JOSÉ EDUARDO MALTA DE SÁ BRANDÃO

COMPOSIÇÕES DE IDSs: VIABILIZANDO O MONITORAMENTO DE SEGURANÇA EM AMBIENTES DE LARGA ESCALA

FLORIANÓPOLIS
2007

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

COMPOSIÇÕES DE IDSs: VIABILIZANDO O MONITORAMENTO DE SEGURANÇA EM AMBIENTES DE LARGA ESCALA

Tese submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Doutor em Engenharia Elétrica.

JOSÉ EDUARDO MALTA DE SÁ BRANDÃO

Florianópolis, Abril de 2007.

COMPOSIÇÕES DE IDSs: VIABILIZANDO O MONITORAMENTO DE SEGURANÇA EM AMBIENTES DE LARGA ESCALA

JOSÉ EDUARDO MALTA DE SÁ BRANDÃO

Esta Tese foi julgada adequada para a obtenção do título de Doutor em Engenharia Elétrica, Área de Concentração em *Sistemas de Informação*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.

Prof. Joni da Silva Fraga, Dr.
Orientador

Prof. Nelson Sadowski, Dr.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

Prof. Joni da Silva Fraga, Dr.
Presidente

Prof. Luciano Paschoal Gaspary, Dr.

Prof. Carlos Alberto Maziero, Dr.

Prof. Jean-Marie Farines, Dr.

Prof. Ricardo José Rabelo, Dr.

A Helana, Izabel e Augusto, que me acompanharam
e me apoiaram nessa grande aventura…

AGRADECIMENTOS

A minha esposa Helana e aos meus filhos Augusto e Izabel, que compartilharam comigo todas as alegrias e momentos difíceis. Sem eles ao meu lado, nada seria possível.

Aos meus pais, Augusto e Maria José, que sempre me deram carinho e apoio.

Aos meus sogros, Helena e Celso, que nos ajudaram nos momentos mais difíceis.

Ao professor Joni Fraga, que me acolheu, orientou e muito me ensinou.

Aos meus colegas de curso, Luiz Otávio, Alysson, Cássia, Michelle, Rafael, Tati, Emerson, Fabio(s), Marcos Vinícius e tantos outros, companheiros de agonias e alegrias, cujo ombro e palavra amiga nunca me faltaram.

Ao Paulo Mafra, que me ajudou muito neste trabalho e nas publicações.

Ao Instituto de Pesquisa Econômica Aplicada (IPEA), que apoiou e tornou possível a realização desta Tese.

Ao CNPq, que financiou parte das publicações resultantes deste estudo.

A banca de qualificação, pelas contribuições que me auxiliaram no desenvolvimento da tese e a banca final, pelas correções sugeridas a este documento.

Resumo da Tese apresentada à UFSC como parte dos requisitos necessários para a obtenção do grau de Doutor em Engenharia Elétrica.

COMPOSIÇÕES DE IDSs: VIABILIZANDO O MONITORAMENTO DE SEGURANÇA EM AMBIENTES DE LARGA ESCALA

José Eduardo Malta de Sá Brandão

Abril/2007

Orientador: Prof. Joni da Silva Fraga, Dr.
Área de Concentração: Sistemas de Informação Industrial
Palavras-chave: Segurança, Detecção de Intrusão, Redes de Computadores
Número de Páginas: 1 + 141

A crescente necessidade de compartilhamento de informações entre organizações parceiras ou membros de organizações virtuais envolve um grande desafio de segurança. Um dos aspectos chave neste desafio é a construção de sistemas de detecção de intrusão (IDSs) que possam operar em ambientes heterogêneos de larga escala. Isto é particularmente difícil devido ao fato de que as diferentes redes envolvidas usam IDSs que não foram projetados para trabalhar de forma cooperativa. O tema central desta tese é a proposta de uma nova abordagem de monitoramento de segurança baseada em composições de IDSs. Uma composição de IDSs é a combinação de elementos de detecção de intrusão, que podem ser IDSs completos ou suas partes, distribuídos entre redes diferentes, mas que operam de forma cooperativa, formando um sistema unificado. Nesta Tese, as composições de IDSs são construídas usando uma arquitetura orientada a serviços baseada na tecnologia de *Web Services*. A interoperabilidade entre os diversos elementos de uma composição é obtida a partir do amplo emprego de esforços de padronização, sobretudo da IETF, W3C e OASIS. As composições dinâmicas são suportadas pelo uso da orquestração de serviços. Para viabilizar as composições de IDSs, é proposta neste documento uma infra-estrutura de serviços, capaz de suportar elementos de IDs baseados em softwares prontos (*commercial off-the-shelf* – COTS), uma necessidade fundamental para prover a interoperabilidade e facilitar a implementação. Nesta Tese também são descritas as implementações de protótipos da infra-estrutura proposta e analisados os resultados obtidos por meio de experimentos com estes protótipos.

Abstract of Thesis presented to UFSC as a partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering.

IDSs COMPOSITIONS: MAKING POSSIBLE THE SECURITY MONITORING IN LARGE-SCALE ENVIRONMENTS

José Eduardo Malta de Sá Brandão

April/2007

Advisor: Prof. Joni da Silva Fraga, Dr.
Area of Concentration: Information Systems
Key words: Security, Intrusion Detection, Computer Networks
Number of Pages: 1 + 141

The growing need for information sharing among partnering organizations or members of virtual organizations poses a great security challenge. One of the key aspects of this challenge is deploying intrusion detection systems (IDS) that can operate in heterogeneous, large-scale environments. This is particularly difficult because the different networks involved generally use IDSs that have not been designed to work in a cooperative fashion. This Thesis presents a model for integrating intrusion detection systems in such environments. The main idea of this Thesis is a new security monitoring approach, based on IDSs compositions. A IDS composition is a combination of intrusion detection elements (which can be complete IDS systems or their components) distributed across different networks so that they operate in a cooperative fashion, in order to provide a unified service. On this Thesis, the IDSs compositions are constructed using a service-oriented architecture (SOA), based on the Web Services technology. The necessary interoperability among the elements of the compositions is achieved through the use of standardized specifications, mainly those developed by IETF, W3C and OASIS. Dynamic compositions are supported through service orchestration. To make possible the IDSs compositions, is proposed on this document a services infrastructure that is capable of supporting commercial off-the-shelf (COTS) IDS elements, a crucial feature for providing interoperability and deployability. On this Thesis are also described prototypes implementations, tests and results.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A popularização da Internet, principalmente para a interação entre parceiros comerciais ou mesmo entre partes de uma mesma organização, dispersas geograficamente, fez nascer uma nova geração de sistemas distribuídos cooperativos e de larga escala. Neste contexto, uma das tendências emergentes são as organizações virtuais, que são formadas quando grupos de organizações compartilham recursos e informações a fim de atingir um objetivo comum. Tal compartilhamento requer um certo nível de integração entre as redes pertencentes a diferentes organizações. Porém, esta integração esbarra freqüentemente em questões de segurança, já que muitas das redes envolvidas foram projetadas para operar de forma isolada, normalmente por trás de barreiras de segurança, como *firewalls* e outros bloqueadores de tráfego. Isto é especialmente problemático quando consideramos que as interações entre organizações virtuais são estabelecidas dinamicamente, o que significa que os ajustes devem ser efetuados "*on-the-fly*".

Um dos desafios envolvidos no monitoramento de segurança, mais especificamente na detecção de intrusão, é justamente a adequação dos atuais modelos de monitoramento aos novos ambientes que envolvem as organizações virtuais ou as múltiplas partes de uma mesma organização. Como ocorre com as redes nas quais os sistemas de detecção de intrusão (*Intrusion Detection Systems - IDSs*) operam, tais IDSs raramente são projetados para funcionar de forma cooperativa com outros IDSs, estando em uma mesma rede ou em redes diferentes. Isso é especialmente incômodo quando os administradores de segurança precisam coletar e correlacionar dados provenientes de redes diferentes a fim de identificar, analisar e bloquear tentativas de ataque. Esta necessidade dificilmente é satisfeita com os modelos atuais de detecção de intrusão. Portanto, novas alternativas precisam ser criadas para suportar os desafios relacionados ao monitoramento de segurança nestes novos ambientes.

Nessa Tese é proposta uma nova abordagem, a qual chamamos de composição de sistemas de detecção de intrusão, ou simplesmente de composição de IDSs. **Uma composição de IDSs é a combinação de elementos de detecção de intrusão, que podem ser IDSs completos ou suas partes, distribuídos entre redes diferentes, mas que operam de forma cooperativa, formando um sistema unificado**.

As composições de IDSs coletam e analisam dados de forma distribuída e oferecem a flexibilidade da configuração dinâmica para atender novas situações, mesmo que temporárias. As composições de

IDSs fazem uso extensivo de esforços de padronização e estão fundamentadas em uma infra-estrutura de serviços e suportes. A adoção desses padrões torna possível a interoperabilidade e a comunicação entre elementos de uma composição. Os IDSs materializados a partir da infra-estrutura proposta seguem a arquitetura orientada a serviços suportada pela tecnologia de *Web Services* [W3C, 2004], com o amplo uso de textos XML [Bray et al., 2004].

1.1 Motivações

1.1.1 Questões Iniciais

Nas últimas duas décadas são observados diversos avanços na área de monitoramento de segurança, sobretudo na construção de sistemas de detecção de intrusão. Tais avanços produziram novas técnicas de análise e novas abordagens de detecção de intrusão. Apesar dos avanços e do sucesso comercial do tema, algumas questões permanecem sem uma solução apropriada, principalmente para enfrentar os novos desafios gerados pelo uso da Internet no relacionamento entre organizações. Infelizmente, as abordagens atuais ainda não tratam de forma eficiente essa questão. Entre os desafios a serem enfrentados, a literatura destaca o desenvolvimento de mecanismos de resposta a ataques, de arquiteturas para sistemas de detecção de intrusão altamente distribuídos, de padrões de interoperabilidade e de novos paradigmas para detecção de intrusão.

1.1.2 Resposta a Incidentes de Segurança

A resposta a incidentes de segurança esbarra na dificuldade em se conseguir informações precisas sobre incidentes em andamento e no perigo potencial de danos que ações de respostas mal fundamentadas possam causar. Sistemas de detecção de intrusão de larga escala permitem a realização de correlações de incidentes de segurança com informações que não podem ser obtidas em uma única rede, como, por exemplo, para a detecção de ataques de negação de serviço distribuídos. Também pode ser útil para identificar com maior precisão os ataques que se originam das redes que estão sendo monitoradas, corrigindo com maior rapidez os problemas. Portanto, um modelo que permita a integração dinâmica de IDSs em ambientes distintos é fundamental para o desenvolvimento e a aplicação de novas técnicas de resposta a ataques.

1.1.3 Diversidade

Outra questão a ser levantada está relacionada à diversidade e redundância no uso de sistemas de detecção de intrusão, a qual envolve aspectos de desempenho e segurança destes IDSs. A utilização de algoritmos de detecção diferentes em um mesmo ambiente de forma redundante aumenta as chances de que uma maior quantidade de intrusões possa ser detectada e com menos alarmes falsos. A decomposição de um sistema de detecção de intrusão em componentes distintos distribuídos já é aplicada para melhorar o desempenho dos IDSs. Com o crescimento do número de sistemas de detecção

de intrusão, verifica-se que alguns deles se especializam em identificar vulnerabilidades, na detecção de determinados tipos de ataques ou no monitoramento de certos aspectos de segurança, enquanto que outros são construídos de forma a armazenar e/ou correlacionar dados de origens diferentes. Há também os que primam pela interface com o usuário.

Vulnerabilidades e ameaças fazem parte do dia a dia das organizações que utilizam sistemas distribuídos, refletindo também nos sistemas de monitoramento. Uma das formas de contornar este problema é o uso de diversidade e redundância de componentes, pois é esperado que as vulnerabilidades de um dos diferentes sistemas não afetem da mesma forma a todos os outros. Portanto, o emprego de uma arquitetura que permita a diversidade de técnicas e mecanismos irá contribuir para melhorar a confiabilidade do modelo de segurança como um todo.

Portanto, em um ambiente ideal, sistemas de segurança de fabricantes diferentes poderiam ser usados a fim de se obter o desempenho máximo de cada um, em sua especialidade. As vulnerabilidades e limitações de uma ferramenta seriam compensadas por outras. A combinação de tecnologias apropriadas seria saudável na solução de situações em ambientes complexos. O usuário ainda poderia selecionar os sistemas de gerenciamento mais adequados a seu ambiente e a interface gráfica que melhor lhe conviesse. Infelizmente, os sistemas de detecção de intrusão atuais carecem de uma infra-estrutura que permita tal diversidade.

1.1.4 Padronização

Novos padrões que auxiliem na prevenção, detecção e resposta a intrusões, estão em desenvolvimento, principalmente pelo IETF (*Internet Engineering Task Force*). Contudo, a aplicação destes padrões ainda ocorre timidamente, tanto na indústria, quanto na área científica. Em muitas ocasiões estes padrões acabam sendo distorcidos a fim de reduzir custos de comunicação ou para implementar novas funcionalidades, acarretando em uma inevitável incompatibilidade. Portanto, a adoção dos padrões originais é imprescindível para se conseguir a tão almejada interoperabilidade entre sistemas de segurança.

1.1.5 Novos Paradigmas

Alguns paradigmas, como o uso de agentes móveis para detecção de intrusão e outras aplicações, na prática não se mostraram viáveis, principalmente por questões de segurança. Por outro lado, o uso da tecnologia de *Web Services* para implementação de sistemas envolvendo organizações virtuais vem ganhando força e indica uma tendência para o desenvolvimento de novos aplicativos, com grandes apostas por parte da indústria de software. Tal crescimento reflete-se na quantidade de padrões que estão em estudo e de ferramentas de desenvolvimento que estão surgindo. Portanto, a associação da tecnologia de *Web Services* com a detecção de intrusão em redes de larga escala parece uma solução adequada.

A opção de uso e compartilhamento de elementos de detecção de intrusão dispersos geograficamente entre organizações e redes distintas, mas parceiras, abre um novo leque de possibilidades

para o desenvolvimento de novas ferramentas e técnicas de detecção de intrusão, abrindo caminho para um novo paradigma nesta área. Contudo, algumas questões ainda precisam ser consideradas no desenvolvimento de uma nova abordagem para detecção de intrusão.

1.2 Visão Geral da Tese

1.2.1 Hipóteses para o Desenvolvimento do Trabalho

Dado o contexto do ambiente característico das organizações virtuais e da expansão das redes de computadores nas organizações tradicionais, as soluções para a criação e implantação de sistemas de detecção de intrusão devem seguir novos modelos. Sobre esta questão, assumimos como hipótese fundamental desta Tese, que um destes novos modelos para viabilizar o monitoramento de segurança em ambientes de larga escala é a composição de sistemas de detecção de intrusão. Desta forma, buscaremos apresentar elementos teóricos e experimentais para fundamentar as propostas apresentadas nesta Tese.

O desenvolvimento deste novo modelo de detecção de intrusão requer ainda que sejam assumidas algumas suposições (ou hipóteses associadas), que aqui são apresentadas na forma de requisitos mínimos.

Requisito 1: Detecção de Intrusão Distribuída

A detecção de intrusão distribuída prioriza a distribuição de elementos envolvidos na detecção de intrusos, que devem trocar informações entre si sem a necessidade de elementos centrais, evitando os gargalos ou pontos únicos de falha que se formam com a centralização de funções ou componentes de um IDS.

Requisito 2: Uso de Elementos de Detecção de Intrusão Heterogêneos

Nem sempre um IDS é capaz de coletar, analisar e gerenciar informações provenientes de diversos níveis e de diferentes ambientes. O modelo de detecção de intrusão proposto deve permitir a integração de ferramentas previamente existentes, mesmo que de diferentes fabricantes. O modelo deve permitir, tanto o uso de IDSs completos e independentes, quanto a integração de componentes de um IDS. Os componentes de um IDS poderão ser, por exemplo, baseados em sistemas abertos ou softwares prontos (*commercial off-the-shelf – COTS*), que implementam ferramentas para segurança de perímetro e IDSs comerciais (monolíticos ou distribuídos), além de IDSs baseados em agentes (móveis ou fixos) etc.

Requisito 3: Composição Dinâmica de Sistemas de Detecção de Intrusão

Os sistemas de detecção de intrusão poderão ser compostos de forma dinâmica, envolvendo mesmo o compartilhamento, entre IDSs, de componentes que poderão ser entidades autônomas. Uma composição pode ser permanente ou temporária, definida com a finalidade, por exemplo, de coletar dados de determinados sensores, de pesquisar diversas bases de dados de eventos ou de compartilhar informações sobre um ataque em andamento.

Requisito 4: Interoperabilidade

A composição de sistemas de detecção de intrusão deve atender às necessidades de ambientes fechados de médias e grandes empresas, mas principalmente as de ambientes abertos que fazem uso da Internet, como as organizações virtuais. Para isso, a interoperabilidade é um fator imprescindível, visto que tais ambientes possuem características e ambientes operacionais diversificados, muitas vezes fornecidos por diferentes fabricantes. Portanto, é necessário nestas composições que seus elementos "falem a mesma língua", ou seja, possuam formas padronizadas de comunicação e de integração. Para viabilizar tais sistemas, enfocamos a necessidade da adoção de padrões que permitam a interoperabilidade nestas composições distribuídas.

Requisito 5: Segurança dos Elementos e da Composição

A segurança dos próprios IDSs é obviamente crítica para qualquer sistema sendo monitorado. Para tal, é necessário o uso de mecanismos que possam garantir as propriedades de segurança das informações trocadas ou manipuladas nessas composições distribuídas de IDSs.

1.2.2 Objetivos

O objetivo geral deste trabalho é **propor soluções que permitam viabilizar a detecção de intrusão em ambientes de larga escala**. A partir deste objetivo principal e considerando os requisitos apresentados anteriormente, são identificados os seguintes objetivos específicos:

1. Definir e desenvolver uma infra-estrutura de integração que permita a composição de sistemas de detecção de intrusão distintos e dispersos geograficamente.
2. Propor na infra-estrutura mecanismos de comunicação padronizados que permitam a interoperabilidade entre IDSs distintos.
3. Agilizar a composição dinâmica de novos sistemas de detecção de intrusão, a partir do suporte da infra-estrutura proposta.
4. Adotar especificações baseadas nos esforços de padronização da IETF, W3C e OASIS a fim de atingir a interoperabilidade necessária.
5. Fazer uso extensivo da linguagem XML e da tecnologia *Web Services* de programação distribuída orientada a serviços.

1.2.3 Resultados Esperados

Como resultado deste estudo, pretendemos apresentar um *framework* mínimo para a composição de sistemas de detecção de intrusão, baseado na tecnologia de *Web Services* e levando em conta os requisitos citados anteriormente.

1.2.4 Validação da Proposta

Na primeira parte do trabalho, foi necessário identificar e avaliar os diversos caminhos e tecnologias passíveis de apliação para o desenvolvimento de um novo modelo de detecção de intrusão. Para isso, foi adotada a metodologia de gestão de riscos, que será apresentada no próximo capítulo.

A validação do framework proposto é feita usando métodos quantitativos e qualitativos. Foram desenvolvidos alguns protótipos e realizados testes de funcionamento com avaliações qualitativas dos mesmos. Também foram calculadas algumas medidas de desempenho destes protótipos a fim de verificar sua viabilidade.

1.3 Organização do Texto

Neste capítulo foi mostrado o contexto no qual são ambientadas as propostas apresentadas nessa Tese. Também foram descritos os objetivos dessa Tese e apresentadas as motivações que levaram à sua elaboração. O restante desse documento está organizado conforme a descrição a seguir.

No capítulo seguinte serão introduzidos os principais conceitos usados ao longo desse documento. Entre os conceitos apresentados, destacam-se aqueles relacionados à segurança em ambientes de tecnologia da informação (TI) e seus desdobramentos. Também são apresentados conceitos e tecnologias relacionadas ao uso de *Web Services*. Neste capítulo também serão apresentados os conceitos relacionados à metodologia de gestão de riscos.

No terceiro capítulo são discutidos os conceitos, as arquiteturas, os padrões e os principais trabalhos relacionados à detecção de intrusão distribuída, foco desta Tese. Os trabalhos relacionados são contextualizados e comparados à proposta aqui apresentada. Também é apresentada uma taxonomia para classificação de elementos de detecção de intrusão, a qual será adotada na proposta.

No quarto capítulo é apresentada uma proposta para a composição de sistemas de detecção de intrusão em ambientes de larga escala. Para descrever a proposta é definido um modelo geral para composição de IDSs, suportado por uma infra-estrutura de serviços e pelo uso de *Web Services*. As características dos serviços que compõem essa infra-estrutura também são detalhadas.

No quinto capítulo é apresentada a materialização da proposta de composição de IDSs. Nele são detalhados os protótipos e experimentos desenvolvidos e são analisados os resultados dos testes realizados.

Finalmente, o sexto capítulo traz as considerações finais desse documento, com os principais resultados e conclusões da Tese.

Em anexo são apresentados os resultados da aplicação da metodologia de gestão de riscos.

Capítulo 2

Principais Conceitos

2.1 Introdução

Diversos conceitos tratados neste estudo são descritos neste capítulo. Entre esses conceitos podemos destacar aqueles relacionados à segurança em tecnologia da informação (TI), ao monitoramento de segurança e à tecnologia de *Web Services*. Dois desses assuntos são abordados em maior profundidade neste capítulo: a segurança em TI e a tecnologia de *Web Services*. Os conceitos relacionados à detecção de intrusão serão aprofundados no capítulo seguinte.

Na seção 2.2 são apresentados os principais conceitos relacionados à segurança. Nesta seção serão apresentadas as propriedades de segurança e definidos os conceitos de políticas de segurança, modelos de segurança, vulnerabilidades, ameaças, ataques, intrusão, risco, mecanismos de segurança, auditoria e detecção de intrusão. Na seção 2.3 é mostrado um resumo dos conceitos sobre a tecnologia de *Web Services*, com destaque à arquitetura orientada a serviços, às principais tecnologias relacionadas e à própria segurança para as interações entre *Web Services*. Outros detalhes e padrões associados a *Web Services* adotados nesta Tese serão apresentados nos próximos capítulos.

2.2 Conceitos de Segurança

2.2.1 Propriedades de Segurança

Segurança em sistemas informáticos é identificada como a capacidade de assegurar a prevenção ao acesso e à manipulação ilegítima da informação, ou ainda, de evitar a interferência indevida na sua operação normal [ISO/IEC, 2005a].

A segurança é fundamentada em três propriedades básicas [Bishop, 2003] [ISO/IEC, 2005a]:

- **Integridade**: garante que a informação não será alterada ou destruída sem a autorização adequada.

- **Confidencialidade**: garante que a informação não será revelada sem a autorização adequada.
- **Disponibilidade**: garante que a informação estará acessível aos usuários legítimos quando solicitada.

A propriedade de integridade inclui também, mas não exclusivamente, a **autenticidade** e a **não repudiação** [US Department of Homeland Security, 2002] [Barker e Lee, 2004]. A propriedade de autenticidade garante que a identidade de um sujeito ou recurso é aquela alegada, sendo aplicada a entidades como usuários, processos, sistemas e informações [ISO, 1989]. A não repudiação garante que uma parte neutra possa ser convencida de que uma transação particular ou um evento tenha ou não ocorrido.

Para o Governo Norte Americano, os objetivos da segurança são preservar a integridade, a disponibilidade e a confidencialidade dos recursos dos sistemas de informações, incluindo: hardware, software, *firmware*, informação/dados e telecomunicações [Ross et al., 2005].

Quando há a quebra de uma ou mais propriedades de segurança, há uma **violação de segurança**. Portanto, como as violações estão relacionadas com as três propriedades básicas, as mesmas podem ser classificadas também em três categorias:

- Revelação não autorizada da informação (violação de confidencialidade);
- Modificação não autorizada da informação (violação de integridade);
- Negação de serviço (violação de disponibilidade).

2.2.2 Política de Segurança

Uma **política de segurança** forma a base para o estabelecimento do que é ou não permitido [Bishop, 2003] . Formalmente, uma política de segurança é uma afirmação que divide os estados do sistema em um conjunto de estados autorizados, ou seguros, e em um conjunto de estados não autorizados, ou não seguros. Dessa forma, um sistema seguro é um sistema que se inicia em um estado autorizado e evolui entre estados seguros.

Infelizmente, na prática, tal definição dificilmente pode ser garantida ou mantida. Os sistemas tornaram-se complexos nos dias de hoje e também evoluem em configuração e funcionalidade, tornando árdua a garantia de um sistema seguro. Mas, em síntese, podemos afirmar que as políticas de segurança normalmente descrevem em linguagem natural, o que usuários e equipes técnicas podem fazer.

2.2.3 Modelo de Segurança

Um modelo que represente uma política particular ou conjunto de políticas é conhecido como **modelo de segurança** [Bishop, 2003]. Quando usados para descrever o comportamento de um sistema diante de políticas de autorização, modelos matemáticos formais de segurança permitem, de

certa maneira, verificações de que a política é coerente, e servem de guia para implementações de esquemas de autorização correspondentes às especificações contidas no modelo.

2.2.4 Vulnerabilidade

Uma **vulnerabilidade** é um defeito ou fraqueza no design ou na implementação de um sistema de informações (incluindo procedimentos de segurança e controles de segurança associados ao sistema), que pode ser intencionalmente ou acidentalmente explorada, afetando a confidencialidade, integridade ou disponibilidade [Ross et al., 2005].

2.2.5 Ameaças, Ataques e Intrusão

A vulnerabilidade, por si só, não representaria perigo se não houvesse a possibilidade da mesma ser explorada. Portanto, uma **ameaça** é qualquer circunstância ou evento com o potencial intencional ou acidental de explorar uma vulnerabilidade específica em qualquer sistema computacional, resultando na perda de confidencialidade, integridade ou disponibilidade [Barker e Lee, 2004]. Atos intencionais que podem produzir violações de segurança são chamados de **ataques**. Finalmente, quando um ataque é bem sucedido, afirmamos que houve uma **intrusão**.

Os ataques podem ser divididos entre **passivos** e **ativos** [Stallings, 2000]. Os ataques passivos possibilitam a obtenção de informações que estão sendo transmitidas ou processadas, sem afetá-las. Nos ataques ativos, os dados são inseridos, adulterados ou ficam indisponíveis.

Um exemplo dessa classificação, que serve de base para entender os ataques mais comuns em sistemas distribuídos, está na Figura 2.1 [Stallings, 2000]. Neste exemplo, os ataques passivos, divididos em revelação do conteúdo de mensagens e análise de tráfego, representam algum tipo de espionagem (*eavesdroping*) e podem acarretar a quebra da propriedade de confidencialidade. Já as ameaças ativas podem ser representadas pelo mascaramento, *replay*, modificação e negação de serviço. A revelação de conteúdo, a análise de tráfego, a modificação de mensagens e a negação de serviço são ameaças bastante conhecidas. O mascaramento corresponde à personificação ou disfarce de uma entidade a fim de obter privilégios atribuídos a uma outra entidade, enquanto o *replay* envolve a captura passiva de uma unidade de dados e sua subseqüente retransmissão para produzir algum tipo de violação.

2.2.6 Risco

O **risco** é o impacto negativo da exploração de uma vulnerabilidade, considerando a probabilidade do uso do mesmo e o impacto da violação [Stoneburner et al., 2002]. Ou seja, o risco é uma tentativa de quantificar as possibilidades de violação e os prejuízos decorrentes do mesmo. Os riscos podem ser identificados e reduzidos, mas nunca totalmente eliminados [Garfinkel et al., 2003].

O risco pode ser expressado matematicamente como uma função da probabilidade de uma origem de ameaça (ou atacante) explorar uma vulnerabilidade potencial e do impacto resultante deste evento adverso no sistema e, conseqüentemente, na empresa ou organização.

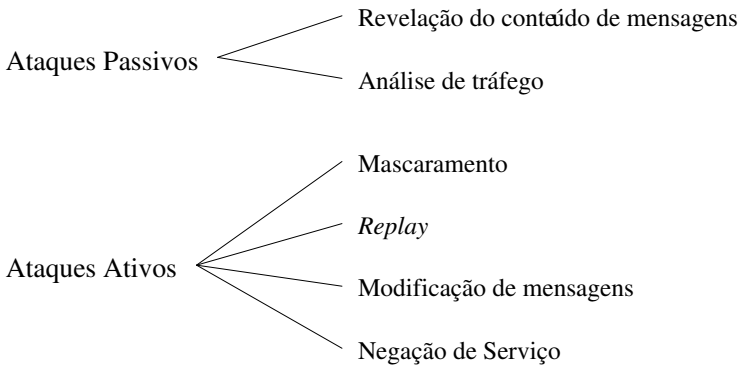

Figura 2.1: Exemplo de classificação de ataques ativos e passivos

2.2.7 Mecanismos de Segurança

Um **mecanismo de segurança** é um método, uma ferramenta ou um procedimento para implementar uma política de segurança [Bishop, 2003] . A distinção entre políticas de segurança e mecanismos de segurança é útil quando estamos projetando sistemas seguros. Normalmente, é desejável que as políticas sejam independentes dos mecanismos de segurança.

Os mecanismos de segurança, da maneira como são concebidos, têm por objetivo prevenir e detectar ataques a sistemas, além de recuperar o alvo do ataque [Bishop, 2003] . A **prevenção** deve evitar que ataques sejam bem sucedidos. A **detecção** é útil quando um ataque não pode ser prevenido, identificando a ocorrência do mesmo com relação à sua natureza, severidade e resultado. A detecção será alvo de maior discussão nos próximos capítulos.

A **recuperação** possui duas formas. A primeira interrompe o dano e tenta reparar o mesmo. Na segunda forma de recuperação, o sistema continua funcionando corretamente enquanto o ataque está sendo realizado e sua recuperação é providenciada. Essa forma de recuperação coincide com a definição de **tolerância a intrusões** [Fraga e Powell, 1985], na qual um sistema deve continuar funcionando sob ataque, ainda que com seu desempenho reduzido.

Os mecanismos de segurança que envolvem a prevenção compõem a primeira linha de defesa. Cada propriedade de segurança é associada a controles específicos de prevenção. Um destes controles de segurança que procura manter a confidencialidade das informações, por exemplo, é o provido pela **criptografia**. Além da confidencialidade, os controles criptográficos podem fornecer meio de checagem da integridade das informações e da não repudiação de transações. Outro destes controles é o **controle de acesso**, que é usado no sentido de prover a integridade e também a confidencialidade das informações. Os controles referentes à negação de serviço são mais complicados, pois envolvem a checagem de qualquer recurso compartilhado.

É de responsabilidade dos controles de segurança a geração de evidências que demonstrem como uma transação específica foi iniciada, processada e encerrada. Tais dados são conhecidos como **trilhas de auditoria**. A trilha de auditoria também pode corresponder a registros cronológicos que sinalizam quando usuários entram no sistema, por quanto tempo permanecem engajados em atividade-

des e se está ocorrendo ou ocorreram tentativas de violação da segurança nas suas ações no sistema. Os registros contidos em trilhas de auditoria devem ser suficientes para permitir a reconstrução, a revisão e o seqüenciamento das ações no ambiente durante uma transação, desde o seu início até a saída dos resultados finais [NCSC, 1996]. Portanto, as trilhas de auditoria são utilizadas para detectar e dissuadir possíveis violações de segurança nos sistemas computacionais e identificar um mau uso dos mesmos [NCSC, 1988].

2.2.8 Monitoramento de Segurança

O estudo de planejamento de tecnologia de segurança computacional feito por Anderson [1972] realizava uma tentativa de definição do que seria necessário para se estabelecer uma vigilância dos sistemas computacionais. As questões levantadas naquele trabalho, sobre o que analisar, como analisar e como proteger o sistema de vigilância e suas informações, permanecem como objeto de estudo até hoje [McHugh, 2001].

Infelizmente, não há um consenso com relação à definição de **Auditoria de Segurança** em sistemas relacionados à tecnologia da informação e comunicação. Neste documento será utilizado o conceito de Auditoria de Segurança definida por Sandhu e Samarati [1996], no qual o processo de auditoria colhe dados sobre atividades em um sistema, analisando-os para descobrir violações de segurança ou diagnosticar suas causas.

Nesse contexto, o documento *A Guide to Understanding Audit in Trusted Systems* [NCSC, 1988] define cinco objetivos que devem nortear a auditoria de segurança e os mecanismos que possam efetivá-la:

1. Permitir a revisão de padrões de acesso a recursos individuais, de históricos de acessos de processos e indivíduos específicos e do uso dos vários mecanismos de proteção suportados pelo sistema, bem como suas eficácias.
2. Permitir a descoberta de tentativas de burlar os mecanismos de proteção.
3. Permitir a descoberta de uso indevido de privilégios.
4. Agir como elemento desencorajador de tentativas de burlar os mecanismos de proteção.
5. Prover uma segurança adicional aos usuários, garantindo-lhes que tentativas de burlar os mecanismos de proteção sejam registradas e descobertas.

A auditoria de segurança pode ser motivada pela ocorrência de algum fato ou demanda, analisando, portanto, dados gerados antes da auditoria. Nesses casos, a auditoria geralmente é realizada off-line utilizando técnicas de investigação de arquivos. Quando a auditoria é realizada on-line, ela normalmente recebe a denominação de **detecção de intrusão** [Sandhu e Samarati, 1996]. Os sistemas que implementam essa forma de auditoria são conhecidos como **Sistemas de Detecção de Intrusão (IDS – *Intrusion detection Systems*)**. Mais detalhes sobre IDSs serão apresentados no próximo capítulo.

2.3 Gestão de Riscos

O aumento da complexidade dos atuais sistemas de detecção de intrusão, com códigos cada vez maiores, interações diversificadas e o uso de componentes externos, torna a tarefa de avaliá-los cada vez mais difícil. Na literatura científica são apresentadas diversas tentativas de avaliar a eficiência de sistemas de detecção de intrusão [Puketza et al., 1996] [Debar et al., 1998] [Durst et al., 1999] [Lippmann et al., 2000a] [Lippmann et al., 2000b] [McHugh, 2000] [Athanasiades et al., 2003]. Nestes casos, uma implementação é avaliada por baterias de testes para verificar o comportamento do IDS e suas reações. Porém, nenhuma destas experiências foca o desenvolvimento seguro dos sistemas de detecção de intrusão ou avalia a segurança dos mesmos.

Infelizmente a avaliação de segurança não pode provar que um sistema é invulnerável a ataques, mas somente que o mesmo mostra um certo grau de confiança nos propósitos a que se destina.

A avaliação de segurança em projetos científicos é particularmente difícil, pois os mesmos são freqüentemente desenvolvidos como provas de conceito e não como produtos acabados. Quando um projeto de software científico, como o apresentado nesta Tese, envolve a integração de múltiplas tecnologias, a identificação e a minimização das vulnerabilidades é ainda mais complicada. Tratar este tipo de projeto apenas no ponto de protótipo pode ser extremamente dispendioso e, em alguns casos, os resultados podem inviabilizar o próprio projeto.

Alguns esforços para padronização de metodologias para avaliação de sistemas de tecnologia da informação vêm sendo realizados. A avaliação por testes baseia-se na verificação de *checklists* e na execução de testes para verificar se determinado sistema ou produto pronto encontra-se de acordo com especificações mínimas de segurança, estabelecidas previamente. Outros tipos de avaliação buscam acompanhar de forma sistemática o projeto de um sistema ou produto, garantindo que o mesmo foi desenvolvido seguindo especificações e boas práticas de segurança. Enquanto a avaliação por testes se aplica em sistemas prontos, a avaliação analítica ocorre durante todas as etapas do processo de desenvolvimento.

O *Common Criteria* (CC) [ISO/IEC, 2005a] [ISO/IEC, 2005b] [ISO/IEC, 2005c] é um exemplo de uma metodologia de testes e acompanhamento de projeto. Ele busca avaliar produtos de segurança, fornecendo também subsídios para uma certificação de segurança destes produtos. Para cada classe de produtos são definidos critérios que devem ser verificados. Entre os produtos que podem ser certificados estão os sistemas de detecção de intrusão.

Um outro exemplo de metodologia para avaliação é a gestão de riscos. A gestão de riscos baseia-se em princípios e boas práticas de gerenciamento e segurança [Swanson e Guttman, 1996], para auxiliar na tomada de decisões. Duas ferramentas metodológicas estão disponíveis para o desenvolvimento da gestão de riscos. São elas a especificação SP800-30 [Stoneburner et al., 2002] desenvolvida pelo NIST[1] e a especificação AS/NZ4360 [AS/NZS, 2004a] desenvolvida pelos governos da Austrália e Nova Zelândia.

[1] http://www.nist.gov

A metodologia de gestão (ou gerenciamento) de riscos vai além da análise de vulnerabilidades de um produto ou protótipo. Ela envolve um processo criterioso e recursivo de documentação, avaliação e decisão durante todas as fases do ciclo de vida do projeto. Nesta Tese, a gestão de riscos foi aplicada para fornecer subsídios ao desenvolvimento do projeto e à avaliação de segurança.

A seguir serão apresentados resumos das principais metodologias de avaliação de sistemas de tecnologia da informação.

2.3.1 Common Criteria (CC)

O conjunto de especificações ISO [ISO/IEC, 2005a] [ISO/IEC, 2005b] [ISO/IEC, 2005c] que formam o *Common Criteria* são derivados de padrões desenvolvidos anteriormente: o TCSEC (livro laranja) desenvolvido pelo governo dos EUA; o CTCPEC, desenvolvido pelo governo canadense; e o ITSEC, criado pelos países europeus. Criado inicialmente em 1995, o CC sofreu revisões recentes.

O CC se baseia em uma linguagem e numa estrutura comuns para expressar requisitos de segurança de sistemas e produtos de tecnologia da informação (TI). Tais sistemas e produtos são chamados de **alvos da avaliação** (*target of evaluation* - ***TOE***). Baseado no CC são desenvolvidos **perfis de proteção** (*protection profiles* - ***PP***) e **alvos de segurança** (*security targets* - ***ST***), que através de requisitos especificam o que o sistema deve fazer. O PP especifica um conjunto de requisitos de segurança, independentes de implementação, para uma categoria de TOEs, como por exemplo, sistemas de detecção de intrusão. O ST define um conjunto de requisitos e especificações para ser usado como base para avaliação de um TOE específico, como por exemplo um IDS de determinado fabricante. Um ST de um produto pode incorporar requisitos ou declarar conformidade com um ou mais PPs.

O TOE, após ter seu ST avaliado com relação aos PPs próprios, recebe uma certificação de nível de garantia (*Evaluation Assurance Level* - *EAL*), que o classifica segundo uma escala progressiva (de EAL1 a EAL7) de características de segurança. O nível EAL1 certifica que o TOE teve seu funcionamento testado. O nível EAL2 estabelece que o sistema teve sua estrutura testada e envolve a cooperação do fabricante. O nível EAL3 certifica que o TOE foi metodicamente testado e checado. O nível EAL4 define que o sistema foi metodicamente projetado, testado e checado. O nível EAL5 prevê que o sistema seja projetado e testado de maneira semiformal. O nível EAL6 sustenta que o TOE foi projetado, verificado e testado de maneira semiformal. Por último, o nível EAL7 certifica que o sistema foi projetado, verificado e testado de maneira formal.

O CC considera que a segurança pode ser obtida durante as fases de desenvolvimento, avaliação e operação do TOE. No desenvolvimento, a segurança é obtida com refinamentos dos requisitos de segurança, gerando uma especificação sumária do TOE presente em um ST. Na fase de operação, podem surgir vulnerabilidades no TOE, exigindo modificações no sistema e a reavaliação da segurança. Na fase de avaliação, o TOE é verificado com base no ST e envolve análise e testes do produto.

No caso específico de sistemas de detecção de intrusão, existem dois conjuntos de perfis de proteção disponíveis. O primeiro conjunto padronizado de PPs é aplicado a ambientes com requisitos de segurança baixos e são classificados com nível EAL2. O segundo conjunto está em fase de

desenvolvimento e é direcionado a ambientes que requerem segurança média (*Medium Robustness Environments*), recebendo nível EAL4. Neste último caso, são considerados os valores dos ativos monitorados e dos dados analisados.

Para efeitos de classificação no CC, os IDSs são divididos em quatro componentes, cada um com seu próprio perfil de proteção: sistemas de IDS (*Intrusion Detection System System*), que envolve o IDS como um todo; sensores (*Intrusion Detection System Sensor*), que correspondem a componentes de um IDS que coletam eventos que podem ser indicativos de vulnerabilidades ou mau uso de recursos de TI; sistemas de varredura (*Intrusion Detection System Scanner*), que correspondem a componentes de um IDS que coletam informações estáticas de configuração que possam indicar um potencial para uma intrusão futura ou a ocorrência de uma intrusão anterior em um sistema de TI; e analisadores (*Intrusion Detection System Analyzer*), que recebem e analisam dados provenientes de sensores e sistemas de varredura.

Existem atualmente (Agosto/2006) 16 produtos de segurança validados na categoria de "sistemas de detecção de intrusão e sistemas de prevenção de intrusos"[2]. Destes, apenas três possuem conformidade com PPs de detecção de intrusão, todos classificados com nível EAL2.

2.3.2 NIST SP800-30

O NIST (*National Institute of Standards and Technology*) disponibiliza uma série de publicações relacionadas à tecnologia da informação. Entre estas publicações está a recomendação SP800-30 (*Risk Management Guide for Information Technology Systems*)[Stoneburner et al., 2002]. O documento fornece a fundamentação para o desenvolvimento de um programa de gestão de riscos, contendo as definições e as direções necessárias para avaliar e atenuar os riscos identificados em sistemas de TI.

A metodologia de gestão de riscos da especificação SP800-30 consiste de dois processos: avaliação de riscos (ou determinação dos riscos) e atenuação de riscos. Além destas duas etapas, é recomendada a revisão periódica de todo o processo. O processo de avaliação de riscos segue um fluxo de atividades, conforme ilustrado na Figura 2.2.

O primeiro passo da metodologia consiste em caracterizar o sistema implementado, descrevendo o ambiente ao qual está vinculado, a sua missão, os seus requisitos funcionais, as suas interfaces, as pessoas envolvidas no suporte, os dados e informações etc. As etapas 2,3,4 e 6 podem ser conduzidas em paralelo após a caracterização do sistema.

A segunda etapa visa identificar as ameaças que podem explorar vulnerabilidades do sistema. A terceira etapa consiste na identificação de vulnerabilidades ou falhas que podem ser exploradas no sistema. Este passo também inclui a identificação da origem das vulnerabilidades e como elas podem ser exploradas.

O objetivo da quarta etapa é analisar os controles de segurança que estão implementados ou pretende-se implantar. os controles podem ser preventivos, quando inibem as tentativas de violação

[2] http://niap.bahialab.com/cc-scheme/vpl/vpl_type.cfm#ids

de segurança, ou podem visar a detecção de possíveis ataques. Os controles também podem ser classificados como técnicos e não técnicos. Os controles técnicos podem ser incorporados no hardware ou software dos sistemas. Os controles não técnicos envolvem controles de gerenciamento e operacionais, como: políticas de segurança, procedimentos operacionais, gestão de pessoal, controles físicos ou ambientais.

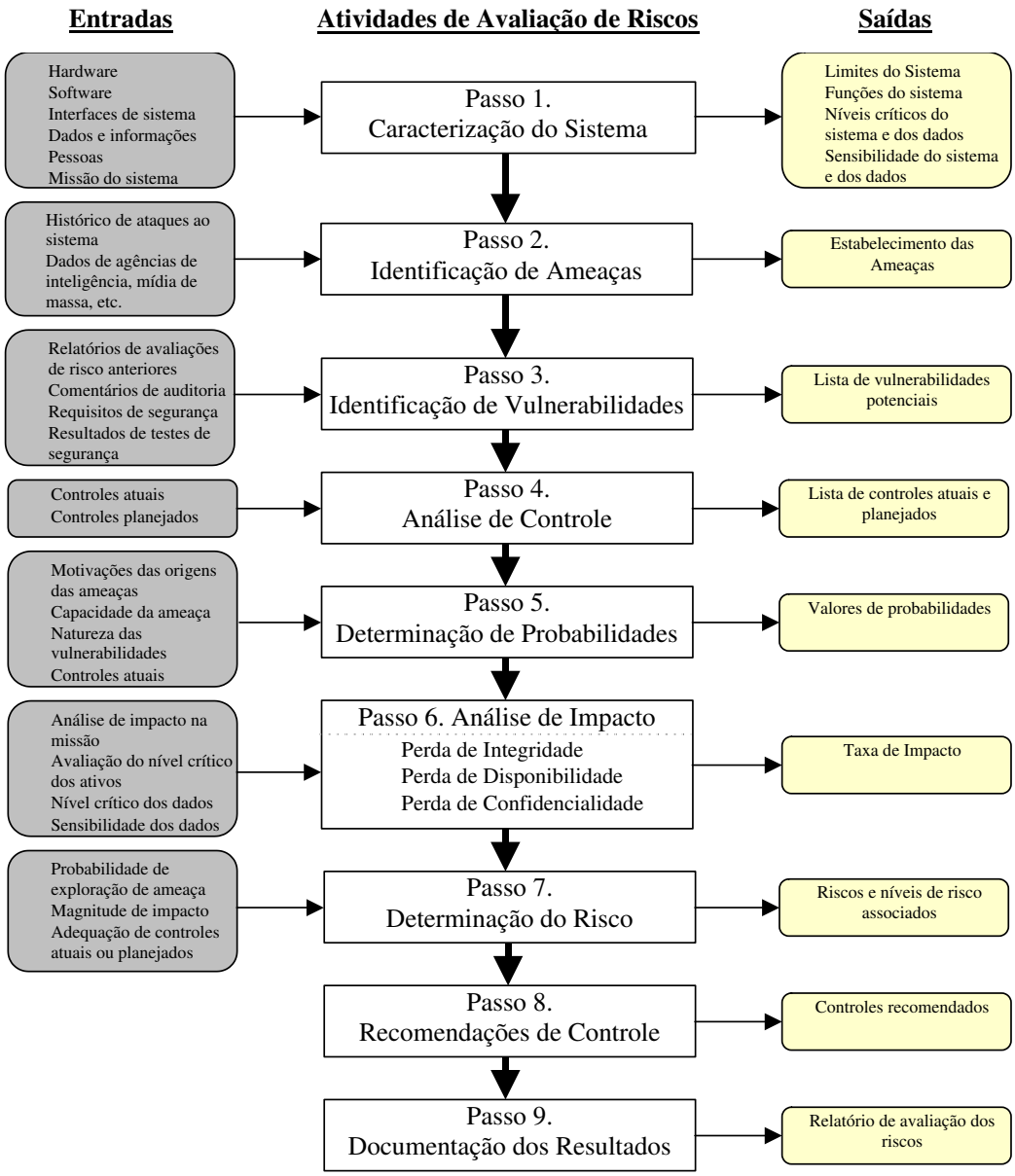

Figura 2.2: Metodologia de gestão de riscos da especificação SP800-30 [Stoneburner et al., 2002]

A quinta etapa define os valores das probabilidades de exploração de uma potencial vulnerabilidade, por uma ameaça. A metodologia define três níveis subjetivos de probabilidade: alta, média e baixa. A probabilidade será alta se o possível atacante estiver altamente motivado e for suficientemente capaz de explorar uma vulnerabilidade cujos controles forem ineficazes. A probabilidade será média se o possível atacante estiver motivado e capaz para explorar uma vulnerabilidade, mas os controles podem impedir com sucesso que a mesma seja explorada. A probabilidade baixa será atribuída

se o possível atacante não estiver motivado ou não for capaz de explorar uma vulnerabilidade ou se os controles podem prevenir ou impedir que a mesma seja explorada.

A análise de impacto corresponde à sexta etapa e visa determinar os danos potenciais que o resultado adverso de um ataque ou violação bem sucedida causa ao sistema. O impacto de um evento de segurança pode ser descrito em termos de perda ou degradação de qualquer uma, ou de uma combinação de quaisquer, das propriedades de segurança: integridade, disponibilidade e confidencialidade. A análise do impacto pode ser feita utilizando avaliações quantitativas ou qualitativas.

A determinação dos níveis de risco é a sétima etapa. Nela é determinado o grau de suscetibilidade ao risco que cada vulnerabilidade representa, considerando a probabilidade da mesma ser explorada, a magnitude do impacto adverso que o fato causaria e a adequação dos controles para reduzir ou eliminar os riscos. Matrizes ou gráficos podem ser utilizados para combinar estes fatores e determinar quantitativamente ou qualitativamente os níveis de risco.

A oitava etapa consiste em relacionar o conjunto de controles que podem reduzir ou eliminar os riscos identificados. O objetivo da recomendação de controles é reduzir o nível de risco de um sistema de TI a um patamar aceitável.

A última etapa corresponde à produção de documentação com os resultados do processo de análise de risco. Também são documentados os resultados parciais de todas as etapas anteriores.

O segundo processo da metodologia de gestão de riscos é a atenuação dos riscos, que envolve a priorização, avaliação e implementação dos controles para redução dos níveis de risco, recomendados no processo de avaliação de riscos. Como a eliminação dos riscos é geralmente impraticável ou próxima do impossível, cabe aos gestores da empresa usar uma abordagem de custo mínimo e implementar os controles mais apropriados para reduzir os riscos a um nível aceitável, com um impacto adverso mínimo na organização.

A metodologia SP800-30 vem sendo adotada com freqüência na segurança de projetos de TI. Porém, seu uso em projetos científicos é raro.

2.3.3 AS-NZ4360

As fases de identificação, análise e avaliação dos riscos da especificação AS/NZ4360 [AS/NZS, 2004a] possuem similaridade com o processo de avaliação de riscos (ou determinação dos riscos) da especificação SP800-30. A etapa de tratamento também é similar à etapa de atenuação dos riscos. A especificação AS-NZ4360 e seu guia [AS/NZS, 2004b] são bem mais amplos que as demais normas de gestão de risco e não estão restritas à segurança de sistemas ou mesmo à tecnologia da informação. Por isso, sua aplicação pode ser estendida a diversas áreas, como, por exemplo, meio ambiente e saúde.

Esta especificação define o processo de gestão de riscos através de 7 elementos (ou fases) principais, conforme ilustrado na Figura 2.3: comunicar e consultar; estabelecer o contexto; identificar os riscos; analisar os riscos; avaliar os riscos; tratar os riscos; e monitorar e rever.

A gestão de riscos pode ter diversas partes interessadas. Estas partes devem ser identificadas e seus papéis e responsabilidades delimitados na fase de **comunicação e consulta**. É importante desenvolver um plano de comunicação que permita a cada uma destas partes conhecer o andamento do processo e fornecer subsídios para seu desenvolvimento.

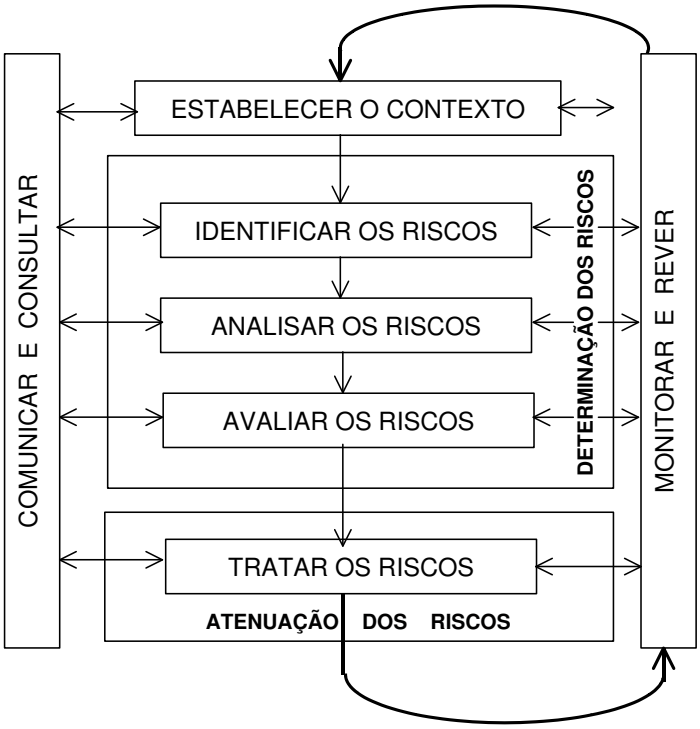

Figura 2.3: Visão Geral do Processo de Gestão de Riscos

A etapa de **estabelecimento do contexto** define os parâmetros básicos, através dos quais serão identificados os riscos que precisam ser geridos e qual será o escopo do restante do processo de gestão de riscos. Também são definidos os critérios os quais serão utilizados na identificação e análise dos riscos. O primeiro passo para se obter o contexto está na descrição dos objetivos do projeto e dos ambientes nos quais eles estão contextualizados. Outro importante aspecto é a definição dos critérios que serão usados na determinação dos riscos do projeto. Estes critérios envolvem a determinação das conseqüências de segurança e os métodos usados para a análise e avaliação dos riscos.

A **identificação dos riscos** é uma das etapas mais críticas, pois os riscos não identificados não serão analisados nem tratados. A identificação deve conter os riscos que estão e os que não estão sob controle do projeto de pesquisa. Na prática, a identificação de riscos é bem mais complexa, pois as informações, quase sempre, são baseadas em experiências e critérios subjetivos dos próprios responsáveis pelo projeto.

Para identificar as vulnerabilidades associadas ao projeto é desejável uma ampla revisão bibliográfica sobre o assunto. Quando há alguma literatura científica sobre as questões levantadas, tal literatura deve ser avaliada e citada. Se a literatura não é suficiente para a identificação dos riscos, pode ser necessária uma consulta à comunidade científica, como a submissão de trabalhos a con-

gressos e periódicos. As questões levantadas na definição do contexto são revistas e colocadas aos participantes interessados.

As revisões obtidas e a apresentação pública do projeto em congressos científicos auxiliaram na identificação e revisão dos riscos. Usando também a literatura relacionada aos objetivos foi possível identificar possíveis ameaças, o que pode ocorrer (conseqüências), como podem acontecer e quais os possíveis tratamentos.

Na etapa da **análise de riscos**, são produzidos dados que irão auxiliar na decisão sobre quais riscos serão tratados e as formas de tratamentos com melhor eficiência de custos. Isso envolve considerações sobre a origem dos riscos, suas conseqüências e as probabilidades de ocorrência das mesmas. As conseqüências e probabilidades são combinadas para produzir o nível de risco.

O objetivo da **avaliação dos riscos** é tomar decisões, baseadas nos resultados da análise de risco. É necessário definir as prioridades e a real necessidade de tratamento dos riscos analisados. A avaliação envolve a comparação dos níveis de risco encontrados, com os critérios estabelecidos quando o contexto foi considerado.

O **tratamento dos riscos** envolve a identificação de opções de tratamento, avaliação destas opções e a preparação para a implementação dos tratamentos selecionados. Esta etapa é equivalente à etapa de atenuação de riscos da especificação SP800-30.

O **monitoramento e a revisão** são partes essenciais da gestão de riscos. Os riscos devem ser monitorados a fim de verificar a eficácia das estratégias de implementação e mecanismos de gerenciamento utilizados no tratamento dos riscos. Portanto, o processo de monitoramento deve ser contínuo e dinâmico. Além do monitoramento contínuo, mudanças organizacionais ou externas podem alterar o contexto da análise, levando a uma revisão completa da gestão de riscos. Revisões também podem ser iniciadas periodicamente ou realizadas por terceiros.

Cada estágio do processo de gestão de riscos deve ser documentado de forma apropriada. Suposições, métodos, origens de dados, análises, resultados e justificativas das decisões tomadas devem ser registrados. Os relatórios produzidos devem ser o mais sucintos e objetivos quanto possível.

Para acompanhar o desenvolvimento da proposta de Tese, adotamos a norma AS/NZ4360 [AS/NZS, 2004a], combinada a outras metodologias de análise para criar um processo de gestão de riscos aplicável a projetos científicos. Esta metodologia foi escolhida por ser a base de uma metodologia de gestão de riscos em desenvolvimento pela ISO (*International Organization for Standardization*). Ela também é flexível e permite adaptações que a torna aplícável à diversas áreas de conhecimento. Em anexo está documentado o processo de gestão de riscos aplicado à presente Tese.

2.3.4 Common Vulnerability Scoring System (CVSS)

Complementando a metodologia de gestão de riscos, é necessário definir critérios para medir os níveis de risco do projeto. Para o cálculo do impacto de possíveis vulnerabilidades, neste projeto

adotamos a metodologia *Common Vulnerability Scoring System* (CVSS)[3], usada pelo NIST para a classificação de vulnerabilidades no *National Vulnerability Database* (NVD)[4]. A base de dados de vulnerabilidades NVD é integrada *ao Common Vulnerabilities and Exposures* (CVE)[5] [Mell e Grance, 2002].

A metodologia do CVSS utiliza uma série de parâmetros e calcula uma pontuação (*score*) que irá definir o grau de risco de uma determinada vulnerabilidade. São utilizados critérios qualitativos para a caracterização das vulnerabilidades. Tais critérios são agrupados em três áreas: um grupo base, um grupo temporal e um grupo ambiental. O grupo base contém todas as características que são intrínsecas e fundamentais para determinada vulnerabilidade e que são invariáveis ao longo do tempo ou em ambientes diferentes. O grupo temporal contém as características que podem mudar ao longo do tempo. No grupo ambiental estão as características que são atreladas a implementações e ao ambiente. As características qualitativas recebem valores e são processadas quantitativamente para obter uma pontuação final ajustada, que irá representar as ameaças que uma vulnerabilidade apresenta em determinado instante de tempo para um ambiente específico. A pontuação representa um valor entre 0 (sem riscos) e 10 (maior risco). O NIST realiza uma classificação de riscos[6] das vulnerabilidades baseado nesta pontuação: Baixo (valor entre 0 e 3,9), Médio (valor entre 4 e 6,9) e Alto (valor entre 7 e 10).

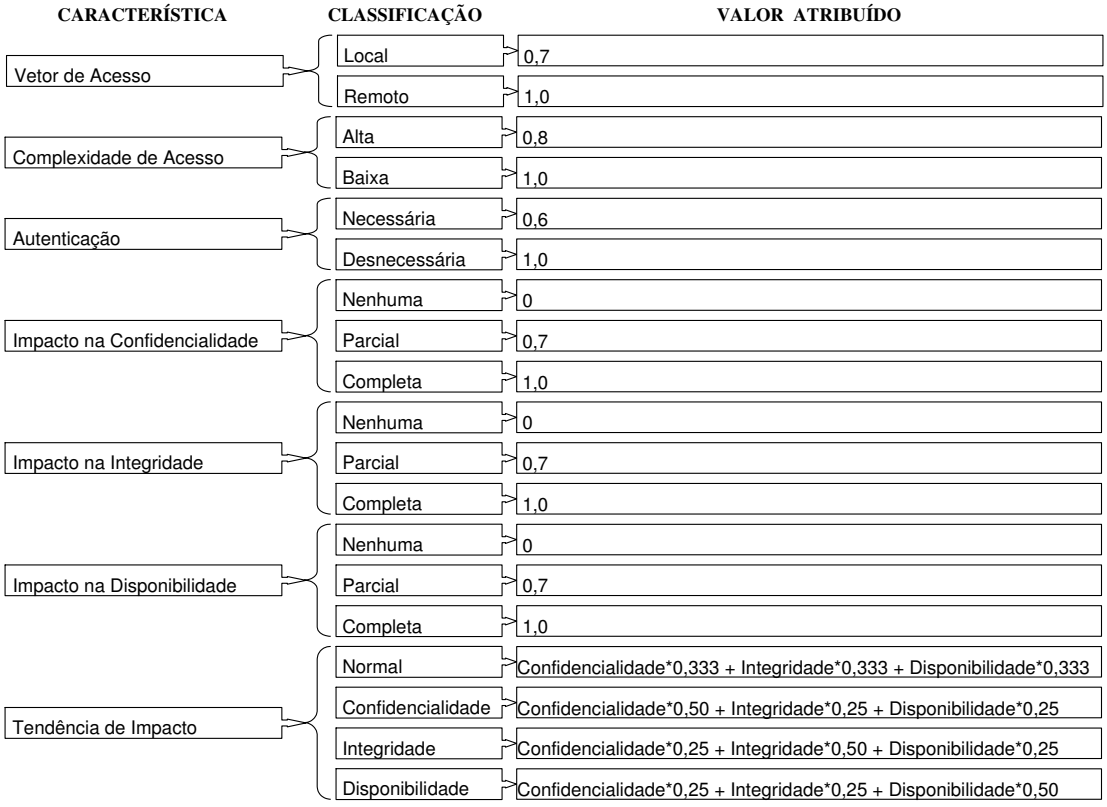

Figura 2.4: Métricas para análise de riscos de segurança, conforme o CVSS

[3]http://www.first.org/cvss/
[4]http://nvd.nist.gov/
[5]http://cve.mitre.org/
[6]http://nvd.nist.gov/cvss.cfm

O grupo base de características é composto de sete critérios, conforme ilustrado na Figura 2.4: vetor de acesso, complexidade de acesso, autenticação, impacto na confidencialidade, impacto na integridade, impacto na disponibilidade e a tendência de impacto. O **vetor de acesso (AC)** identifica se a vulnerabilidade pode ser explorada remotamente (classificação Remoto) ou requer acesso físico ou *login* autenticado no sistema alvo (classificação Local). A **complexidade de acesso (AC)** mede o esforço necessário para explorar a vulnerabilidade. Se forem necessárias circunstâncias muito específicas ou condições especiais de acesso, a complexidade é considerada alta. Se não, a complexidade é baixa[7].

A métrica de **autenticação (AU)** mede se é necessária ou desnecessária a autenticação do atacante no sistema alvo para que a vulnerabilidade seja explorada. As métricas de **confidencialidade (CI)**, **integridade (II)** e **disponibilidade (AI)** medem o grau de impacto em cada um destes requisitos de segurança, caso a vulnerabilidade seja explorada. O impacto pode ser completo se houver comprometimento total do requisito. Pode ser parcial se houver danos consideráveis sem que haja controle total do que possa ser obtido, modificado ou totalmente interrompido. O impacto também pode ser nenhum. A métrica de **tendência de impacto** é utilizada para atribuir um peso maior a uma das métricas de confidencialidade **(CIT)**, integridade **(IIT)** e disponibilidade **(AIT)**. Ou, então, mantê-las com pesos eqüitativos.

Para a análise de riscos de segurança do projeto, foi adotado apenas o grupo base de critérios de caracterização de vulnerabilidades, já que não estão sendo tratadas vulnerabilidades em softwares específicos. Para calcular a pontuação é utilizada a seguinte fórmula sobre os valores atribuídos a cada vulnerabilidade, de acordo com suas características:

Pontuação = 10 * AI * AC * AU * ((CI * CIT) + (II * IIT) + (AI * AIT)))

Os resultados e exemplos da aplicação desta metodologia são apresentados em anexo.

2.4 *Web Services*

Segundo a especificação do W3C [W3C, 2004], o objetivo da **Arquitetura *Web Services*** (WSA - *Web Services Architecture*) é estabelecer uma forma padrão de interoperabilidade entre aplicativos de software diferentes, funcionando em uma variedade de plataformas e/ou *frameworks*.

Um ***Web Service*** é descrito como um sistema de software projetado para suportar a interoperabilidade da interação máquina-a-máquina através de uma rede. Ele possui uma interface descrita em um formato processável por computador, o WSDL (*Web Services Description Language*) [W3C, 2005]. Outros sistemas interagem com o *Web Services* de uma forma determinada, através de sua descrição usando mensagens SOAP [W3C, 2003], geralmente transportadas usando o protocolo HTTP, com um documento XML [Bray et al., 2004] em conjunto com outros padrões relacionados.

[7] Em uma versão draf mais recente do CVSS é considerada também uma complexidade intermediária, redimensionando os pesos. Porém, isto ainda não foi adotado pelo NIST e não foi usada neste projeto.

Um *Web Service* é uma noção abstrata que precisa ser implementada por um agente concreto. O **agente** é a parte concreta de um software ou hardware que envia e recebe mensagens, enquanto o **serviço** é o recurso caracterizado pelo conjunto abstrato de funcionalidades que são fornecidas. Portanto, um mesmo *Web Service* pode ser implementado de formas diversas, sem que as funcionalidades oferecidas sejam alteradas.

A forma de prover serviços em *Web Services* é baseada no **modelo cliente-servidor**, no qual um **agente fornecedor** (*provider agent*) oferece um serviço que provê alguma funcionalidade em nome de seu proprietário (pessoa ou organização), denominado "**entidade fornecedora**" (*provider entity*). O cliente deste serviço é a "**entidade requisitante**" (*requester entity*), que utiliza um **agente requisitante** (*requester agent*) para interagir com o agente fornecedor.

O mecanismo de troca de mensagens é documentado em uma **descrição *Web Service*** (WSD – *Web service Description*). O WSD é uma especificação processada por computador da interface do *Web Service*, escrita na linguagem WSDL. Esta linguagem define os formatos das mensagens, os tipos de dados, os protocolos de transporte e os formatos de serialização de transporte que devem ser usados entre os agentes requisitante e fornecedor. A linguagem WSDL também especifica uma ou mais localizações na rede, através das quais um agente fornecedor pode ser invocado. Em essência, a descrição *Web Service* representa um acordo (ou contrato) que governa o mecanismo de interação com o serviço.

A **semântica** do *Web Service* é a expectativa sobre o comportamento de um serviço, em particular, em resposta às mensagens que são enviadas a ele. Enquanto a descrição representa o contrato que governa os mecanismos de interação com um serviço particular, as semânticas representam o contrato que governa o propósito e as conseqüências dessa interação.

Existem várias formas pelas quais uma entidade requisitante pode contratar e usar um *Web Service*. A forma mais utilizada é ilustrada na Figura 2.5 [W3C, 2004]. (1) as entidades requisitantes e o fornecedor começam conhecendo-se; (2) as entidades requisitantes e o fornecedor de alguma forma concordam com a descrição do serviço e a semântica que irão governar a interação entre os agentes; (3) a descrição do serviço e a semântica são concretizados pelos agentes; (4) os agentes interagem através da troca de mensagens, assim realizando alguma tarefa de interesse das entidades.

2.4.1 Arquitetura Orientada a Serviços

Uma **Arquitetura Orientada a Serviço** (SOA – *Services Oriented Architecture*) é uma forma de arquitetura de sistemas distribuídos tipicamente caracterizada pelas seguintes propriedades:

- Visão Lógica: o serviço é uma visão abstrata lógica dos programas reais, bases de dados, processos de negócios etc, definida em termos do que eles fazem.
- Orientação à mensagem: o serviço é formalmente definido em termos das mensagens trocadas entre agentes fornecedores e agentes requisitantes. A estrutura interna de um agente, incluindo

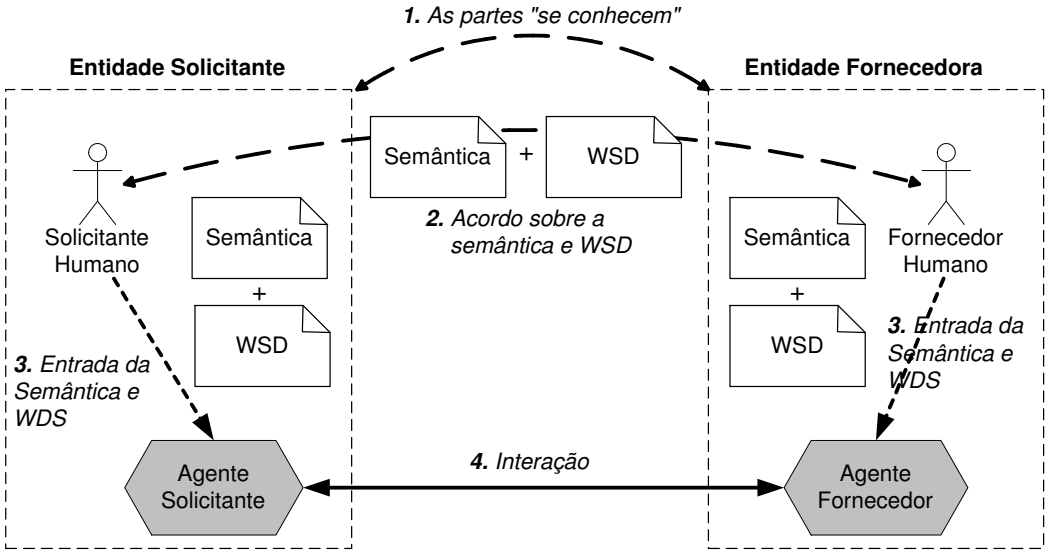

Figura 2.5: Processo de contratação de um *Web Service*

as características como a linguagem de implementação, a estrutura de processos e até as estruturas de bases de dados, é deliberadamente abstraída fora do SOA.

- Orientação à descrição: o serviço é descrito por meta-dados processáveis por máquina. A descrição suporta a natureza pública do SOA: somente aqueles detalhes que estão expostos ao público e que são importantes para o uso do serviço devem ser incluídos na descrição. A semântica de um serviço deve estar documentada diretamente ou indiretamente em sua descrição.
- Granularidade: os serviços tendem a utilizar um pequeno número de operações com mensagens relativamente grandes e complexas.
- Orientação à rede: os serviços tendem a ser orientados para o uso em rede, embora isso não seja um requisito absoluto.
- Neutralidade a plataformas: as mensagens são enviadas através de interfaces em um formato padrão independente de plataforma. O XML é o formato mais óbvio que atende a essa restrição.

2.4.2 Tecnologias *Web Services*

A arquitetura de *Web Services* envolve diversas tecnologias organizadas em camadas e interrelacionadas. Existem diversas formas de visualizar essas tecnologias, da mesma forma que existem vários meios de construir e usar *Web Services*. A Figura 2.6 ilustra algumas destas famílias de tecnologias. Entre estas tecnologias, destacamos o uso da linguagem XML, do protocolo SOAP e da linguagem WSDL, que serão resumidas a seguir.

XML

A linguagem XML (*Extensible Markup Language*) tem como principal objetivo a descrição de informações de forma flexível e simplificada. Derivada do SGML (*Standard Generalized Markup*

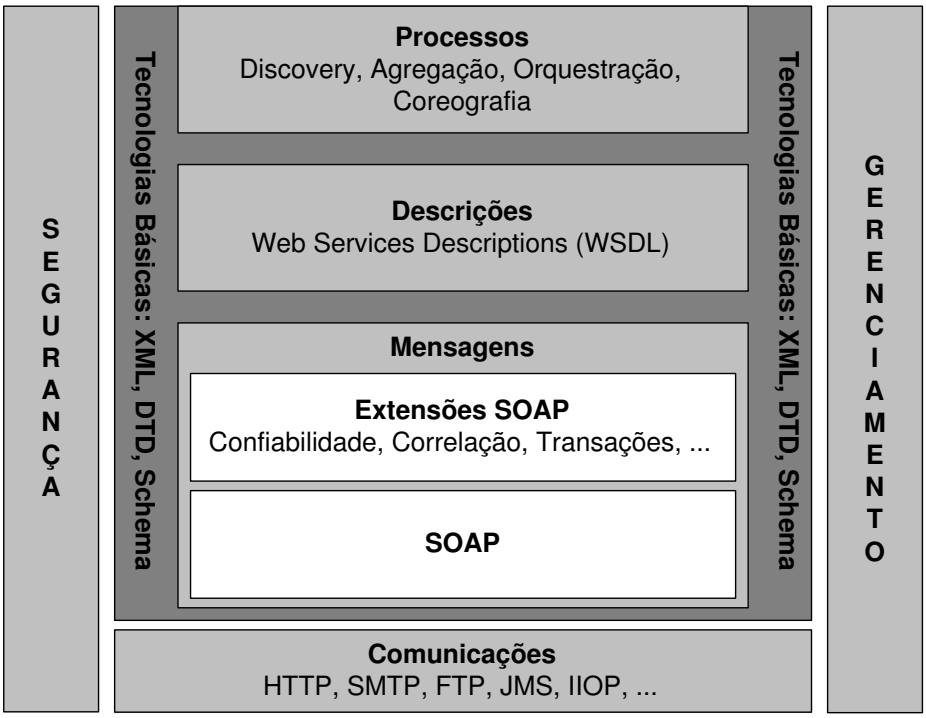

Figura 2.6: Pilha da Arquitetura Web Service

Language) [Smith e Stutely, 1988], foi originalmente projetada para enfrentar os desafios das publicações eletrônicas. A linguagem XML é atualmente uma peça importante na troca de uma grande variedade de dados, principalmente nas tecnologias que envolvem a Internet. Nos documentos XML, os dados e metadados são descritos de forma unificada, ou seja, um mesmo documento armazena os dados e sua descrição.

A linguagem XML oferece um formato de dados padrão, flexível e inerentemente extensível, reduzindo a dificuldade de desenvolvimento de muitas tecnologias necessárias para garantir o sucesso dos *Web Services*.

SOAP

A especificação SOAP 1.2 [W3C, 2003] provê um *framework* padrão, extensível e composto para o empacotamento e troca de mensagens XML. Um *framework* é definido com base em mensagens XML: um modelo de processamento e um modelo de extensão. No contexto da arquitetura WS, ele também provê um mecanismo conveniente para referenciar habilidades (*capabilities*). As mensagens SOAP podem ser transportadas por uma variedade de protocolos de rede, como: o HTTP, SMTP, FTP, RMI/IIOP ou um protocolo proprietário. Em uma interpretação mais geral, uma mensagem SOAP (*Service Oriented Architecture*) contém a informação necessária para invocar um serviço ou transporta os resultados de uma invocação de serviço. Em outra interpretação, o SOAP RPC (*Simple Object Access Protocol*), uma mensagem carrega uma invocação de método em um objeto remoto, com a lista de argumentos que precisam ser transferidos do ambiente local para o ambiente remoto.

WSDL

A especificação WSDL (*Web Services Description Language*) [W3C, 2005] define uma linguagem para descrever *Web Services*. O WSDL começa descrevendo o *Web Services* com as mensagens que são trocadas entre os serviços requisitantes e o fornecedor. As mensagens são descritas abstratamente e então limitadas a um protocolo de rede concreto e um formato de mensagem. Assim, a definição de um *Web Service* pode ser mapeada para qualquer linguagem de implementação, plataforma, modelo de objetos ou sistema de mensagem.

2.4.3 Descoberta de *Web Services*

Se a entidade requisitante deseja iniciar uma interação com uma entidade fornecedora e não sabe com qual serviço fornecedor deseja se engajar, então a entidade requisitante precisa descobrir um candidato adequado. A descoberta é o ato de localizar uma descrição em linguagem processável por máquina de um *Web Service* que pode ter sido anteriormente desconhecido e que se enquadra em certos critérios funcionais. Nesta Tese, a descoberta de serviços é implementada pelo Serviço de Registro e Pesquisa, detalhado nos próximos capítulos.

2.4.4 Segurança em *Web Services*

As ameaças a *Web Services* envolvem ameaças a *hosts*, a aplicação e a toda infra-estrutura de rede. Para prover segurança aos *Web Services*, uma gama de mecanismos de segurança baseados em XML são necessários. Tais mecanismos devem resolver problemas relacionados com a autenticação, o controle de acesso baseado em papéis, a aplicação de políticas de segurança distribuídas e a segurança na camada de mensagens, que acomode a presença de intermediários. Tudo isso sem perder de vista a interoperabilidade.

Os mecanismos de segurança tradicionais, como: TLS/SSL (*Transport Layer Security*) [Dierks e Rescorla, 2006], Redes Privadas Virtuais (*Virtual Private Networks - VPNs*) [Bishop, 2003], IPSec (*Internet Protocol Security*) [Kent e Seo, 2005] e S/MIME (*Secure Multipurpose Internet Mail Exchange*) [Ramsdell, 2004] são tecnologias ponto-a-ponto. Apesar dessas tecnologias serem usadas na segurança de *Web Services*, elas não são suficientes para prover um contexto de segurança fim-a-fim, como requerido pela granularidade dos *Web Services*. Em geral eles utilizam uma abordagem baseada em mensagens que possibilita interações complexas que podem incluir o roteamento de mensagens entre (e através de) vários domínios confiáveis. Por isso, a segurança na camada de mensagem é mais importante.

Os requisitos para prover segurança fim-a-fim em *Web Services* são resumidas no documento *Web Services Architecture* da W3C [W3C, 2004]. Outros mecanismos são apresentados no anexo A desta Tese.

Os *Web Services* podem fluir através de *firewalls* e podem ser transportados através de túneis de comunicação usando portas e protocolos existentes. O uso da transferência segura de mensagens garante a privacidade, a confiabilidade e a integridade das interações. As assinaturas digitais podem ser

usadas para auxiliar no não repúdio. Os canais seguros podem ser usados para proteger as mensagens. Infelizmente, quando há a necessidade de retransmissão das mensagens, situação típica em *Web Services*, tais mecanismos nem sempre são adequados. Nos casos mais gerais, a assinatura e a criptografia do *payload* das mensagens podem ser usadas no roteamento e confiabilidade das mensagens.

A segurança em *Web Services* também requer o uso de políticas corporativas que precisariam ser integradas a políticas externas que extrapolam as empresas e envolvem resoluções de confiança. Diante deste quadro, as organizações necessitariam implementar habilidades tais como: identificações que cruzam domínios, políticas distribuídas, políticas de confiança, mecanismos de descoberta seguros e confiáveis, além da segurança nas mensagens.

A Figura 2.7 [W3C, 2004] exemplifica o modelo de segurança desejado para *Web Services*, no qual políticas são acompanhadas pela segurança das mensagens e por serviços de descoberta confiáveis.

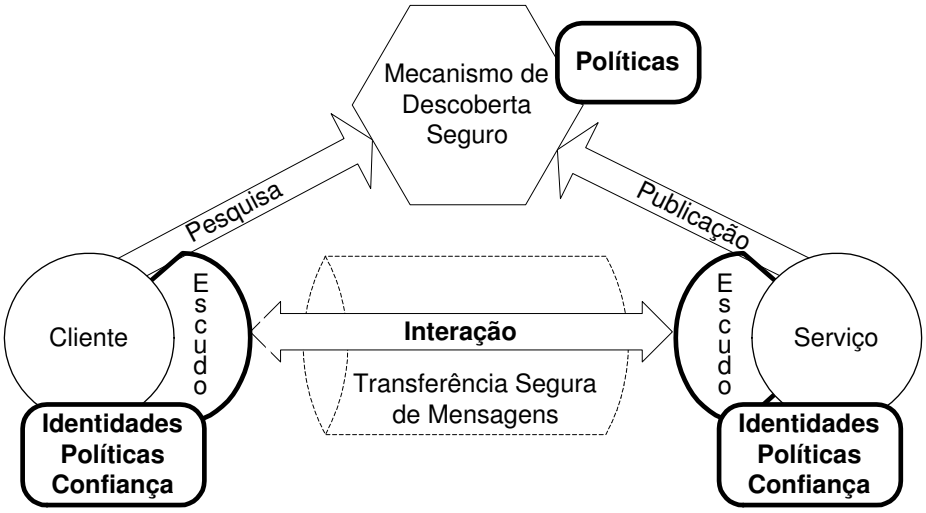

Figura 2.7: Descoberta e interação segura

2.4.5 Tecnologias de Segurança Associadas a *Web Services*

As assinaturas XML (***XML-Signature***) [Eastlake et al., 2002][Reagle, 2000] são definidas para uso em transações XML. O padrão *XML-Signature* define um esquema para assinar digitalmente dados arbitrários contidos em documentos XML. Assinaturas XML dão suporte à autenticação, à integridade de dados e ao não repúdio aos dados assinados.

O ***XML-Encryption*** [Reagle, 2002][Imamura et al., 2002] especifica um processo de criptografia de dados representados em XML. O elemento cifrado pode ser um dado arbitrário (inclusive um documento XML), um elemento XML ou um índice do elemento. O resultado do dado cifrado é um elemento XML *Encryption* que contém ou referencia um dado cifrado.

Desenvolvido pelo OASIS, o ***Web Services Security*** (*WSS* ou *WS-Security*) [OASIS, 2004c] define a extensão SOAP que provê qualidade de proteção pela integridade, pela confidencialidade e pela

autenticação das mensagens. Os mecanismos WSS podem ser usados para acomodar uma ampla variedade de modelos de segurança e tecnologias de criptografia. A integridade das mensagens é fornecida pelo uso de *XML-Signature* em conjunto com *tokens* de segurança para garantir que as mensagens sejam transmitidas sem alteração. A confidencialidade da mensagem é garantida com o *XML-Encryption* em conjunto com *tokens* de segurança para manter partes das mensagens SOAP confidenciais.

2.4.6 Gerenciamento de *Web Services*

Para o gerenciamento de *Web Services* está em desenvolvimento o conjunto de padrões WSDM (Web Services Distributed Management) [OASIS, 2004b]. A especificação WSDM provê interfaces comuns de gerenciamento que são agrupadas segundo habilidades específicas (*capabilities*). O WSDM fornece habilidades relacionadas à identificação, estado de funcionamento, disponibilidade do componente, configuração, métricas para monitoramento e notificação de eventos. Cada *Web Service* define em seu documento WSDL quais habilidades estão disponíveis.

Associada ao WSDM está o conjunto de especificações WSN (*Web Services Notification*) [Graham et al., 2006] da OASIS, que fornece o suporte à notificação de eventos entre *Web Services*. Relacionada ao WSN, a especificação WS-Topics [Vambenepe et al., 2006] define mecanismos para organizar e categorizar itens (tópicos) de interesse para seleção de eventos.

2.4.7 Considerações Sobre o Uso de *Web Services*

Conforme explicado no documento Web Services Architecture [W3C, 2004], nem sempre o uso de *Web Services* traz benefícios que justifiquem seus custos em desempenho. Em geral, o SOA e a tecnologia de *Web Services* são mais apropriados para aplicações:

- que precisam operar sobre a Internet, na qual a confiabilidade e velocidade não podem ser garantidas;
- nas quais existe a habilidade de gerenciar o desenvolvimento, de forma que todos os requisitantes e fornecedores são atualizados de uma vez;
- nas quais os componentes do sistema distribuído rodam em diferentes plataformas e produtos comerciais;
- nas quais uma aplicação existente precisa ser exposta para uso sobre a rede e pode ser "empacotada" como um web service.

2.5 *Conclusões do Capítulo*

Neste capítulo foram apresentados os principais conceitos de segurança, que estão sempre relacionados à manutenção das três propriedades básicas: integridade, confidencialidade e disponibilidade. Por-

tanto, quando uma ou mais dessas propriedades são quebradas, ocorre uma violação de segurança. As violações de segurança, em geral, são possibilitadas pela existência de defeitos ou fraquezas em um sistema, conhecidas como vulnerabilidades, podendo ser exploradas intencionalmente ou acidentalmente, o que gera ameaças aos sistemas. Quando acontece uma tentativa de exploração intencional de uma vulnerabilidade, há então um ataque. Se o ataque for bem sucedido, ocorre uma intrusão no sistema.

A fim de se estabelecer regras do que é ou não permitido em um sistema informático, é necessária uma política de segurança. Os mecanismos de segurança concretizam as políticas. O objetivo desses mecanismos é evitar, detectar e recuperar de ataques os sistemas.

Nenhum sistema computacional é invulnerável. Por isso, avaliar os riscos associados a um sistema ou projeto é essencial. Um método para a avaliação dos riscos de um projeto ou produto é a gestão de riscos.

Neste capítulo também foram apresentados os principais conceitos relacionados à tecnologia de *Web Services*. Uma série de especificações e tecnologias que são agrupadas para permitir a interoperabilidade e a segurança necessária para a comunicação entre os mais diversos aplicativos.

Capítulo 3

Detecção de Intrusão Distribuída

Neste capítulo são apresentadas as principais experiências em detecção de intrusão distribuída, cujos resultados positivos ou negativos auxiliaram no desenvolvimento desta Tese. A evolução dos sistemas de detecção de intrusão é marcada pelo desenvolvimento de diversos sistemas comerciais e inúmeras propostas científicas, sobre as quais foi sendo fundamentado o alicerce dos IDSs atuais. Portanto, para propor uma infra-estrutura para a construção de sistemas de detecção de intrusão de larga escala é necessário entender a evolução dos sistemas de detecção de intrusão.

Na primeira seção deste capítulo são apresentados um breve histórico da detecção de intrusão e conceitos básicos de construção de IDSs distribuídos, que são adotados nesta Tese. Na seção 3.2 será abordada uma taxonomia para classificação dos elementos de detecção de intrusão. Na seção 3.3 são resumidos os novos padrões para detecção de intrusão. Na seção 3.4 os principais trabalhos relacionados à detecção de intrusão distribuída são discutidos. Na seção 3.5 são traçadas algumas considerações sobre a construção de IDSs distribuídos e a relação destes sistemas com esta Tese. Finalmente, na seção 3.6 são apresentadas as conclusões deste capítulo.

3.1 Histórico e Modelo Geral de Detecção de Intrusão

O estudo de planejamento de tecnologia de segurança computacional feito por Anderson [1972] realizava uma tentativa de definição do que seria necessário para se estabelecer uma vigilância dos sistemas computacionais. As questões levantadas naquele trabalho, sobre o que analisar, como analisar e como proteger o sistema de vigilância e suas informações, permanecem como objeto de estudo até hoje [McHugh, 2001].

Diversos estudos relatam a evolução histórica da segurança computacional e das técnicas e mecanismos de detecção de intrusão [Bace, 2000][McHugh, 2001][Axelsson, 2000][Sherif e Dearmond, 2002]. Os primeiros estudos formalmente documentados sobre possíveis intrusões em sistema computacionais datam do início da década de 70, quando Karger e Schell [1974, 2002] executaram uma avaliação do sistema operacional Multics e Stryker [1974] analisou a segurança do sistema Univac.

Em ambos os casos foram descobertas vulnerabilidades que poderiam permitir a um usuário não confiável acessar ou modificar informações protegidas.

Apesar de existirem discussões anteriores sobre questões de segurança, foi somente com Anderson [1980] que se teve o primeiro *framework* coerente sobre a detecção de intrusões. Preocupado com o nível das informações de segurança registradas nas trilhas de auditoria, procurou identificar as ameaças existentes até então para criar um mecanismo de monitoramento que complementasse as informações disponíveis em um sistema, com o objetivo de detectar possíveis ataques e invasões ao mesmo. Contudo, somente em meados da década de 80, Denning definiu o que seria o primeiro modelo de detecção de intrusão [Denning, 1986][Denning, 1987]. Os métodos de análise usados atualmente em detecção de intrusão fogem pouco daquele modelo, que se baseia em: **sujeitos** que iniciam uma atividade em um sistema alvo; **objetos** que são entidades gerenciadas pelo sistema; **registros de auditoria** que são gerados pelo sistema em resposta a ações dos sujeitos sobre os objetos; **perfis** que caracterizam o comportamento de sujeitos em relação a objetos; **registros de anomalias** que são gerados quando um comportamento anormal é observado e; por fim, **regras de atividade** que definem as **ações** a serem realizadas quando uma determinada condição é satisfeita.

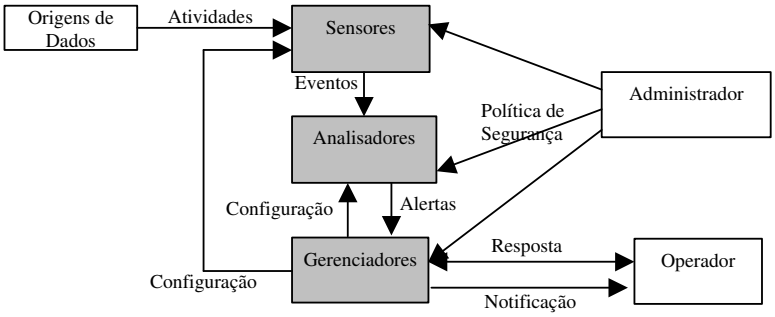

Figura 3.1: Modelo básico de um IDS

Ainda não existe uma padronização para construção de sistemas de detecção de intrusão. Contudo, o modelo de relacionamento definido pelo grupo *Intrusion Detection Working Group* (IDWG) do IETF (*Internet Engineering Task Force*)[1] em Wood e Erlinger [2002], representado na Figura 3.1, serve como parâmetro para entender o relacionamento entre os elementos de um sistema de detecção de intrusão, que é formado basicamente por: **origem de dados** onde ocorrem as **atividades** de sujeitos sobre objetos; **sensor**(es) que gera(m) **eventos** (ou registros de auditoria); **analisadores** que verificam nos eventos indícios de anomalias, gerando **alertas**. Os **gerenciadores** são responsáveis pela **configuração** do sistema e pelo envio de **notificações** aos **operadores** a partir de alertas. Além dos operadores, há também a interferência humana no papel de **administradores** do sistema, que definem as políticas de segurança. É sempre desejável a existência de mecanismos de **resposta** que façam o rastreio dos eventos suspeitos para a correta identificação e **responsabilização** do sujeito atacante. A resposta também envolve a notificação de eventos e pode ser executada automaticamente pelo IDS ou manualmente pelo operador. Os elementos deste modelo podem estar contidos em um único sistema monolítico ou podem estar distribuídos em vários sistemas de uma rede de computadores.

[1] http://www.ietf.org

Tabela 3.1: Classificação dos sensores para detecção de intrusão

Origem da Informação	Dados Brutos	Externa	Escuta de pacotes na rede
		Interna	Sensor de rede integrado
			Sensor de aplicação integrado
	Dados de Registros	Nível de sistema	Registros de auditoria do sistema operacional
			Registros de utilização (*accounting log*)
		Nível de aplicação	Registros de Aplicações
		Metadados	Alertas de sistemas de detecção de intrusão

3.2 Taxonomia para Sistemas de Detecção de Intrusão

A tentativa de classificação de técnicas e arquiteturas de sistemas de detecção de intrusão vem sendo feita desde a década de 80 [Neumann e Parker, 1989]. Com o aprimoramento das pesquisas na área, a cada avaliação, novas técnicas são introduzidas e modelos mais abrangentes são sugeridos.

Esse tipo de classificação é necessária para facilitar a seleção de determinado tipo de elemento para as composições de IDSs propostas nesta Tese. Como não existe uma ontologia específica para a detecção de intrusão, os elementos sensores seguem a taxonomia proposta em Alessandri et al. [2001], representada na Tabela 3.1. No caso dos analisadores e IDSs monolíticos, são adotadas em suas classificações as mais recentes taxonomias de IDSs [Axelsson, 2000][Debar et al., 1999][Debar et al., 2000] [McHugh, 2001], combinadas de acordo com a arquitetura e o método de detecção adotado, conforme apresentado na Tabela 3.1. A seguir, serão apresentados maiores detalhes sobre a taxonomia.

3.2.1 Origem da Informação dos Sensores

A classificação da origem da informação é dividida em duas classes principais: **Dados Brutos** (*Raw Data*); e **Dados de Registros** (*log data*). Os dados brutos representam uma visão não transformada dos dados gerados em uma atividade. Em contraste, os dados de registros fornecem uma visão interpretada dos dados, pelo sensor.

Os dados brutos podem ser obtidos de forma **Externa** ou de forma **Interna**. A origem de informações externa provê uma visão dos dados antes deles atingirem o sistema monitorado, enquanto os dados internos são obtidos nos sistemas monitorados. Os dados externos podem ser obtidos por sensores com escuta de pacotes na rede. Já os dados internos podem ser obtidos por sensores de rede, embutidos na pilha de protocolos do sistema monitorado ou então por sensores contidos na própria aplicação.

Os dados de registro podem ser classificados em três categorias: **Nível de sistema**, **Nível de Aplicação** ou **Metadados**. No nível de sistema, os dados podem ser fornecidos pelo sistema operacional (*audit logs*) ou por registros de uso (*accounting log*), como, por exemplo, *logins* de usuários. No nível de aplicação os dados são fornecidos pelos registros dos próprios aplicativos. Já no nível "metadados", os dados são alertas de segurança provenientes de outros sistemas de monitoramento, como por exemplo *firewalls* ou IDSs.

Tabela 3.2: Classificação dos sistemas de detecção de intrusão

<table>
<tr><td rowspan="11">Método de Detecção</td><td rowspan="6">Comportamento / Anomalia</td><td rowspan="3">Auto-aprendizagem</td><td rowspan="2">Não usam séries temporais</td><td>Modelagem por regras</td></tr>
<tr><td>Estatísticas descritivas</td></tr>
<tr><td rowspan="2">Séries temporais</td><td>Redes Neurais</td></tr>
<tr><td>Sistemas Imunológicos</td></tr>
<tr><td rowspan="3">Programada</td><td rowspan="2">Estatísticas descritivas</td><td>Estatísticas simples</td></tr>
<tr><td>Baseado em regras simples</td></tr>
<tr><td>Limite</td></tr>
<tr><td rowspan="2">Negação de padrão</td><td>Modelagem de séries de estados</td></tr>
<tr><td rowspan="5">Conhecimento / Assinatura ou abusos</td><td rowspan="5">Programada</td><td rowspan="2">Modelagem de estado</td><td>Transição de estado</td></tr>
<tr><td>Rede de Petri</td></tr>
<tr><td colspan="2">Sistemas especialistas</td></tr>
<tr><td colspan="2">Combinação de strings</td></tr>
<tr><td colspan="2">Baseado em regras simples</td></tr>
<tr><td rowspan="3">Comportamento Pós-detecção / Resposta</td><td rowspan="2">Ativo</td><td colspan="3">Ação de controle sobre o alvo do ataque</td></tr>
<tr><td colspan="3">Ação de controle sobre a origem do ataque</td></tr>
<tr><td colspan="4">Passivo</td></tr>
<tr><td rowspan="4">Origem dos dados</td><td colspan="4">Host</td></tr>
<tr><td colspan="4">Rede</td></tr>
<tr><td colspan="4">Aplicação</td></tr>
<tr><td colspan="4">Alertas de detecção de intrusão de outros IDSs</td></tr>
<tr><td rowspan="2">Tempo de detecção</td><td colspan="4">On-line (tempo real)</td></tr>
<tr><td colspan="4">Off-line</td></tr>
<tr><td rowspan="2">Freqüência de Uso</td><td colspan="4">Monitoramento contínuo</td></tr>
<tr><td colspan="4">Análise periódica / batch</td></tr>
<tr><td rowspan="3">Arquitetura de Análise</td><td colspan="4">Centralizada</td></tr>
<tr><td rowspan="2">Distribuída</td><td colspan="3">Hierárquica</td></tr>
<tr><td colspan="3">Totalmente distribuída</td></tr>
</table>

3.2.2 Métodos de Detecção de Intrusão

Os métodos de detecção são divididos em duas classes. A primeira classe refere-se aos sistemas de detecção de anomalias, enquanto a segunda classe diz respeito à detecção de abusos (ou detecção por assinaturas). A Figura 3.2, adaptada de Cachin et al. [2000] representa bem a diferenciação entre essas duas classes.

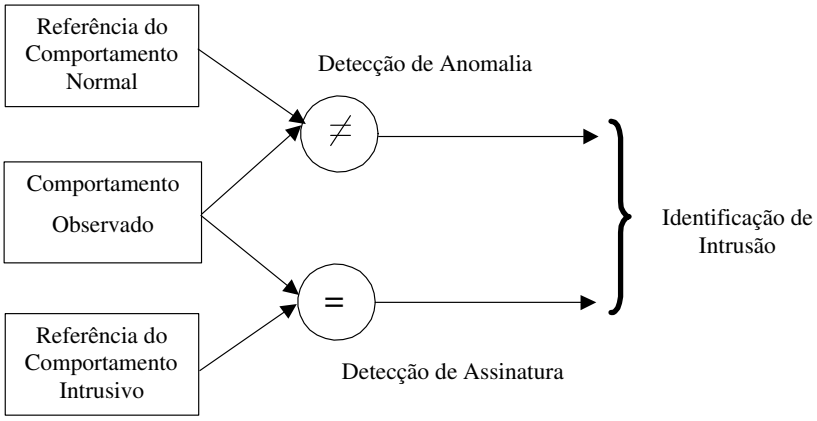

Figura 3.2: Diferenciação entre as classes de detecção de intrusão

A detecção por anomalias procura por situações que destoem do comportamento normal. A construção desses detectores inicia com a formação de uma opinião sobre o que é considerado normal para o comportamento de um sujeito, para então decidir qual a porcentagem de uma atividade é considerada anormal. Com isso é possível identificar novos padrões de comportamento malicioso (ataques). A dificuldade desse tipo de abordagem está na geração de um grande número de alarmes falsos, necessitando ainda de uma grande quantidade de recursos computacionais.

A detecção por anomalias pode ser dividida em duas classes. A primeira é a de sistemas com auto-aprendizado, a qual é capaz de construir sua própria base de conhecimento do que é normal. Esses sistemas de auto-aprendizagem também são subdivididos em sistemas que não utilizam séries temporais e os que as utilizam. A técnica de análise de séries temporais é o estudo estatístico de um conjunto discreto numerável de valores de uma variável de estado de um sistema dinâmico. A técnica parte do pressuposto de que os fatores que influenciaram o comportamento da série no passado continuarão a fazê-lo no futuro. Entre os sistemas que não utilizam séries temporais estão: os que fazem modelagem por regras, no qual o IDS analisa o tráfego e formula um número de regras que descrevem a operação normal do sistema; e os que usam estatísticas descritivas, no qual o sistema coleta estatísticas simples a partir de certos parâmetros para um perfil e constrói um vetor de distância entre o tráfego observado e o perfil criado. Nos sistemas que usam séries temporais, estão incluídas técnicas como as de redes neurais e as técnicas inspiradas nos sistemas imunológicos.

A segunda classe de técnicas de detecção por anomalias é a programada. Nessa classe, o sistema é programado ou ensinado por um indivíduo a fim de formar uma opinião do que é considerado anormal o suficiente para sinalizar uma violação de segurança. Essa subclasse também pode ser dividida entre as técnicas que usam estatísticas descritivas e as que usam negação de padrão. As estatísticas

descritivas são usadas para definir um perfil de comportamento por meio do uso de medidas estatísticas calculadas a partir de certos parâmetros. Essas medidas podem ser estatísticas simples, na qual a própria medida define a ação; podem ser estatísticas baseadas em regras que serão aplicadas sobre os valores; ou baseadas em limites definidos previamente para a ativação de alarmes.

Na negação de padrão, a idéia é estabelecer explicitamente as circunstâncias nas quais o sistema opera de maneira segura, ativando um alarme de intrusão se o comportamento for alterado. A modelagem de séries de estado é a forma de se implementar essa técnica, onde a operação é codificada em forma de um conjunto de estados e as transições entre os estados estão implícitas no modelo. A limitação deste comportamento modelado está na complexidade das semânticas das aplicações envolvidas.

Na detecção por assinaturas, a decisão é formada com base no conhecimento de perfis de processos intrusivos, comparados com o comportamento observado de um processo em execução. Tal abordagem gera poucos alarmes falsos, mas requer constante manutenção para identificar novos comportamentos intrusivos.

A classe de detecção por assinaturas é programada, sendo subdividida em quatro subclasses. A primeira subclasse corresponde à modelagem de estados, na qual a intrusão é codificada como um número de diferentes estados, que devem ser observados para identificar uma intrusão. A segunda subclasse se refere aos sistemas especialistas, que descrevem regras de um comportamento intrusivo. A terceira subclasse é a de combinação de *strings*, na qual se deve pesquisar *substrings* que identifiquem ataques no texto de uma mensagem. A quarta subclasse é baseada em regras simples, bem menos complexas que as dos sistemas especialistas, porém executadas mais rapidamente.

3.2.3 Características dos IDSs

Os sistemas de detecção de intrusão também podem ser divididos de acordo com suas características. Oito características são identificadas por Axelsson [2000]: **tempo de detecção**, **freqüência de uso**, **origem dos dados**, **comportamento pós-detecção**, **arquitetura de análise**, **arquitetura de coleta dos dados**, **segurança e interoperabilidade**.

O **tempo de detecção** identifica se o sistema funciona em tempo real (*on-line*) ou não (*off-line*). A **freqüência de uso** identifica se o sistema irá funcionar continuamente ou será ativado somente em determinadas situações de exceção.

Os IDSs têm como **origem de dados** um dos diferentes níveis do sistema: *host*, rede ou aplicação. Infelizmente, os sistemas de detecção de intrusão normalmente são especializados em apenas um desses níveis. Os IDS baseados em rede monitoram o tráfego que passa pelos sensores na rede, enquanto os sistemas baseados em *host* verificam o comportamento da atividade computacional na máquina hospedeira. Os IDS de aplicações verificam os eventos decorrentes do uso de uma aplicação específica. Os três tipos de IDS possuem vantagens e desvantagens que são tratadas por Bace e Mell [2001]. Um quarto tipo de origem de dados é citado por Debar et al. [1999], como sendo os alertas de detecção de intrusão gerados por outros IDSs.

O **comportamento pós-detecção, ou resposta a ataques**, seria a capacidade de identificar uma atividade maliciosa e então realizar ações para bloqueá-la ou minimizar suas conseqüências. A forma mais comum de resposta a ataques é a passiva, onde há a notificação ou geração de alarmes a um operador. As respostas ativas podem resultar na ativação de novos sensores, mudanças nas políticas e nos mecanismos de controle no ambiente ou, ainda, na concretização de contra-ataques ao intruso.

Com relação à **arquitetura de análise dos dados**, estes podem ser processados de forma centralizada ou distribuída. A análise centralizada é realizada em um único ponto da arquitetura, enquanto na análise distribuída, os dados podem ser processados em diversos locais.

A **coleta dos dados** também pode ser centralizada ou distribuída. Quando a coleta é distribuída, os dados são coletados em diversos locais. Os dados coletados em um único ponto podem ser transmitidos e analisados em vários locais, enquanto que os dados coletados em vários pontos podem ser processados em um analisador central.

A **segurança** do IDS e sua habilidade de se proteger ainda são áreas pouco estudadas. A classificação utilizada [Axelsson, 2000] aplica uma escala subjetiva de alta e baixa segurança. Pode também ser aplicado um padrão de avaliação, como o *Common Criteria* [ISO/IEC, 2005a][ISO/IEC, 2005b][ISO/IEC, 2005c] para definir o grau de segurança do IDS. Porém, dada a possibilidade de ocorrência de novas vulnerabilidades, tal classificação seria muito volátil.

A **interoperabilidade** do IDS reflete o quanto este é capaz de operar com outros sistemas de detecção de intrusão, de aceitar dados de origens diferentes etc. Como a interoperabilidade não é um critério parametrizado, não há uma classificação definida para ela. Portanto, para efeitos de seleção de elementos em uma composição de IDSs, tanto a interoperabilidade, quanto a segurança foram suprimidas da taxonomia.

3.3 Padrões para Detecção de Intrusão

A necessidade de correlação de informações para melhorar a eficiência dos sistemas de detecção de intrusão distribuídos levou organizações conhecidas a trabalharem em modelos que padronizem a comunicação entre detectores de intrusão. Uma das primeiras iniciativas foi o CIDF(*Common Intrusion Detection Framework*) [Kahn et al., 1998][Staniford-Chen et al., 1998][Porras et al., 1999][Tung, 2000], financiada pela DARPA[2] (Agência de Projetos de Pesquisa Avançados de Defesa americana, na sigla em inglês). O CIDF estabelece quatro componentes: geradores de eventos (E-boxes); analisadores de eventos (A-boxes); base de dados de eventos (D-boxes); e unidade de resposta (R-boxes). Estes componentes se comunicam usando mensagens denominadas GIDOS (*Generalizzed Intrusion Detection Objetcs*), que são representadas por um formato padronizado na linguagem CISL (*Common Intrusion Specification Language*) [Feiertag et al., 1999]. Apesar de bastante citado na literatura científica, o CIDF não chegou a se transformar em um padrão nem teve grande repercussão no mercado, mas encorajou a criação do grupo *Intrusion Detection Exchange Format* (IDWG), pelo IETF, cujos resultados serão discutidos a seguir.

[2] http://www.darpa.mil

O IETF também mantém um grupo de estudos para o desenvolvimento dos padrões necessários para o intercâmbio de informações e tratamento de incidentes de segurança, o INCH[3] (*Extended Incident Handling*). Esse grupo dá continuidade ao trabalho iniciado pelo projeto Terena (*Trans-European Research and Education Networking Association*)[4] e que deu origem à RFC 3067 [Arvidsson et al., 2001]. As experiências do grupo INCH também serão apresentadas a seguir. Todas estas especificações são baseadas na linguagem XML [Bray et al., 2004].

3.3.1 Syslog

Um dos modelos públicos mais conhecidos e utilizados para a geração e transmissão de eventos de auditoria é o protocolo *syslog*, desenvolvido na década de 80 (primeira versão está datada em 1983) por Eric Allman, da Universidade de Berkeley e padronizado pelo IETF na RFC 3164 [Lonvick, 2001]. O protocolo *syslog* foi projetado como um protocolo de transporte de notificações de eventos, seja para apresentação na console, para envio a um centralizador de eventos através de uma rede IP ou para armazenamento em arquivos. Sua aceitação se deve principalmente a sua simplicidade, que resulta em fácil implementação e configuração.

Apesar do protocolo *syslog* prover mecanismos para troca de mensagens de eventos, sua utilização para trocar informações entre sistemas de detecção de intrusão distintos é bastante limitada. Como cada IDS pode escrever seus alertas no formato que quiser, mesmo utilizando um formato do *syslog* mais estruturado [Gerhards, 2006] [Keeni, 2006] e seguro [Kelsey et al., 2006] [Miao e Yuzhi, 2006], ainda fica difícil agregar informações mais precisas sobre alertas de segurança que possam ser processados automaticamente, impossibilitando que sistemas de detecção diferentes se comuniquem de forma eficiente. Porém, sua aplicação no envio de dados brutos entre sensores diversos e elementos de análise é bastante viável.

3.3.2 IDMEF

Inicialmente o grupo IDWG definiu o modelo geral de um sistema de detecção de intrusão (Figura 3.3), a terminologia utilizada e os requisitos necessários para a interoperabilidade dos sistemas [Wood e Erlinger, 2002]. Entre os requisitos está o desenvolvimento de um formato e de um protocolo específico para a troca de informações entre sistemas de detecção de intrusão. A fim de separar a semântica do mecanismo de comunicação, o formato das mensagens de detecção de intrusão deve ser independente do protocolo de comunicação utilizado. O formato definido é o *Intrusion Detection Message Exchange Format* (IDMEF) [Debar et al., 2006] e como protocolo de comunicação foi sugerido o *Intrusion Detection Exchange Protocol* (IDXP) [Feinstein et al., 2002]. Ambas especificações são baseadas na linguagem XML [Bray et al., 2004].

[3] http://www.ietf.org/html.charters/inch-charter.html

[4] Entidade de pesquisa e educação em tecnologia, formada em 1994 e mantida pela Comunidade Européia. http://www.terena.nl

Ao contrário do formato IDMEF, o protocolo IDXP não obteve grande aceitação no mercado. Porém, é permitido que o formato IDMEF seja transportado por qualquer protocolo, como o adotado nesta Tese.

A principal aplicação do formato IDMEF está na comunicação de alertas entre os componentes de análise e o componente de gerenciamento do sistema de detecção de intrusão. Porém, outras aplicações também seriam possíveis. A troca de informações para o cruzamento e correlação de dados, além da criação de um banco de dados padronizado, são outras formas de utilização.

O modelo de dados do IDMEF é definido como uma série de classes modulares usadas para segmentar os dados. Esse modelo é definido através de DTD (*Document Type Definition*) XML. Outra proposta de especificação, utilizando XML *Schema* foi proposta por Ohta [2004][5]. A Figura 3.3. representa o modelo UML (*Unified Modeling Language*)[6] e a hierarquia das classes IDMEF [Debar et al., 2006], sem todos os níveis de detalhamento.

Há inicialmente dois tipos de mensagens que agregam uma série de classes: *Alert* e *Heartbeat*. A *Heartbeat* é uma mensagem de controle usada para que um analisador possa informar ao gerenciador que ele continua funcionando. Na mensagem *Heartbeat*, o analisador é identificado na classe *Analyser* e envia dois *timestamps* com a etiqueta de tempo da criação da mensagem (*CreateTime*) e tempo no relógio do analisador (*AnalyzerTime*), junto com o intervalo em que as mensagens são geradas (*HeartbeatInterval*) e outras informações adicionais (*AditionalData*).

A mensagem *Alert* descreve um evento de segurança. Obrigatoriamente deve conter a descrição do analisador (*Analyser*), o momento de criação da mensagem (*CreateTime*) e uma possível identificação do que trata a mensagem (*Classification*). Podem ser incluídas informações adicionais (*AditionalData*), o *timestamp* em que o evento foi detectado (*DetectTime*), o *timestamp* de envio da mensagem (*AnalyserTime*), a identificação da(s) possível(is) origem(ns) dos eventos (*Source*) e a identificação do(s) possível(is) alvo(s) do evento (*Target*), além da informação sobre o impacto do evento, ações realizadas pelo analisador e confiança na avaliação (*Assessment*).

Associada à mensagem *Alert* há ainda outras classes estendidas. Isso permite ao modelo incluir novas especificações sempre que necessário ou associá-lo a outros modelos. A classe *ToolAlert* é usada para identificar a ferramenta que gerou os ataques e provocou inúmeros alertas. A Classe *CorrelationAlert* é usada para agrupar alertas relacionados. A Classe *OverflowAlert* fornece informações sobre um ataque específico de *buffer overflow*.

Não serão discutidos aqui detalhes sobre os objetos utilizados nas classes IDMEF. Maiores informações sobre o modelo podem ser obtidas em Debar et al. [2006].

O formato IDMEF não possui qualquer especificação sobre a segurança das mensagens, deixando essa tarefa para o protocolo de comunicação. Como as mensagens podem carregar informações sensíveis, é obrigatório o uso de protocolos que garantam a autenticação mútua entre os envolvidos na comunicação, além da integridade e confidencialidade das mensagens. Nos próximos capítulos serão discutidos os métodos adotados para proteger as mensagens IDMEF.

[5] Esta especificação não foi atualizada e perdeu sua validade.

[6] http://www.uml.org

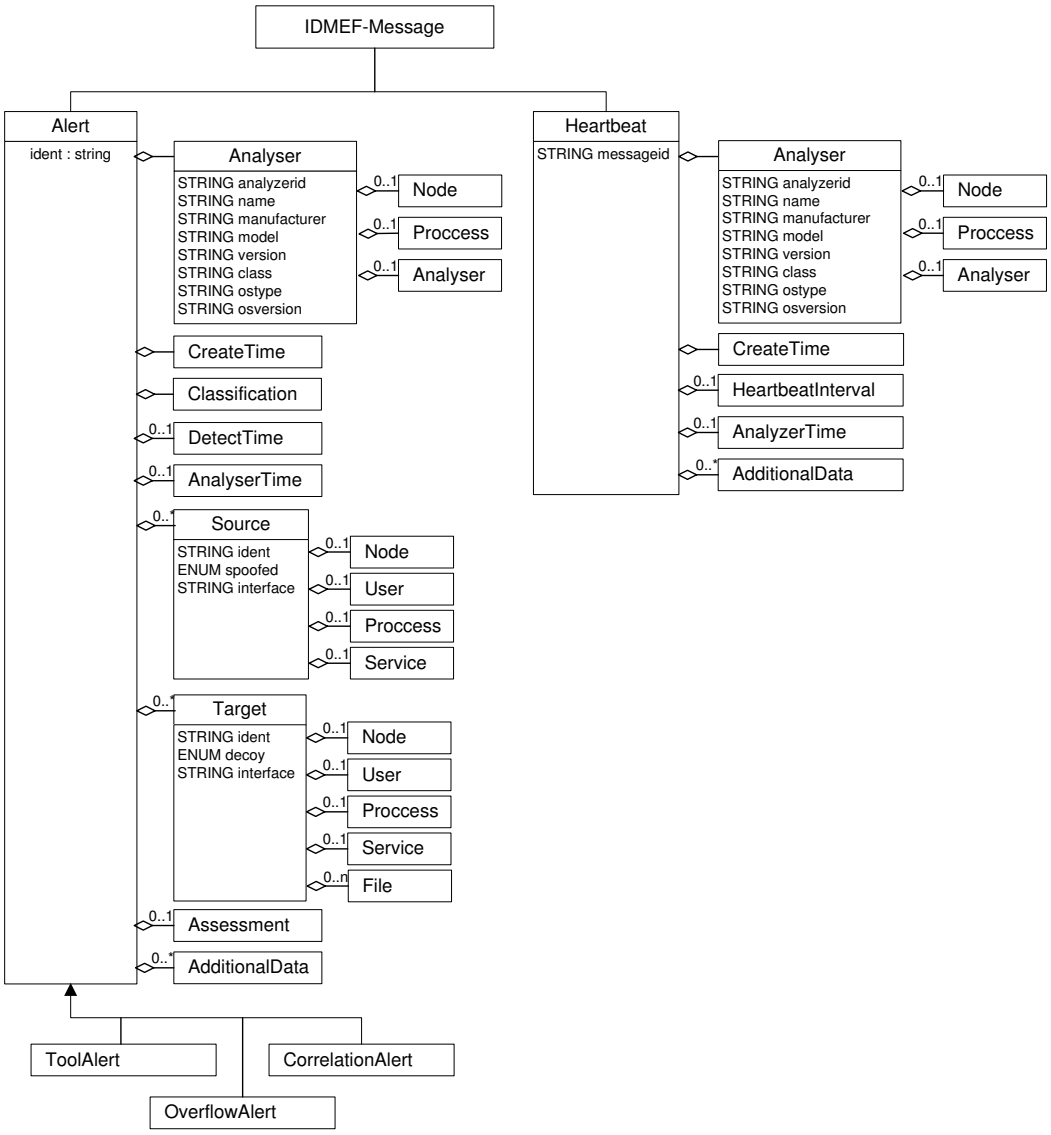

Figura 3.3: Modelo de dados do IDMEF

3.3.3 FINE

A resposta a incidentes de segurança ainda é uma atividade incipiente. Respostas automáticas ainda são pouco confiáveis e podem acarretar negação de serviço no alvo ou no suposto atacante. O entendimento dos ataques e vulnerabilidades é essencial para que essa área de pesquisa possa avançar.

Para a análise de incidentes de segurança e a criação de novas defesas, é necessário que se entenda como ocorrem os ataques, seus alvos e suas origens. Os responsáveis pela coleta e organização das informações são as equipes de resposta a incidentes de segurança (CSIRTs – *Computer Security Incident Response Teams*) [Brownlee e Guttman, 1998]. Tais equipes mantêm bancos de dados com as informações coletadas.

Como eventos de segurança podem envolver domínios administrativos diferentes, há a necessidade de cooperação entre CSIRTs para a complementação de informações e realização de ações

conjuntas. Essa cooperação só será possível se os dados forem tratados de forma padronizada, ou seja, se houver um modelo mínimo para coleta e formatação dos dados.

A cooperação entre CSIRTs é definida pelo modelo FINE (*Format for INcident information Exchange*) [Keeni et al., 2006]. O propósito do modelo FINE é facilitar a troca de estatísticas e informações de incidentes de segurança entre CSIRTs, envolvendo as partes em uma análise reativa das atividades de intrusão em andamento e proativamente identificar suas tendências a fim de facilitar a prevenção de incidentes. O modelo operacional do FINE está representado na Figura 3.4.

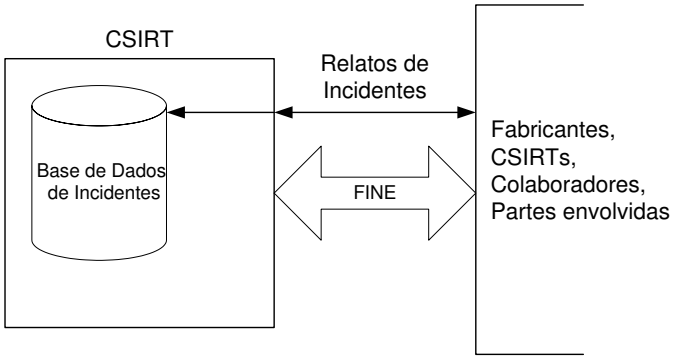

Figura 3.4: Modelo operacional do FINE

Os requisitos do FINE são implementado pelo IODEF (*Incident Object Description and Exchange Format*)[Danyliw et al., 2006]. O IODEF tem seus próprios requisitos definidos na RFC 3067 [Arvidsson et al., 2001]. Esses requisitos englobam a criação de um formato padrão para descrição, armazenamento e troca de informações sobre incidentes entre CSIRTs. Enquanto o IDMEF (seção 3.3.2) é especificado para uso em sistemas automatizados, o foco do IODEF é a comunicação humana. Como no IDWG, as especificações também são baseadas na linguagem XML [Bray et al., 2004].

Na questão de segurança, a especificação IODEF está um passo à frente da especificação IDMEF, pois explicita o uso de mecanismos de segurança na própria mensagem. É definido o uso de XML-Encription e XML-Signature para garantir a integridade, a confidencialidade e a autenticidade das mensagens IODEF. Estes mecanismos também serão usados no modelo de segurança das comunicações de alertas abordado nesta Tese.

3.3.4 Considerações sobre os padrões

Tomando como base o modelo de detecção de intrusão de Wood e Erlinger [2002], a Figura 3.5 dá uma visão geral de como podem ser empregados os formatos padrões aqui analisados. A comunicação entre sensores e analisadores pode ser feita pelo protocolo *syslog* ou usando o formato IDMEF. A comunicação entre analisadores e gerenciadores é feita usando o formato padrão IDMEF. Esse formato também pode ser utilizado como base para construção da caracterização de um incidente de segurança.

A integração entre os formatos IDMEF e IODEF é prevista pelo grupo INCH. Os alertas de intrusão servem como base para que um operador humano modele um incidente de segurança, gerando

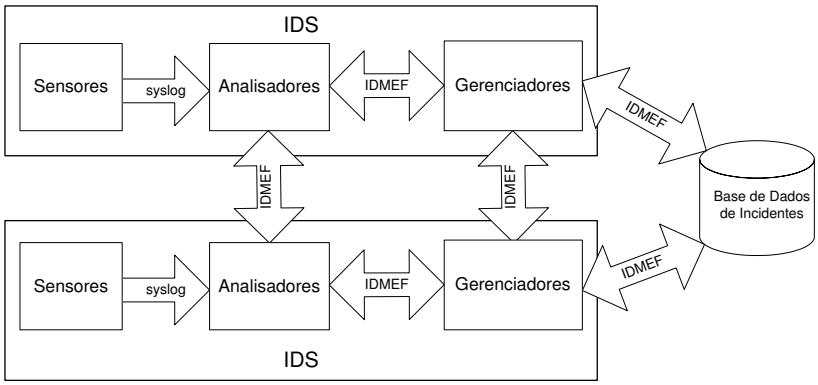

Figura 3.5: Integração dos formatos padrões em um IDS

um documento IODEF, conforme ilustrado na Figura 3.6. No documento IODEF, a classe de alerta de incidentes (*Alert*) do formato IDMEF, que aqui recebe o nome *IncidentAlert*, é agrupado com a classe *Incident* do formato IODEF.

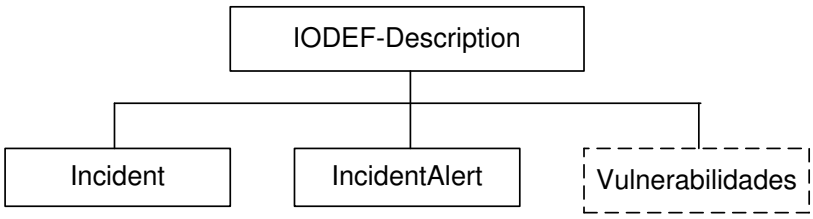

Figura 3.6: IODEF/IDMEF

Nesse modelo integrado, também é prevista a inclusão de um formato padrão para alertas de vulnerabilidades. Assim, os incidentes poderiam ser correlacionados com as vulnerabilidades exploradas pelo atacante, permitindo respostas mais rápidas a ataques. As informações provenientes dos alertas IDMEF podem ser parcialmente usadas em algumas classes do IODEF ou integralmente incluídas como informações adicionais dos documentos.

A integração dos modelos e protocolos de transmissão de eventos de IDSs e alertas de segurança não é uma tarefa trivial. Como se pôde observar, cada modelo possui seu próprio formato, mas atualmente permitem o uso de protocolos de transporte variados. No caso do formato IDMEF, questões importantes relacionadas à segurança são “empurradas” para os protocolos de transmissão que, por sua vez, possuem seus próprios problemas de segurança.

Portanto, apesar dos esforços que vêm sendo desenvolvidos pelos grupos de trabalho do IETF, ainda há muito que avançar para garantir que mensagens contendo eventos de segurança sejam transmitidas de forma segura pela Internet. Em face desta questão, algumas soluções são propostas nos próximos capítulos desta Tese.

3.4 O Estado da Arte em IDSs Distribuídos

Nesta seção são apresentadas as principais experiências e produtos para detecção de intrusão distribuída. De acordo com o enfoque da Tese, é dada maior ênfase às arquiteturas do que às técnicas de detecção. A fim de padronizar a análise destes trabalhos, as figuras e modelos originais foram adaptados para se adequar a uma visão do modelo geral de detecção de intrusão apresentado na seção 3.1 (ver Figura 3.1).

3.4.1 IDSs Distribuídos Tradicionais

Um dos principais trabalhos relacionados à cooperação e autonomia em detecção de intrusão é apresentado em White e Pooch [1996] e White et al. [1996], com a introdução do conceito de gerenciadores de segurança cooperativos (*Cooperating Security Managers – CSM*) para detecção de intrusão baseada em parceria (*peer-based*). Essa abordagem foca na necessidade dos sistemas de detecção de intrusão trabalharem de forma cooperativa, sem precisar de um mecanismo centralizado de análise. Ao invés disso, cada sistema de detecção de intrusão assume esse papel, trocando informações com os demais, recebendo e enviando informações. Apesar do CSM ter sido específico para rastreamento de intrusões através de *logins* de usuários em sistemas Unix, a idéia serviu de base para outros mecanismos, como o sistema de detecção de intrusão AAFID [Balasubramaniyan et al., 1998].

O AAFID [Balasubramaniyan et al., 1998][Spafford e Zamboni, 2000] introduz o conceito de agentes de software autônomos em detecção de intrusão. Os agentes autônomos do AAFID são sensores que funcionam de forma independente de outros processos. Os agentes autônomos podem ser adicionados ou removidos de um sistema sem alterar outros componentes ou reinicializar o IDS. Um agente pode também ser parte de um grupo de agentes que realizam funções simples diferentes, mas podem trocar informações e produzir resultados mais complexos do que um agente sozinho.

Diversas vantagens são atribuídas por Balasubramaniyan et al. [1998] ao uso de agentes. Como são independentes, se um agente para de funcionar, somente seus resultados são perdidos. Os agentes podem ser facilmente atualizados, simplesmente incluindo outros novos ao sistema e desativando os antigos. Cada agente pode ser escrito em uma linguagem diferente que aproveite melhor as características do monitoramento desejado e do sistema no qual ele funcionará.

O AAFID é um IDS baseado em *host* e possui uma arquitetura de controle e análise distribuída de forma hierárquica. A Figura 3.7 representa a distribuição física do modelo. Outros IDSs hierárquicos serão apresentados na próxima seção.

As entidades mais básicas do AAFID são os “agentes”, que operam como sensores que vigiam aspectos do funcionamento dos *hosts* e reportam eventos aos *transceivers*. Um *transceiver* é um analisador que funciona no mesmo *host* que seus “agentes”, sendo responsável também por atividades de controle sobre esses “agentes”. Os *transceivers* também funcionam como interfaces externas de cada *host*, recebendo comandos e enviando informações aos monitores. Os agentes não enviam alertas, somente *transceivers* e monitores podem enviar alertas. Os monitores são as entidades de mais alto

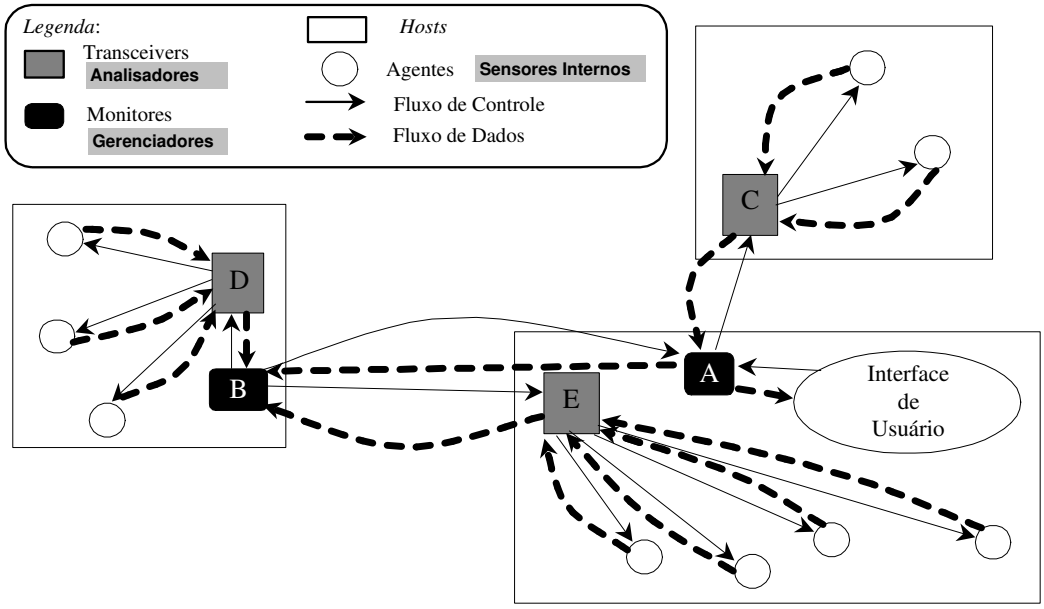

Figura 3.7: Layout físico dos componentes do AAFID

nível na arquitetura AAFID, funcionando como gerenciadores. Eles têm controle sobre os analisadores (*transceivers*) e sensores (agentes) subordinados a estes analisadores. Os monitores agrupam as informações recebidas dos *analisadores* em diversos *hosts* e trocam informações entre si a fim de detectar ataques de forma distribuída. A última peça do AAFID é a interface de usuário. Essa interface é independente das demais entidades e se comunica com os monitores para realizar as tarefas de controle e configuração das entidades do sistema, além de disponibilizar os alertas aos seus operadores. A interface de usuário não chega a ser detalhada.

Em seus papéis de controle sobre os analisadores, os gerenciadores são pontos únicos de falha. Para contornar esse problema, os autores sugerem que monitores em nível superior da hierarquia detectem a falha e ativem monitores de níveis inferiores. Outra solução apresentada seria a redundância de monitores. Porém, nenhuma das duas propostas chega a ser desenvolvida nos testos descritivos. Outro problema relatado é a falta de mecanismos de autenticação e de controle de acesso que permitam diferentes níveis de acesso ao IDS.

Outro IDS distribuído que adota o conceito de cooperação é o Indra (*Intrusion Detection and Rapid Action*) [Janakiraman et al., 2003]. Como o CSM, ele não possui qualquer modelo de hierarquia, sendo totalmente distribuído. O objetivo amplo do projeto Indra é distribuir qualquer informação de tentativa de intrusão (obtida pelas pretensas vítimas) entre todos os parceiros interessados, via uma rede *peer-to-peer* (P2P).

Cada *host* possui um IDS completo, o qual tanto observa tentativas de intrusão, quanto reforça o controle de acesso, baseado em sua memória de tentativas anteriores de ataques ao próprio *host* ou reportados pelos vizinhos. A resposta a incidentes do Indra está focada no *host* que o abriga, respondendo prontamente a uma tentativa de ataque com o bloqueio de recursos locais ao atacante.

Localmente, é adotada uma implementação baseada em *plugins* que, neste caso, são programas Java acoplados ao IDS em tempo de execução. Esses *plugins* são enviados por um administrador e após sua autenticação são carregados no espaço de execução do IDS. As tarefas que podem ser executadas pelos *plugins* não estão restritas à detecção de intrusão, podendo também realizar correções de segurança ou outras funções relacionadas ao gerenciamento do *host*.

Segundo os autores do Indra, as mensagens transmitidas entre os *hosts* são criptografadas e assinadas usando chaves assimétricas e têm sua autenticidade e integridade verificadas no destino. O formato das mensagens não é mencionado nos artigos. Enquanto o mecanismo de assinatura de mensagens está claro, o mecanismo de criptografia para grupo é dúbio. Como o modelo define que cada mensagem seja transmitida em *multicast* apenas uma vez, seria necessário o uso de chaves de grupo e não de chaves privativas individuais. Apesar do mecanismo distribuído de funcionamento, a distribuição de chaves criptográficas, na prática, é feita por um mecanismo centralizado, destoando de todo o discurso *peer-to-peer* adotado no projeto. Outro problema está na falta de mecanismos que garantam a segurança das mensagens.

As propostas de Bass [2002, 2004] partem da premissa de que uma arquitetura de processamento de dados baseada no modelo *publish/subscribe* pode ser aplicada para resolver vários desafios associados ao processamento de dados cooperativo distribuído. Nesta abordagem, um elemento que está enviando uma mensagem não precisa conhecer o destino físico da mensagem. O processo apenas rotula a mensagem com um determinado assunto e passa a mesma ao sistema de comunicação para a transmissão. O processo que recebe a mensagem também não precisa saber a localização do emissor. O processo receptor informa apenas ao sistema de comunicação quais são os seus assuntos de interesse.

Em Bass [2002], os grupos de sensores de um mesmo tipo de informação formam federações de serviços distribuídos que publicam e se inscrevem para receber informações relevantes e conjuntos de dados. No modelo proposto, cada elemento possui a capacidade de se comunicar com qualquer outro nó para obter a informação necessária para executar sua missão.

Em [Bass, 2004] é proposta uma infra-estrutura para suportar o modelo de comunicação. Essa arquitetura é baseada na coordenação distribuída e no uso de serviços que fazem a adaptação de protocolos e a transformação de dados dos mecanismos hoje existentes para uso no modelo proposto. Os elementos do modelo podem ser ferramentas de mercado que usam tais serviços para participar do processo de geração e análise dos dados.

Apesar desta preocupação com um modelo comum para a transmissão dos dados, não são adotadas especificações de padrões para protocolos ou formatos de mensagens. A falta destes padrões dificultaria bastante a interoperabilidade pretendida. Também é considerado o uso de mecanismos de segurança, porém tais mecanismos não são especificados, deixando em aberta a questão da segurança na troca de mensagens e na autenticação dos elementos envolvidos na detecção de intrusão.

Uma das experiências mais citadas e copiadas em detecção de intrusão distribuída é o EMERALD [Porras e Neumann, 1997][Neumann e Porras, 1999]. No EMERALD, a detecção é direcionada a um

"alvo" que será monitorado. Uma estrutura implementada localmente funciona como um sensor e analisador, sendo chamada de monitor de serviço (*Service Monitor*).

Tanto internamente, quanto externamente, os monitores utilizam um mesmo modelo de troca de mensagens assíncrono baseado no modelo cliente-servidor. Um módulo cliente pode se inscrever para receber dados de eventos ou resultados de análise dos servidores. Uma vez que seja aceita a solicitação de inscrição, o módulo servidor passa a enviar automaticamente os dados aos clientes.

É adotada também uma abordagem hierárquica de análise distribuída, com três níveis, conforme ilustrado na Figura 3.8. O monitor de serviço incorpora o papel de sensor no primeiro nível, podendo funcionar independentemente dos demais elementos da hierarquia. O segundo nível é a análise no escopo de um domínio, no qual estão envolvidos diversos serviços. O terceiro nível da hierarquia é o empresarial, agindo como um gerenciador que coordena e analisa eventos que cruzam diversos domínios em uma mesma organização.

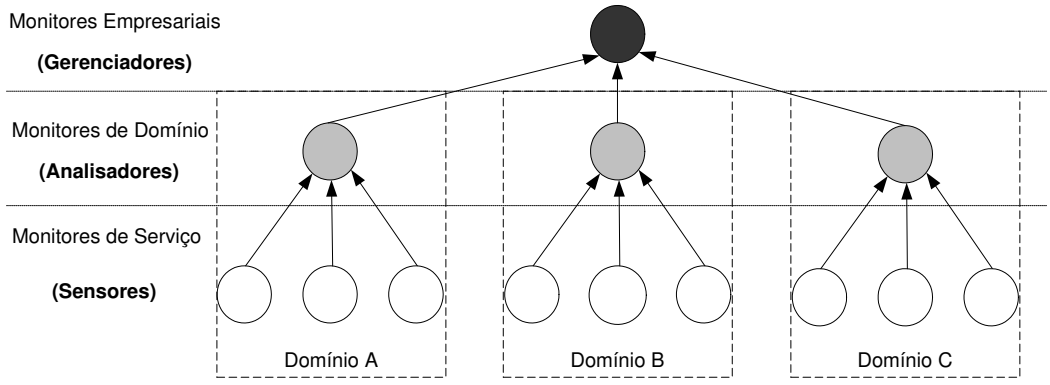

Figura 3.8: Hierarquia do EMERALD

Monitores de serviço produzem resultados da análise local que são passados ao monitor de domínio. Os monitores de domínio correlacionam os resultados dos monitores de serviço, produzindo novos resultados, que são então propagados aos monitores empresariais. Monitores empresariais correlacionam e respondem aos resultados de análises produzidas pelos monitores de domínio. Embora esse modelo não escape do *overhead* necessário para uma comunicação confiável, ele reduz a necessidade de solicitações e respostas repetitivas.

Apesar de pregar a interoperabilidade, esta questão se restringe a comparações com o modelo CIDF. Infelizmente, não há informações sobre os mecanismos de comunicação adotados para garantir tal premissa. O mesmo ocorre com os mecanismos de segurança necessários à troca de mensagens.

O sistema MADHID (*Multi-Agent based Distributed Hierarchical Intrusion Detection*) [Zhang et al., 2003] possui um modelo semelhante ao AAFID (ver seção 3.4.1). Porém, sua arquitetura é baseada em um modelo hierárquico, combinado com o modelo centralizado. Na arquitetura do MADHID, conforme ilustrado na Figura 3.9, cada *host* possui um ou mais sensores, chamados de agentes de detecção (*Detection Agent – DA*). Estes sensores enviam suas notificações a um analisador, chamado de agente coordenador (*Coordination Agent - CA*). Este analisador é o responsável pela efetivação da colaboração entre os agentes para a detecção de intrusão distribuída, podendo estar instalado no mesmo *host* que os sensores ou em outro *host*.

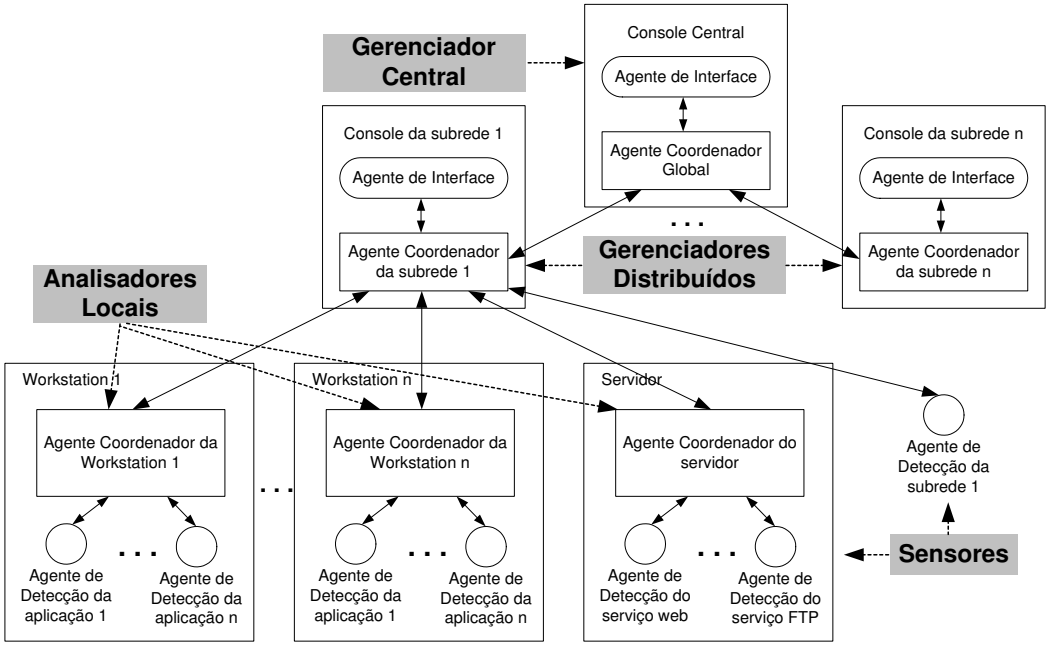

Figura 3.9: Modelo de hierarquia de agentes do MADHID

Os sensores são autônomos na detecção de intrusão e se comunicam com o coordenador local apenas quando não conseguem identificar positivamente uma intrusão. Um gerenciador central, chamado de agente coordenador global, analisa as mensagens de detecção submetidas pelos analisadores locais em diferentes *hosts* e cooperam com os gerenciadores distribuídos.

A arquitetura hierárquica permite a comunicação eficiente. Porém, ataques direcionados aos gerenciadores impediriam facilmente que o sistema efetuasse conclusões a respeito de ataques distribuídos. Um mecanismo de autenticação entre os elementos é previsto no trabalho, mas não é feita qualquer referência à segurança das mensagens entre analisadores e gerenciadores.

A família de IDSs STAT [Vigna et al., 2003] define diversos tipos de sensores próprios que coletam e analisam informações de forma distribuída. Como nas propostas de Bass, sensores de fabricantes diferentes também podem ser usados. Para isso, os alertas devem ser gerados no formato IDMEF (ver seção 3.3.2).

Para gerenciar estes sensores, parte-se do princípio que os sistemas de detecção de intrusão que operam em diferentes ambientes e com diferentes níveis de abstração (detecção e correlação), compartilham uma mesma arquitetura e possuem premissas de controle similares. Como conseqüência, uma única infra-estrutura de controle e configuração pode ser usada para gerenciar um grande número de componentes heterogêneos. Assim, foi criado um elemento chamado *MetaSTAT* que atua como analisador e gerenciador. O *MetaSTAT* também faz o roteamento dos alertas entre os sensores e outras instâncias da infra-estrutura, usando um mecanismo de comunicação chamado *CommSTAT*. O formato IDMEF foi estendido para incluir as mensagens de controle usadas na configuração e atualização dos sensores.

Os sensores são conectados a um *proxy* que funciona como uma interface com a infra-estrutura *MetaSTAT*. O *proxy* executa: o pré-processamento das mensagens, a autenticação dos elementos envolvidos na comunicação e a integração com aplicações de terceiros. Ao receber uma mensagem de controle, o *proxy* a repassa aos sensores a ele conectados, que então executam os comandos.

Não fica claro como são efetivados o controle dos elementos distribuídos de gerência e a análise, nem os mecanismos usados para implementar o *CommSTAT*. As ferramentas para autenticação e controle de acesso aos elementos, além da segurança das mensagens, também não são descritas.

3.4.2 IDSs de Larga-Escala

Podemos afirmar que os sistemas de detecção de intrusão convencionais não são estruturados para a troca de informações de segurança entre diferentes organizações. Estas informações normalmente são mantidas restritas ao escopo da organização onde foram coletadas. Contudo, como se pode perceber pela análise da literatura, uma tendência na detecção e intrusão é o uso de IDSs de larga escala que fazem a correlação de informações provenientes de múltiplas origens na Internet.

Exemplos bem conhecidos desses sistemas são o DShield.org[7] (*Distributed Intrusion Detection System*) e o myNetWatchman[8]. Essas duas iniciativas disponibilizam agentes de coleta (sensores) que são instalados em *firewalls*, roteadores ou outros dispositivos de filtragem e controle de acesso. Os dados coletados são enviados a um gerenciador centralizado para a correlação e análise. Como resultado, são disponibilizados alertas e estatísticas que irão auxiliar na prevenção de ataques.

Infelizmente, tais mecanismos de correlação possuem os inconvenientes relacionados à centralização da análise e os IDSs correspondentes não utilizam formatos e protocolos padronizados de comunicação, restringindo o uso mais amplo de sensores. Também não é possível a cooperação ou o compartilhamento direto de informações entre as organizações envolvidas na coleta dos dados.

O IDF (*Intrusion Detection Force*) [Teo et al., 2003] é um dos poucos trabalhos nesta área que propõe mecanismos para a troca de informações sobre incidentes de segurança entre organizações por meio da Internet. O artigo sugere um sistema de detecção de intrusão em escala global para a internet, que seria uma infra-estrutura virtual sobre a Internet atual, que possibilitaria a troca de informações de segurança, análise inteligente de dados e resposta a intrusões. O principal objetivo seria defender as organizações e proteger a Internet como um todo.

Segundo os autores da proposta do IDF, a arquitetura escolhida para o sistema é um modelo híbrido entre a hierárquica e a totalmente distribuída, combinando as vantagens de ambos os modelos. Sendo assim, foi decidido por uma hierarquia com dois níveis com apenas dois tipos de entidades a serem implementadas: os “nodos” e “supernodos”. O nodo corresponde ao primeiro nível da hierarquia, enquanto os supernodos estão no segundo nível, conforme ilustrado na Figura 3.10.

Cada nodo do IDF é um IDS completo que coleta dados e os analisa, além de implementar mecanismos de respostas no sistema ao qual está associado. Nodos próximos são agrupados em conjuntos

[7] http://www.dshield.org
[8] http://www.mynetwatchman.com

chamados “coletivos”. Os coletivos permitem a redundância para suportar a segurança e *survivability*. Para isso, os dados obtidos em um nodo são replicados aos demais nodos do coletivo. O nodo também funciona como sensor, compartilhando informações com os supernodos que realizam a correlação e análise das informações obtidas. Um único nodo “eleito” do coletivo fica responsável pela comunicação com os supernodos. Se ele falhar, outro do coletivo o substitui dinamicamente. Os supernodos também são agrupados formando um “supercoletivo” que compartilha informações com outros supercoletivos. A área sob autoridade de um supercoletivo é chamada de zona, que facilita o gerenciamento e administração, além de escalabilidade.

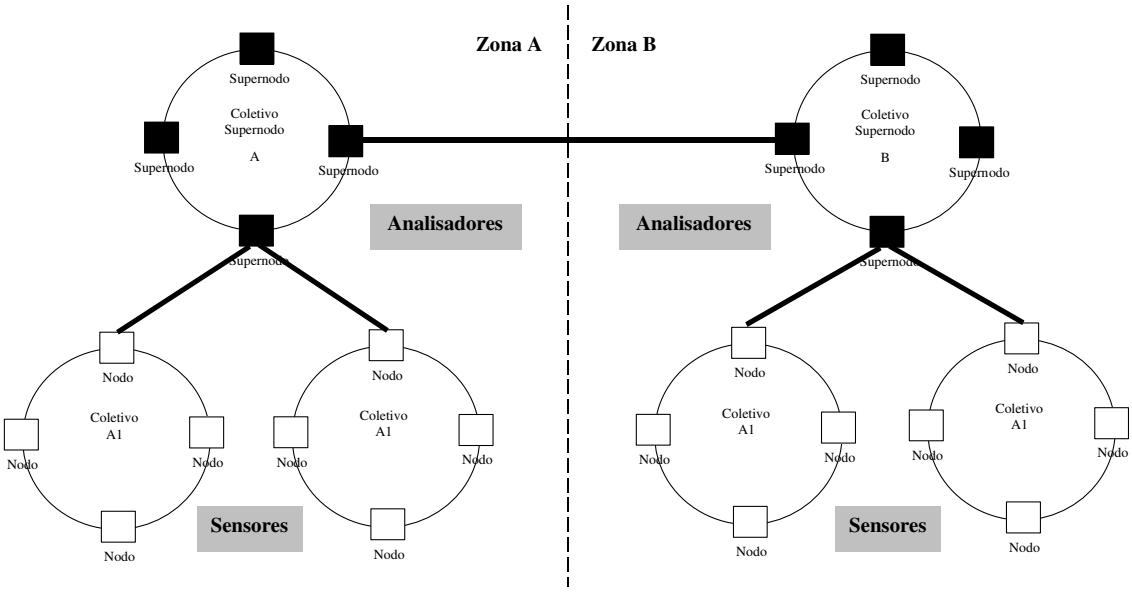

Figura 3.10: Entidades na arquitetura IDF

Dentro do modelo de detecção de intrusão distribuída, os nodos agem como sensores e os supernodos como analisadores. O IDF não define como os elementos serão gerenciados.

Não há informações sobre o protocolo de comunicação entre os nós ou o formato das mensagens que são transmitidas. Também não são mencionados mecanismos de segurança para autenticação dos elementos ou troca de mensagens. Não há outras referências sobre o IDF e aparentemente o projeto está parado. Porém, as idéias apresentadas, sobretudo a arquitetura escalar, permitem pensar em sistemas de detecção de intrusão e gerenciamento interorganizacionais mais restritos, ao invés de sistemas em escala global, nos quais haja menos restrições de segurança e a infra-estrutura necessária seja mais modesta.

Outro IDS de larga-escala é proposto pelo projeto DOMINO [Yegneswaran et al., 2004]. O objetivo do DOMINO é construir uma infra-estrutura para detecção de intrusão e resposta global distribuída por meio da combinação de dados provenientes de origens distintas. Os dados coletados são compartilhados com a finalidade de prevenir ataques de *worms*, criando uma “lista negra” de endereços que potencialmente podem ser usados em ataques.

A arquitetura do DOMINO, ilustrada na Figura 3.11, é composta de elementos analisadores (*Axis*) que fazem o cruzamento e correlação de dados provenientes de uma hierarquia de sensores

(*Satellites*) ou de outros dispositivos (*Terrestrial Nodes*), como *firewalls* e IDSs de rede. Os sensores *Sattellites* são *honey-pots* para atrair e monitorar o tráfego malicioso (*scans*) destinado a endereços não utilizados na rede.

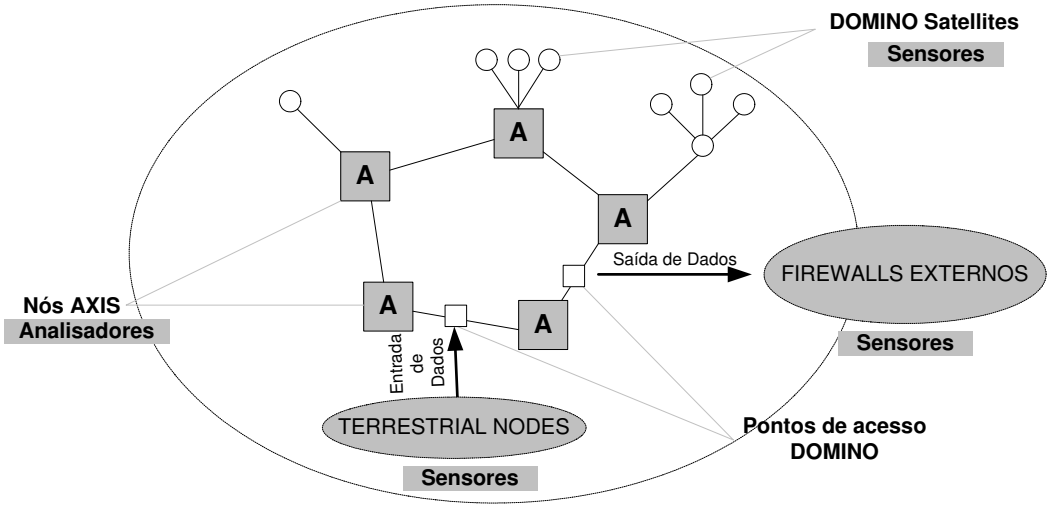

Figura 3.11: Organização dos nós DOMINO em uma rede *overlay peer-to-peer*

Os *Axis* formam uma rede *overlay peer-to-peer* que utilizam mecanismos de comunicação seguros para a troca de informações entre si. Esses mecanismos incluem a autenticação mútua, além da criptografia e a assinatura de mensagens.

Segundo os autores, o mecanismo de compartilhamento de dados (alertas) é baseado em um "espaço de tuplas único e acessível a todos" (*flat tuple space*). Porém não são dados detalhes do mesmo ou de como é formada a rede *overlay*. Também não há menção sobre a segurança destes espaços de tuplas.

No DOMINO, os sensores e analisadores são disponibilizados e gerenciados apenas pelos administradores das redes aos quais estão conectados. Estes elementos podem ser configurados, atualizados e inquiridos usando mensagens específicas. Não é definido um elemento específico de gerenciamento.

As mensagens do DOMINO são formatadas em XML usando uma extensão do formato IDMEF (ver seção 3.3.2). Esta extensão inclui novos tipos de mensagens e alterações na classificação dos alertas, que não são totalmente compatíveis com a especificação do IDMEF.

3.4.3 IDSs Usando o Paradigma de Agentes Móveis

Os agentes móveis são a forma mais poderosa de código móvel. Esses agentes de software podem migrar de forma autônoma pela rede a fim de realizar uma tarefa levando consigo seu estado de execução, além de dados e certos recursos necessários para a sua execução. As outras formas de mobilidade de código são: o código sob demanda e a interpretação remota [Fuggetta et al., 1998]. No modelo de código sob demanda, é solicitado o código a um servidor e executado no ambiente do

Tabela 3.3: Comparativo entre as propostas de MAIDS na literatura

Abordagem	Ambiente	Sensor		Analisador		Gerenciador	Arquitetura de Gerenciamento
		Mobilidade	Dados	Mobilidade	Técnica	Mobilidade	
Krügel e Toth, 2001, 2002	Fechado	Estático	Híbrido	Móvel	Abusos	Estática	Centralizada
Tripathi et al., 2002a, 2002b	Aberto	1 *hop*	*Host*	1 *hop*	Anomalias e Abusos	Estática	Distribuída
Foukia et al. 2001	Fechado	Móvel	*Host*	Móvel	Anomalias	-	Distribuída
Queiroz et al. 1999	Aberto	1 *hop*	*Host*	Móvel	Anomalias e Abusos	Móvel	Distribuída
Mell e McLarnon, 1999	Fechado	Estático	-	Móvel	-	Móvel	Distribuída
Kachirski e Guha 2002	Aberto	Móvel	*Host e* Rede	Móvel	Anomalias	Móvel	Distribuída
Guangchun et al., 2003	Fechado	1 *hop*	*Host*	Móvel	-	Estática	Centralizada
Asaka et al., 1999	Fechado	Estático	*Host*	Móvel	-	Estática	Centralizada

cliente. Os *applets* são atualmente a forma mais popular desse modelo. Na interpretação remota (ou execução remota), o cliente envia para execução esse código a um dado servidor, como ocorre nas requisições SQL.

O paradigma de agentes móveis tem se tornado nos últimos anos bastante conhecido e difundido. Sua aplicação traz vantagens que são bem conhecidas [Jansen e Karygiannis, 1999]. Levar o processamento até os dados, ao invés dos dados ao processamento, pode reduzir a carga e os atrasos (latência) na rede. A autonomia no movimento e nas decisões dos agentes enquanto desempenham suas tarefas pode tornar os sistemas independentes de um mecanismo central de controle (execução assíncrona), possibilitando também adaptação e reação a ambientes hostis.

Infelizmente, o paradigma de agentes móveis implica também desvantagens [Jansen e Karygiannis, 1999] [Vigna, 2004]. O tamanho do código do agente e dos dados que este deve transportar influe no desempenho do sistema. A carência de uma linguagem/ontologia comum, bem como a insuficiência de metodologias e de ferramentas para projeto, teste e depuração afetam o desenvolvimento de aplicações baseadas nesta abordagem. Já as ameaças à segurança limitam a maior aceitação deste paradigma. Tais vantagens e desvantagens também se aplicam ao uso do paradigma de agentes móveis nos sistemas de detecção de intrusão (MAIDS – *Mobile Agents Intrusion Detection System*).

Observa-se na Tabela 3.3 [Brandão e Fraga, 2004], um resumo de como essas experiências são estruturadas com relação ao contexto do ambiente e como cada componente do IDS é implementado. Para os sensores são verificadas a mobilidade empregada e a origem dos dados coletados. Para analisadores são observadas a mobilidade e a técnica de análise usada. No gerenciamento do IDS é evidenciado também o emprego de mobilidade. Por fim, é identificada a arquitetura de gerenciamento adotada.

A maioria dos trabalhos usando agentes móveis em detecção de intrusão segue o modelo tradicional de detecção de intrusão distribuída (Figura 3.1), com exceção de Foukia et al. [2001], que implementa IDSs completos em um único agente móvel.

Contudo, é importante observar que o uso de agentes móveis não é apropriado para alguns dos elementos de um IDS distribuído. Em Brandão e Fraga [2004], é verificada a viabilidade do uso

Tabela 3.4: Aplicações viáveis de código móvel em detecção de intrusão

Componente	Situação de Uso	Tipo de Código Móvel
Sensor	Monitoramento contínuo	Agentes Móveis de 1 *hop*
	Monitoramento não contínuo / Ambiente perigoso	Interpretação Remota
Analisador	Detecção contínua baseada em *host* ou aplicação	Agentes Móveis de 1 *hop*
	Análise Distribuída	Agentes Móveis
	Análise periódica, intermitente ou sob demanda	Interpretação Remota
Gerenciador	Localização fixa / Tolerância a Falhas	Agentes Móveis de 1 *hop* com Replicação
Resposta	Perseguição e coleta de informações	Interpretação Remota

de código móvel em cada elemento de um IDS, sugerindo, quando necessário, alguma adequação a este uso. A Tabela 3.4 apresenta um resumo dos tipos de mobilidade de código que podem ser empregados nos componentes de detecção de intrusão e em quais situações. A questão da segurança dos IDSs usando agentes móveis é muito pouco explorada na literatura. Também não são abordados os problemas relacionados à interoperabilidade.

3.4.4 Detecção de Intrusão Usando *Grids*

O uso de grades computacionais (*grids*) é uma nova abordagem para a detecção de intrusão distribuída. A proposta de [Choon e Samsudin, 2003] apresenta um IDS no qual os sensores ficam distribuídos nos nodos de uma *grid* e enviam dados a um elemento centralizado que analisa os dados e executa a tarefa de gerência. Infelizmente, a abordagem centralizada acaba por comprometer o desempenho do IDS. Uma evolução é observada nos projetos GIDA [Tolba et al., 2005], PGIDS [Leu et al., 2005a] e GIGS [Leu et al., 2005b]. Estes IDSs distribuem a tarefa de análise dos dados entre nodos de uma grade computacional. Enquanto o GIDA usa um conjunto de nodos predefinidos para distribuir a análise, o GIDS e o PGIDS utilizam balanceamento de carga para selecionar dinamicamente os nodos.

Infelizmente nenhuma destas propostas se preocupa com a padronização dos formatos para comunicação entre os elementos ou com a segurança desta comunicação.

3.4.5 IDSs de Código Aberto

O *Snort* [Beale, 2004][9] é um dos mais conhecidos e difundidos sistemas de detecção de intrusão baseado em rede, sendo também constantemente citado na literatura científica. Seu emprego nas propostas científicas é, normalmente, no papel de sensor, enviando dados a elementos de correlação e análise. Para isso, vários mecanismos de formatação e comunicação são adotados. Entre estes mecanismos de comunicação está o uso do formato IDMEF. Ao *Snort* pode ser acoplado um *plugin* que permite a conversão dos alertas originais em formato texto para o formato IDMEF. Infelizmente, o *plugin* usa uma versão desatualizada do IDMEF, que não é totalmente compatível com o padrão atual.

[9] http://www.snort.org

Outro IDS bastante conhecido é o *Prelude-Ids* [Vandoorselaere, 2006][10]. O *Prelude* é um IDS híbrido que agrega e correlaciona alertas gerados por sensores de diversos tipos e fabricantes, distribuídos em uma rede de computadores. No *Prelude*, é utilizado um modelo hierárquico de análise. Os dados coletados dos sensores são enviados a um ou mais gerenciadores. O gerenciador pode também compartilhar seus alertas com outros gerenciadores.

O *Prelude* usa uma grande quantidade de sensores disponíveis no mercado, como Snort, programas de anti-virus, firewalls, Sistemas Gerenciadores de Banco de Dados, servidores Web etc. Para que estes sensores possam se comunicar com os gerenciadores, é implementado um agente de software específico para cada sensor e que funciona de forma independente. Este software recebe os dados originais, os formata e transmite ao *Prelude*. Porém, o agente não possui capacidade de gerenciamento sobre os sensores e o gerenciador do *Prelude* não interfere nos agentes. Portanto, a ativação e configuração dos sensores e dos agentes devem ser feitas manualmente. No caso de falhas, a recuperação também é manual.

As mensagens trocadas entre os elementos do *Prelude* são enviadas sobre conexões SSL usando uma variação do formato IDMEF, que não é baseada em XML. Como o grupo IDWG padroniza apenas o uso da linguagem XML para a formatação das mensagens, o formato nativo utilizado pelo *Prelude-ids* para a comunicação com os sensores seria incompatível com o padrão original do IDMEF. No *Prelude-ids* a análise de alertas IDMEF em XML só é possível *off-line*, pela importação de arquivos com o uso de um produto comercial. Portanto, fica comprometida a adoção de sensores baseados no IDMEF padrão, para envio de alertas ao *Prelude-ids* para detecção *on-line*.

O *Prelude*-ids e o *Snort* são usados como analisadores e sensores em nosso protótipo. Para isso, foram feitas modificações para torná-los totalmente compatíveis com as especificações originais do IDMEF.

3.4.6 IDSs e Web Services

A associação de tecnologia de *Web Services* e a detecção de intrusão ainda é uma área quase inexplorada na literatura. Um IDS que emprega o modelo de detecção de intrusão do IDWG (ver seção 3.1) e *Web Services* é introduzido resumidamente em [Park et al., 2003]. O sistema de detecção de intrusão consiste em um conjunto de elementos que executam a tarefa de coleta e análise dos dados de rede e enviam alertas a um gerenciador centralizado. Os alertas são enviados por um protocolo específico do IDS, adotando o formato IDMEF e a linguagem XML. Não há informações sobre o protocolo de comunicação nem como os sensores são ativados ou configurados. Infelizmente, a proposta usa apenas um método de detecção, centraliza a análise e, apesar de adotar o IDMEF internamente, não permite a integração com outros IDSs, por usar um protocolo próprio.

Outra proposta foi apresentada em Fagundes e Gaspary [2006]. A idéia básica é semelhante à nossa, onde IDSs baseados em *Web Services* são invocados dinamicamente, seguindo um modelo para detecção de ataques específicos. Para transmitir alertas entre os sensores e o analisador é usado o

[10] http://wwww.prelude-ids.org

mecanismo de notificação de eventos para *Web Services* WSN (Web Services Notification) [Graham et al., 2006]. Infelizmente, a proposta ainda não usa um formato padronizado para a formatação dos alertas, como o IDMEF. Também não é feita qualquer referência sobre a segurança do sistema ou como será a coordenação do IDS ou o gerenciamento dos elementos. A linguagem desenvolvida para a modelagem dos ataques poderia ser substituída por linguagens padronizadas que descrevem fluxos de processos e invocações de serviços, como a adotada nesta Tese.

3.5 Considerações sobre IDSs Distribuídos

Trabalhos envolvendo a detecção de intrusão distribuída em ambientes de larga escala e o uso de padrões de *Web Services* ainda são raros na literatura relacionada. Contribui para isto o fato de que a detecção de intrusão em sistemas distribuídos de larga escala é uma área de pesquisa recente. O mesmo ocorre com as definições e padrões envolvendo *Web Services*.

Ainda são raras as propostas de detecção de intrusão em redes de larga escala. Sistemas como *DShield.org* e o *myNetWatchman* centralizam a análise e não permitem o compartilhamento direto de informações entre organizações ou redes distintas. Um dos poucos trabalhos nesta área é o IDF [Teo et al., 2003], que propõe mecanismos para a troca de informações sobre incidentes de segurança entre organizações por meio da Internet. Outra proposta é o DOMINO[Yegneswaran et al., 2004], no qual os dados coletados são compartilhados com a finalidade de prevenir ataques específicos de *worms*.

Propostas recentes de IDSs distribuídos [Zhang et al., 2003][Teo et al., 2003] [Tolba et al., 2005][Leu et al., 2005a][Leu et al., 2005b], em geral, ainda não usam formatos e protocolos padrões para a comunicação entre os elementos envolvidos na detecção. Contudo, a necessidade de se utilizar formatos comuns para a comunicação entre IDSs está presente na literatura. Um exemplo disso é a proposta de Bass [2002, 2004], na qual notificações de segurança são geradas em formatos nativos e transformadas em um formato único. Porém, infelizmente, este formato único não segue os padrões existentes.

Esforços recentes de padronização relacionados à troca de informações de segurança estão sendo desenvolvidos, principalmente, pelo IETF, através dos grupos de trabalho IDWG e INCH. Todas estas especificações são baseadas na linguagem XML [Bray et al., 2004].

São poucas as experiências na literatura ou mesmo de produtos que se utilizam destes padrões emergentes. Um dos IDSs mais populares, o *Snort*, pode utilizar um *plugin* que o permite enviar alertas no formato IDMEF. O DOMINO [Yegneswaran et al., 2004] e a família de IDSs STAT [Vigna et al., 2003] estendem o formato IDMEF para atender suas necessidades. O uso de padrões nestes casos é limitado e as extensões realizadas não são completamente compatíveis com a especificação original.

Um IDS que emprega o modelo do IDWG e *Web Services* é descrito em Park et al. [2003]. Infelizmente, o modelo usa apenas um método de detecção, centraliza a análise e, apesar de adotar o IDMEF internamente, não permite a integração com outros IDSs. Em outro modelo, apresentado em

Fagundes e Gaspary [2006], também é adotada a tecnologia de *Web Services*. Neste caso, a idéia de modelar mecanismos de análise usando componentes distribuídos é bastante interessante e pode ser potencializada se adotados os mecanismos propostos nesta Tese.

Um IDS que se aproxima da nossa proposta é o *Prelude-ids*. É um IDS híbrido que agrega e correlaciona alertas gerados por sensores de diversos tipos e fabricantes, distribuídos em uma rede de computadores. Porém o formato utilizado na comunicação com os sensores não é totalmente incompatível com o padrão original do IDMEF. Apesar disso, o gerenciador do *Prelude-ids* é capaz de formatar alertas em XML, o que facilita seu uso como sensor ou analisador em IDSs de larga escala, como o que propomos.

Ao contrário dos IDSs apresentados nesta seção, nossa proposta utiliza apenas formatos padronizados originais para a comunicação, possibilitando a integração de qualquer sensor, analisador ou gerenciador a uma composição de IDSs. O *Prelude*-ids e o *Snort* são usados como analisadores e sensores em nosso protótipo. Para isso, foram feitas modificações para torná-los totalmente compatíveis com os padrões originais.

Novas propostas para IDSs distribuídos incluem o uso de grades computacionais (*Grids*), como os projetos GIDA [Tolba et al., 2005], PGIDS [Leu et al., 2005a] e GIGS [Leu et al., 2005b]. Tais propostas distribuem a tarefa de análise dos dados coletados por sensores entre nodos de uma grade computacional. Apesar de não fazermos o uso de *Grids*, nossa proposta também pode ser aplicada em tal ambiente.

A abordagem de agentes móveis em detecção de intrusão, apesar de bastante estudada, demonstrou possuir graves restrições práticas, principalmente na questão de segurança, inviabilizando seu uso em ambientes de larga escala.

A segurança dos elementos dos IDSs e a das comunicações entre estes elementos é rara na maioria dos sistemas de detecção de intrusão distribuídos.

A Tabela 3.5 ilustra as diferenças entre os IDSs distribuídos analisados e a nossa proposta, com relação aos mecanismos de comunicação utilizados e à possibilidade de interoperabilidade com outros IDSs.

3.6 Conclusões do Capítulo

Neste capítulo traçamos um breve histórico sobre a origem e o desenvolvimento dos sistemas de detecção de intrusão (IDSs). Foram discutidos os problemas relacionados à detecção de intrusão e apresentadas as principais soluções atualmente disponíveis na literatura científica. A necessidade de cooperação entre sistemas de detecção de intrusão, o uso de elementos heterogêneos e a combinação dinâmica destes elementos indicam a direção a ser seguida para a construção dos novos IDSs.

Apesar dos avanços na área de detecção de intrusão, a implantação de IDSs em ambientes de larga escala ainda carece de uma infra-estrutura condizente com tais redes. A falta de formatos padronizados de comunicação e a carência de mecanismos de segurança são quase uma unanimidade. Também

Tabela 3.5: Interoperabilidade nos IDSs Distribuídos

IDS	Comunicação	Interoperabilidade com outros IDSs
CSM	Formato Próprio	Não
AAFID	Formato Próprio	Não
INDRA	Formato Próprio	Não
EMERALD	Formato Próprio	Não
MADHID	Formato Próprio	Não
IDF	Formato Próprio	Não
GIDA	Formato Próprio	Não
GIDS	Formato Próprio	Não
PGIDS	Formato Próprio	Não
[Bass, 2002][Bass, 2004]	Formato Próprio	Não
DOMINO	IDMEF alterado	Uso de sensores
STAT	IDMEF alterado	Uso de sensores
Snort	IDMEF desatualizado	Exportação
Prelude-ids	IDMEF alterado	Uso de Sensores e Exportação
[Park et al., 2003]	IDMEF	Não
DShield.org	Formato Próprio	Uso de sensores
myNetWatchman	Formato Próprio	Uso de sensores
Agentes Móveis	Formato Próprio	Não
Composição de IDSs	**Diversos padrões (implementado IDMEF)**	**Sim**

não há um gerenciamento padronizado que permita a coordenação dos elementos de detecção de intrusão nestes ambientes de larga escala.

Portanto, a infra-estrutura proposta nesta Tese vem ao encontro das necessidades pertinentes à construção dos sistemas de detecção de intrusão em ambientes de larga escala. Tal infra-estrutura será apresentada no próximo capítulo.

Capítulo 4

Composição de IDSs

4.1 Introdução

Este capítulo detalha a proposta de uma nova abordagem para a construção de sistemas de detecção de intrusão de larga escala, que chamamos de composição de IDSs. Essas composições de IDSs envolvem a combinação de diversos sistemas de monitoramento que coletam e analisam dados de forma distribuída e oferecem a flexibilidade da configuração dinâmica para atender a novas situações, mesmo que temporárias. As composições de IDSs, nesta abordagem, fazem uso extensivo de esforços de padronização e estão fundamentadas em uma infra-estrutura de serviços e suportes. A adoção destes padrões torna possível a interoperabilidade e a comunicação entre elementos de uma composição e, mesmo, entre IDSs completos. Os IDSs materializados a partir da infra-estrutura proposta seguem a arquitetura orientada a serviços suportada pela tecnologia de *Web Services* [W3C, 2004], com o amplo uso de textos XML [Bray et al., 2004].

A próxima seção descreve o modelo geral para composição de IDSs e a infra-estrutura de serviços necessária. Na seção 4.3 é apresentado o modelo adotado para a comunicação segura nas composições de IDSs. Na seção 4.4 está descrito o processo de criação e gerenciamento dos elementos de detecção de intrusão. Na seção 4.5 é apresentado o Serviço de Registro e Pesquisa. A seção 4.6 trata do mecanismo para a compatibilização de formatos. A seção 4.7 mostra a proposta para a criação e gerenciamento das composições. Na seção 4.8 são abordados os mecanismos necessários à segurança das composições. Na seção 4.9 são apresentados os requisitos para a auditoria nas composições de IDS. Algumas considerações sobre o modelo de composição de IDSs são traçadas na seção 4.10. Na última seção são apresentadas as conclusões deste capítulo.

4.2 Modelo Geral para Composições de IDSs

Nesta seção é apresentado um modelo geral para a composição de sistemas de detecção de intrusão. Para tanto são introduzidas as características desejáveis para o modelo proposto e a infra-estrutura projetada para atender a tais requisitos.

4.2.1 Características das Composições de IDSs

Uma composição de IDSs é efetivada com o auxílio de uma infra-estrutura de serviços. A composição de IDSs deve permitir tanto a integração de IDSs completos e independentes, quanto a configuração de novos sistemas de detecção a partir de elementos de IDSs. São adotados como elementos básicos de um IDS aqueles definidos no modelo de detecção de intrusão do IETF [Wood e Erlinger, 2002], apresentado na seção 3.1: sensores, analisadores e gerenciadores. A escolha da arquitetura orientada a serviços se deve às facilidades de integração que caracterizam as tecnologias que seguem tal paradigma.

A composição de sistemas de detecção de intrusão, a partir dos serviços do modelo proposto, deve atender às necessidades de ambientes fechados de médias e grandes empresas, mas principalmente às de ambientes abertos que fazem uso da Internet. As composições de IDSs, segundo o modelo, podem se estender por diferentes organizações, permitindo, por exemplo, o compartilhamento de alertas de segurança. Esta troca de informações pode estar sujeita a políticas que limitam o fluxo que sai de cada organização na comunicação entre elas. Para lidar com esta dificuldade, tanto os elementos de uma composição de IDSs, quanto a infra-estrutura de serviços, são representados como *Web Services*[W3C, 2004].

Em geral, os IDSs são especializados e não são capazes de tratar com informações provenientes de diversos níveis e de diferentes ambientes. O modelo que propomos enfatiza o uso de elementos heterogêneos e distribuídos, permitindo a integração de ferramentas previamente existentes, mesmo que de diferentes fabricantes. É necessário nessas composições que seus elementos compartilhem formas padronizadas de comunicação e de integração. Portanto, a interoperabilidade é fundamental nas composições de IDSs. O presente trabalho se concentra nos esforços dos organismos de padronização IETF[1], OASIS[2] e W3C[3].

Uma composição de IDSs pode ser permanente ou temporária, sendo formada para coletar dados de determinados sensores, pesquisar diversas bases de dados de eventos ou para compartilhar informações sobre um ataque em andamento. A composição dinâmica permite a adaptação a situações novas em um ambiente distribuído de larga escala.

A segurança dos próprios IDSs é obviamente um ponto crítico para qualquer sistema de monitoramento. Para tal, é necessário o uso de mecanismos que possam garantir as propriedades de segurança das próprias informações trocadas ou manipuladas nestas composições distribuídas de IDSs.

4.2.2 Infra-estrutura de Serviços para Composição de IDSs

A composição de IDSs envolve o uso de um conjunto de serviços e suportes que são apresentados em uma forma simplificada na Figura 4.1. Os elementos de uma composição de IDSs são vistos como serviços e disponibilizados por meio de *Web Services* (Elementos de IDS com suporte WS).

[1] http://www.ietf.org
[2] http://www.oasis-open.org
[3] http://www.w3c.org

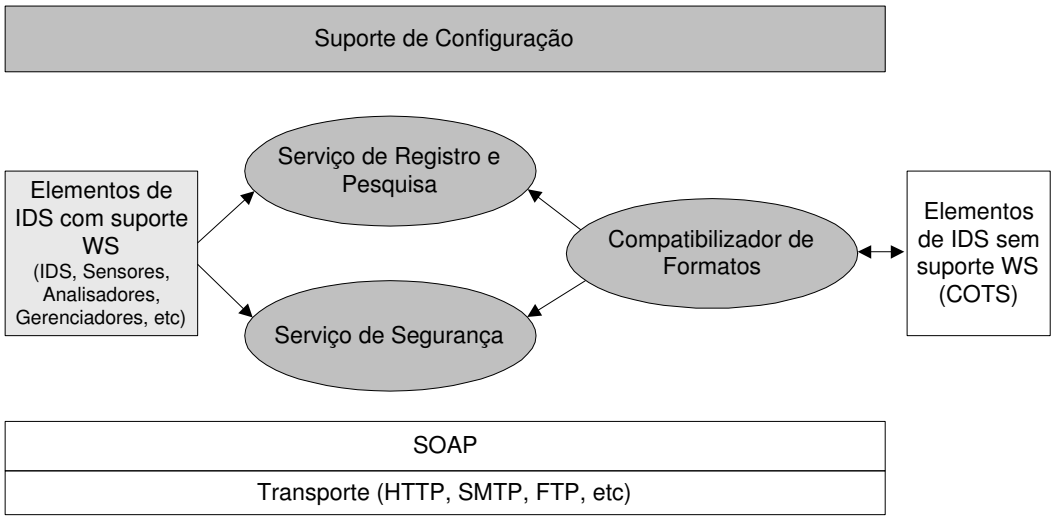

Figura 4.1: Infra-estrutura de Serviços para Composição de IDSs

As informações referentes aos serviços prestados por elementos de composições de IDSs, necessárias nas interações com os mesmos, são acessadas por meio do Serviço de Registro e Pesquisa (SRP), disponível na infra-estrutura proposta. Tal serviço está fundamentado na especificação UDDI (*Universal Description, Discovery and Integration specification*) [OASIS, 2004a]. A descrição no SRP de um elemento de detecção de intrusão, na forma de *Web Service*, além de conter a identificação do serviço e a sua localização, precisa fornecer informações relacionadas ao acesso do serviço e às políticas que o governam.

Qualquer elemento de uma composição de IDSs pode funcionar no modo requisitante (cliente), no modo fornecedor (servidor) ou em ambos os modos. As interações entre *Web Services* normalmente seguem o modelo cliente/servidor, trocando requisições (*requests*) e respostas (*replies*). Contudo, o fornecimento de serviços entre elementos de composições de IDS está sujeito às suas atividades, o que implica que as respostas dos provedores (modo fornecedor) não são imediatas a uma requisição e muitas vezes não são únicas. Por exemplo, quando uma atividade suspeita é identificada por um sensor, este como prestador de serviço envia ao cliente uma mensagem contendo um alerta de segurança. Esse monitoramento pode ter um tempo determinado de duração, após o qual o serviço deixa de ser prestado, como, por exemplo, na primeira ocorrência do evento. Mas também pode durar indefinidamente, gerando inúmeras notificações.

Os elementos de composição de IDSs se comunicam enviando notificações de eventos e alertas. Portanto, as operações de *Requests* e *Replies* são usadas para enviar as notificação de eventos e alertas entre os elementos da composição.

O Serviço de Segurança trata da autenticação, do controle de acesso dos elementos envolvidos na composição e do gerenciamento dos certificados usados na composição.

Para definir elementos “serviços” a partir de partes dos IDSs convencionais é necessário introduzir um nível de funções que formam o que chamamos de “Compatibilizador de Formatos” (CF), cuja principal atividade é tratar com os formatos usados nas trocas de mensagens das composições de

serviços. Essas funções, por exemplo, atuam também na intermediação da comunicação entre partes usadas de IDSs convencionais (sem suporte *web*) e os *Web Services* que os disponibilizam para as composições. O Compatibilizador de Formatos atua ainda na configuração dos elementos serviços e na segurança das mensagens (estabelecimento do contexto de segurança das mensagens, cifragem e assinaturas das mensagens XML, etc).

4.2.3 Interações nas Composições de IDSs

No modelo proposto, um IDS pode ser composto, por exemplo, por sensores que fornecem serviços de geração de eventos e elementos que executam análise, redução ou correlação destes eventos. Sensores e analisadores podem ser autônomos[4] ou dependentes[5], assumindo papéis de requisitantes e fornecedores de serviços dependendo do tipo de interação que esteja participando. Outro aspecto que estamos explorando é a possibilidade do compartilhamento destes componentes elementares entre IDSs; na Figura 4.2, as composições "A" e "C" compartilham um mesmo sensor, enquanto que as composições "B" e "C" compartilham um mesmo analisador. Os gerenciadores usam serviços de sensores e analisadores, podendo também interagir com outros gerentes no nosso modelo. Na figura, as setas indicam o sentido das mensagens contendo notificações de segurança.

Diferentemente do que ocorre com o modelo de detecção de intrusão do IDWG (seção 3.1), na figura, alguns elementos sensores trocam informações diretamente com o gerenciador. Isto é possível, pois os sensores da composição também podem ser IDSs completos, que agregam as funcionalidades de sensor e analisador.

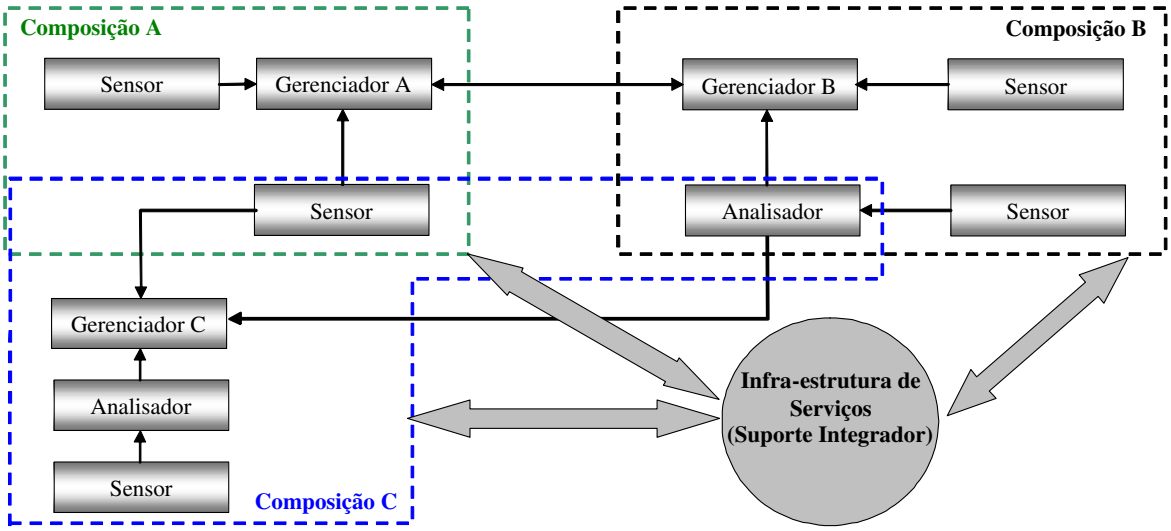

Figura 4.2: Distribuição física das composições e a comunicação entre seus elementos

A Figura 4.3 expande as composições da Figura 4.2 a fim de ilustrar as relações e interações entre seus elementos e sua distribuição física. No exemplo da Figura 4.3 as composições possuem

[4]Funcionam de forma independente de outros componentes do IDS.

[5]Dependem de outro(s) componente(s) do IDS para seu funcionamento, sendo ativado somente se requisitado por outro componente.

elementos distribuídos em quatro domínios administrativos distintos. A composição “B” é formada por elementos contidos exclusivamente no domínio “2”. A composição “A” é formada por elementos pertencentes aos domínios “1” e “3”, enquanto que a composição “C” usa elementos dos domínios “1”, “2”, “3”, e “4”.

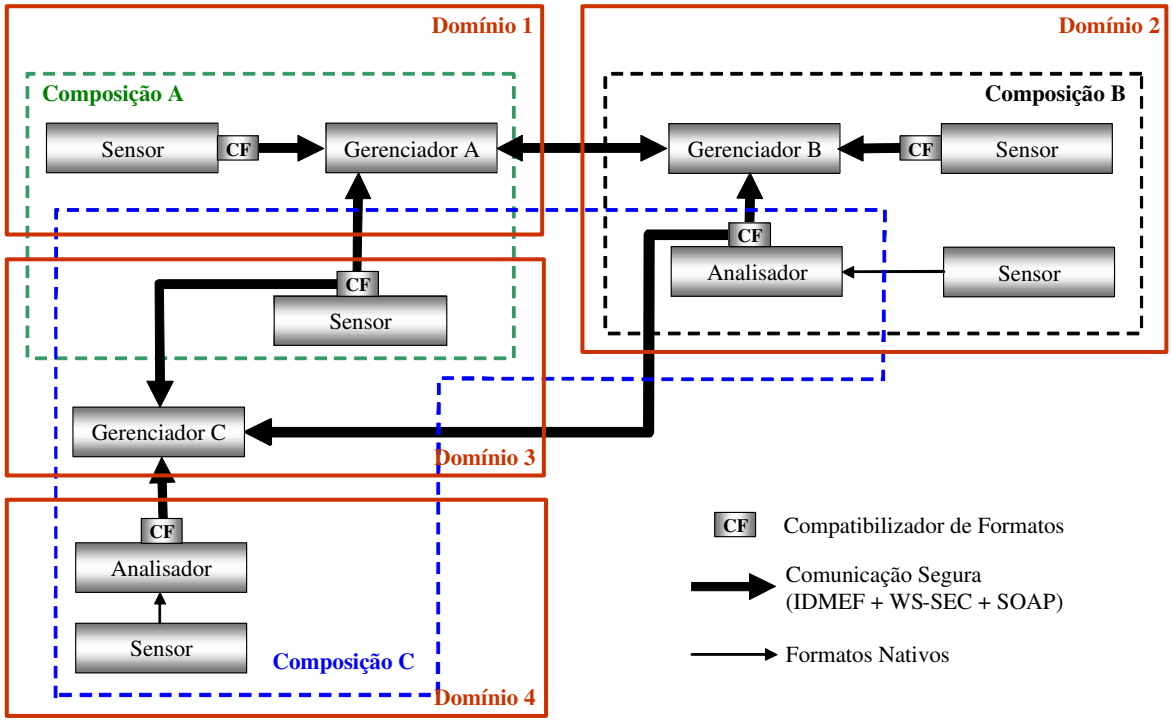

Figura 4.3: Possibilidades de Composições

Conforme ilustrado no exemplo da Figura 4.3, as mensagens entre elementos de IDS são enviadas por elementos serviços, utilizando os formatos suportados para as comunicações da composição. Quando a comunicação se dá no escopo de uma rede local, os elementos podem enviar notificações usando formatos originais dos elementos. Nas interações que envolvem domínios administrativos distintos, a comunicação está baseada em formatos padrões, como o IDMEF (Intrusion Detection Message Exchange Format) [Debar et al., 2006], seguindo um modelo de comunicação segura e padronizada, que será descrito na seção 4.3.

4.3 Comunicação Segura

Conforme apresentado no capítulo anterior, a especificação IDMEF não define mecanismos de segurança para a comunicação das mensagens. Nesta Tese adotamos a especificação *WS-Security* [OASIS, 2004c] para garantir a segurança fim-a-fim das mensagens IDMEF trocadas nas composições. Uma vez formatadas nos padrões especificados, as notificações codificadas em SOAP [W3C, 2003] são assinadas conforme a especificação *XML-Signature* [Eastlake et al., 2002][Reagle, 2000] e cifradas conforme as recomendações da especificação *XML-Encryption* [Reagle, 2002][Imamura et al., 2002]. O processo de assinatura e cifragem das mensagens faz uso de chaves assimétricas,

baseadas no modelo X.509 [ITU-T, 1993]. O encapsulamento das mensagens é ilustrado na Figura 4.4.

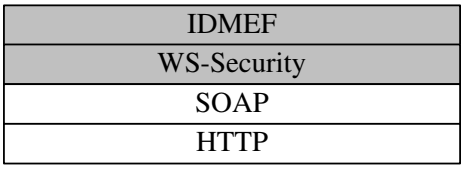

Figura 4.4: Encapsulamento de Mensagens de Detecção de Intrusão

O processo de comunicação segura é ilustrado na Figura 4.5. Neste processo, o elemento de detecção de intrusão servidor envia alertas a um elemento de IDS cliente. O elemento servidor formata o alerta IDMEF (1) e o encapsula em uma mensagem SOAP (2). Em seguida, verifica se o certificado do cliente é válido (3). O certificado do cliente é obtido pelo servidor no momento de sua ativação na composição, conforme será descrito na próxima seção. Se o certificado perdeu sua validade ou se o tempo entre a última verificação do mesmo for maior do que o definido pelo administrador, o Serviço de Segurança é consultado para a obtenção de um certificado válido. De posse do certificado do cliente, o servidor assina a mensagem usando o padrão XML-Signature (4). Uma vez assinada, a mensagem é cifrada usando o padrão XML-Encryption (5). A mensagem é então transmitida usando um protocolo de transporte padronizado (HTTP, FTP, SMTP etc) (6). O cliente, ao receber a mensagem (7), verifica se o servidor que a enviou possui autorização para interagir com o cliente (8). Se a mensagem não for autorizada, ela é descartada. Como ocorre com o servidor, o cliente verifica a validade do certificado do servidor, obtido durante sua ativação na composição (9). Caso seja necessário, um novo certificado é obtido junto ao Serviço de Segurança. De posse do certificado do servidor, o cliente decifra a mensagem (10) e verifica se a mesma é autêntica e está íntegra (11). Finalmente, é extraído da mensagem o alerta IDMEF (12).

O processo de configuração do elemento de detecção de intrusão e de inclusão dos mesmos em uma composição serão apresentados na próxima seção.

4.4 Criação e Configuração de Elementos de IDS

Os elementos que podem ser usados em composições de detecção de intrusão são registrados e descritos no SRP para serem oferecidos como *Web Services*. Suas interfaces são descritas em um formato processável, fornecido pela linguagem WSDL (*Web Services Description Language*) [W3C, 2005]. A localização da descrição destas interfaces também é mantida no Serviço de Registro e Pesquisa. Uma vez disponibilizados e devidamente registrados, os serviços destes elementos podem ser localizados para que interajam com outros serviços na composição.

Os elementos de IDS também fornecem interfaces especificadas segundo o *Web Services Distributed Management* (WSDM) [OASIS, 2004b]. A definição de interfaces segundo este padrão permite que qualquer serviço seja configurado utilizando padrões de gerenciamento específicos para *Web Services*. Tais interfaces são agrupadas segundo habilidades específicas (*capabilities*), conforme apre-

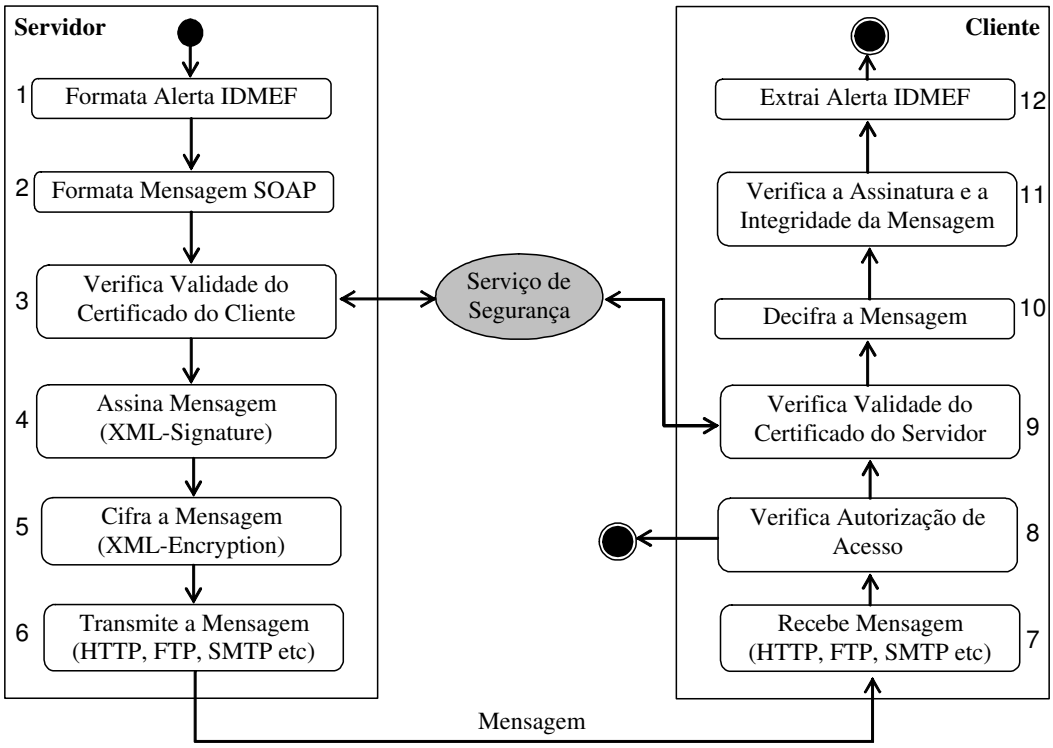

Figura 4.5: Processo de Comunicação Segura

sentado no capítulo 2, seção 2.4.6. Apesar de todas as habilidades serem desejáveis, por enquanto, apenas a de configuração é essencial para a composição de IDSs.

Cabe exclusivamente ao administrador responsável por um elemento de serviço a configuração do mesmo. Portanto, nas composições de IDSs, são selecionados os parâmetros de detecção pré-configurados, não cabendo ao responsável pela composição alterar as características de funcionamento dos elementos.

4.4.1 Inclusão de Elementos na Composição

Um operador ou responsável pela segurança de uma organização virtual obtém junto ao Serviço de Segurança um *token* que será usado para executar o processo de composição de serviços (orquestração) e interagir com os elementos de detecção de intrusão que serão adicionados à composição.

Os elementos clientes e servidores que fazem parte de uma composição são configurados para interagir entre si. O estabelecimento das relações entre estes elementos é realizado dinamicamente durante o processo de composição. O diagrama da Figura 4.6 ilustra como ocorre esse processo. Nesse exemplo, um elemento servidor (por exemplo, um sensor) é configurado para enviar alertas de segurança a um elemento cliente (por exemplo, um analisador). O servidor recebe uma solicitação de inscrição contendo um *token* de autorização (1). O servidor verifica junto ao Serviço de Segurança a autorização (2) e responde ao solicitante (3) se a solicitação foi ou não autorizada. Se foi autorizada, o servidor busca as informações necessárias para interagir com o cliente junto ao Serviço de Registro

e Pesquisa (4), como o *accessPoint*, certificados etc. Obtidas as informações necessárias, o servidor passa a enviar as notificações de acordo com a configuração solicitada. O elemento cliente também é configurado para receber alertas provenientes do elemento servidor e o método de configuração é análogo ao método de configuração do servidor.

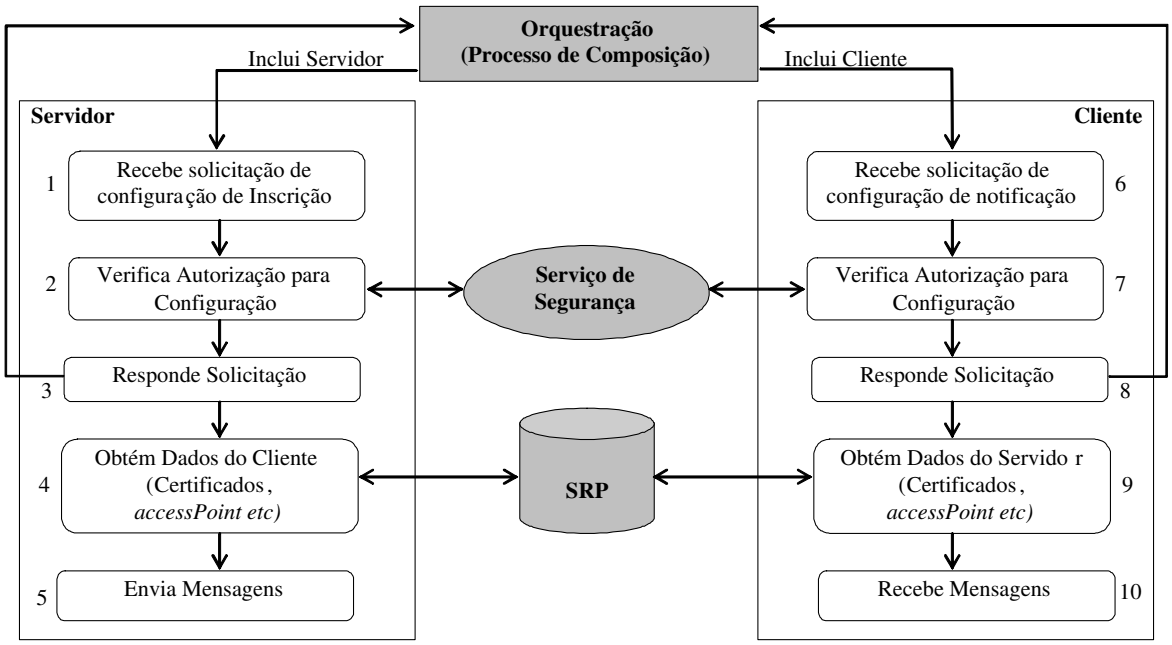

Figura 4.6: Processo de Configuração de Elementos da Composição

4.5 Serviço de Registro e Pesquisa (SRP)

O Serviço de Registro e Pesquisa (SRP) é uma peça fundamental de nossa proposta. O SRP é baseado na especificação UDDI (*Universal Description, Discovery and Integration specification*) [OASIS, 2004a]. Esta especificação usa a abordagem de registro [W3C, 2004], com ênfase na criação de domínios administrativos para o armazenamento de informações. A especificação define também os mecanismos para a associação de diversos servidores UDDI e a replicação dos dados, provendo a tolerância a falhas e a escalabilidade necessárias ao SRP.

Para a composição de IDSs, o SRP está fundamentado em dois tipos de UDDIs: primária e de consulta. A UDDI primária é aquela na qual o SRP inclui, altera e elimina registros de serviços e composições. A UDDI de consulta é usada apenas para busca por elementos públicos de detecção de intrusão e o SRP não possui permissão de incluir ou alterar registros na mesma. Os procedimentos de registro e busca serão detalhados ao longo desta seção.

A seguir, será descrita a estrutura da UDDI e como os serviços e as composições são representados utilizando tal estrutura. Também é descrito o funcionamento do SRP na busca de elementos de detecção de intrusão.

4.5.1 Estrutura da UDDI

A UDDI registra informações sobre empresas e serviços por meio de estruturas de dados em XML, chamadas de "entidades". Também é fornecida uma API padrão para a inclusão, manutenção e consulta dessas entidades. Atualmente, uma UDDI registra seis tipos de entidades, que são: *businessEntity*, *businessService*, *bindingTemplate*, *tModel*, *publisherAssertion* e *subscription*. A *businessEntity* descreve a empresa ou a organização que provê uma coleção de serviços. A *businessService* detalha a coleção de *Web Services* oferecidos pela organização descrita na *businessEntity*. A *bindingTemplate* contém as informações técnicas de um *Web Service* em particular. A *publisherAssertion* descreve os relacionamentos que a *businessEntity* tem com outra *businessEntity*. Uma *subscription* representa, no serviço de registro, uma inscrição de um interessado que deseja se manter a par de mudanças nos registro das entidades de negócios e seus serviços. As entidades *publisherAssertion* e *subscription* são utilizadas, principalmente, para a replicação de informações e mesmo associação entre UDDIs. As Estruturas do tipo *tModel* definem tipos de *Web Services*, protocolos utilizados pelos mesmos ou categorias de sistemas. As interfaces do serviço, descritas em documento WSDL, são obtidas a partir das URLs disponíveis nos *tModels*. Cada entidade possui também subestruturas.

4.5.2 Registro de Elementos de IDS

Para os registros de serviços na UDDI primária, devem ser construídas estruturas do tipo *businessServices*, conforme ilustrado na Figura 4.7. Nesse exemplo, é apresentado o registro de um elemento baseado no popular IDS *Snort* [Beale, 2004], conectado em algum ponto da rede, cujos serviços são disponibilizados por um *Web Service* denominado "*Snort WS1*". Uma chave de identificação única para a UDDI, representada pela *TAG serviceKey*, segue um formato padronizado e será usada como referência ao serviço. As subestruturas *name* e *description* descrevem o serviço de forma textual. O acesso ao componente se dá pela URL indicada no *accessPoint* contido em uma estrutura *bindingTemplate*. No exemplo da Figura 4.7 foram suprimidos alguns detalhes do registro a fim de facilitar a explicação das estruturas da UDDI.

Em geral para *Web Services*, a localização e detalhes de implementação não são relevantes. Porém, no caso de serviços relacionados à detecção de intrusão, tais informações podem ser indispensáveis para a composição dinâmica de IDSs. Isso ocorre, por exemplo, quando é preciso localizar um sensor em um ponto específico de uma rede. Em nosso caso, adotamos a classe *Analyzer* da especificação IDMEF para representar tais informações. As Informações sobre o elemento de detecção de intrusão e sua localização são incluídas no registro do serviço na UDDI em uma estrutura do tipo *bindingTemplate*. Uma *Tag* do tipo *instanceParms* permite incluir textos XML genéricos dentro desta estrutura.

Para facilitar a localização de determinado tipo de elemento para as composições de IDSs, adotamos uma classificação para os mesmos de acordo com seus papéis, conforme a taxonomia apresentada no capítulo 3, seção 3.2. Estas categorias também são armazenadas na UDDI. No registro do serviço na UDDI, a estrutura *categoryBag* indica a categoria do elemento. O IDS exemplificado na Figura

```
<businessService businessKey="104522bf-f411-0452-a82b-8c7512184005" serviceKey="10452c21 -8ae1-0452-e610-300113d34ddc">
  <name xml:lang="en">Snort WS1</name>
  <description xml:lang="en">Snort IDS 1</description>
  <bindingTemplates>
    <bindingTemplate bindingKey="10452331 -b831-0452-738c-5c24e08d8f95">
      <description xml:lang="en">Service API endpoint</description>
      <accessPoint URLType="http" useType="endpoint">http://snortws1.das.ufsc.br/ </accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="uuid:1045c387 -bf31-045c-bcc5-0fd90cc105f5" />
      </tModelInstanceDetails>
    </bindingTemplate>
    <bindingTemplate bindingKey="10452331 -b841-0452-124c-8eb6a6a02a3d">
      <description xml:lang="en">Component Details</description>
      <accessPoint URLType="other">10.1.2.3</accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="uuid:10452307 -fdc1-0452-3aca-edf37f538328">
          <instanceDetails>
            <overviewDoc>
              <description xml:lang="en">IDMEF Analyzer Structure</descript ion>
              <overviewURL>http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-14.txt</overviewURL>
            </overviewDoc>
            <instanceParms>
              <![CDATA[
                <?xml version="1.0" encoding="utf-8" ?>
                  <ids:instanceParmsContainer xmlns="urn:domain.com:analyzer_structure">
                    <Analyzer ident="uuid:das.ufsc.br:ids:snort1">
                      <Node category="dns">
                        <location>LIDAS – Terreo BU</location>
                        <name>Snort WS1</name>
                        <Address category="ipv4-addr">
                          <address>10.1.2.3</address><netmask>255.255.255.0</netmask>
                        </Address>
                      </Node>
                      <manufacturer>snort.org</manufacturer><model>Snort</model><version>2.3.3</version>
                      <ostype>Linux</ostype><osversion>2.6.11-6mdk</osversion>
                    </Analyzer>
                  </ids:instanceParmsContainer>
              ]]>
            </instanceParms>
          </instanceDetails>
        </tModelInstanceInfo>
      </tModelInstanceDetails>
    </bindingTemplate>
    …
  </bindingTemplates>
  <categoryBag>
    <keyedReference keyName="Meta" keyValue="uddi:ids:sensor:informationsource:log:meta"/>
  </categoryBag>
</businessService>
```

Figura 4.7: Exemplo de registro de um componente

4.7 desempenha o papel de um sensor. Contudo, em outros casos, um elemento pode desempenhar o papel de sensor e de analisador.

4.5.3 Registro de Composições

Uma composição de IDSs é efetivada a partir de sua representação que é dada por meio de seu registro na UDDI. Esse registro é feito na forma de um *tModel.* Documentos indicados em *tags* do tipo *overview_Doc* contidas no *tModel* irão identificar e descrever a composição. Entre esses documentos está o arquivo WSDL que descreve as interfaces da composição e a estrutura XML que descreve as interações e a organização dos elementos da composição, conforme ilustrado na Figura 4.8. A partir desses documentos, a composição poderá ser implementada.

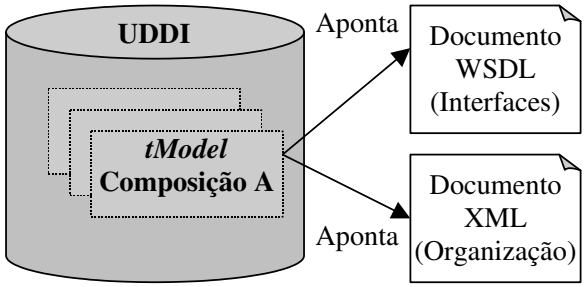

Figura 4.8: Registro de Composições

O Serviço de Registro e Pesquisa oferece uma interface que auxilia o registro da composição na UDDI primária. Os detalhes sobre esta interface serão mostrados no próximo capítulo. Os detalhes sobre os documentos e a organização da composição são apresentados na seção 4.7.

4.5.4 Busca de Elementos

O Serviço de registro e pesquisa cuida da busca de elementos de detecção de intrusão registrados nas UDDIs. A UDDI primária é cadastrada no SRP e será usada para a busca de elementos de IDS. Além deste servidor de busca, as UDDIs de consulta também podem ser usadas na pesquisa. Para isso basta que as UDDIs também estejam cadastradas no SRP e esse possua autorização de acesso às mesmas.

Para ampliar a funcionalidade do SRP desenvolvemos um mecanismo de busca recursiva que permite ao SRP consultar servidores de registro que não estavam configurados inicialmente. Para isso basta manter nas próprias UDDIs o registro de outras UDDIs. Estes registros são identificados por uma categoria, que permite ao SRP buscar de forma recursiva a lista de UDDIs disponíveis. O processo de busca por servidores ocorre a cada solicitação de busca por elementos de IDS enviada ao SRP, conforme o procedimento ilustrado na Figura 4.9.

```
SearchService(searchParms, UDDIList, stopParams){
     Para cada Servidor em UDDIList faça {
           Se Atingir(stopParms) Retorna(serviceList)
           serviceList = serviceList  + SearchService(searchParms, Servidor)
           new_UDDIList = SearchUDDI(Servidor)
           Atualiza(stopParams)
           SearchService(searchParms, new_UDDIList, stopParms)
     }
     Retorna(serviceList)
}
```

Figura 4.9: Algoritmo de busca recursiva

O SRP recebe uma solicitação de pesquisa (*SearchService*) por um elemento de IDS. Essa solicitação inclui os parâmetros de pesquisa, a lista de UDDIs que serão pesquisadas e as condições de parada da pesquisa. O SRP realiza inicialmente a busca na UDDI primária, de acordo com os parâmetros da pesquisa solicitada. O SRP envia à UDDI primária uma mensagem de busca por ser-

viços classificados como UDDI. A busca por serviços e UDDIs é repetida em todos os servidores cadastrados no SRP. Este procedimento é identificado como o primeiro **ciclo de busca**.

De posse da lista, o SRP realiza em cada um dos novos servidores descobertos uma nova busca por elementos de IDS e UDDIs. Esse procedimento é identificado como segundo ciclo de busca. Os ciclos de busca continuam até que alguma condição de parada seja atingida. As condições de parada podem ser: a descoberta de uma determinada quantidade de elementos de IDS; a quantidade de ciclos de busca; o número de servidores pesquisados; o tempo máximo de duração da pesquisa; um erro; ou a busca completa em todos as UDDIs disponíveis.

No capítulo seguinte serão dados maiores detalhes sobre a implementação do SRP e os testes realizados.

4.6 Compatibilizador de Formatos (CF)

Para permitir o uso de IDSs prontos (*commercial off-the-shelf – COTS*)[6] ou outros elementos de detecção que não possuam suporte a Web Services, disponibilizamos um serviço chamado de "Compatibilizador de Formatos" (CF), que faz a ligação entre esses elementos e a composição. O CF é responsável tanto pela comunicação de alertas, quanto pela ativação e configuração dos elementos, conforme ilustrado na Figura 4.10.

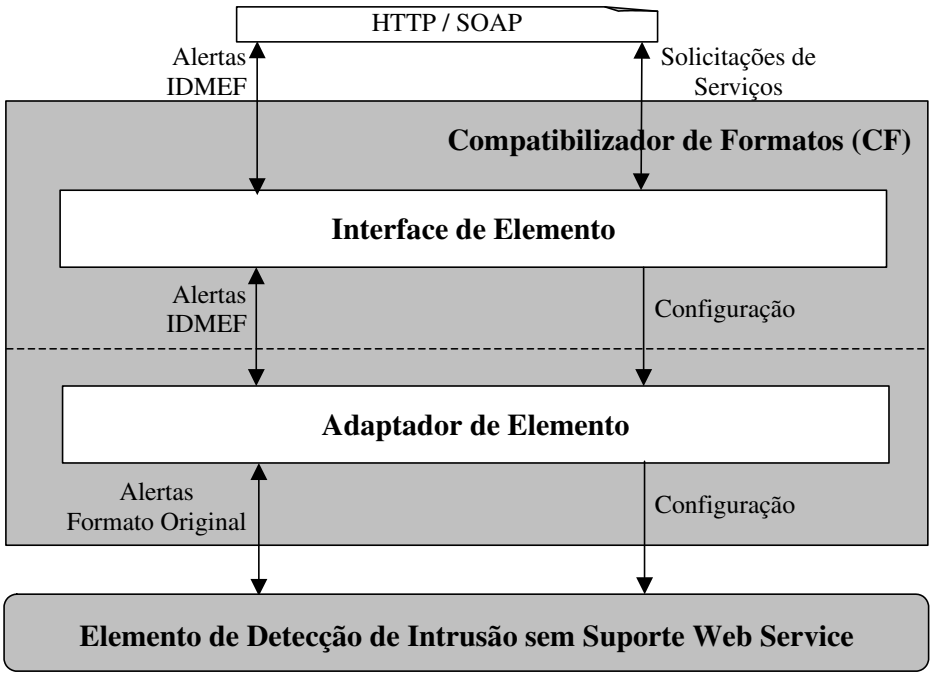

Figura 4.10: Compatibilizador de Formatos

O compatibilizador de formatos é dividido em duas partes: Interface de Elemento (IE) e Adaptador de Elemento (AE). O Processador de Mensagens é a parte do CF que faz o encapsulamento

[6] O termo COTS aplica-se a softwares prontos, que podem ser comerciais, livres ou científicos.

e a interpretação das mensagens SOAP. Ele também efetua os procedimentos para a segurança das mensagens.

O Adaptador de Elemento é funcionalmente dependente do sistema de detecção de intrusão ao qual o CF está relacionado. Os alertas nos formatos originais destes IDSs são reestruturados, pelo AE, para o formato padrão usado na interação com a composição. O AE também é responsável pela ativação e configuração do elemento, recebendo as requisições de serviço e configurando o IDS para atendê-las. O AE deve operar no mesmo *host* do elemento.

4.7 Criação e Gerenciamento de Composições de IDSs

Os modelos tradicionais de gerenciamento de *Web Services*, como o WSDM, são eficazes para a manutenção individual de um serviço. Porém, não oferecem recursos suficientes para implementar as composições de IDSs. Nesta seção analisamos brevemente as alternativas disponíveis e descrevemos o modelo para composição de IDSs adotado nesta ese.

4.7.1 Modelos para Composição de Serviços

Os termos "orquestração" (*orchestration*) e "coreografia" (*choreography*) são normalmente usados para descrever aspectos da criação de composições de *Web Services* [Peltz, 2003]. Ambos os termos são empregados para representar processos de negócios. Um processo de negócio é um conjunto de passos parcialmente ordenados, cujo objetivo é atingir uma meta como, por exemplo, a construção de um IDS de larga escala. A orquestração descreve como *Web Services* podem interagir com outros *Web Services* internos e externos a um domínio administrativo. A orquestração inclui a lógica do negócio (ou seja, o comportamento desejado da composição) e a ordem das execuções definidas a partir de fluxos de controles que atravessam organizações e aplicações. A orquestração sempre representa o fluxo de eventos da composição a partir da perspectiva de uma das partes envolvidas. No caso específico da composição de IDSs, a visão é a do administrador responsável pela segurança.

A coreografia [Austin et al., 2004] [Peltz, 2003], outro dos termos usados para composições, está relacionada às interações observáveis entre os serviços e seus usuários. Uma coreografia é um contrato multiparte que descreve o comportamento observável externo entre o serviço e seus clientes. O "comportamento externo observado" é definido pela presença ou ausência de mensagens que são trocadas entre os *Web Services* e seus clientes. Enquanto a orquestração possui o processo de negócio centrado apenas em uma das partes envolvidas, a coreografia corresponde a um acordo entre as partes envolvidas em cada possível interação. Tal modelo é colaborativo do que a orquestração. Porém, acaba se tornando mais complexo do ponto de vista do administrador que deseja formar uma composição. Coreografia e orquestração são formas complementares de compor *Web Services*.

Nesta Tese, optamos por adotar a orquestração ao invés da coreografia, pois a orquestração fornece a flexibilidade necessária para tratar com uma composição, como um todo. A orquestração per-

mite ao administrador definir o fluxo de atividades dentro de uma composição. Tal processo(fluxo) permanece inalterado quando elementos de IDS são adicionados ou removidos. Em compensação, a coreografia falha ao tratar tal evolução. Se novos elementos são adicionados, o comportamento observado muda e a coreografia precisa ser alterada. Contudo, o uso de coreografia não está descartado em uma futura investigação.

4.7.2 Orquestração de IDSs

A orquestração de serviços precisa ser flexível e adaptável para tratar com mudanças eventuais em composições de IDSs. A flexibilidade pode ser obtida realizando a separação entre a lógica do processo de composição e os *Web Services* usados. Tal separação normalmente pode ser conseguida por eio de um motor (*engine*) de orquestração. O motor cuida de todo o fluxo de processos, chamando os *Web Services* apropriados e determinando as próximas etapas que serão executadas.

A Figura 4.11, adaptada de Peltz [2003], ilustra uma orquestração simplificada da criação de uma composição de IDSs. Os passos internos são interpretados por um motor de orquestração que os executará seqüencialmente ou em paralelo, conforme definido pelo administrador. Os serviços de suporte à composição são invocados na ordem estabelecida no fluxo do processo definido. O fluxo do processo estabelece também a ordem em que os elementos da composição de IDSs são agregados à composição. Na figura os passos paralelos representam, por exemplo, a invocação simultânea de elementos sensores.

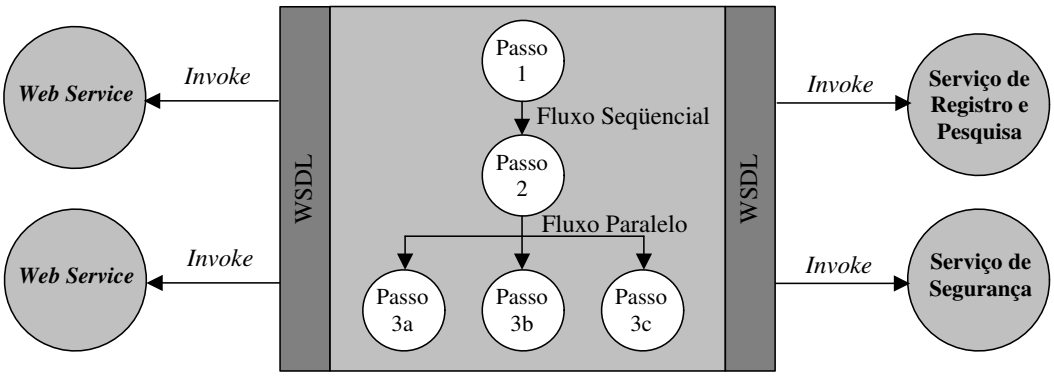

Figura 4.11: Orquestração

Toda composição de IDSs possui um ou mais Serviços Gerenciadores que são os elementos que permitem a um administrador (ou responsável) interagir com toda a composição de IDSs para a visualização de alertas e para a reconfiguração da composição. Esta reconfiguração ocorre pela adição e exclusão de elementos, além de mudanças na seleção de eventos que deverão ser reportados pelos elementos servidores.

Os detalhes sobre as ferramentas para implementação da orquestração serão apresentados no próximo capítulo.

4.7.3 Processo de Composição Dinâmica

Nesta Tese foram estudados dois tipos de orquestração: orquestração fechada e orquestração aberta. Na orquestração fechada, é criada uma composição a partir de uma descrição de orquestração, na qual as interações entre os elementos e a ordem de inclusão dos mesmos na composição são pré-estabelecidos. Porém, os elementos que farão parte da composição não são previamente conhecidos, apenas seus papéis e suas características são descritos na orquestração. Quando a orquestração é implementada, é feita a busca e a seleção dinâmica dos elementos que irão fazer parte da composição. Por exemplo, é especificado que a composição deve conter um sensor de rede localizado em um determinado ambiente. O endereço e o ponto de acesso deste sensor só serão descobertos no momento da implementação da composição.

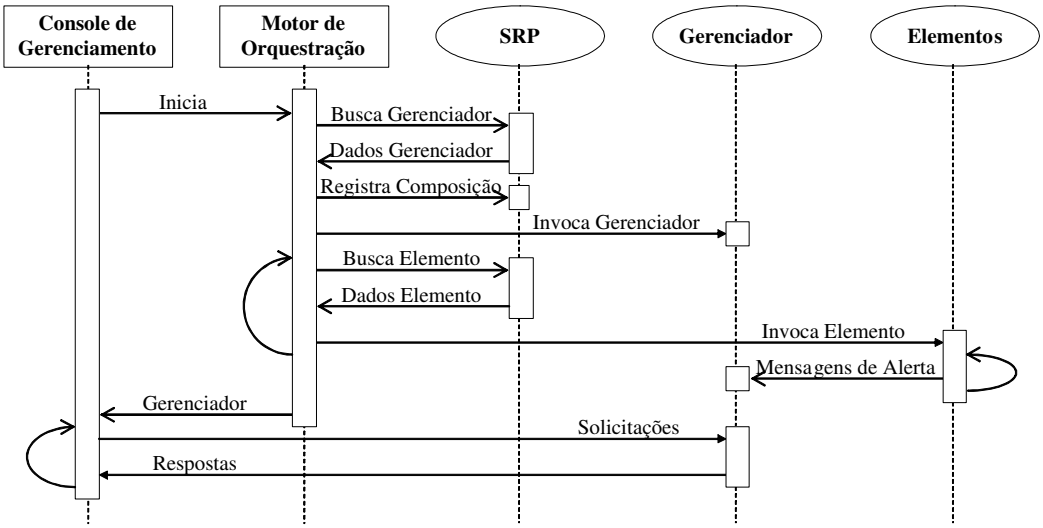

Figura 4.12: Diagrama de seqüência de uma orquestração dinâmica fechada

O processo de orquestração fechada é ilustrado na Figura 4.12. Nessa figura, o administrador executa uma orquestração por meio de uma console de gerenciamento. De acordo com a seqüência de eventos da figura, é enviada uma requisição ao Serviço de Registro e Pesquisa (SRP), a fim de localizar um elemento gerenciador, obter suas interfaces e seu ponto de acesso. Com este primeiro elemento definido, a composição é oficialmente criada e registrada no SRP. O próximo passo é ativar o serviço gerenciador. O fluxo passa então ao processo de ativação dos demais elementos da composição, buscando seus dados no SRP e invocando cada um dos elementos, que passam a enviar seus alertas a um ou mais clientes. Usando a console de gerenciamento associada ao serviço gerenciador, o administrador pode visualizar os alertas e interagir com a composição. Infelizmente, este tipo de orquestração não permite a inclusão de elementos que não foram previstos inicialmente.

Na orquestração aberta, as composições de IDSs são criadas a partir de uma descrição básica inicial contendo um modelo geral de composição. Este modelo é executado por um motor de orquestração que inclui os elementos de detecção de intrusão na composição. O diagrama da Figura 4.13 ilustra o mecanismo dinâmico da composição. Na orquestração aberta, além da busca dinâmica, as interações e as características dos elementos também são especificadas dinamicamente, por um procedimento externo, usando a console de gerenciamento. Portanto, a composição é ativada de acordo

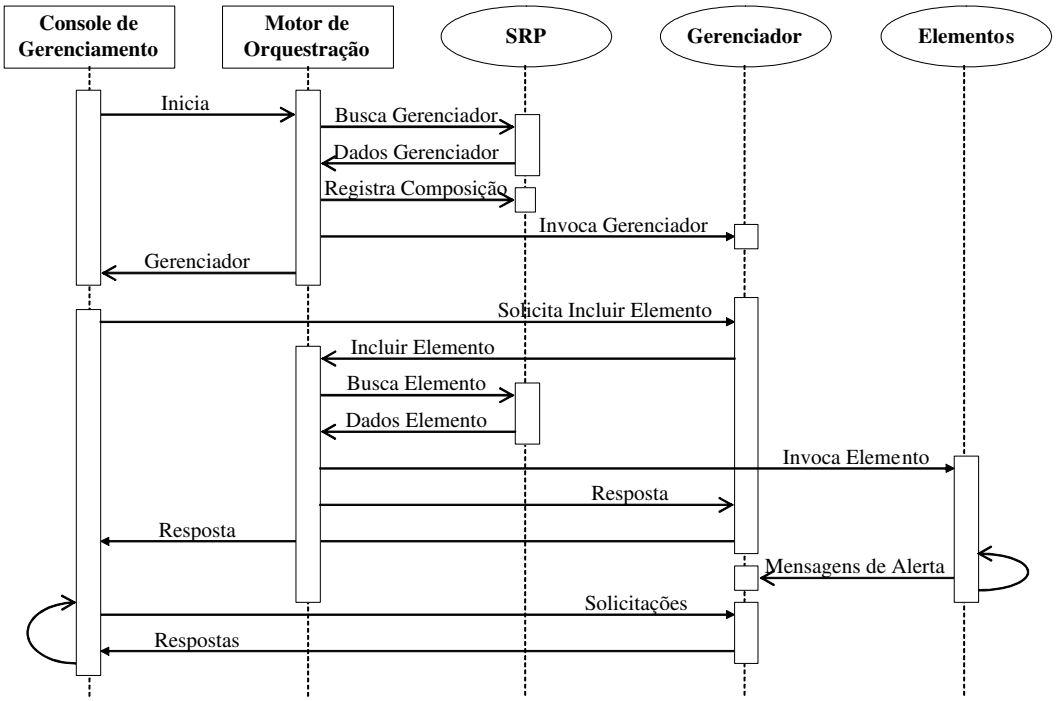

Figura 4.13: Diagrama de seqüência de uma orquestração dinâmica aberta

com os parâmetros informados por esse procedimento externo e não estão descritos na especificação da orquestração básica inicial. Apesar de propiciar maior flexibilidade, a orquestração aberta não mantém o estado da configuração da composição, impossibilitando sua reconstrução em caso de falhas.

Para solucionar esse problema, adotamos um modelo híbrido, no qual as associações configuradas na orquestração aberta são armazenadas como o registro em uma nova orquestração, que poderá ser importada e executada novamente em caso de falhas. A Figura 4.14 ilustra o funcionamento desse modelo. Na figura, a composição é iniciada a partir de uma descrição predefinida. Depois de concluída a ativação dos elementos, a orquestração passa a interagir com o gerenciador e com a console de gerenciamento para incluir, alterar ou excluir elementos da composição. O estado da configuração é armazenado dinamicamente no arquivo que descreve a composição. Para reiniciar a composição, basta executá-la novamente no motor de orquestração, a partir de seu registro no SRP.

4.7.4 Manutenção da Consistência da Composição

Um dos maiores problemas de configuração dinâmica em sistemas distribuídos é a manutenção da consistência da aplicação distribuída. A necessidade de retirada e a substituição de componentes é sempre problemática do ponto de vista da aplicação. No caso do nosso modelo de orquestração aberta, estes problemas de consistência são controlados pela facilidade de uma visão única da composição (a composição básica inicial e registros subseqüentes) e pela centralização do processo de negócio. Porém, são necessários mecanismos para identificar possíveis falhas de componentes e exclusões forçadas dos mesmos.

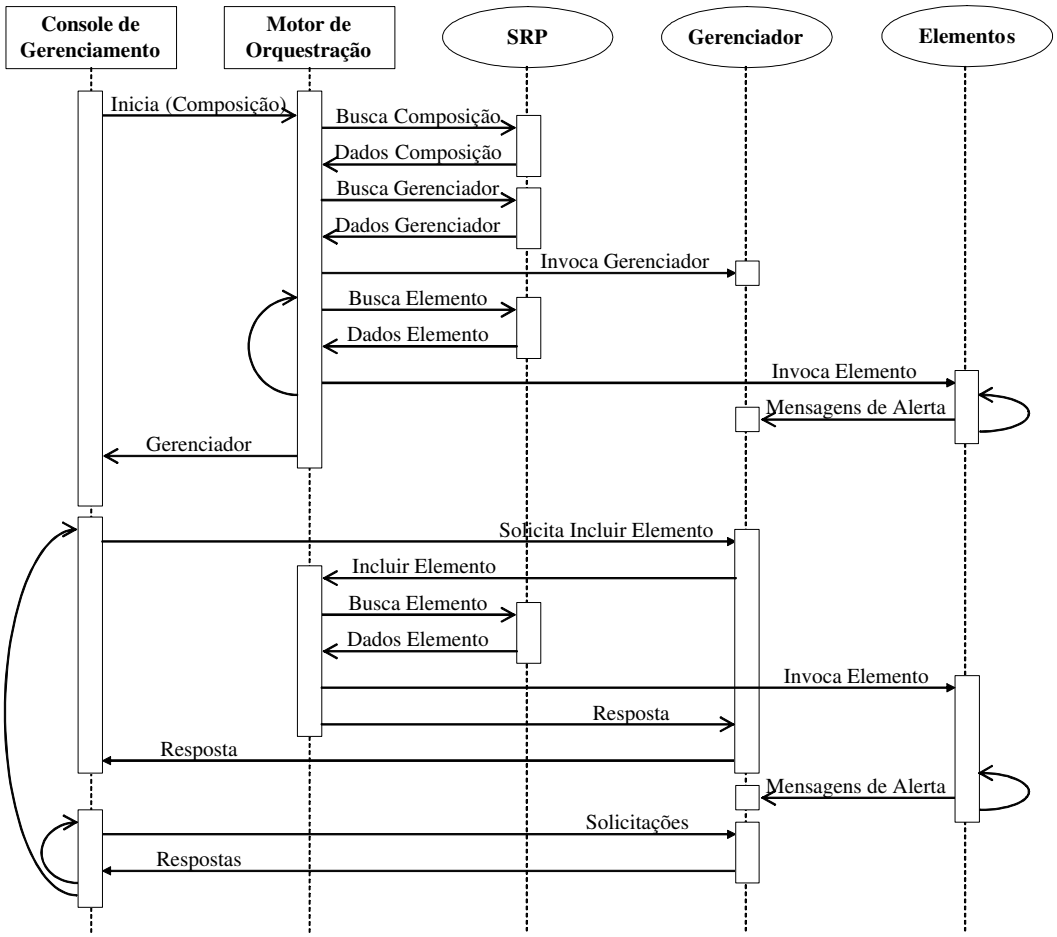

Figura 4.14: Diagrama de seqüência da orquestração dinâmica proposta

A composição de IDS cria uma série de dependências entre os elementos, seja no papel de provedor de serviço, seja no papel de consumidor desses serviços. A exclusão de um elemento na extremidade de uma composição (por exemplo, um sensor) é mais simples, pois basta configurar este elemento provedor para parar de enviar mensagens à composição e configurar os elementos consumidores do serviço para que interrompam a recepção de mensagens daquele provedor.

Para a exclusão de um elemento consumidor, localizado no meio de uma composição hierárquica, é preciso executar de forma recursiva a exclusão de todos os elementos provedores de serviço ao elemento a ser excluído. Também é preciso configurar os elementos consumidores do serviço para que deixem de receber mensagens daquele provedor.

Para detectar erros nos provedores de serviço, é usada a mensagem *Heartbeat* do protocolo IDMEF, que é enviada periodicamente aos consumidores do serviço a fim de verificar se o elemento continua operando. Se o elemento falhar ou for retirado da composição, a mensagem deixa de ser enviada e a falha é detectada pelo elemento consumidor. Uma mensagem de erro reportada pelo consumidor do serviço ativa o processo de reorganização da composição.

O procedimento de reorganização é diferente do procedimento de exclusão, pois podem haver disponíveis outros elementos de IDS capazes de executar as tarefas do elemento faltoso. Nesse caso, é realizada uma nova busca no SRP a fim de localizar tais elementos. Uma vez localizado um novo

elemento, aquele faltoso é substituído e os elementos que interagiam com ele são reconfigurados para interagir com o seu substituto.

Caso não seja possível substituir um elemento faltoso por outro similar, a composição deve ser revista. O procedimento de revisão consiste em eliminar da composição o elemento faltoso, de forma semelhante à exclusão de elementos descrita anteriormente.

4.8 Serviço de Segurança

O Serviço de Segurança realiza ações que auxiliam na autenticação e no controle de acesso dos elementos envolvidos na composição. O Serviço de Segurança também está baseado na UDDI. As chaves públicas a serem usadas nas comunicações entre os elementos da composição também seguem a especificação *WS-Security* e são obtidas na UDDI, junto com o registro destes elementos, por meio do Serviço de Segurança. No registro do elemento, uma estrutura *bindingTemplate* armazena a chave pública ou indica sua localização. As chaves públicas e chaves privadas são baseadas no modelo X.509 [ITU-T, 1993]. A Figura 4.15 mostra um exemplo da estrutura *bindingTemplate* que aponta para uma URL, na qual pode ser obtida a chave pública do serviço.

```
<businessService businessKey="104522bf-f411-0452-a82b-8c7512184005" serviceKey="10452c21-8ae1-0452-e610-300113d34ddc">
   <name xml:lang="en">Snort WS1</name>
   <description xml:lang="en">Snort IDS 1</description>
   <bindingTemplates>
      •••
      <bindingTemplate bindingKey="10452331-b841-0452-124c-8eb6a6a02a3d">
         <description xml:lang="en">Public Key</description>
         <accessPoint URLType="http">http://ftp.das.ufsc.br/pubkeys/snortws1</accessPoint>
         <tModelInstanceDetails>
            <tModelInstanceInfo tModelKey="uuid:10452307-fdc1-0452-3aca-edf37f538328">
            </tModelInstanceInfo>
         </tModelInstanceDetails>
      </bindingTemplate>
      •••
</businessService>
```

Figura 4.15: Exemplo de Registro de Chave Pública na UDDI

As etapas de segurança para o registro de um elemento de detecção de intrusão, na forma de *Web Service* são as seguintes: (1) a geração das chaves pública e privada que serão utilizadas nas interações do elemento; (2) o registro na UDDI da chave pública do elemento ou o endereço (URL), no qual a chave pode ser obtida; e (3) o registro na UDDI da política de segurança para acesso ao elemento.

Cada domínio administrativo de uma organização virtual deve possuir pelo menos um Serviço de Segurança, onde elementos de IDSs possuem suas listas de controle de acesso. O administrador responsável pela segurança de uma organização virtual precisa se autenticar junto ao Serviço de Segurança do seu domínio para poder ativar uma composição de IDSs. Ele receberá as credenciais (certificados e demais informações) que serão repassadas ao gerenciador da composição.

Neste trabalho assumimos que já existe uma relação de confiança pré-estabelecida entre as organizações envolvidas no processo de composição de IDSs, de forma que as informações necessárias para a autorização de acesso possam ser obtidas por meio do Serviço de Segurança.

4.9 Auditoria

Além das mensagens de alerta de detecção de intrusão, os *Web Services* que oferecem os serviços dos elementos de IDS precisam gerar evidências de suas atividades administrativas que demonstrem como uma transação específica foi iniciada, processada e encerrada. Sendo assim, a ativação de um serviço, as solicitações de novos serviços, as alterações na configuração do serviço e os erros nas transações precisam ser registradas para auditoria futura.

Para suportar a auditoria, o *Web Service* é configurado para enviar notificações sobre suas atividades administrativas. Estas notificações são enviadas a um sistema de armazenamento de eventos de auditoria ou a uma console de administração ou mesmo a um sistema de detecção de intrusão associado a uma composição. As interfaces necessárias para realizar tal operação são semelhantes às utilizadas para o envio de alertas de detecção de intrusão.

4.10 Considerações sobre Composições de IDSs

Nesta seção traçamos algumas considerações sobre o modelo proposto para a composição de IDSs. O uso de composições de IDSs dinâmicas, suportadas pela infra-estrutura de serviços proposta nesse texto, torna possível a construção de soluções mais adequadas para o monitoramento de segurança em redes de larga escala. O uso de *Web Services* permite que elementos de composições para detecção de intrusão possam ser acessados atravessando domínios de redes distintas ou, ainda, segmentadas. Usando esta infra-estrutura proposta, elementos para composição de IDSs podem ser implementados em ambientes de larga escala, registrados e ativados sob demanda. O uso da UDDI com uma classificação específica destes elementos torna a busca de serviços especializados mais fácil e precisa.

Aparentemente, as composições de IDSs poderiam ser implementadas segundo um modelo tradicional de gerenciamento, no qual agentes de software interagem com os elementos de IDSs para configurá-los e transmitir suas informações aos demais elementos. Porém, tal abordagem acabaria gerando restrições para a interoperabilidade entre os elementos e não resolveria o problema de comunicação entre redes distintas, sobretudo se os elementos estiverem disponibilizados em redes com endereços administrativos que não podem ser atingidos por uma rede externa.

O uso de *Web Services* e das especificações que padronizam as comunicações entre IDSs resolvem parte do problema. Para as composições é necessário descrever o fluxo dos dados, bem como o próprio processo de composição. A orquestração de serviços surge como a alternativa ideal para gerenciar as composições, pois permite descrever tanto o processo de composição, quanto a troca de mensagens entre seus elementos. A forma com que os registros são descritos na UDDI foi modificada para simplificar o processo e foi reduzida a quantidade de descrições de configuração, criando uma visão unificada, padronizada e portável da composição.

Assim, nesse novo modelo, o suporte necessário às composições são simplificados. Eles não interferem na configuração do elemento, mas no comportamento da composição.

A configuração dos elementos, por outro lado, pode passar a ser realizada por ferramentas genéricas de gerenciamento, como o *Tivoli* da IBM[7], CA *Unicenter*[8] e o HP *Openview*[9], que já começam a oferecer suporte à especificação WSDM.

4.10.1 Novas Possibilidades de Aplicações

A cooperação entre empresas e organizações é outro ponto importante que pode ser viabilizado com a presente proposta. Os elementos de detecção de intrusão podem ser disponibilizados entre parceiros para permitir a investigação de atividades suspeitas provenientes de suas diversas redes de computadores. Isto, porém, requer a especificação e o monitoramento de políticas de segurança precisas entre as partes envolvidas.

Esta proposta favorece também a terceirização da tarefa de análise de alertas de segurança. Serviços analisadores mantidos por centros de resposta a incidentes de segurança ou empresas prestadoras de serviço podem ser ativados para receberem alertas e correlacioná-los com alertas recebidos de outros clientes. Como retorno, cada cliente pode obter alertas globais, estatísticas precisas sobre incidentes e parâmetros para configuração de suas redes.

Redes de sensores do tipo "*Honey Nets*[10]" também podem facilmente ser formadas usando o modelo de composição de IDSs e a infra-estrutura de serviços proposta.

4.11 Conclusões do Capítulo

Neste capítulo foi apresentado um modelo de uma nova abordagem para a composição dinâmica de sistemas de detecção de intrusão. O modelo permite a integração de elementos de IDSs ou mesmo de sistemas monolíticos para criar sistemas de detecção de intrusão distribuídos, em ambientes de larga escala. Para conseguir flexibilidade e interoperabilidade nestas composições dinâmicas, é proposta uma infra-estrutura baseada em serviços que deve servir de suporte a essas composições. Essa infra-estrutura está fortemente fundamentada na tecnologia de *Web Services* e em padrões para comunicação de alertas de segurança.

A infra-estrutura é apresentada neste texto como uma estratificação de serviços. Esses serviços contribuem nos vários níveis para viabilizar as composições de IDSs e tornar as comunicações entre seus elementos interoperáveis e seguras. Também é proposta a criação e o gerenciamento das composições utilizando o conceito de orquestração de *Web Services*.

Apesar da proposta enfatizar a integração dos diversos padrões adotados neste trabalho, isso na prática não é tão trivial. As várias organizações que definem os padrões nem sempre estão de acordo

[7] http://www-306.ibm.com/software/tivoli/

[8] http://www3.ca.com/solutions/Solution.aspx?ID=315

[9] http://www.managementsoftware.hp.com/

[10] *Honey Nets* são redes de sensores de detecção de intrusão (*honeypots*), que tem por objetivo serem atacadas para prover aos administradores informações que levem a um maior conhecimento sobre vulnerabilidades e mecanismos de ataques (http://www.honeynet.org)

e especificações que seriam, aparentemente, complementares acabam sendo incompatíveis. Outra dificuldade está relacionada a trabalhos de padronização incipientes. Apesar de tudo, foi possível encontrar um meio termo que permitiu integrar especificações incipientes com modelos e padrões reconhecidos.

Portanto, em vista do que foi abordado, acreditamos que a proposta de modelo e a infra-estrutura de serviços apresentadas neste capítulo representem uma contribuição significativa para a detecção de incidentes em sistemas de larga escala.

Capítulo 5

Protótipos e Experimentos

5.1 Introdução

A infra-estrutura de serviços proposta para a criação de composições de IDSs é materializada na forma de protótipos. O ambiente de desenvolvimento, as ferramentas utilizadas e os protótipos implementados são apresentados neste capítulo.

Os protótipos têm por objetivo estabelecer evidências da exigüidade do modelo para detecção de intrusão em sistemas de larga escala proposto nesta Tese. Após sua implementação, os protótipos são avaliados na prática, através de alguns experimentos que serão apresentados neste capítulo. Uma avaliação mais detalhada, focada na segurança dos modelos implementados e seguindo a metodologia de gestão de riscos, é apresentada no anexo A.

5.2 Protótipos

Para testar as funcionalidades propostas nesta Tese, foram desenvolvidos alguns protótipos relacionados à infra-estrutura de serviços e aos elementos de detecção de intrusão, conforme ilustrado na figura 5.1. Na infra-estrutura de serviços destacamos os seguintes protótipos: Serviço de registro e Pesquisa; Serviço de Segurança; e Compatibilizador de Formatos. Para o teste da infra-estrutura, desenvolvemos um elemento gerenciado e adaptamos dois sistemas de detecção de intrusão existentes no mercado, para realizar os papéis de sensores e analisadores. Na orquestração de serviços foi usada uma linguagem de orquestração padronizada e um ambiente para a sua execução. A seguir, estão descritos alguns detalhes sobre os protótipos.

5.2.1 Serviço de Registro e Pesquisa

Para a implementação do Serviço de Registro e Pesquisa foram selecionadas inicialmente três implementações de UDDI: a UDDI inclusa no *BEA WebLogic Workshop*; a *JUDDI*[1] do projeto *Apache*;

[1] http://ws.apache.org/juddi/

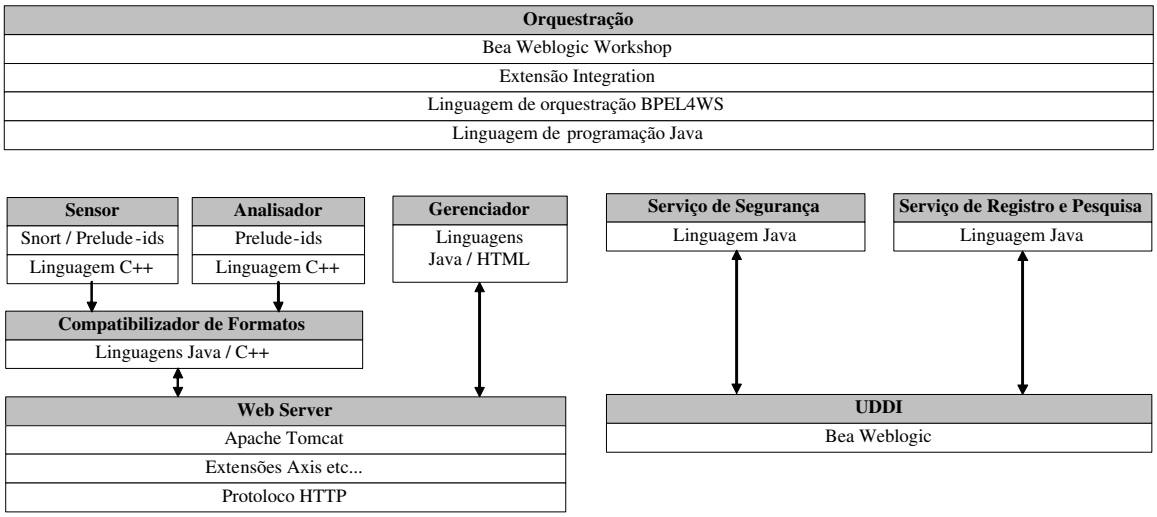

Figura 5.1: Protótipos do modelo

e a UDDI integrante do pacote Java *WSDP*[2] (*Java Web Services Developer Pack*). Tais implementações foram escolhidas por serem as mais usadas e por disponibilizarem versões para o sistema operacional *Linux*.

As implementações do *WSDP* e a *JUDDI* são bastante simples e fáceis de usar. Porém, o tempo de acesso experimentado foi bastante lento e não há mecanismos para a validação dos registros, permitindo o envio de requisições XML com erros. Por outro lado, a UDDI inclusa no *Weblogic Workshop* possui um mecanismo de validação dos registros e mais recursos de gerenciamento do que as demais, inclusive com a possibilidade de integração com outras UDDIs e criação de *clusters*. Por esses motivos, optamos por adotá-la. A primeira versão testada do *WebLogic Workshop* (a versão 8.1) apresentava erro no uso de estruturas do tipo *instanceParms*[3], que foi solucionado nas versões subseqüentes.

Antes de usar os procedimentos de registro e consulta, foi necessário introduzir na UDDI a taxonomia que classifica os elementos de detecção de intrusão. Todas as implementações fornecem recursos para configuração ou importação de taxonomias. A Figura 5.2 mostra parte do *tModel* com a taxonomia que foi importada na UDDI do *Weblogic Workshop*.

Interfaces do SRP

Para facilitar o registro e pesquisa dos elementos e das composições na UDDI, além das interfaces padrões de uma UDDI [OASIS, 2004a], foram criadas as seguintes interfaces de serviço:

1. *ClientRequestDiscovery* – executa o processo de descoberta por serviços na UDDI, a partir dos parâmetros *categoryBag* e *Address*. O parâmetro *categoryBag* contém as informações sobre a categoria do elemento desejada, de acordo com a taxonomia implementada. O parâmetro *Address* define o endereço ou máscara de rede do elemento. Como resultado da execução desta

[2] http://java.sun.com/webservices/downloads/webservicespack.html
[3] http://forums.bea.com/bea/thread.jspa?threadID=600004490&tstart=75

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance" xmlns:xsd="http://www.w3.org/1999/XMLSchema">
  <SOAP-ENV:Body>
    <Taxonomy checked="true" type="categorization" xmlns="urn:uddi-org:api_v2" >
      <applicability>
        <scope>businessService</scope>
        <scope>bindingTemplate</scope>
        <scope>tModel</scope>
      </applicability>
      <tModel tModelKey="uuid:1dbd1d96-4908-470f-8c16-593f1fa23d4b" >
        <name>Intrusion Detection Classification</name>
        <description xml:lang="en">Intrusion Detection System Sensor Taxonomy</description>
    <overviewDoc>
          <description xml:lang="en">
            This tModel describes taxonomy for intrusion detection components.
            The main components are: sensor, analyzer and manager.
            It is not a standard taxonomy.
            The analyzer taxonomy is based on Debar et al.(1999), Alexon(2000) and McHugh(2001) papers.
            The sensor taxonomy is based on MAFTIA deliverable D3 Version 1.01 document.
            Until now, there is not a wide accepted IDS manager taxonomy.
          </description>
          <overviewURL>http://www.das.ufsc.br/~jemsb/composition/ids_taxonomy.html</overviewURL>
        </overviewDoc>
        <categoryBag>
          <keyedReference keyName="TModel categorization type" keyValue="categorization" tModelKey="uuid:C1ACF26D-
9672-4404-9D70-39B756E62AB4"/>
        </categoryBag>
      </tModel>
      <categories>
        <category keyName="Sensor" keyValue="uddi:ids:sensor">
          <category keyName="Information Source Type" keyValue="uddi:ids:sensor:informationsource">
            <category keyName="Raw Data" keyValue="uddi:ids:sensor:informationsource:raw">
              <category keyName="External" keyValue="uddi:ids:sensor:informationsource:raw:external">
                <category keyName="Network Packet Sniffer"
keyValue="uddi:ids:sensor:informationsource:raw:external:sniffer"/>
              </category>

...
      </categories>
    </Taxonomy>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figura 5.2: Exemplo de registro da taxonomia de IDSs na UDDI

operação, o SRP retornará uma lista com os *accessPoints* dos elementos de IDS que cumprem as exigências especificadas nos parâmetros.

2. *ClientRequestCreate* – efetua a criação do registro de uma nova composição na UDDI primária do SRP. O resultado desta operação será o parâmetro *tModelKey*, que identifica a composição na UDDI.

As demais interfaces seguem a mesma nomenclatura das interfaces padrão da UDDI.

Procedimentos básicos de registro e consulta

Para o registro e busca na UDDI podem ser usadas ferramentas disponíveis nas implementações das UDDIs testadas. Porém, tais ferramentas são limitadas e não suportam todos os parâmetros necessários ao registro e pesquisa dos elementos de detecção de intrusão. O ideal seria a disponibilidade de uma interface gráfica que auxiliasse nas tarefas de criação e manutenção de registros na UDDI. Porém, não chegamos a criar tal interface gráfica. Optamos por introduzir no protótipo do gerencia-

dor um mecanismo simplificado que lê um arquivo XML contendo a operação desejada e executa a submissão desta operação diretamente à UDDI. O resultado da operação é apresentado na tela.

Para ilustrar a busca tomamos como base um cliente tentando localizar um determinado tipo de elemento de IDS a partir do SRP, conforme ilustrado Figura 5.3. Este cliente pode ser um operador em uma console de gerenciamento, um elemento gerenciador ou um motor de orquestração criando uma composição.

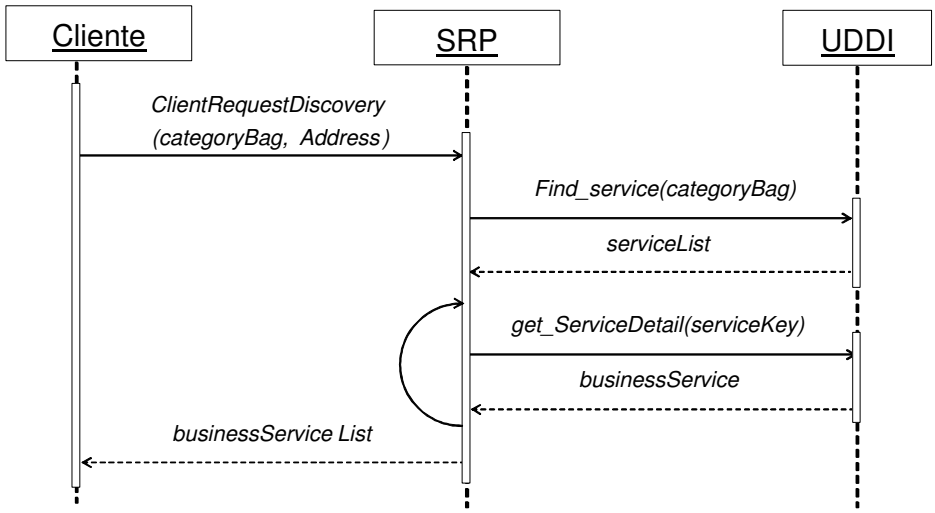

Figura 5.3: Diagrama de seqüência de uma pesquisa por componentes

Nesta pesquisa, o cliente envia ao SRP uma operação *ClientRequestDiscovery* contendo uma estrutura *categoryBag* e o parâmetro *Address*. O SRP envia à UDDI uma solicitação de operação *find_service*, incluindo a estrutura *categoryBag*.

Como resultado desta operação de pesquisa, é retornada uma estrutura *serviceList* com a descrição e as chaves de identificação dos serviços (*serviceKey*), que disponibilizam os elementos de IDSs e empresas (*businessKey*) que os oferecem.

De posse da *serviceKey*, a operação *get_serviceDetail* é enviada à UDDI, que retorna a estrutura *businessService* contendo as estruturas *bindingTemplates* necessárias para acessar o serviço do elemento de IDS desejado, que incluem seu endereço de rede. A operação *get_serviceDetail* é repetida até que sejam obtidas as informações de todos os elementos encontrados.

Após filtrar os elementos que combinem com todos os parâmetros da busca, o SRP retorna ao cliente uma lista dos elementos encontrados, contendo as estruturas *businessService* completas dos mesmos.

As composições também podem ser localizadas na UDDI. Para isso buscamos pela identificação da composição (nome ou chave) ou por um *tModel* que represente a composição, a partir da categoria do mesmo. No caso de uma composição, usamos uma *categoryBag* com a chave *keyValue="uddi:ids:composition"*.

Busca Recursiva

A busca por elementos de detecção de intrusão é feita recursivamente a partir dos registros encontrados na UDDI primária do SRP. O mecanismo de busca recursiva segue o procedimento descrito no capítulo anterior.

No protótipo, configuramos no SRP uma lista de candidatos a servidores de busca. Estes servidores de busca podem ser tanto SRPs, quanto UDDIs. Se a UDDI primária não estiver disponível, os servidores de busca (UDDIs de consulta) cadastrados na lista são consultados. Se a primeira UDDI de consulta da lista não responder a uma solicitação de busca, os servidores seguintes são consultados seqüencialmente até que um deles responda à solicitação ou a lista se esgote. Caso nenhum servidor da lista responda, uma mensagem de erro é reportada e o processo de composição é interrompido.

A Figura 5.4 ilustra uma busca recursiva com dois ciclos de consulta, incluindo três servidores de registro. Na UDDI "A", configurada como primária, foram registradas as UDDIs "B" e "C". Ao iniciar a busca na UDDI "A", o SRP obteve os registros das demais UDDIs e também as usou consulta. Ao término da consulta, o SRP retorna os serviços encontrados nas três UDDIs que satisfaziam o critério de pesquisa.

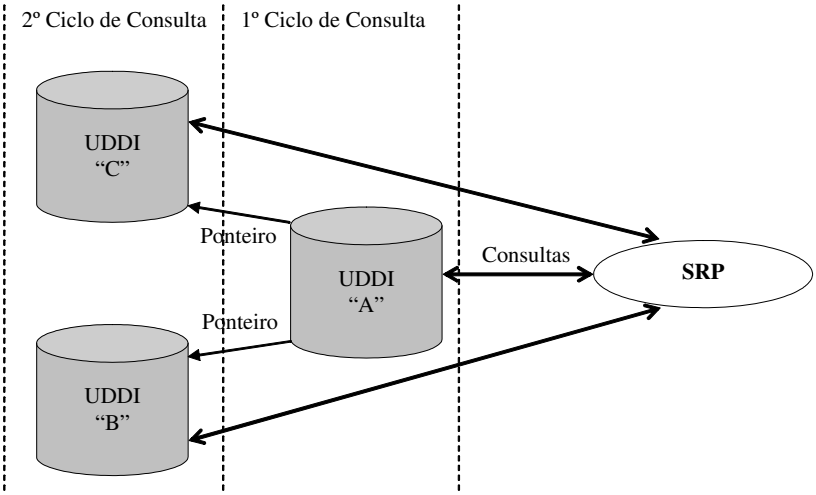

Figura 5.4: Topologia de busca recursiva com dois ciclos de consulta

A Figura 5.5 exemplifica uma busca com três ciclos de consulta a três servidores de registro. A UDDI "A" continha o registro da UDDI "B" e, esta última, o registro da UDDI "C".

Para evitar que a busca por elementos de detecção de intrusão se estenda demasiadamente, a quantidade de UDDIs pesquisadas pelo protótipo do SRP é limitada por quatro parâmetros: *MaxServices*, *MaxHops*, *MaxServers* e *MaxSearchTime*. O parâmetro *MaxServices* define a quantidade de elementos de IDS que podem ser encontrados. O parâmetro *MaxHops* limita a quantidade de ciclos de consulta que poderão ser executados. O parâmetro *MaxServers* limita a quantidade de UDDIs que podem ser pesquisadas durante uma busca. O parâmetro *MaxSearchTime* define o tempo máximo, em segundos, que pode durar uma busca. Caso o limite de um dos parâmetros seja atingido ou ocorra um erro, a busca é interrompida.

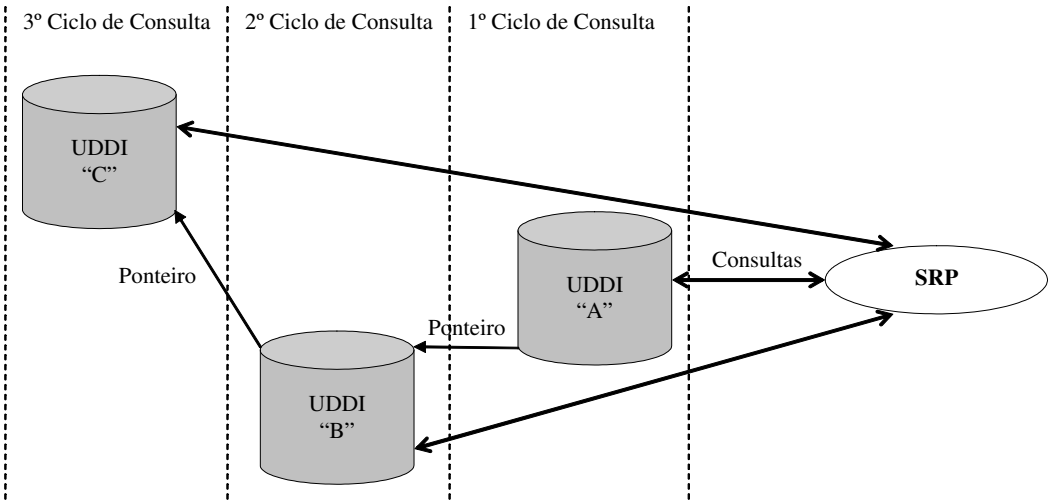

Figura 5.5: Topologia de busca recursiva com dois ciclos de consulta

5.2.2 Serviço de Segurança

O serviço de segurança foi implementado, basicamente, usando os mecanismos disponíveis na UDDI. Como o servidor UDDI possui mecanismos para autenticação de usuários por meio do uso de *tokens* (ver seção 5.4.2), usamos o mesmo mecanismo para autenticar os usuários.

A localização das chaves públicas, baseadas no modelo X.509, usadas nas interações entre os elementos, são indicadas no registro do elemento na UDDI, conforme ilustrado no exemplo da seção 3.7. Uma estrutura *bindingTemplate* indica a localização da chave. Opcionalmente, a própria chave também pode ser incluída na UDDI, usando o *container* da *tag instanceParms*, na mesma estrutura *bindingTemplate*.

O mecanismo, apesar de bastante simplificado, é suficiente para realizar a autenticação e o controle de acesso à UDDI e aos elementos de detecção de intrusão adotados nos testes.

Na política de segurança foram implementados dois tipos de papéis: Operador e Administrador. Os Operadores têm permissão para acessar os serviços e selecionar os elementos para composição. Os Administradores têm permissão privilegiada para efetuar registros na UDDI e configurar os elementos. Os Administradores de um domínio de segurança não têm permissão privilegiada sobre elementos de outro domínio.

As operações de registro e alteração de dados na UDDI são sempre precedidas de um passo de autenticação no servidor. Essa autenticação é própria da API da UDDI e consiste do envio de uma credencial por parte do cliente e o recebimento de um *token* que será usado a partir desse momento. A Figura 5.6 ilustra a mensagem encapsulada em SOAP, com a solicitação de um *token* para o usuário "**username**", repassando a senha "**password**". Na mesma figura está a mensagem de resposta, que retorna o *token* "**406682D9-128C-9897-6D22-C1EEC96E067B**".

Na UDDI, as consultas em geral não exigem autenticação dos usuários. Em nosso caso, isso é indesejado, pois um atacante poderia utilizar informações coletadas na UDDI para identificar, loca-

Requisição de Token:

```
<get_authToken xmlns="urn:uddi-org:api_v2" generic="2.0" userID="username" cred="password"/>
```

Resposta da UDDI:

```
<authToken generic="2.0" operator="test" xmlns="urn:uddi-org:api_v2">
  <authInfo>406682D9-128C-9897-6D22-C1EEC96E067B</authInfo>
</authToken>
```

Figura 5.6: Autenticação junto a UDDI

lizar e, conseqüentemente, burlar os sistemas de detecção de intrusão. A versão de UDDI usada no protótipo dispõe apenas de um mecanismo simples de autenticação com registro manual de usuários. Porém, uma versão comercial de UDDI, a BEA *AquaLogic Service Registry*[4], baseada na versão 3.02 da especificação, permite o uso de diversos sistemas de autenticação disponíveis no mercado, como o *Kerberos*[5], facilitando sua integração com os demais sistemas implementados na rede do usuário. Testes com uma cópia de demonstração desta UDDI comercial mostraram que tal integração é possível, inclusive com chaves no padrão X.509.

Em nosso protótipo, as mensagens são encapsuladas em SOAP e transportadas utilizando o protocolo HTTP. Como as implementações UDDI não suportam diretamente o *XML-Encryption*, por enquanto, estamos utilizando o SSL para garantir a confidencialidade dos dados. A estrutura opcional *dsig:Signature* pode ser utilizada na UDDI como assinatura do serviço, prevenindo que registros adulterados, que apontem para serviços maliciosos, sejam incluídos na UDDI. Porém, tal opção não chegou a ser implementada nos testes.

5.2.3 Compatibilizador de Formatos

Conforme descrito no capítulo anterior, o Compatibilizador de Formatos é dividido em dois módulos. O Adaptador de Elemento interage diretamente com o elemento de detecção de intrusão, enquanto a Interface de Elemento fornece os serviços na forma de *Web Services*.

A Interface de Elemento disponibiliza as interfaces de serviço e se comunica com o Adaptador de Elementos por troca de mensagens. Se os módulos estiverem dispostos em *hosts* distintos de uma mesma rede, a comunicação entre eles é feita usando SSL.

A Interface de elemento oferece interfaces de composição e configuração, as quais são descritas em um documento WSDL. As interfaces de composição são padrão para todos os tipos de elementos de detecção de intrusão. Porém, há uma diferenciação entre as interfaces nos elementos que agem como clientes e as interfaces naqueles que são fornecedores de serviços.

[4] http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/products/aqualogic/service_registry/
[5] http://web.mit.edu/Kerberos/

Interfaces dos Fornecedores de Serviços

Os fornecedores de serviço tornam disponíveis interfaces compatíveis com a especificação WSN (capítulo 2, seção 2.4.6), cujas funcionalidades são implementadas diretamente no protótipo:

- ***Subscribe***: solicita a inscrição para receber determinados tópicos de eventos;
- ***GetCurrentMessage***: solicita que seja encaminhada a última mensagem produzida sobre determinado tópico;
- ***PauseSubscription***: solicita uma pausa no envio de mensagens;
- ***ResumeSubscription***: solicita o reinício da transmissão de mensagens após uma pausa (*PauseSubscription*);
- ***Destroy***: solicita a interrupção definitiva do envio de mensagens.

Interfaces de Consumidores de Serviços

Os consumidores de serviço também devem possuir pelo menos uma interface compatível com a especificação WSN. A interface ***Notify*** recebe os eventos enviados por um elemento fornecedor de serviços.

Além dessa interface, também é necessária uma interface de ativação do elemento na composição. Através dessa interface, denominada ***RequestNotify***, o elemento cliente é requerido para iniciar a recepção de alertas de segurança provenientes de outro elemento. Nessa operação, é enviada a identificação do provedor de serviços, que é validada junto ao Serviço de Segurança. Essa operação é necessária para evitar ataques de negação de serviço e de mascaramento. Como resposta a esta mensagem, o cliente pode aceitar (*accept*) ou se negar (*deny*) a participar da composição recebendo mensagens do fornecedor indicado. Outra Operação, a ***CancelSendAlert***, emite uma ordem cancelando a recepção de mensagens de um ou mais gerenciadores.

Interfaces de Configuração

As interfaces de configuração são usadas pelos administradores de sistema para configurar e atualizar os elementos de detecção de intrusão. Essas interfaces são inspiradas na especificação WSDM, na habilidade de configuração (capítulo 4 seção 4.4). Como as operações dessa habilidade são específicas para cada produto, não há uma padronização das mesmas. Como esta especificação é omissa, foram criadas as seguintes operações de configuração para o modelo de composição de IDSs:

- ***SystemUpgrade***: substitui o código executável do sistema de detecção de intrusão, mas não reinicializa o IDS;
- ***GetConfig***: solicita uma cópia do arquivo de configuração do IDS;
- ***ConfigUpdate***: substitui o arquivo de configuração do IDS, mas não reinicializa o IDS;

- ***StopSystem***: interrompe o funcionamento do IDS, cancelando todas as interações ativas como cliente ou servidor;
- ***RestartSystem***: reinicializa o IDS, mantendo todas as interações ativas.

Como o Adaptador de Elementos é independente da Interface de Elementos, o módulo que disponibiliza o serviço via *Web* continua funcionando quando a operação *RestartSystem* é executada. Porém, se houver erro na execução dessa operação, as interações são canceladas. A partir deste erro, somente as operações de configuração poderão ser executadas no IDS, até que o elemento volte a funcionar normalmente.

5.2.4 Sensores

Para o protótipo, foram adaptados os IDSs *Snort* e *Prelude-ids* para funcionarem como sensores. Ambos os IDSs tiveram seu código fonte adaptado para produzir mensagens no formato IDMEF padrão e atualizado. Isso foi necessário, pois o *plugin* IDMEF do *Snort* estava desatualizado e o *Prelude-ids* não usa o formato padrão XML para formatação das mensagens.

O Compatibilizador de Formatos, que disponibiliza os sensores na forma de *Web Service*, implementa as interfaces de fornecedores de serviço, para enviar os alertas aos elementos analisadores e gerenciadores.

5.2.5 Analisadores

O *Prelude-ids* tem a capacidade de receber alertas de diversas fontes, correlacioná-los e enviar as informações a outro destino. Por isso, também usamos o *Prelude-ids* para fazer o papel de analisador. Nesse caso, foi preciso fazer uma adaptação para que o mesmo pudesse receber e enviar alertas no formato XML padrão do IDMEF.

Para receber as informações, o Compatibilizador de Formatos que disponibiliza o analisador na forma de *Web Service* implementa as interfaces de consumidores de serviços e de fornecedores de serviço. Como consumidor de serviços, ele recebe dados de sensores ou outros analisadores. Como fornecedor de serviços, ele envia dados a outros analisadores e a gerenciadores.

5.2.6 Gerenciador

Para executar as tarefas de gerenciamento da composição, desenvolvemos um elemento gerenciador bastante simplificado. O elemento foi implementado como um *Web Service* e também oferece uma interface HTML para interagir com o responsável pela segurança. A Figura 5.7 apresenta a interface web do protótipo de gerenciador, com destaque à opção na qual são apresentadas as notificações recebidas (Visualizar Notificações).

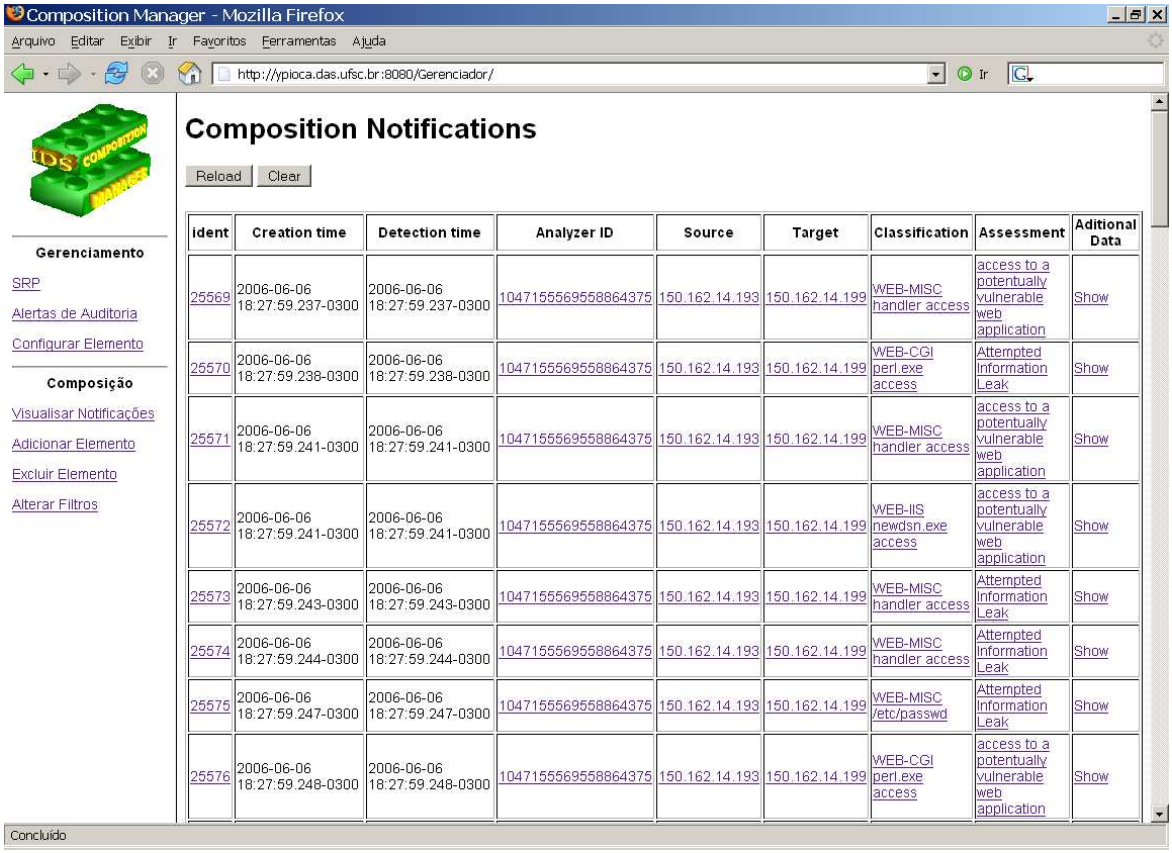

Figura 5.7: Tela do elemento gerenciador "*IDS Composition Manager*"

O protótipo do gerenciador também permite: interagir com o Serviço de Registro e Pesquisa; visualizar alertas de auditoria vindos de elementos da composição; configurar os elementos que estão em seu domínio administrativo; adicionar novos elementos à composição; excluir elementos; e alterar os filtros de notificações. Elementos fora do domínio administrativo podem ser incluídos nas composições, mas não podem ter sua configuração alterada por gerenciadores de outros domínios.

5.2.7 Orquestração de IDSs

Para a orquestração de IDSs foram selecionadas ferramentas para sua implementação. Tais ferramentas incluem uma linguagem, o servidor para executar a orquestração (motor de orquestração) e os recursos necessários para criar e editar a mesma.

Também foi desenvolvido um procedimento geral para a criação e a manutenção dinâmica de composições de IDSs. Esse procedimento foi implementado e executado por um motor de orquestração.

Ferramentas de orquestração

Existem atualmente diversas linguagens e ferramentas que auxiliam na orquestração de *Web Services* [Peltz, 2003][Wang et al., 2004]. Neste trabalho foi adotado o BPEL4WS (*Business Process Execution Language for Web Services*) [OASIS, 2005], também chamado de BPEL. Essa escolha

deve-se, principalmente, à disponibilidade de ótimas ferramentas no mercado e à vasta documentação disponível.

O BPEL4WS visa a composição de *Web Services* em um ambiente distribuído, principalmente entre múltiplas organizações. Sua especificação provê uma gramática baseada em XML para descrever a lógica de controle necessária para coordenar uma composição. Esta gramática pode ser interpretada e executada por motores de orquestração que são controlados por uma das partes envolvidas. Os blocos de construção dos processos BPEL4WS descrevem atividades executadas dentro de uma composição. Há atividades básicas e atividades estruturadas. As atividades básicas são instruções de interação entre os *Web Services*. As atividades estruturadas descrevem o fluxo de execução do processo.

Uma vez descrita (via *BEA Weblogic Workshop*), a composição pode ser implementada a partir de serviços *web* ou exportada para o formato BPEL, junto com a descrição WSDL. Estes documentos são registrados e representam, respectivamente, as interfaces da composição e o fluxo de execução da mesma. A partir desses documentos é possível recuperar (construir) a composição de IDSs e sua correspondente orquestração.

Procedimento geral para composição de IDSs

A Tabela 5.1 descreve os passos necessários para a criação de composições de IDSs genéricas utilizando a infra-estrutura desenvolvida. Nesse procedimento, é criada uma orquestração que possibilita a inclusão dinâmica de elementos de detecção de intrusão a partir das interações da orquestração com um de seus elementos, o gerenciador, conforme descrito no capítulo anterior.

Tabela 5.1: Passos para a criação de composições de IDSs

Passo	**Operação**
1	Localizar na UDDI o Serviço Gerenciador, de acordo com os parâmetros de busca especificados utilizando a operação *ClientRequestDiscovery*
2	Executar a operação *ClientRequestCreate* para criar um registro da composição na UDDI.
3	Inicializar o Serviço Gerenciador (*perform).*
4	Associar o Serviço Gerenciador à composição, usando a operação *ClientRequestRegistry.*
5	Localizar na UDDI o Serviço Analisador, de acordo com os parâmetros de busca especificados utilizando a operação *ClientRequestDiscovery*.
6	Executar a operação *ClientRequestSubscribe* no Serviço Analisador para subscrição dos alertas, enviando-os ao Serviço Gerenciador.
7	Associar o Serviço Analisador à composição, usando a operação *ClientRequestRegistry.*
8	Repetir os passos 5, 6 e 7 para cada um dos demais elementos envolvidos na composição (Sensores 1, 2 e 3).

Os passos descritos na tabela serão detalhados a seguir.

5.2.8 Implementação da Orquestração

A partir dos passos pré-definidos, foi modelada e implementada uma orquestração usando a interface gráfica do sistema *BEA Weblogic Workshop*. A Figura 5.8 mostra as operações básicas de uma orquestração, definidas por este procedimento.

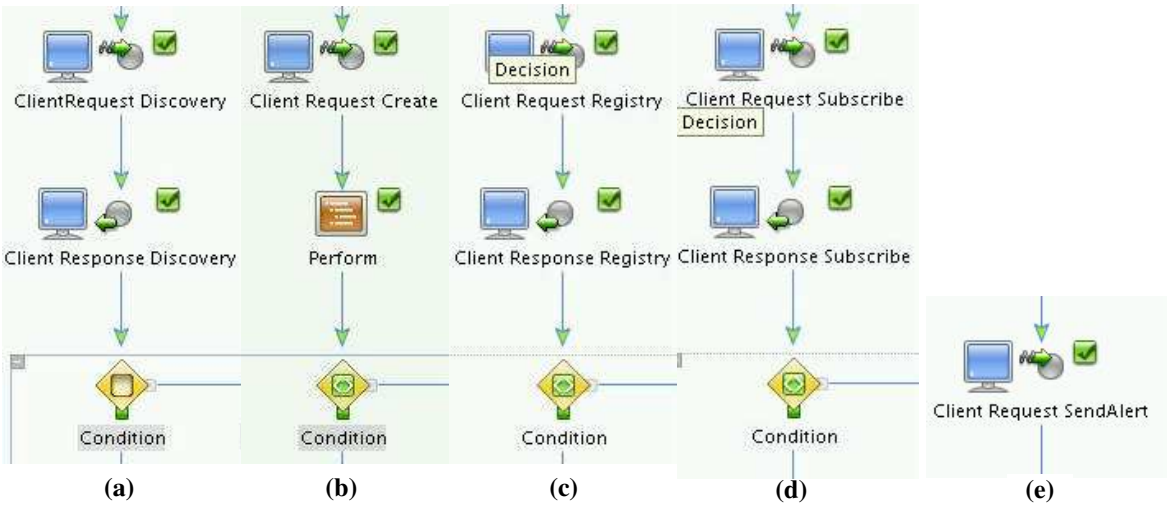

Figura 5.8: Modelo geral de composição

A princípio é conhecida apenas a funcionalidade desejada dos elementos e não a sua localização. A Operação *ClientRequestDiscovery* (Figura 5.8a) é responsável por executar a busca por serviços na UDDI utilizando o Serviço de Registro e Pesquisa, de acordo com parâmetros estabelecidos na operação. Nos testes realizados no protótipo, foram usados dois parâmetros: as características do elemento de detecção de intrusão descritas na estrutura *categoryBag*; e o endereço da rede (ou sub-rede) em que o elemento deve estar operando (esta última contida na estrutura da classe *Analyzer* do formato IDMEF).

A operação *ClientRequestCreate* (Figura 5.8b) é responsável pela requisição para a criação do registro da composição. A operação *Perform* faz a verificação dos dados e a inserção do registro da composição na UDDI. As operações *ClientRequestRegistry* e *ClientResponseRegistry* (Figura 5.8c) são responsáveis respectivamente pela requisição e resposta da inclusão de um novo serviço à composição.

As operações *ClientRequestSubscribe* e *ClientResponseSubscribe* (Figura 5.8d), por sua vez, fazem a subscrição de serviços. Elas são baseadas, respectivamente, nas operações *SubscribeRequest* e *SubscribeResponse* das interfaces padrões da especificação WSN (*Web Service Notification*) [Graham et al., 2006]. Por último, a operação *ClientRequestSendAlert* (Figura 5.8), é usada para configurar um cliente para o recebimento de mensagens.

A orquestração de composições segue uma seqüência lógica e cronológica dos eventos representados na Figura 5.8. A partir de cada evento é possível continuar ou ir para o final do fluxo de execução (representado pelo *Condition*). Para executar a operação *ClientRequestSendAlert*, por exemplo, não é

necessário executar os eventos anteriores, desde que eles já tenham sido executados pelo menos uma vez.

5.3 Ambiente de Desenvolvimento e Testes

5.3.1 Linguagens de Programação

Os protótipos descritos neste capítulo foram desenvolvidos usando as linguagens de programação Java e C++. A linguagem Java foi usada para a implementação dos *Web Services* e do Compatibilizador de Formatos, mais precisamente a Interface de Elemento. Essa linguagem foi escolhida por sua portabilidade e pela disponibilidade de bibliotecas e ferramentas de desenvolvimento para a criação de *Web Services*. A Linguagem C++ foi usada para a implementação de adaptações nos IDSs *Snort* e *Prelude-ids*, adotados nos protótipos.

Para a interpretação e compilação dos programas na linguagem Java, foi usado o pacote JDK 5.0[6]. Para a compilação dos programas em C++ foi usado o compilador GCC 4.0.2.

Na formatação das mensagens na linguagem C++ foi adotada a biblioteca *LibIDMEF*[7]. Para a formatação das mensagens IDMEF na linguagem Java, foi usada a biblioteca *JavaIDMEFv0.93beta*[8].

Para a implementação da segurança das mensagens, foram usadas as bibliotecas Java *WSS4J*[9] e *XML Security*[10].

5.3.2 Sistemas Operacionais

Para o desenvolvimento dos protótipos foram usados os sistemas operacionais Linux *Ubuntu*[11], *Mandriva*[12] e *OpenBSD*[13], além do *Windows XP*[14]. Para uniformizar o ambiente de testes, os mesmos foram executados exclusivamente em máquinas com o sistema operacional *Ubuntu*.

5.3.3 Ferramentas de Desenvolvimento

Para o desenvolvimento dos protótipos em linguagem Java, optamos por usar Ambientes de Desenvolvimento Integrado (*Integrated Development Enviroment - IDE*). Duas ferramentas foram usadas: o *WebLogic Workshop*, da BEA[15] e o NetBeans IDE[16]. O Web *Logic Workshop* é uma ferramenta

[6] http://java.sun.com/javase/downloads/index.jsp
[7] http://sourceforge.net/projects/libidmef/
[8] http://sourceforge.net/projects/javaidmef/
[9] http://ws.apache.org/wss4j/
[10] http://xml.apache.org/security/
[11] http://www.ubuntu.com/
[12] http://www.mandriva.com/
[13] http://www.openbsd.org/
[14] http://www.microsoft.com/windowsxp/default.mspx
[15] http://www.bea.com
[16] http://www.netbeans.org/

comercial que disponibiliza uma versão gratuita para desenvolvimento. O *NetBeans* é uma ferramenta de código aberto, bastante usada para o desenvolvimento de aplicativos baseados na linguagem Java.

Ambas as ferramentas permitem a edição, a compilação e a distribuição dos programas gerados para a criação de *Web Services*. As versões da ferramenta *WebLogic Workshop* empregadas no protótipo (8.1 e 9.2) possuem ainda a opção de criação de orquestrações e a execução das mesmas na própria ferramenta, além de controlar o desenvolvimento das estruturas XML (DTD). Também há recursos para visualização da troca de mensagens SOAP, facilitando a depuração dos protótipos. Recentemente, o *NetBeans* também incluiu facilidades para o desenvolvimento integrado de Web Services e a criação de orquestrações. Porém, não chegamos a testar profundamente tais opções.

O *NetBeans* foi usado na criação e depuração dos programas em Java, principalmente no desenvolvimento do protótipo do serviço gerenciador. Já o*WebLogic Workshop* foi usado na criação de *Web Services* e na orquestração de serviços.

5.3.4 Ambiente Web

Para a disponibilização dos *Web Services* desenvolvidos nesta Tese, foram testados dois ambientes distintos: o servidor próprio *BEA Web Logic Workshop*; e o servidor *Apache Tomcat*[17]. Ambas opções suportam a execução de *servlets* Java que implementam os *Web Services*, porém há algumas diferenças e incompatibilidades. O *WebLogic Workshop* fornece um ambiente integrado que possibilita o desenvolvimento, a implantação (*deployment*) e o teste dos Web Services, além do suporte ao motor de orquestração.

O *Tomcat*, distintamente, exige um trabalho maior na sua instalação, pois cada ferramenta de apoio, como o interpretador SOAP, deve ser instalada em separado. Contudo, isso permite o uso de padrões incipientes que ainda não estão disponíveis no *WebLogic Workshop*. Exemplos disso são o padrão de gerenciamento WSDM e de notificação WSN. Esses padrões só estão disponíveis por meio de outras ferramentas da BEA, que não são gratuitas.

Portanto, a criação e a execução da orquestração é feita usando o *WebLogic Workshop*. Já os *Web Services* são disponibilizados usando o servidor *Tomcat* e seus agregados, como o *Axis*[18] (SOAP).

O *Muse*[19] (WSDM) e o *Pubscribe*[20] (WSN) foram avaliados. Porém, não chegaram a ser usados diretamente nas versões do protótipo que foram testadas. O *Pubscribe* apresentou alguns problemas de implementação. Além disso, optamos por reduzir ao máximo o *overhead* das mensagens, reduzindo a quantidade de encapsulamentos.

[17] http://tomcat.apache.org/
[18] http://ws.apache.org/axis/
[19] http://ws.apache.org/muse/
[20] http://ws.apache.org/pubscribe/

5.3.5 Equipamentos

Para a execução dos testes, foram usados cinco computadores HP 325 *Business Desktop*, com processador Athlon XP 2600, HD de 40Gbytes, 512Mbytes de memória DDR 333MHz e placa de rede de 10/100Mbps. No desenvolvimento também foram usados computadores Dell Optiplex GX620, com processador Intel Pentium 4 de 3GHz, com HD de 80Gbytes e 1Gbyte de memória DDR2 533MHz e placa de rede de 10/100/1000Mbps.

5.3.6 Ambiente dos Testes

Os testes nos protótipos foram realizados em um ambiente de produção do campus universitário, utilizando elementos de composição de IDSs localizados em redes distintas. O campus universitário possui milhares de computadores com dezenas de sistemas operacionais diferentes, distribuídos em diversas subredes interconectadas, com suas barreiras (como *firewalls*) e riscos de segurança. Portanto, a escolha do ambiente de testes reflete a heterogeneidade e a diversidade das redes de larga escala. No teste de desempenho (seção 5.7) foi usada uma rede isolada para que não houvesse interferência de outros tipos de tráfego.

5.4 Experimento 1: Teste Operacional

O objetivo desse experimento é verificar o uso da orquestração na composição de IDSs, além de testar o funcionamento dos protótipos dos elementos de detecção de intrusão desenvolvidos.

5.4.1 Cenário do Teste

Para a realização do teste, foi usado o cenário ilustrado na figura 5.9a. Em duas redes de computadores foram dispostos elementos de detecção de intrusão, devidamente registrados na UDDI.

Para avaliar o experimento, foram efetuadas simulações de ataque do tipo *port Scan*[21] utilizado o software *Nmap*[22], a partir de um host na rede "A" em direção à rede "B". Acompanhando estas tentativas de ataque, verificamos as trocas de mensagens entre os elementos e a possibilidade de identificação da origem do ataque.

Os elementos de detecção de intrusão foram compostos usando a orquestração básica apresentada na Tabela 5.1, a fim de criar a composição inicial ilustrada na figura 5.9b.

[21] Varredura de portas de um host em busca de vulnerabilidades de segurança.
[22] http://www.insecure.org

5.4.2 Resultados Esperados

Nesse teste espera-se como resultado: (1) criar uma composição a partir de uma orquestração básica; (2) obter alertas de segurança a partir dos elementos da composição e, se possível, identificar a origem de ataques; (3) alterar dinamicamente a topologia da composição inicial; (4) poder retornar à topologia inicial da composição.

5.4.3 Execução do Teste

A execução dos testes foi realizada em duas redes de computadores distintas, conectadas à rede da UFSC, separadas por sistema de *firewall*. O tráfego nas duas redes não foi alterado para a execução dos testes, a fim de verificar o funcionamento da mesma em um ambiente real. A realização dos testes descritos na Tabela 5.2 são ilustrados na Figura 5.9.

Tabela 5.2: Descrição dos testes

a	Foram registrados na UDDI diversos possíveis elementos para composição de IDSs
b	Uma composição foi criada inicialmente na rede "A" com dois IDSs monolíticos baseados no *Snort* (*SnorWS1* e *SnortWS2*) agindo como sensores nesta rede e enviando seus alertas a um analisador baseado no *Prelude-ids* (*PreludeWS1*). O Serviço Gerenciador (*ManagerWS1*) recebe e apresenta as notificações ao administrador.
c	Em um ataque simulado, o elemento *SnortWS1* (na rede "A") detecta uma possível tentativa de ataque procedente da rede "B".
d	A composição é alterada para incluir um novo sensor localizado na rede "B". O serviço *SnortWS3* é localizado e ativado para coletar o tráfego proveniente do *host* suspeito e enviar os eventos diretamente ao *ManagerWS1*.
e	Novas mensagens de alerta confirmam a tentativa de ataque a partir do *host* suspeito.
f	Após as medidas administrativas para contenção do ataque, a composição volta à sua configuração original.

Na simulação de ataque, um usuário no console de um *host* realizou os ataques. Na detecção e resposta a outros ataques, mais serviços poderiam ser ativados para localizar a origem e identificar possíveis danos causados, como por exemplo, por uma base de dados com eventos de auditoria para identificar com maior precisão o usuário que originou o ataque.

5.4.4 Conclusões Sobre o Experimento

Nesse experimento foi mostrado que é possível criar e manipular uma composição de IDSs usando a orquestração de serviços do experimento anterior. Sobre essa composição, foi realizada uma simulação de tentativa de ataque. Como resposta à detecção do ataque, um novo elemento foi adicionado à composição dinâmica, para auxiliar na coleta de evidências. Com esses resultados, atingimos as expectativas do teste.

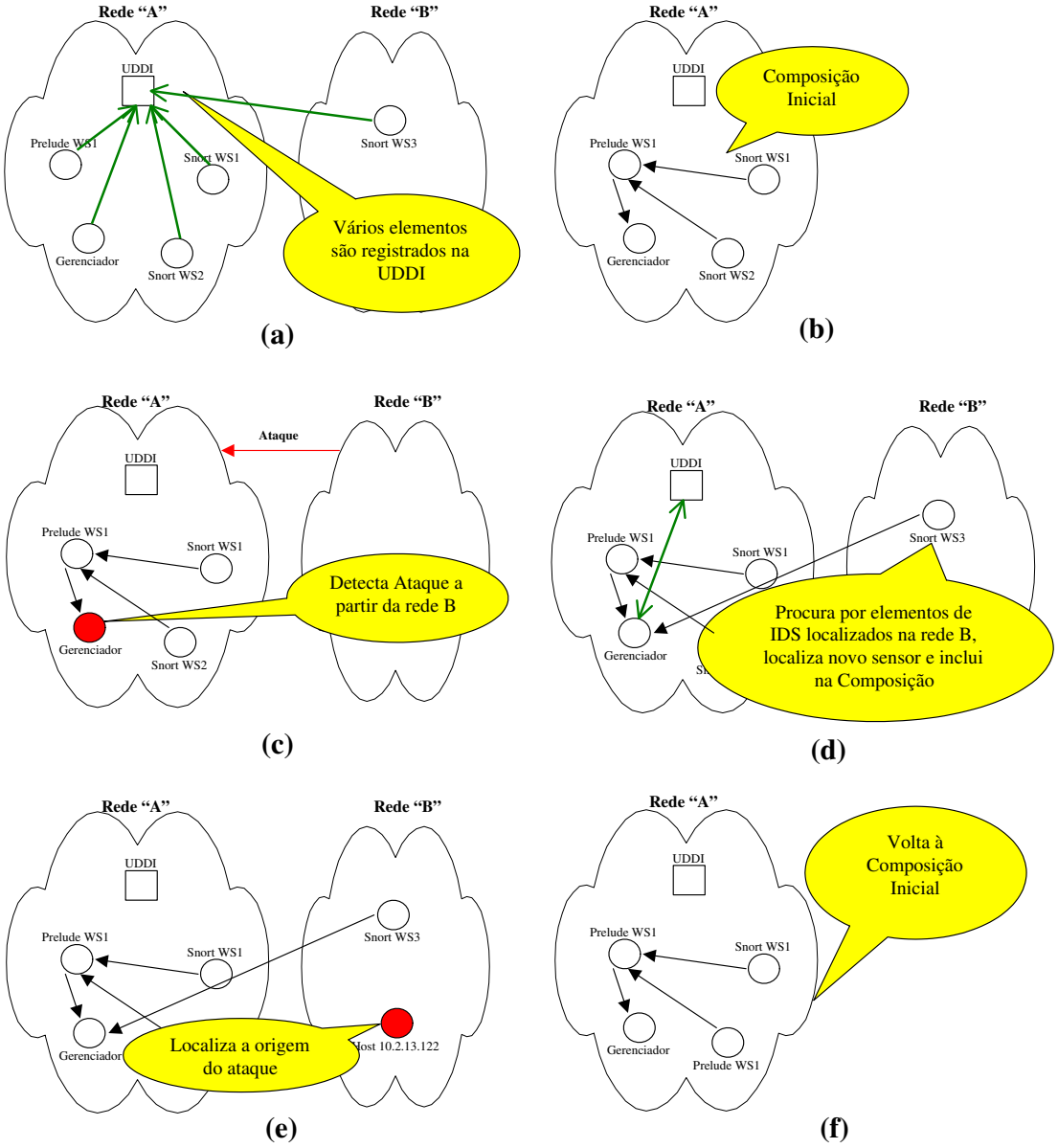

Figura 5.9: Etapas do teste operacional

5.5 Experimento 2: Desempenho e Custos em Ambiente Controlado

O uso dos novos padrões de comunicação baseados na troca de mensagens XML e a inclusão de mecanismos de segurança nas comunicações implicam novos custos de desempenho. Porém, ainda não existem medições que avaliem os custos na transmissão de mensagens de detecção de intrusão usando *Web Services* e nem como a inclusão de mecanismos de segurança influencia o desempenho dos IDSs. Portanto, o objetivo desse teste é verificar os custos de desempenho associados à adoção dos mecanismos de comunicação segura propostos nesta Tese.

5.5.1 Cenário do Teste

Para realizar os testes, foi montado um cenário de testes, conforme ilustrado na Figura 5.10. O ambiente para detecção de intrusão é composto por: um elemento sensor, baseado no IDS *Snort;* um elemento gerenciador, baseado no *Prelude-ids*, que recebe os alertas do sensor; e duas máquinas que simulam ataques simultâneos.

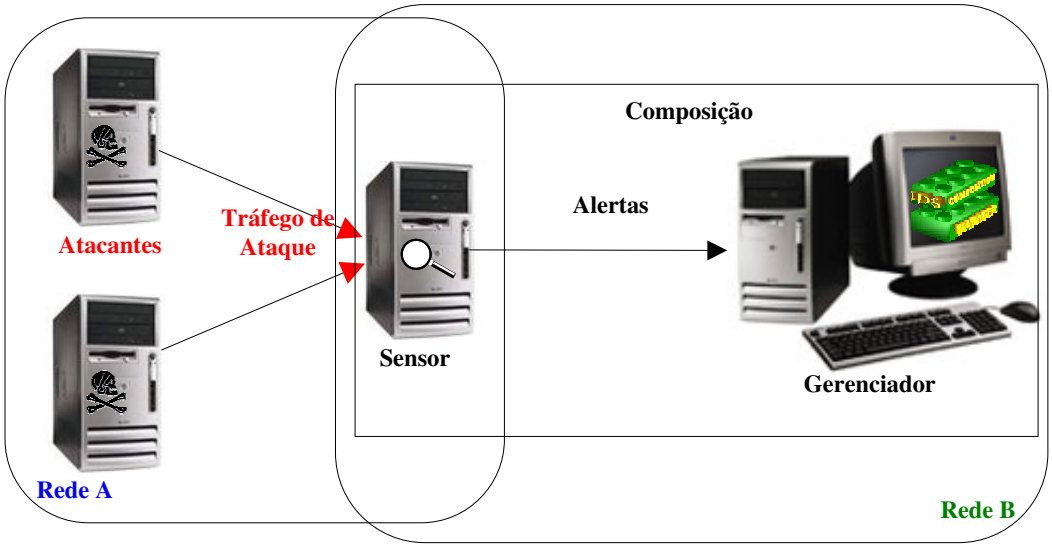

Figura 5.10: Topologia do ambiente de testes de custos

Os atacantes e o sensor foram conectados em uma rede local usando um *switch 3COM Baseline 2824 3C16474*, de 24 portas 10/100/1000Mbps e que possui uma capacidade de *switching* de 32 Gbps. O sensor foi configurado com duas interfaces de rede. A primeira interface está conectada à rede com os atacantes e a segunda interface está conectada à rede com o elemento gerenciador. Tal topologia teve como objetivo evitar ruídos durante as medições, pois permitiu isolar completamente a rede com o tráfego recebido pelos atacantes, da rede com o tráfego dos alertas gerados.

Para efetuar os ataques, utilizou-se a ferramenta *Idswakeup* [23], que utiliza a base de assinaturas do *Snort* para gerar ataques que serão identificados pelo IDS.

Para os testes foram contabilizados os custos para a transmissão de mensagens usando: (1) o formato nativo do Prelude-IDS em conexões SSL, com mensagens IDMEF sem XML; (2) o formato padrão IDMEF em XML, transportadas pelo protocolo HTTP protegido por SSL; e (3) o formato padrão IDMEF em XML, usando o protocolo HTTP e protegidas por *XML-Encryption* e *XML-Signature*. Os alertas encapsulados em mensagens SOAP usam algoritmo *tripledes-cbc* e chaves de 512 bits.

Para medir o desempenho, consideramos taxas diferentes de ataques por minuto: 5.000, 10.000, 15.000 e 30.000. Cada bateria de testes foi repetida 10 vezes e obtida a média dos resultados.

[23] http://www.hsc.fr/ressources/outils/idswakeup

5.5.2 Resultados Esperados

Neste experimento esperamos obter os seguintes resultados:

1. medir a capacidade de detecção de ataques e processamento dos mesmos;
2. medir o *overhead* de tráfego;
3. medir o retardo no processamento dos ataques; e
4. medir o retardo na transmissão das mensagens.

As medidas de desempenho obtidas com estes testes visam definir os limites operacionas dos protótipos desenvolvidos e da própria composição de IDSs.

5.5.3 Capacidade de Detecção

No primeiro teste, para medir a capacidade de detecção, foram efetuadas baterias de testes, aumentando a quantidade de tentativas de ataques até atingir um limite em que o sensor ou o gerenciador não seria mais capaz de realizar suas tarefas (negação de serviço). Foi obtido como limite a taxa de 30.890 tentativas de ataques por minuto, quando o sensor usando *Web Services* falha. Cabe ressaltar que a falha ocorre no Compatibilizador de Formatos e não no IDS, que continua funcionando. Tal falha ocorre devido ao estouro de memória na máquina virtual *Java*, configurada pelo Servidor *Tomcat*. Usando a configuração original de memória, 64Mbytes, o limite operacional foi próximo a 6.000 tentativas de ataque por minuto. Para não afetar o desempenho dos demais processos em execução no computador, optamos por limitar o uso de memória pela máquina virtual Java em 256 Mbytes, correspondendo a 50% da memória disponível no equipamento. Com mais ajuste na configuração de memória, seria possível obter resultados mais favoráveis.

5.5.4 Tamanho das Mensagens

O segundo teste consistiu em verificar o tamanho médio das mensagens de alerta geradas em cada um dos formatos e calcular o *overhead* de tráfego. Os resultados desse teste podem ser visualizados na Figura 5.11. Comparando o formato original com o formato XML usando SSL, constatamos que o *overhead* gerado é, na média, 2,8 vezes maior. Na comparação com o XML usando os mecanismos de proteção baseados na especificação *WS-Security*, o overhead sobe para 6,8 vezes o tempo necessário para enviar alertas no formato original com SSL.

5.5.5 Tempo de Processamento

No terceiro teste é medido o tempo médio de processamento (em milisegundos), necessário para detectar e formatar cada alerta durante as tentativas de ataque. Os resultados do teste podem ser

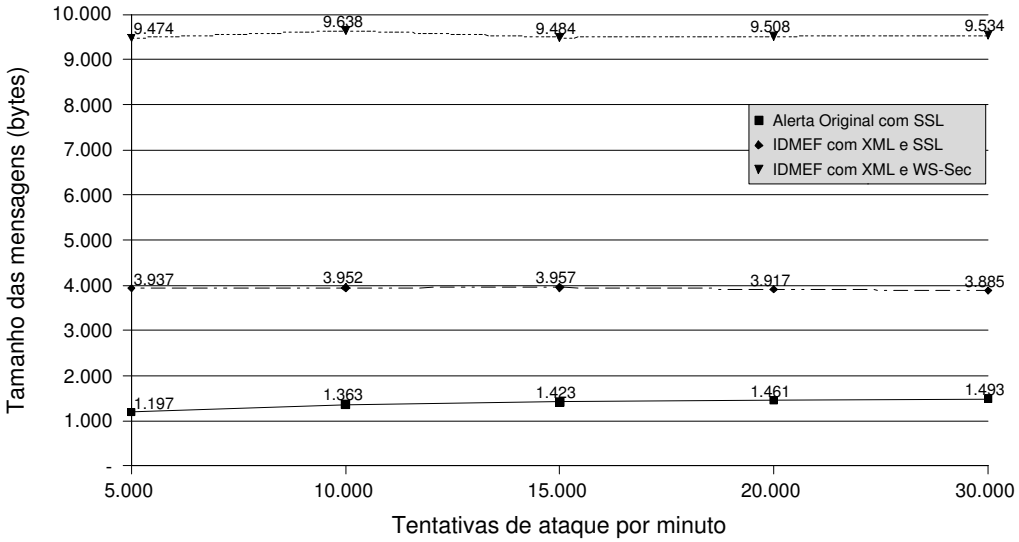

Figura 5.11: Tamanho médio das mensagens de alerta

observados na Figura 5.12. Analisando os dados, verificamos que o tempo necessário para processar cada mensagem para transmissão decresce com o tempo e a tendência é a de que a diferença de tempo não seja significativa. A melhor performance do nosso modelo de comunicação ocorre porque no *WS-Security* é necessário cifrar e assinar cada mensagem individualmente, enquanto que no SSL as mensagens são cifradas em bloco e enviadas em uma única sessão. Já com taxas superiores a 15.000 tentativas de ataque por minuto, há uma degradação do serviço web, devido às limitações do hardware.

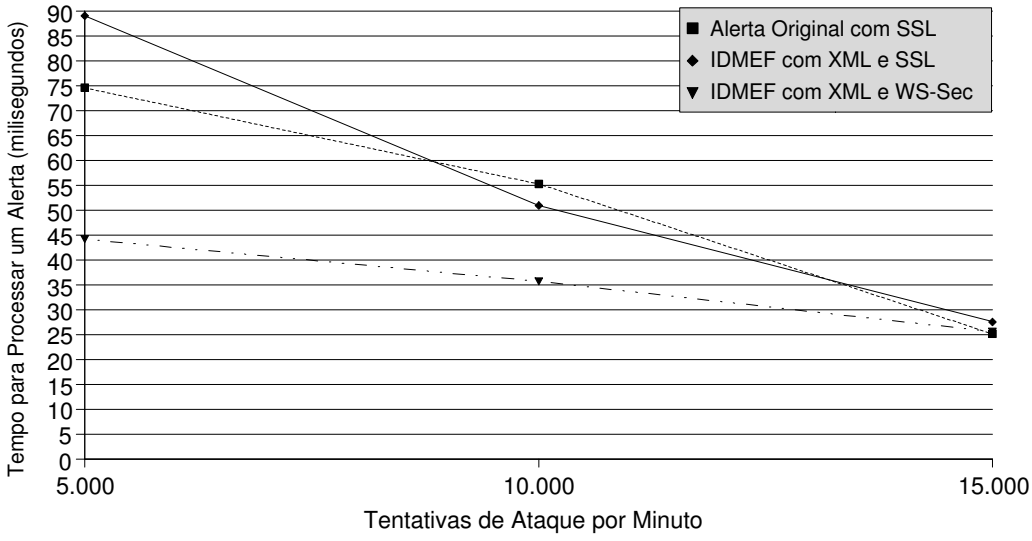

Figura 5.12: Tempo médio para processar um ataque

5.5.6 Tempo de Transmissão

No quarto teste foi verificado o tempo necessário para transmitir cada mensagem de alerta. Na Figura 5.13, observa-se que não há aumento no tempo de transmissão quando a quantidade de ataques é incrementada. Porém, o tempo gasto para a transmissão das mensagens IDMEF com *WS-Security* é maior do que nos outros casos. Isso é conseqüência do tamanho das mensagens, além do *overhead* de segurança e do próprio protocolo HTTP.

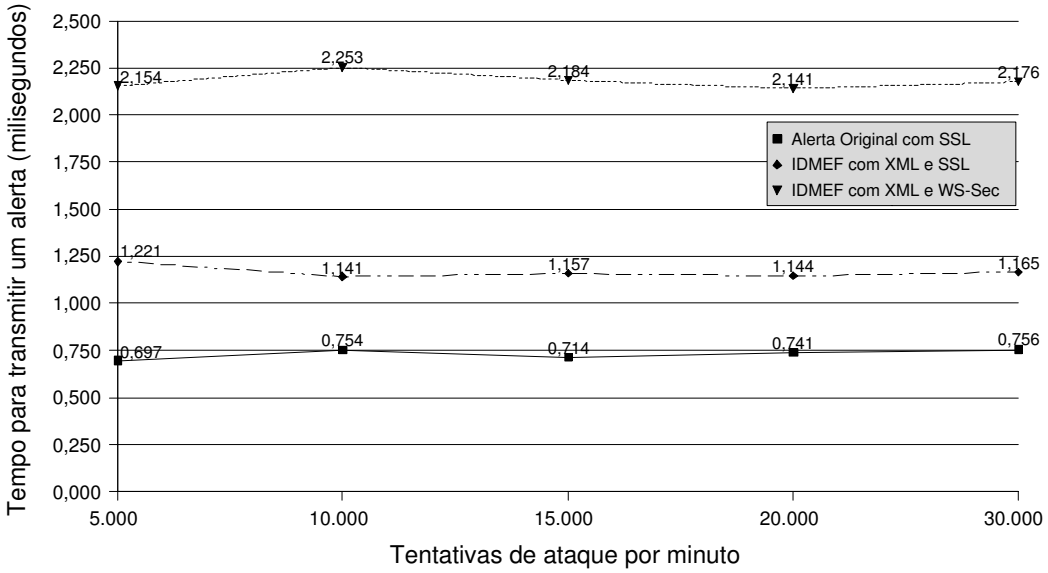

Figura 5.13: Tempo médio para transmitir um alerta

5.5.7 Conclusões Sobre o Experimento

Neste experimento foram realizadas medições de desempenho para avaliar o custo envolvido na adoção dos mecanismos de comunicação segura na composição de IDSs. A partir desses resultados, podem ser otimizados os processos de composição a fim de reduzir os custos e evitar falhas. Considerações sobre as alternativas de otimização são abordadas na próxima seção.

5.6 Experimento 3: Interações Entre Organizações Distintas

Esse experimento visa analisar a criação e o comportamento de uma composição de IDSs, envolvendo instituições distintas e usando a Internet.

5.6.1 Cenário do Teste

O cenário desse teste consiste em elementos de detecção de intrusão instalados nas redes do IPEA[24] e da UFSC. A configuração final da composição está ilustrada na Figura 5.14.

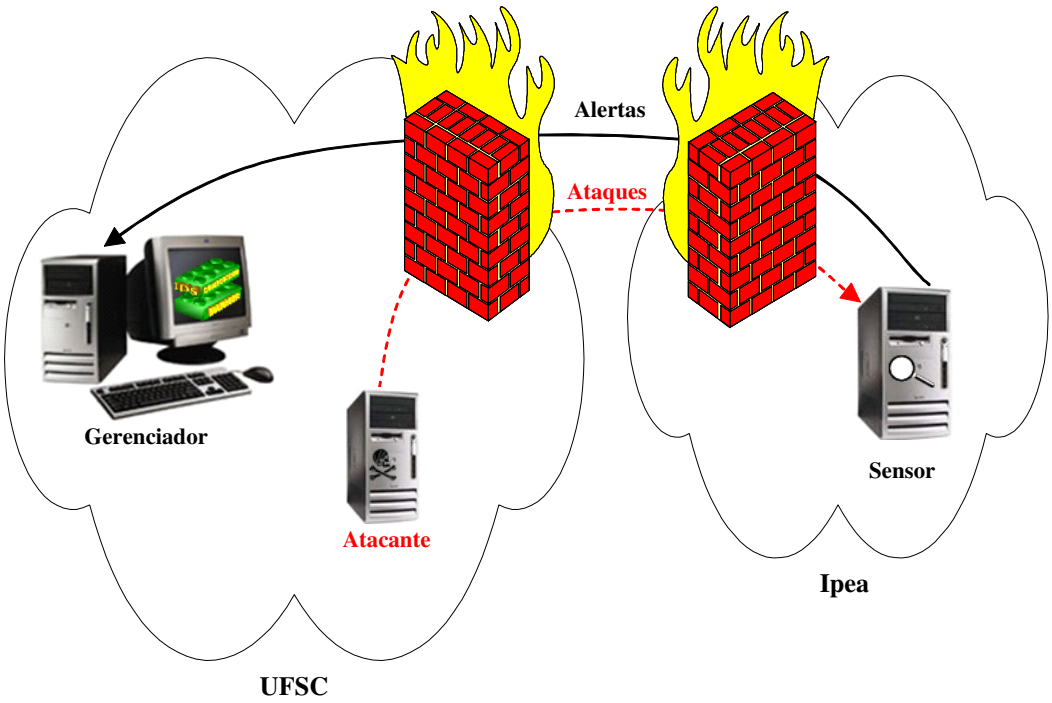

Figura 5.14: Composição entre organizações distintas

No experimento, a composição foi formada com dois elementos. O sensor localizado na rede do IPEA simula o comportamento de um *honeypot*, configurado para enviar alertas de ataques a um gerenciador localizado na rede da UFSC. O sensor foi configurado para filtrar os ataques originados somente na rede da UFSC. Em uma aplicação real, as informações coletadas poderiam ser usadas para barrar os ataques rapidamente na origem, reduzindo o tempo entre a detecção e a contenção de ataques.

Durante o processo de configuração constatamos a necessidade de padronização das portas utilizadas para facilitar a passagem dos dados através dos *firewalls*.

Apesar do fluxo de saída das redes usadas não sofrer filtragem, o fluxo de entrada, mesmo usando o protocolo HTTP na porta TCP 80, tende a ser filtrado. Nesses casos usamos o recurso de *proxy* para padronizar a entrada na rede e redirecionar o tráfego para os serviços dispostos na DMZ[25] ou internamente. Como em ambas as redes também há filtragem de tráfego entre a DMZ e a rede interna, foi necessário reconfigurar os *firewalls* para permitir a troca de mensagens entre o *proxy* e o servidor *Web* interno.

[24] Instituto de Pesquisa Econômica Aplicada - http://www.ipea.gov.br
[25] Zona desmilitarizada, na qual são colocados os serviços abertos à Internet

A Figura 5.15 mostra como foram configuradas as conexões. As conexões que saem das redes das organizações usam o protocolo HTTP padrão, com acesso através da porta TCP 80. Os pontos de acesso aos elementos de detecção de intrusão estão localizados nos *proxies*, onde as mensagens são redirecionadas para os servidores internos que, em nosso experimento, foram padronizados para operar na porta TCP 8080, a fim de diferenciar o tráfego do experimento para facilitar sua identificação.

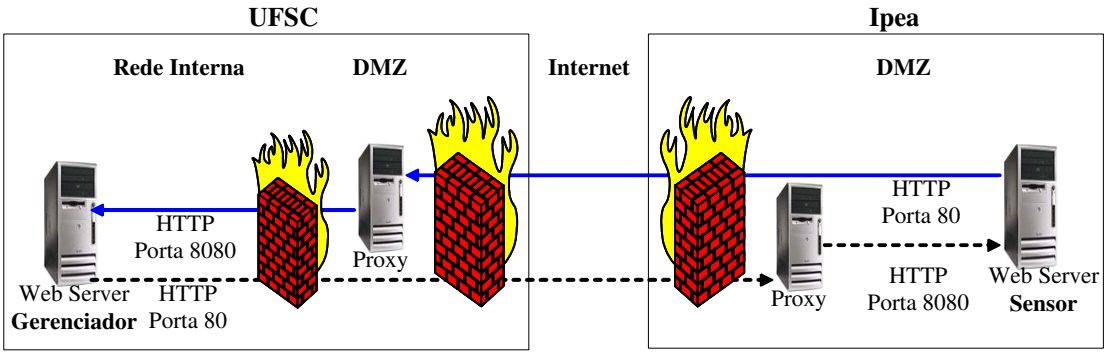

Figura 5.15: Configuração de protocolos para o tráfego de dados

Os testes foram realizados com as redes operando normalmente usando a Internet. As mensagens foram transmitidas usando o padrão IDMEF em XML e os mecanismos de proteção foram baseados na especificação *WS-Security*. Os alertas encapsulados em mensagens SOAP usam algoritmo tripledes-cbc e chaves de 512 bits. A bateria de testes foi repetida 10 vezes e obtida a média dos resultados.

5.6.2 Resultados Esperados

Esperamos com esse teste obter os seguintes resultados: (1) Criar uma composição de IDSs em redes conectadas através da Internet; (2) Complementar o teste anterior, a fim de identificar os custos associados à transmissão das mensagens pela Internet.

5.6.3 Resultados Obtidos

Durante o teste foram coletadas informações sobre a o tempo médio para transmissão das mensagens e a taxa de transmissão. O tempo médio obtido foi de 4,403 ms, já descontado o atraso da rede, que é, em média, de 51,126 ms. O resultado, apesar de corresponder ao dobro do tempo médio obtido nos testes de laboratório, está dentro do retardo esperado em uma conexão Internet.

A taxa de ataques obtida foi de 255 tentativas por minuto. Esse valor baixo reflete a limitação de banda da conexão do IPEA com a Internet (4 Mbps) e à concorrência com os demais fluxos de dados, que mantém o enlace com uma média de carga superior a 80%.

5.6.4 Conclusões Sobre o Experimento

O experimento mostrou que é possível o uso da Internet para a composição de IDSs. Mesmo com um tráfego alto no enlace com a Internet, o atraso experimentado na transmissão das mensagens ainda é baixo e não compromete o comportamento da composição.

5.7 Considerações Finais Sobre os Experimentos

Sobre o Serviço de Registro e Pesquisa, constatamos que as UDDIs avaliadas possuem recursos similares e que o desempenho não é um fator que possa influenciar na escolha do produto a ser usado.

A orquestração de serviços, infelizmente, ainda está amarrada à ferramenta de desenvolvimento. Mesmo usando a linguagem BPEL4WS, a compatibilidade não é total entre ferramentas distintas. O motor de orquestração é executado no ambiente integrado de desenvolvimento e a execução do arquivo em outros servidores não é trivial. Porém, esperamos que com a popularização da linguagem BPEL4WS, como já está ocorrendo, a interoperabilidade deixe de ser um problema.

Entre os experimentos realizados, o que mede os custos de segurança merece aqui uma análise mais aprofundada. Nesse experimento, identificamos alguns cuidados que devem ser tomados na aplicação das propostas desta Tese em futuras implementações.

Verificamos no protótipo um limite operacional para o compatibilizador de formatos. Tal limite está relacionado à execução de programas na linguagem Java, que pode provocar falhas no serviço quando o número de ataques simultâneos é extremamente alto[26].

Para reduzir a possibilidade de falhas, deve ser implementado um mecanismo de proteção a fim de controlar o fluxo de dados e evitar um estouro de memória. A especificação WSN já fornece em suas interfaces parâmetros para configurar o fluxo de dados, bastando implementá-la no *Web Service*. Como o tempo de transmissão das mensagens não varia com o número de ataques, é possível calcular a quantidade máxima de mensagens que podem ser transmitidas por um único elemento e assim definir os parâmetros para o controle de fluxo. No caso do protótipo implementado, o limite máximo seria de aproximadamente 27.500 mensagens por minuto.

Outro mecanismo de controle de falhas é o uso de mensagens *Heartbeat* do protocolo IDMEF, a fim de verificar se o elemento continua operando. No caso de falha, é solicitada a reativação do elemento.

Com relação aos custos de transmissão, verificamos que o tamanho das mensagens cresce bastante com o uso dos padrões de segurança para XML. Porém, o tamanho da mensagem não reflete de forma proporcional o tempo necessário para processar os alertas. O tempo de processamento de mensagens XML com *WS-Security* é menor do que os demais formatos, quando a taxa de tentativas de ataque é inferior a 15.000. Após este limite, o desempenho do *Compatibilizador de Formatos* começa a ser afetado.

[26] Uma taxa de 30.000 tentativas ataques por minuto é considerada alta para ser tratada por um único sensor.

Por limitações de prazo não foram feitas otimizações no código dos protótipos. Porém, é possível que a performance dos elementos possa ser melhorada em novas versões desses protótipos.

Para otimizar os custos de comunicação nas composições de IDSs, sugerimos que nas composições implementadas exclusivamente em redes locais, nas quais é possível a conexão direta entre os elementos, seja adotado o protocolo SSL ao invés do uso dos padrões associados ao *WS-Security*. Contudo, para garantir a interoperabilidade entre os elementos dispostos em organizações ligadas via Internet, deve ser mantida a mensagem no padrão IDMEF com XML.

5.8 Conclusões do Capítulo

Neste capítulo foram apresentados os protótipos desenvolvidos para a implementação do modelo de composição de IDSs e os experimentos usando tais protótipos. Foram desenvolvidos protótipos para implementar a infra-estrutura de serviços proposta e para implementar e adaptar alguns elementos de detecção de intrusão.

Nos experimentos realizados, foram testados os protótipos desenvolvidos e os custos envolvidos na comunicação segura entre os elementos de IDS. Também foi realizado um teste prático da aplicação do modelo de composição de IDSs baseado em *Web Services*.

Os protótipos e experimentos desenvolvidos mostraram a viabilidade do modelo de composição de sistemas de detecção de intrusão proposto nesta Tese.

Capítulo 6

Conclusões

6.1 Resumo da Tese

Neste trabalho foi apresentado um estudo para viabilizar o monitoramento de segurança em ambientes de larga escala, os quais extrapolam o limite físico tradicional das organizações e de suas redes de computadores, envolvendo também organizações parceiras com os mesmos objetivos.

Para permitir a detecção de intrusão nessa concepção organizacional é apresentada uma nova abordagem para a composição dinâmica de sistemas de detecção de intrusão. Essa nova abordagem permite a integração de elementos de IDSs ou mesmo de sistemas monolíticos para criar sistemas de detecção de intrusão distribuídos, em ambientes de larga escala. Para conseguir flexibilidade e interoperabilidade nessas composições dinâmicas, é proposta uma infra-estrutura baseada em serviços, que deve servir de suporte a essas composições. Essa infra-estrutura está fortemente fundamentada na tecnologia de *Web Services* e em padrões para comunicação de alertas de segurança.

A infra-estrutura é apresentada nesse texto como uma estratificação de serviços. Esses serviços contribuem nos vários níveis para viabilizar as composições de IDSs e tornar as interações entre seus elementos interoperáveis e seguras.

Nessa infra-estrutura são necessários novos métodos de gerenciamento. Como solução, a composição dinâmica de IDSs é obtida por meio da orquestração de serviços, a qual permite que composições possam ser iniciadas e reconfiguradas dinamicamente, quando necessário, e assim suportando as mudanças características dos ambientes de larga escala.

A interoperabilidade entre os atuais IDSs é fornecida por uma camada, que chamamos de Compatibilizador de Formatos, a qual oferece os elementos de detecção de intrusão na forma de *Web Services* e é responsável pela conversão das mensagens originais dos IDSs para formatos padronizados e vice-versa.

Para validar as propostas apresentadas, foram implementados alguns protótipos e realizados experimentos usando tais protótipos. Também foi feita uma avaliação da segurança nas composições de IDSs e foram propostas soluções para os problemas identificados.

Acreditamos que as propostas apresentadas nesta Tese possam contribuir para o desenvolvimento da segurança em ambientes de larga escala. Portanto, esperamos melhorar a segurança das organizações por meio do uso de sistemas de detecção de intrusão que sejam escaláveis, redundantes e dispersos geograficamente.

A seguir, nesse capítulo, é apresentada uma revisão dos objetivos iniciais da Tese, as principais contribuições e algumas possibilidades de trabalhos futuros.

6.2 Revisão dos Objetivos

Nessa seção são reapresentados os objetivos definidos no primeiro capítulo deste documento, junto com as soluções encontradas para atingir tais objetivos.

1. **Definir e desenvolver uma infra-estrutura de integração que permita a composição de sistemas de detecção de intrusão distintos e dispersos geograficamente.**

   A infra-estrutura de integração foi definida tomando como base os seguintes componentes: o Serviço de Registro e Pesquisa, o Serviço de Segurança, o Compatibilizador de Formatos e os mecanismos de gerenciamento.

   O Serviço de Registro e Pesquisa é responsável pelas funções de registro e busca de elementos de detecção de intrusão, conforme descrito na seção 4.5. O Serviço de Segurança é responsável pelos mecanismos de autenticação e controle de acesso entre os elementos envolvidos e entre os mesmos e a infra-estrutura de serviços, conforme explicado na seção 4.8. O compatibilizador de Formatos é responsável pela disponibilização dos elementos de IDSs na forma de *Web Services* e da comunicação entre estes elementos, de acordo com o que é mostrado na seção 4.6. Os mecanismos de gerenciamento são responsáveis pela configuração dos elementos de detecção de intrusão e pela coordenação da própria composição, conforme proposto na seção 4.7.

2. **Propor mecanismos de comunicação padronizados que permitam a interoperabilidade entre IDSs distintos.**

   Para viabilizar o uso de elementos de detecção de intrusão existentes foi desenvolvido um Compatibilizador de Formatos, que faz a ponte entre tais elementos e a nova abordagem de composição usando *Web Services*. O Compatibilizador de Formatos também executa a tarefa de padronização dos formatos de comunicação dos alertas de detecção de intrusão, conforme descrito na seção 4.6. No protótipo desenvolvido foi adotado o formato IDMEF do IETF.

3. **Agilizar a composição dinâmica de novos sistemas de detecção de intrusão, a partir do suporte da infra-estrutura proposta.**

   Os modelos tradicionais de gerenciamento de *Web Services*, como o WSDM, são eficazes para a manutenção individual de um serviço. Porém, não oferecem recursos suficientes para implementar as composições dinâmicas de IDSs, conforme discutido no capítulo 4. Por isso, nesta

Tese, a composição dinâmica de IDSs é obtida com o uso da orquestração de serviços. A orquestração permite ao administrador definir um fluxo de processo genérico que possa ser usado para formar uma composição. Tal processo permanece inalterado quando elementos de IDS são adicionados ou removidos. Para descrever as orquestrações foi desenvolvido e testado um protótipo que faz uso da linguagem BPEL4WS, conforme apresentado na seção 5.2.7.

4. **Adotar especificações baseadas nos esforços de padronização da IETF, W3C e OASIS a fim de atingir a interoperabilidade necessária.**

   Para garantir a interoperabilidade entre os elementos da composição, foram adotados padrões e especificações de grandes organizações, mais precisamente o IETF, W3C e OASIS. Do IETF foi usado o formato de mensagens IDMEF a fim de padronizar a troca de mensagens de detecção de intrusão. Também foram adotados padrões de segurança associados ao XML desenvolvidos em conjunto com o W3C. Do W3C e do OASIS foram obtidos os padrões relacionados aos *Web Services*, como a própria arquitetura WSA e as linguagens WSDL e BPEL4WS.

   Cabe ressaltar que a escolha e a integração dos diversos padrões não foi uma tarefa trivial. Algumas especificações que seriam, aparentemente, complementares acabam sendo incompatíveis. Outra dificuldade está relacionada ao uso de esforços de padronização incipientes que são alterados constantemente. As ferramentas de desenvolvimento que adotam os padrões também fazem pequenas "melhorias", que dificultam o uso dos aplicativos desenvolvidos em ambientes diferentes. Apesar de tudo, foi possível encontrar um meio termo que permitiu integrar especificações incipientes com modelos e padrões reconhecidos.

5. **Fazer uso extensivo da linguagem XML e da tecnologia *Web Services* de programação distribuída orientada a serviços.**

   Todas as especificações de padrões adotadas nesta Tese são baseadas na linguagem XML. A infra-estrutura proposta para a viabilização das composições de IDSs é baseada em especificações, serviços e ferramentas que dão suporte à tecnologia de *Web Services*. Os elementos de uma composição de IDSs são vistos como serviços e disponibilizados por meio de *Web Services*.

6.3 Contribuições

O estudo desenvolvido nesta Tese resultou em contribuições inéditas na área de detecção de intrusão, apresentadas a seguir:

1. A identificação e avaliação dos principais trabalhos científicos para a detecção de intrusão distribuída e, principalmente, para ambientes de larga escala foi realizada extensivamente. Nessa avaliação são apresentados os principais problemas e desafios relacionados ao foco desta Tese, os quais são tratados pelas propostas aqui apresentadas. Como destaque deste esforço investigativo, está a adoção de uma taxonomia que permite a classificação de elementos de detecção de intrusão para busca e seleção nas composições de IDSs.

2. É proposta uma nova abordagem, que chamamos de composições de IDSs, para possibilitar a detecção de intrusão em ambientes de larga escala. Essa abordagem tem o potencial de fomentar uma nova gama de pesquisas na área de detecção de intrusão.

3. O uso de elementos de detecção de intrusão já existentes no mercado é viabilizado com a inclusão de uma camada de serviço entre os elementos IDSs atuais e a composição de IDSs. O Compatibilizador de Formatos implementa essa camada, fornecendo aos sistemas atuais as funcionalidades necessárias para suprir os requisitos de integração das composições de IDSs.

4. A adoção da orquestração de *Web Services* para a criação e gerenciamento de composições de IDSs é um trabalho inédito na área de segurança. Até então as aplicações desse mecanismo de gerenciamento estavam focadas na criação de aplicações de negócios, enquanto o monitoramento de segurança seguia o modelo tradicional de gestão.

5. A adoção da norma AS/NZ4360 e a adaptação do método CVSS para a identificação dos níveis de risco associados a áreas de pesquisa e aplicações distintas é um diferencial desta Tese. Apesar de conhecido na área de segurança, a metodologia de gestão de riscos raramente é empregada na avaliação de propostas científicas.

Para divulgar as contribuições aqui apresentadas, os resultados obtidos durante o trabalho foram publicados e apresentados em eventos científicos nacionais e internacionais conhecidos, que possuem comitê de programa e corpo de revisores. Os artigos relacionados ao trabalho estão publicados em [Brandão et al., 2005] [Brandão et al., 2006a] [Brandão et al., 2006b] [Brandão et al., 2006c]. Além desses trabalhos, um estudo preliminar sobre detecção de intrusões usando agentes móveis foi publicado em [Brandão e Fraga, 2004].

6.4 Perspectivas Futuras

Acreditamos que a composição de IDSs para detecção de intrusão em ambientes de larga escala possa fomentar novas oportunidades de estudo na área de monitoramento de segurança. Entre esses novos estudos podemos identificar os seguintes:

1. A criação de novas técnicas de correlação e análise de dados provenientes de ambientes altamente distintos;

2. A extensão das composições para incluir elementos, também baseados em *Web Services*, que possam estar associados a centros de resposta a incidentes de segurança, os quais auxiliem na identificação e descrição de ataques;

3. O uso da orquestração de serviços para modelar métodos de detecção e resposta a incidentes, que serão implementados em composições dinâmicas de IDSs;

4. A associação da infra-estrutura de serviços da composição às grades computacionais a fim de prover redundância e uma maior flexibilidade aos serviços;

5. A experimentação da composição de IDSs em redes de sensores do tipo “*Honey Nets*”.

6. A criação de modelos dinâmicos de confiança que tornem o compartilhamento de elementos de IDS mais ágil e seguro;

7. Melhorias na infra-estrutura de serviços a fim de torná-la mais eficiente.

Referências Bibliográficas

Alessandri, D., Cachin, C., Dacier, M., Deak, O., Julisch, K., Randell, B., Riordan, J., Tscharner, A., Wespi, A., e Wüest, C. "Towards a Taxonomy of Intrusion Detection Systems and Attacks". MAFTIA Deliverable D3, EU Project IST-1999-11583 Malicious- and Accidental-Fault Tolerance for Internet Applications (MAFTIA), sep 2001. URL `http://domino.watson.ibm.com/library/cyberdig.nsf/1e4115aea78b6e7c85256b360066f0d4/5fa980eb952e09d085256ac600535997?OpenDocument`. Version 1.01.

Anderson, J. P. "Computer Security Technology Planning Study". Technical Report ESD-TR-73-51, USAF Electronic Systems Division, Hanscom Air Force Base, Bedford, Massachusetts, October 1972. URL `http://seclab.cs.ucdavis.edu/projects/history/CD/ande72b.pdf`.

Anderson, J. P. "Computer Security Threat Monitoring and Surveillance". Technical Report Contract 79F26400, James P. Anderson Co., Box 42, Fort Washington, PA, 19034, USA, 26 February revised 15 April 1980. URL `http://seclab.cs.ucdavis.edu/projects/history/CD/ande80.pdf`.

Arvidsson, J., Cormack, A., Demchenko, Y., e Meijer, J. "TERENA'S Incident Object Description and Exchange Format Requirements". RFC 3067, Internet Engineering Task Force, February 2001.

AS/NZS. *Risk Management - AS/NZS4360:2004*. Australian/New Zealand Standard, third edition, August 2004a. ISBN 0-7337-5904-1.

AS/NZS. *Risk Management Guidelines Companion to AS/NZS 4360:2004 - HB 436:2004*. Australian/New Zealand Standard, third edition, August 2004b. ISBN 0-7337-5960-2.

Athanasiades, N., Abler, R., Levine, J., Owen, H., e Riley, G. "Intrusion Detection Testing and Benchmarking Methodologies". In *IEEE-IWIA '03: Proceedings of the First IEEE International Workshop on Information Assurance (IWIA'03)*, page 63, Washington, DC, USA, 2003. IEEE Computer Society. ISBN 0-7695-1886-9.

Austin, D., Babir, A., Peters, E., e Ross-Talbot, S. "Web Services Choreography Requirements". W3c working draft 11, W3C, March 11 2004.

Axelsson, S. "Intrusion-Detection Systems: A Taxonomy and Survey". Technical Report 99–15, Department of Computer Engineering, Chalmers University of Technology, SE–412 96, Göteborg, Sweden, March 2000.

Bace, R. e Mell, P. “NIST special publication on Intrusion Detection System.”. Technical report, NIST (National Institute of Standardsand Technology), August 2001. URL http://csrc.nist.gov/publications/nistpubs/800-31/sp800-31.pdf. Special Publication 800-31.

Bace, R. G. *Intrusion detection*. Macmillan Publishing Co., Inc., Indianapolis, IN, USA, 2000. ISBN 1-57870-185-6.

Balasubramaniyan, J. S., Garcia-Fernandez, J. O., Isacoff, D., Spafford, E., e Zamboni, D. “An Architecture for Intrusion Detection Using Autonomous Agents”. In *ACSAC ’98: Proceedings of the 14th Annual Computer Security Applications Conference*, Washington, DC, USA, 1998. IEEE Computer Society. ISBN 0818687894. URL http://portal.acm.org/citation.cfm?id=784589.784631.

Barker, W. C. e Lee, A. “Information Security - Volume II: Appendices to Guide for Mapping Types of Information and Information Systems to Security Categories”. NIST Special Publication 800-60, June 2004.

Bass, T. “The Federation of Critical Infrastructure Information via Publish-Subscribe Enabled Multisensor Data Fusion”. In *the Fifth International Conference on Information Fusion: Fusion 2002*, pages 1076–1083, Annapolis, MD, Jul. 2002. ISBN 0-9721844-1-4.

Bass, T. “Service-Oriented Horizontal Fusion in Distributed Coordination-Based Systems”. In *IEEE MILCOM 2004*, volume 2, pages 615– 621, Monterey, CA, USA, Nov. 2004. ISBN 0-7803-8847-X. URL http://www.valuerocket.com/papers/published.html.

Beale, J. *Snort 2.1 Intrusion Detection, Second Edition*. Syngress Publishing, 2004. ISBN 1931836043.

Bishop, M. *Computer Security: Art and Science*. Addison Wesley, Bonston, MA, 2003. ISBN 0-2014-4099-7.

Brandão, J. E. M. S. e Fraga, J. S. “Abordagens e Emprego de Agentes Móveis em Detecção de Intrusão”. In *6o Simpósio Segurança em Informática (SSI’2004*, São José dos Campos, SP, Nov. 2004. ISBN 8-5879-7805-5.

Brandão, J. E. M. S., Fraga, J. S., e Mafra, P. M. “Composição de IDSs usando web services”. In *Proceedings of the 2005 Simpósio Brasileiro em Segurança da Informação e de Sistemas (SBSeg)*, pages 339–342, Florianópolis, SC, September 2005. ISBN 85-7669-044-6.

Brandão, J. E. M. S., Fraga, J. S., e Mafra, P. M. “A New Approach for IDS Composition”. In *Proceedings of ICC 2006: IEEE International Conference on Communications*, Istanbul, Turkey, June 2006a. ISBN 1-4244-0355-3.

Brandão, J. E. M. S., Fraga, J. S., e Mafra, P. M. “Criação e Gerenciamento de Composições de IDSs”. In *Proceedings of the 2006 Simpósio Brasileiro em Segurança da Informação e de Sistemas (SBSeg)*, Santos, SP, September 2006b. ISBN 85-7669-075-6.

Brandão, J. E. M. S., Fraga, J. S., Mafra, P. M., e Obelheiro, R. R. "A WS-Based Infrastructure for Integrating Intrusion Detection Systems in Large-Scale Environments". In *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE*, volume 4275 of *Lecture Notes in Computer Science*, pages 462–479, Montpellier, France, November 2006c. Springer Berlin / Heidelberg. ISBN 9-7835-4048-2.

Bray, T., Paoli, J., e Sperberg-McQueen, C. M. "Extensible Markup Language (XML) 1.0 (Third Edition)"". W3C Recommendation, February 2004. URL http://www.w3.org/TR/2004/REC-xml-20040204.

Brownlee, N. e Guttman, E. "Expectations for Computer Security Incident Response". RFC 2350, Internet Engineering Task Force, June 1998.

Cachin, C., Camenisch, J., Deswarte, Y., Dobson, J., Horne, D., Kursawe, K., Laprie, J.-C., Lebraud, J.-C., Long, D., McCutcheon, T., Muller, J., Petzold, F., Pfitzmann, B., Powell, D., Randell, B., Schunter, M., Shoup, V., Veríssimo, P., Trouessin, G., Stroud, R. J., Waidner, M., e Welch, I. S. "MAFTIA: Reference Model and Use Cases". DI/FCUL TR 00–5, Department of Informatics, University of Lisbon, August 2000. URL http://www.di.fc.ul.pt/tech-reports/00-5.pdf.

Charfi, A. e Mezini, M. "Using Aspects for Security Engineering of Web Service Compositions". In *ICWS '05: Proceedings of the IEEE International Conference on Web Services (ICWS'05)*, pages 59–66, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2409-5.

Choon, O. T. e Samsudin, A. "Grid-based intrusion detection system". In *The 9th Asia-Pacific Conference on Communications (APCC 2003)*, volume 3, pages 1028–1032, Penang, Malaysia, Sep. 2003. ISBN 0-7803-8114-9.

Dacier, M. "Design of an Intrusion-Tolerant Intrusion Detection System. Maftia Project, deliverable 10". Technical report, IBM Zurich Research Laboratory, August 2002.

Danyliw, R., Meijer, J., e Demchenko, Y. "The Incident Object Description Exchange Format Data Model and XML Implementation". Internet Draft draft-inch-ietf-iodef-08.txt, IETF, August 2006. URL http://www.ietf.org/internet-drafts/draft-ietf-inch-iodef-08.txt.

Debar, H., Curry, D., e Feinstein, B. "The Intrusion Detection Message Exchange Format". Internet Draft draft-ietf-idwg-idmef-xml-16, IETF, Mar. 2006. URL http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-16.txt.

Debar, H., Dacier, M., Wespi, A., e Lampart, S. "An Experimentation Workbench For Intrusion Detection Systems". Technical report, IBM, IBM Research, Zurich Research Laboratory, 1998.

Debar, H., Dacier, M., e Wespi, A. "Towards a taxonomy of intrusion detection systems". *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(8):805–822, April 1999. ISSN 1389-1286. URL http://www.elsevier.com/cas/tree/store/comnet/sub/1999/31/8/2122.pdf.

Debar, H., Dacier, M., e Wespi, A. "A Revised Taxonomy for Intrusion Detection Systems". *Annales des Telecommunications*, 55(7–8):361–378, July-August 2000.

Demchenko, Y., Gommans, L., de Laat, C., e Oudenaarde, B. "Web Services and Grid Security Vulnerabilities and Threats Analysis and Model". In *Proceedings of the "6th IEEE/ACM International Workshop on Grid Computing*, pages 262–267, Seattle, Washington, USA, 2005. IEEE Cat. No. 05EX1210C. ISBN 0-7803-9493-3.

Denning, D. E. "An Intrusion-Detection Model.". In *IEEE Symposium on Security and Privacy*, pages 118–133, 1986.

Denning, D. E. "An Intrusion-Detection Model". *IEEE Transactions on Software Engineering*, Vol. SE-13(No. 2):222–232, February 1987.

Dierks, T. e Rescorla, E. "The Transport Layer Security (TLS) Protocol". IETF RFC 4346, April 2006. http://tools.ietf.org/html/rfc4346.

Durst, R., Champion, T., Witten, B., Miller, E., e Spagnuolo, L. "Testing and evaluating computer intrusion detection systems". *Commun. ACM*, 42(7):53–61, July 1999. ISSN 0001-0782. URL http://portal.acm.org/citation.cfm?id=306571.

Eastlake, D., Reagle, J., e Solo, D. "(Extensible Markup Language) XML-Signature Syntax and Processing". Request for Comments 3275, Internet Engineering Task Force, March 2002. URL http://www.rfc-editor.org/rfc/rfc3275.txt.

Esfandiari, B. e Tosic, V. "Requirements for Web Service Composition Management.". In *Proc. of the 11th Hewlett-Packard Open View University Association (HP-OVUA) Workshop*, Paris, France, 2004. Hewlett-Packard. URL http://www.hpovua.org/PUBLICATIONS/PROCEEDINGS/11_HPOVUAWS/hpov2004/www/ConferenceProgramme.htm.

Esfandiari, B. e Tosic, V. "Towards a Web Service Composition Management Framework.". In *ICWS*, pages 419–426. IEEE Computer Society, 2005. ISBN 0-7695-2409-5.

Fagundes, L. L. e Gaspary, L. P. "Quebrando a Barreira entre Mecanismos de Segurança através da Composição de Serviços Web: Uma Arquitetura para Detecção de Ataques Distribuídos e de Múltiplas Etapas". In *Proceedings of the 2006 Simpósio Brasileiro em Segurança da Informação e de Sistemas (SBSeg)*, Santos, SP, September 2006. ISBN 85-7669-075-6.

Feiertag, R., Kahn, C., Porras, P., Schnackenberg, D., Staniford-Chen, S., e Tung, B. "A Common Intrusion Specification Language (CISL)", June 1999. URL http://gost.isi.edu/cidf/drafts/language.txt.

Feiertag, R., Redmond, T., e Rho, S. "A Framework for Building Composable Replaceable Security Services". In *DARPA Information Survivability Conference & Exposition*, volume 2, pages 391–402, 2000a.

Feiertag, R., Rho, S., Benzinger, L., Wu, S., Redmond, T., Zhang, C., Levitt, K., Peticolas, D., Heckman, M., Staniford, S., e McAlerney, J. "Intrusion detection inter-component adaptive negotiation". *Comput. Networks*, 34(4):605–621, 2000b. ISSN 1389-1286.

Feinstein, B., Matthews, G., e White, J. "The Intrusion Detection Exchange Protocol (IDXP)". Internet Draft draft-ietf-idwg-beep-idxp-07, IETF, Oct. 2002. URL http://www.ietf.org/internet-drafts/draft-ietf-idwg-beep-idxp-07.txt.

Foukia, N., Hulaas, J., e Harms, J. "Intrusion Detection with Mobile Agents". In *proceedings of the 11th Annual Internet Society Conference (INET 2001)*, Stockholm, Sweden, Jun. 2001.

Fraga, J. e Powell, D. "A fault and intrusion-tolerant file system". In *In J. Grimson and H.-J. Kugler, editors, Proceedings of the Third IFIP International Conference on Computer Security (IFIP/Sec'85)*, pages 203–218, Dublin, Ireland, August 1985. ISBN 0-444-87801-7.

Frincke, D. "Balancing cooperation and risk in intrusion detection". *ACM Trans. Inf. Syst. Secur.*, 3 (1):1–29, 2000. ISSN 1094-9224.

Fuggetta, A., Picco, G. P., e Vigna, G. "Understanding Code Mobility". *IEEE Transactions on Software Engineering*, 24(5):342–361, 1998. ISSN 0098-5589.

Garfinkel, S., Spafford, G., e Schwartz, A. *Practical Unix and Internet security (3rd ed.)*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, third edition, 2003. ISBN 0-596-00323-4.

Gerhards, R. "The syslog Protocol". Internet Draft draft-ietf-syslog-protocol-17.txt, IETF, June 2006. URL http://www.ietf.org/internet-drafts/draft-ietf-syslog-protocol-17.txt.

Graham, S., Hull, D., e Murray, B. "Web Services Base Notification 1.3 (WS-BaseNotification)". OASIS Web Services Notification (WSN) TC, May 2006. URL http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-pr-03.pdf.

Han, J. e Zheng, Y. "Security Characterisation and integrity assurance for software components and component-based systems". In *International Conference on Software Methods and Tools*, pages 61–66. IEEE Computer Society Press, 2000.

Imamura, T., Dillaway, B., e Simon, E. "XML Encryption Syntax and Processing". W3c recommendation, W3C, December 2002. URL http://www.w3.org/TR/xmlenc-core/.

ISO. "Information Processing Systems - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture". ISO 7498-2, 1989.

ISO/IEC. "Common Criteria for Information Technology Security Evaluation - Part 1: Introduction and general model". ISO/IEC 15408:2005, August 2005a.

ISO/IEC. "Common Criteria for Information Technology Security Evaluation - Part 2: Security functional requirements". ISO/IEC 15408:2005, August 2005b.

ISO/IEC. "Common Criteria for Information Technology Security Evaluation - Part 3: Security assurance requirements". ISO/IEC 15408:2005, August 2005c.

ITU-T. "ITU-T recommendation X.509", 1993.

Janakiraman, R., Waldvogel, M., e Zhang, Q. "Indra: A peer-to-peer approach to network intrusion detection and prevention". *wetice*, 00:226, 2003. ISSN 1080-1383.

Jansen, W. e Karygiannis, T. “NIST Special Publication 800-19 - Mobile Agent Security”, 1999. URL citeseer.ist.psu.edu/jansen00nist.html.

Kahn, C., Porras, P., Staniford-Chen, S., e Tung, B. “A Common Intrusion Detection Framework”. Submitted to the Journal of Computer Security, available through: http://seclab.cs.ucdavis.edu/cidf/papers/jcs-draft/cidf-paper.ps, July 1998.

Karger, P. A. e Schell, R. R. “Multics Security Evaluation: Vulnerability Analysis - ESD-TR-74-193 Vol. II”, June 1974. URL http://csrc.nist.gov/publications/history/karg74.pdf.

Karger, P. A. e Schell, R. R. “MULTICS Security Evaluation: Vulnerability Analysis”. In *18th Annual Computer Security Applications Conference (ACSAC 2002)*, Las Vegas, NV, USA, December 9-13 2002. IEEE. URL http://domino.watson.ibm.com/library/cyberdig.nsf/papers?SearchView@Query=(multics).

Keeni, G., Danyliw, R., e Demchenko, Y. “Requirements for the Format for Incident Information Exchange (FINE)”. Internet Draft draft-ietf-inch-requirements-08.txt, IETF, June 2006. URL http://www.ietf.org/internet-drafts/draft-ietf-inch-requirements-08.txt.

Keeni, G. M. “Syslog Management Information Base”. Internet Draft draft-ietf-syslog-device-mib-08.txt, IETF, July 2006. URL http://www.ietf.org/internet-drafts/draft-ietf-syslog-device-mib-08.txt.

Kelsey, J., Callas, J., e Clemm, A. “Signed syslog Messages”. Internet Draft draft-ietf-syslog-sign-18.txt, IETF, May 2006. URL http://www.ietf.org/internet-drafts/draft-ietf-syslog-sign-18.txt.

Kent, S. e Seo, K. “Security Architecture for the Internet Protocol”. IETF RFC 4301, December 2005. http://tools.ietf.org/html/rfc4301.

Leu, F.-Y., Lin, J.-C., Li, M.-C., e Yang, C.-T. “A Performance-Based Grid Intrusion Detection System”. In *COMPSAC ’05: Proceedings of the 29th Annual International Computer Software and Applications Conference (COMPSAC’05) Volume 1*, pages 525–530, Washington, DC, USA, 2005a. IEEE Computer Society. ISBN 0-7695-2413-3.

Leu, F.-Y., Lin, J.-C., Li, M.-C., Yang, C.-T., e Shih, P.-C. “Integrating Grid with Intrusion Detection”. *aina*, 01:304–309, 2005b. ISSN 1550-445X.

Lindqvist, U. e Jonsson, E. “A Map of Security Risks Associated with Using COTS”. *Computer*, 31 (6):60–66, 1998. ISSN 0018-9162.

Lippmann, R., Haines, J. W., Fried, D. J., Korba, J., e Das, K. “The 1999 DARPA off-line intrusion detection evaluation”. *Comput. Networks*, 34(4):579–595, 2000a. ISSN 1389-1286.

Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., Weber, D., Webster, S. E., Wyschogrod, D., Cunningham, R. K., e Zissman, M. A. “Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation”. *discex*, 02:1012, 2000b.

Lonvick, C. “The BSD Syslog Protocol”. RFC 3164, Internet Engineering Task Force, August 2001. URL http://www.rfc-editor.org/rfc/rfc3164.txt.

McHugh, J. “Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory”. *ACM Trans. Inf. Syst. Secur.*, 3(4):262–294, November 2000. ISSN 1094-9224. URL http://portal.acm.org/citation.cfm?id=382923.

McHugh, J. “Intrusion and intrusion detection.”. *Int. J. Inf. Sec.*, 1(1):14–35, 2001.

Mell, P. e Grance, T. “Use of the Common Vulnerabilities and Exposures (CVE) Vulnerability Naming Scheme”. NIST Special Publication 800-51, September 2002.

Mell, P., Marks, D., e McLarnon, M. “A denial-of-service resistant intrusion detection architecture”. *Comput. Networks*, 34(4):641–658, 2000. ISSN 1389-1286.

Miao, F. e Yuzhi, M. “TLS Transport Mapping for SYSLOG”. Internet Draft draft-ietf-syslog-transport-tls-02.txt, IETF, June 2006. URL http://www.ietf.org/internet-drafts/draft-ietf-syslog-transport-tls-02.txt.

NCSC. “A Guide to Understanding Audit in Trusted Systems”. National Computer Security Center NCSC-TG-001, 1988.

NCSC. “Auditing Issues in Secure Database Management Systems”. National Computer Security Center Techinical Report - 005 Volume 4/5, 1996.

Neumann, P. G. e Parker, D. B. “A Summary of Computer Misuse Techniques”. In *Proceedings of the 12th National Computer Security Conference*, pages 396–407, Baltimore, Maryland, 10–13 October 1989.

Neumann, P. G. e Porras, P. A. “Experience with EMERALD to Date”. In *Proceedings of the Workshop on Intrusion Detection and Network Monitoring*, pages 73–80, Berkeley, CA, USA, 1999. USENIX Association. ISBN 1-880446-37-5.

OASIS. “UDDI Version 3.0.2”. OASIS UDDI Spec Technical Committee Draft, 10 2004a. URL http://uddi.org/pubs/uddi-v3.0.2-20041019.htm.

OASIS. “Web Services Distributed Management: Management Using Web Services (MUWS 1.0) Part 2 - Web Services Distributed Management: Management of Web Services (WSDM-MOWS) 1.0”. OASIS Web Services Distributed Management (WSDM) TC, December 2004b. URL http://www.oasis-open.org/committees/tc%95home.php?wg%95abbrev=wsdm.

OASIS. “Web Services Security: SOAP Message Security 1.0”, Março 2004c. http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf.

OASIS. “Business Process Execution Language for Web Services”, Sep 2005. URL http://www.oasis-open.org/committees/download.php/14616/wsbpel-specification-draft.htm. Version 2.0 - Committee Draft, 01 September 2005.

Ohta, K. “XML Schema definition for IDMEF message”. Internet Draft IETF Internet-Draft draft-kohei-idmef-schema-00, IETF, February 2004. URL http://xml.coverpages.org/draft-kohei-idmef-schema-00.txt.

Parastatidis, S. e Webber, J. “Assessing the Risk and Value of Adopting Emerging and Unstable Web Services Specifications”. In *SCC ’04: Proceedings of the 2004 IEEE International Conference on Services Computing*, pages 65–72, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2225-4.

Park, S., Kim, K., Jang, J., e Noh, B. “Supporting Interoperability to Heterogeneous IDS in Secure Networking Framework”. *The 9th Asia-Pacific Conference on Communications (APCC 2003)*, 2 (21-24):844–848, Sep. 2003.

Peltz, C. “Web Services Orchestration and Choreography”. *IEEE Computer*, 36(10):46–52, Oct 2003. URL http://jmvidal.cse.sc.edu/library/peltz03a.pdf.

Porras, P. A. e Neumann, P. G. “EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances”. In *Proc. 20th NIST-NCSC National Information Systems Security Conference*, pages 353–365, 1997. URL http://citeseer.ist.psu.edu/porras97emerald.html.

Porras, P., Schnackenberg, D., Staniford-Chen, S., Davis, Stillman, M., e Wu, F. “The Common Intrusion Detection Framework Architecture”, June 1999. URL http://gost.isi.edu/cidf/drafts/communication.txt.

Ptacek, T. H. e Newsham, T. N. “Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection”. Technical report, Secure Networks,Inc., January 1998. URL http://www.snort.org/docs/idspaper/.

Puketza, N. J., Zhang, K., Chung, M., Mukherjee, B., e Olsson, R. A. “A Methodology for Testing Intrusion Detection Systems”. *IEEE Trans. Softw. Eng.*, 22(10):719–729, 1996. ISSN 0098-5589.

Ramsdell, B. “Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification”. IETF RFC 3851, July 2004. http://www.ietf.org/rfc/rfc3851.txt.

Reagle, J. “XML Signature Requirements”. RFC 2807, IETF, July 2000.

Reagle, J. “XML Encryption Requirements”. Note 04, W3C, Mar. 2002.

Ross, R., Katzke, S., Johnson, A., Swanson, M., Stoneburner, G., Rogers, G., e Lee, A. “Recommended Security Controls for Federal Information Systems”. NIST Special Publication 800-53, February 2005.

Sandhu, R. e Samarati, P. “Authentication, access control, and audit”. *ACM Comput. Surv.*, 28(1): 241–243, 1996. ISSN 0360-0300.

Sherif, J. S. e Dearmond, T. G. “Intrusion Detection: Systems and Models”. *wetice*, 00:115, 2002. ISSN 1080-1383.

Smith, J. M. e Stutely, R. *SGML: the user's guide to ISO 8879*. Halsted Press, New York, NY, USA, 1988. ISBN 0-470-21126-1.

Spafford, E. H. e Zamboni, D. "Intrusion detection using autonomous agents". *Comput. Networks*, 34(4):547–570, October 2000. ISSN 1389-1286. URL http://dx.doi.org/10.1016/S1389-1286(00)00136-5.

Stallings, W. *Network Security Essentials: Applications and Standards*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2000. ISBN 0-1301-6093-8.

Staniford-Chen, S., Tung, B., e Schnackenberg, D. "The Common Intrusion Detection Framework (CIDF)". In *1998 Information Survivability Workshop (ISW '98)*, Orlando, FL, October 1998. IEEE Press. URL www.cert.org/research/isw/isw98/all_the_papers/no33.html.

Stoneburner, G., Goguen, A., e Feringa, A. "Risk Management Guide for Information Technology Systems". NIST Special Publication 800-30, July 2002.

Stryker, D. "Subversion of a "secure"operating system.". Technical report, Naval Research Laboratory, Washington, 1974. Memorandum Report 2821.

Swanson, M. e Guttman, B. "Generally Accepted Principles and Practices for Securing Information Technology Systems". NIST Special Publication 800-14, September 1996.

Teo, L., Zheng, Y., e Ahn, G.-J. "Intrusion Detection Force: An Infrastructure for Internet-Scale Intrusion Detection". In *First IEEE International Information Assurance Workshop (IWIA 2003)*, pages 73–88, Germany, March 2003.

Tolba, M., Abdel-Wahab, M., Taha, I., e Al-Shishtawy, A. "GIDA: Toward Enabling Grid Intrusion Detection Systems". *5th IEEE International Symposium on Cluster Computing and the Grid*, May 2005. URL http://dsg.port.ac.uk/events/conferences/ccgrid05/wip/schedule/.

Tung, B. "The Common Intrusion Specification Language: A Retrospective". *discex*, 02:1036, 2000.

US Department of Homeland Security. "Federal Information Security Management Act of 2002, H.R. 2458-48", 2002. URL http://www.cio.gov/archive/e_gov_act_2002.pdf. (Public Law 107-347).

Vambenepe, W., Graham, S., e Niblett, P. "Web Services Topics 1.3 (WS-Topics)". OASIS Web Services Notification (WSN) TC, July 2006. URL http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-cs-01.pdf.

Vandoorselaere, Y. "Prelude: an Open Source, Hybrid Intrusion Detection System", Aug. 2006. URL http://www.prelude-ids.org/article.php3?id_article=66.

Vigna, G. "Mobile Agents: Ten Reasons For Failure". *mdm*, 00:298, 2004.

Vigna, G., Valeur, F., e Kemmerer, R. A. "Designing and implementing a family of intrusion detection systems". In *Proceedings of the 9th European Software Engineering Conference*, pages 88–97, Helsinki, Finland, September 2003. ISBN 1-58113-743-5.

W3C. “SOAP Version 1.2”. W3C World Wide Web Consortium, June 2003. URL http://www.w3.org/TR/soap/.

W3C. “Web Services Architecture”. W3C Working Group Note 11, February 2004. URL http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/.

W3C. “Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language”. W3C Working Draft, August 2005. URL http://www.w3.org/TR/wsdl20/.

Wang, H., Huang, J. Z., Qu, Y., e Xie, J. “Web services: problems and future directions.”. *Journal of Web Semantics*, 1(3):309–320, 2004.

White, G. e Pooch, V. “Cooperating Security Managers: Distributed intrusion detection systems”. *Computers & Security*, Vol. 15(No. 5):441–450, 1996. Elsevier Science Ltd.

White, G. B., Fisch, E. A., e Pooch, U. W. “Cooperating security managers: a peer-based intrusion detection system”. *Network, IEEE*, 10(1):20–23, 1996. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=484228.

Wood, M. e Erlinger, M. “Intrusion Detection Message Exchange Requirements”. Internet Draft draft-ietf-idwg-requirements-10, IETF, Oct. 2002. URL http://www.ietf.org/internet-drafts/draft-ietf-idwg-requirements-10.txt.

Yegneswaran, V., Barford, P., e Jha, S. “Global Intrusion Detection in the DOMINO Overlay System”. In *NDSS*, San Diego, California, USA, Feb. 2004. The Internet Society. ISBN 1-891562-18-5, 1-891562-17-7.

Yu, D. e Frincke, D. “Towards Survivable Intrusion Detection System”. *hicss*, 09:90299a, 2004. ISSN 1530-1605.

Yu, W. D., Supthaweesuk, P., e Aravind, D. “Trustworthy Web Services Based on Testing”. *sose*, 0: 167–177, 2005.

Zhang, R., Qian, D., Chen, H., e Wu, W. “Collaborative intrusion detection based on coordination agent”. In *Proceedings of the Fourth International Conference on Parallel and Distributed Computing, Applications and Technologies, 2003. PDCAT’2003*, pages 175– 179, Aug. 2003. ISBN 0-7803-7840-7.

Anexos

Anexo A

Acompanhamento e Avaliação do Trabalho: a Gestão de Riscos

A.1 Comunicar e Consultar

No desenvolvimento de pesquisas acadêmicas, podem ser identificados pelo menos cinco tipos de papéis:

1. **Membros do projeto de pesquisa** – pessoas diretamente relacionadas ao desenvolvimento do projeto;

2. **Membros do grupo de pesquisa** – pessoas que pertencem ao mesmo grupo de pesquisa, mas não estão diretamente relacionados à pesquisa em desenvolvimento;

3. **Comunidade científica** – pessoas interessadas nos resultados da pesquisa, como membros de comitês de programa e revisores de simpósios e periódicos, participantes de congressos científicos e leitores dos trabalhos publicados;

4. **Instituições e órgãos de pesquisa** – instituições e órgãos de pesquisa aos quais o projeto de pesquisa está vinculado;

5. **Instituições e órgãos de fomento** – responsáveis pelo custeio do projeto.

Para a elaboração do plano de comunicação e consulta neste projeto de Tese foram considerados apenas os três primeiros papéis, conforme a Tabela A.1. Para cada papel (participantes) são traçados os objetivos da consulta e comunicação, a perspectiva dos participantes na consulta, os métodos utilizados e como serão avaliados os resultados obtidos.

Tabela A.1: Plano de comunicação e consulta

Objetivos	Participantes	Perspectivas dos Participantes	Métodos Usados	Avaliação
Estabelecimento de diretrizes e revisão contínua do projeto	Membros do projeto de pesquisa	Processo contínuo de avaliação dos riscos	Reuniões periódicas e apresentação de relatórios técnicos.	Auto-avaliação
Identificação de possíveis falhas, troca de experiências e obtenção de críticas e sugestões	Membros do grupo de pesquisa	Conhecimento de novas tecnologias	Seminários e encontros.	Análise periódica das contribuições apresentadas.
Obtenção de críticas e sugestões, identificação de novas aplicações, troca de experiências e avaliação do projeto	Comunidade científica	Divulgação e conhecimento de novas tecnologias	Submissão de artigos científicos para prospecção, publicação de resultados e apresentação de artigos.	Compilação e análise das revisões, sugestões e críticas dos artigos.

A.2 Estabelecer o Contexto

A.2.1 Definição dos Objetivos

Tomando como base os requisitos do projeto de Tese, apresentados no primeiro capítulo, foram identificados e descritos cinco objetivos iniciais:

O1: **Detecção de Intrusão Distribuída**

O2: **Uso de Elementos Heterogêneos**

O3: **Composição Dinâmica de IDSs**

O4: **Adoção de Padrões de Interoperabilidade**

O5: **Segurança dos Elementos e da Composição**

No restante do texto, tais objetivos serão referenciados com a ordem e numeração apresentada acima (O1, O2, O3 e O4). A análise do Objetivo 5 (Segurança) é diluída entre os demais objetivos.

A determinação dos ambientes de desenvolvimento e execução do projeto tem por objetivo identificar os riscos desses ambientes que possam afetar os objetivos da pesquisa. No estudo de caso, por se tratar de um projeto envolvendo redes de larga escala, são considerados os ambientes interno e externo.

A.2.2 Definição de Critérios

Conseqüências de Segurança

São definidas na Tabela A.2 as conseqüências da exploração de determinada ameaça à segurança e que possam afetar os objetivos do projeto. Essas conseqüências serão utilizadas para a identificação e a análise dos riscos de segurança do projeto.

Tabela A.2: Conseqüências de segurança

Conseqüência	Descrição
Confidencialidade	Informações críticas são reveladas a usuários não autorizados
Integridade	Informações críticas são alteradas ou eliminadas por usuários não autorizados
Disponibilidade	Elementos de software ou hardware têm sua performance reduzida ou seu funcionamento interrompido.

Critérios de Identificação e Análise dos Riscos de Segurança

Podemos resumir o processo de identificação dos possíveis fatores de riscos, ao conjunto de respostas a três questões: qual é a **ameaça** (exploração de vulnerabilidades); **o que pode ocorrer** (conseqüências); e **como pode ocorrer** (ataque).

Para o cálculo dos riscos, nesse projeto adotamos a metodologia *Common Vulnerability Scoring System* (CVSS)[1], usada pelo NIST para a classificação de vulnerabilidades no *National Vulnerability Database* (NVD)[2]. A base de dados de vulnerabilidades NVD é integrada *ao Common Vulnerabilities and Exposures* (CVE)[3] [Mell e Grance, 2002].

Para a análise de riscos de segurança do projeto, foi adotado apenas o grupo base de critérios de caracterização de vulnerabilidades, já que não estão sendo tratadas vulnerabilidades em softwares específicos.

A.3 Identificar os Riscos

Para identificar as vulnerabilidades associadas ao projeto é desejável uma ampla revisão bibliográfica sobre o assunto. Quando há alguma literatura científica sobre as questões levantadas, tal literatura deve ser avaliada e citada. Se a literatura não é suficiente para a identificação dos riscos, pode ser necessária uma consulta à comunidade científica, como a submissão de trabalhos a congressos e periódicos. As questões levantadas na definição do contexto são revistas e colocadas aos participantes interessados.

As revisões obtidas e a apresentação pública do projeto em congressos científicos auxiliaram na identificação e revisão dos riscos. Usando também a literatura relacionada aos objetivos foi possível identificar possíveis ameaças, o que pode ocorrer (conseqüências), como podem acontecer e quais os possíveis tratamentos.

A seguir, os riscos são separados de acordo com os objetivos. Cabe ressaltar que os controles identificados sugerem os possíveis métodos de tratamento dos riscos e não uma solução ou tecnologia específica.

[1] http://www.first.org/cvss/
[2] http://nvd.nist.gov/
[3] http://cve.mitre.org/

A.3.1 Sistemas de Detecção de Intrusão Distribuídos

Na primeira identificação representada na Tabela A.3, foram consideradas as ameaças associados aos sistemas de detecção de intrusão distribuídos. As ameaças e vulnerabilidades a tais IDSs são conhecidas na literatura, facilitando a determinação dos riscos e possíveis tratamentos. Nesta identificação, destacamos os trabalhos de Lindqvist e Jonsson [1998], Ptacek e Newsham [1998], Mell et al. [2000], Dacier [2002], Yegneswaran et al. [2004] e Yu e Frincke [2004].

A.3.2 Elementos de Detecção de Intrusão Heterogêneos

O segundo objetivo a ter seus riscos identificados é o uso de elementos heterogêneos de detecção de intrusão ou de comunicação entre diferentes IDSs, conforme ilustrado na Tabela A.4. Por se tratar de uma área nova de pesquisa, não há estudos específicos sobre tal assunto. Adaptando o modelo de análise de segurança para composições de software proposto em Han e Zheng [2000], identificamos os riscos associados à segurança dos componentes, à arquitetura do sistema composto e ao processo de design da arquitetura e da composição.

Como a proposta do projeto prevê o uso de ferramentas previamente existentes, mesmo que de diferentes fabricantes, consideramos, principalmente, os riscos relacionados ao uso de softwares prontos (*commercial off-the-shelf – COTS*) [Lindqvist e Jonsson, 1998]. Também foram identificados ameaças na comunicação entre elementos independentes de detecção de intrusão [Bass, 2004].

A.3.3 Composição Dinâmica de IDSs

A maioria das ameaças pertinentes à composição dinâmica estão relacionados à composição de serviços [Esfandiari e Tosic, 2004][Esfandiari e Tosic, 2005] [Wang et al., 2004] [Charfi e Mezini, 2005]. Os trabalhos de Feiertag et al. [2000a, 2000b] e Frincke [2000] auxiliam na identificação dos requisitos de IDSs dinâmicos e das etapas de composição. Novas questões são identificadas a partir destas informações: a disponibilização de elementos de IDSs; a localização desses elementos; a escolha dos elementos; e a reconfiguração da composição. Cada uma dessas questões possui problemas e riscos específicos que foram agrupados na Tabela A.5.

A.3.4 Padrões de Interoperabilidade

O quarto escopo a ser analisado é bastante amplo, pois são combinados o uso de padrões de interoperabilidade com a aplicação da linguagem XML e o uso de *Web Services*. Cada uma dessas questões merece uma análise separada.

Como pretendemos adotar padrões de interoperabilidade emergentes, a análise de Parastatidis e Webber [2004] é bastante esclarecedora. Nela, os riscos associados à adoção de padrões emergentes, principalmente aqueles relacionados aos padrões de Web Services, são identificados e são sugeridas algumas contramedidas. A Tabela A.6 apresenta tais riscos.

Tabela A.3: Registro de riscos: IDSs distribuídos

Ameaça	O que pode ocorrer	Como pode ocorrer	Possíveis Controles	Referências
Negação de Serviço	Afeta a disponibilidade do sistema	Desativação de Elementos por ataques diretos, explorando vulnerabilidades	Uso de mecanismos automáticos para detecção de falhas de funcionamento e Reativação automática dos elementos	[Yegneswaran et al. 2004] [Ptacek and Newsham 1998] [Dacier 2002] [Yu e Frincke, 2004]
			Replicação de elementos	
			FIltragem de tráfego	
			Seleção dinâmica de novos elementos	
		Elementos maliciosos enviam grande quantidade de dados falsos	Controle do fluxo e filtragem da quantidade de mensagens que excedam determinado limite	
Mascaramento	Afeta a integridade e privacidade do sistema	Um elemento malicioso pode se passar por um elemento verdadeiro	Autenticação mútua dos elementos	[Yegneswaran et al. 2004]
Ofuscação	Afeta o desempenho do sistema	Elemento malicioso envia grande quantidade de dados falsos para ofuscar o processo de detecção	Controle do fluxo e filtragem da quantidade de mensagens que excedam determinado limite	[Yegneswaran et al. 2004]
		Elemento malicioso envia pequena quantidade de dados falsos para ofuscar o processo de detecção	Mecanismos de correlação de dados eficientes	
		Tentativa de localização (*scanning*) furtiva ou coordenada dos elementos	Controle de acesso nos mecanismos de registro e pesquisa para localização dos elementos	
			Uso do maior número possível de elementos a fim de detectar tentativas de *scanning*	
Ataque de Inserção	Afeta o desempenho do sistema	O IDS aceita pacotes que são rejeitados pelo sistema alvo. Atacante envia pacotes diretamente ao sensor a fim de iludi-lo.	Uso de sensores baseados em aplicação	[Ptacek and Newsham 1998]
			Uso de diversidade de sensores	[Yu e Frincke, 2004]
Ataque de Evasão	Afeta o desempenho do sistema	O sistema alvo aceita pacotes que o IDS rejeita. Atacante envia pacotes truncados ou com uma ordem trocada para iludir o sensor	Uso de sensores baseados em aplicação	[Ptacek and Newsham 1998]
			Uso de diversidade de sensores	[Yu e Frincke, 2004]
Espionagem	Afeta a confidencialidade do sistema	Atacantes interceptam as mensagens de alerta trocadas entre os elementos de IDS	Uso de canais de comunicação exclusivos	[Mell et al 2000] [Dacier 2002]
			Uso de criptografia	
		Elementos comprometidos são usados para coletar e enviar informações sigilosas	Controle de transmissão	[Lindqvist and Jonsson 1998]
Filtragem de Alertas	Afeta o desempenho, a integridade e a confidencialidade do sistema	Atacantes interceptam mensagens com alertas sobre suas atividades, descartando-as, redirecionando-as ou alterando-as seletivamente	Associação de criptografia e assinatura de mensagens	[Dacier 2002]
Interrupção ou desvio de conexão (*hijacking*)	Afeta o desempenho, a integridade e a confidencialidade do sistema	Atacantes interceptam ou desviam uma conexão entre elementos de IDS	Uso de canais seguros, com criptografia e controle de seção	[Dacier 2002]
Comprometimento de Elementos de IDS	Afeta o desempenho, a integridade e a confidencialidade do sistema	Atacantes alteram o código ou a configuração do elemento	Replicação de elementos em diversas plataformas e comparação de alertas	[Dacier 2002] [Yu e Frincke, 2004]

Tabela A.4: Registro de riscos: elementos heterogêneos

Ameaça	O que pode ocorrer	Como pode ocorrer	Possíveis Controles	Referências
Elementos vulneráveis	Atacantes exploram vulnerabilidades conhecidas de um determinado elemento de IDS	Falhas de *design* e implementação	Aplicação de técnicas de tolerância à intrusão, com uso de diversidade de software	[Lindqvist and Jonsson 1998]
			Atualização continua com a aplicação de correções	
Códigos maliciosos são introduzidos no elemento.	Afeta a integridade e confidencialidade	Não realização de processo de validação do software no momento da aquisição	Validação do software, verificando se ele está de acordo com os requisitos de segurança necessários	[Lindqvist and Jonsson 1998] [Han e Zheng, 2000]
		Interceptação e adulteração durante o processo de entrega do software	Autenticação da origem e uso de canais seguros de entrega	
Confiar em elementos não confiáveis	Inclusão de novas Vulnerabilidades	Elemento não foi projetado para funcionar de forma segura	Limitar o uso a processos sem requisitos de segurança	[Lindqvist and Jonsson 1998]
			Confinamento a ambientes que forneçam segurança ao elemento	
Dificuldade de Gerenciamento	Falhas de operação	Falhas de instalação e de configuração	Aplicação de técnicas e protocolos padronizados de gerenciamento. Qualificação de recursos humanos	[Lindqvist and Jonsson 1998]
	Aumento dos custos operacionais	Altos custos para administrar e manter atualizada uma base de software e hardware muito heterogênea		
	Inclusão de novas Vulnerabilidades	Atualizações inseguras		
		Efeitos inesperados		
		Backdoors de gerenciamento		
Dificuldade de Interoperabilidade	Falha de comunicação	Os elementos utilizam protocolos de comunicação ou formatos de mensagens incompatíveis	Estabelecimento de um formato padrão de comunicação	[Bass 2004]
	Inclusão de novas vulnerabilidades	Níveis de segurança diferentes entre os elementos	Políticas de segurança com um nível mínimo de segurança que deverá ser comum a todos os elementos	[Lindqvist and Jonsson 1998]

Tabela A.5: Registro de riscos: composição dinâmica de IDSs

Ameaça	O que pode ocorrer	Como pode ocorrer	Possíveis Controles	Referências
Dificuldade de localização e escolha dos elementos	Falha no processo de descoberta dos componentes	Incompatibilidade nas descrições dos componentes	Uso de linguagens e ferramentas de descrição padronizadas	[Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]
		Falhas na caracterização das interfaces		
		Descrições imprecisas dos componentes	Uso de taxonomias, ontologia e semântica para caracterização dos componentes	
Elementos indisponíveis	Indisponibilidade no momento da criação ou reconfiguração da composição	Falha nos mecanismos de busca	Mecanismos de busca distribuída e replicação de informações	
		Falha nos componentes	Redundância, mecanismos de detecção de falhas e mecanismos de reativação automática	
Redução da confiabilidade da composição	Afeta a integridade, a confidencialidade e a disponibilidade do sistema	Novo elemento altera as características de segurança da composição	Definir os requisitos de segurança da composição e avaliar os requisitos de segurança do elemento antes de incluí-lo na composição	[Charfi e Mezini, 2005] [Feiertag et al., 2000a, 2000b] [Frincke, 2000]
Exposição de informações confidenciais	Afeta a confidencialidade do sistema	Informações confidenciais são repassadas a novos elementos que não possuem autorização para recebê-las	Definir mecanismos que permitam filtrar e retransmitir informações que estejam de acordo com a política de segurança da composição e dos seus elementos	[Feiertag et al., 2000a, 2000b] [Frincke, 2000]
Negação de Serviço	Afeta a disponibilidade do sistema	Um elemento provedor de serviço aceita mais requisições de serviço do que é capaz de atender	Controlar e limitar o número de requisições	[Feiertag et al., 2000b]
		Um elemento cliente de serviço recebe mais mensagens do que é capaz de processar	Controlar o número de mensagens recebidas e renegociar com os elementos provedores de serviço	

Tabela A.6: Registro de riscos: adoção de padrões

Ameaça	O que pode ocorrer	Como pode ocorrer	Possíveis Controles	Referências
Dificuldade de interoperabilidade	O sistema não consegue integrar com novos elementos de IDS ou de fabricantes distintos	A especificação usada não se torna um padrão	Não adotar especificações muito incipientes	[Parastatidis and Webber 2004]
		Os padrões adotados não são compatíveis entre si.	Adotar padrões que reconhecidamente sejam compatíveis, como os relacionados no WS-Interoperability Profile	
		Alto custo de integração dos padrões		
		Os padrões dependem da tecnologia adotada ou são disponibilizados por um único fabricante	Adotar padrões que sejam independentes de tecnologia	
		O padrão adotado no projeto pode não ser adotado pelo mercado	Usar somente padrões de organizações conhecidas e associadas mercado, como o IETF, OASIS e W3C	
Dificuldade de Implementação	A composição não pode ser implementada em determinados ambientes	Componentes, ferramentas e implementações dos padrões não são completamente compatíveis com a especificação original	Evitar o uso de ferramentas e implementações que disponibilizem recursos não compatíveis com o padrão original	
			Utilizar padrões que possuam a maior variedade de implementações possíveis	
			Dar preferência a implementações e ferramentas amplamente utilizadas no mercado	
	Atrasos na implementação	Falta de pessoal qualificado	Qualificação de pessoal	
			Adotar padrões que já possuam ferramentas no mercado e boa documentação	

Os riscos da aplicação da linguagem XML em conjunto com a tecnologia de web service são analisados em Demchenko et al. [2005] e Yu et al. [2005] e estão representados na Tabela A.7.

A.3.5 Analisar os Riscos

A.3.6 Agrupamento dos Riscos

Os riscos similares levantados na fase anterior foram agrupados e os riscos de menor impacto foram excluídos. Após analisar os resultados da fase anterior do processo de gestão de riscos, verificamos a necessidade de dividi-los em duas categorias: **riscos de segurança** e **riscos operacionais**. Os riscos de segurança referem-se às vulnerabilidades que podem ser exploradas para afetar os requisitos de confidencialidade, integridade e disponibilidade de um sistema de detecção de intrusão de larga escala. Os riscos operacionais não são provocados por vulnerabilidades, mas envolvem decisões que podem afetar o resultado final do projeto de pesquisa.

Alguns eventos, apesar de serem idênticos, são explorados de forma diferente e, portanto, são analisados separadamente. Cada risco recebe uma identificação única que será usada para referenciá-lo nas etapas subseqüentes do processo de gestão de riscos. A Tabela A.8 apresenta os riscos de segurança combinados. A Tabela A.9 apresenta os riscos operacionais combinados.

A.3.7 Agrupamento dos Controles

As técnicas, mecanismos e ferramentas de controle são agrupadas para simplificar o processo de definição de custos e tratamento dos riscos. A Tabela A.10 apresenta os controles de segurança, suas referências e os custos. Os custos foram atribuídos de acordo com a disponibilidade de ferramentas, a necessidade de pessoal especializado, a documentação, a multiplicação de recursos e a facilidade de implementação dos controles. Os controles de baixo custo são aqueles amplamente difundidos, que possuem boa documentação e são de fácil implementação. Os controles de custo médio são pouco difundidos, possuem algumas ferramentas, são de implementação mais complicada ou envolvem redundância de recursos. Já os controles de alto custo são em geral incipientes, necessitam de pessoal especializado, são de difícil implementação ou necessitam de muito mais recursos para serem efetivados.

A Tabela A.11 apresenta os controles relacionados aos riscos operacionais, junto com seus respectivos custos. Os custos dos controles operacionais dependem do esforço necessário para implementá-los. São considerados de baixo custo os controles operacionais que usam pouca mão-de-obra ou podem ser executados em pouco tempo. Os controles de custo médio necessitam de mão-de-obra especializada ou da aplicação de técnicas de gerenciamento. Os controles de alto custo envolvem o uso de métodos ou tecnologias incipientes ou de difícil implementação.

Tabela A.7: Registro de riscos: uso de XML

Ameaça	O que pode ocorrer	Como pode ocorrer	Possíveis Controles	Referências
Sondagem às interfaces dos Web Services	Afeta a confidencialidade do sistema	Atacante identifica interfaces, operações e parâmetros dos serviços que são publicados indevidamente em documentos WSDL ou não estão publicados	Uso de ferramentas de análise de vulnerabilidades para validar os documentos WSDL e as interfaces dos serviços	[Demchenko et al 2005] [Yu et al., 2005]
			Controle de acesso aos documentos WSDL e aos serviços.	
Ataque ao analisador gramatical XML	Afeta a confidencialidade, integridade e disponibilidade do sistema	O analisador gramatical XML é iludido para subjugar a capacidade de processamento ou executar códigos móveis maliciosos	Autenticação, assinatura de mensagens, controle de acesso e análise de conteúdo.	
Conteúdo XML malicioso	Afeta a confidencialidade, integridade e disponibilidade do sistema	Textos XML contém códigos maliciosos que exploram vulnerabilidades ou são executados pelas aplicações		
Ataques por referências externas	Afeta a confidencialidade, integridade e disponibilidade do sistema	Documentos XML contêm ponteiros maliciosos para referências externas que são chamadas ou executadas indevidamente		
Ataques aos protocolos SOAP/XML	Afeta a disponibilidade do sistema (negação de serviço)	Um atacante tenta sobrecarregar o sistema com mensagens SOAP (SOAP *Flooding*)	Autenticação e controle de acesso	
	Afeta a confidencialidade, integridade e disponibilidade do sistema	Mensagens são interceptadas e retransmitidas como se fossem mensagens legítimas (*Replay)*	Uso de *timestamps* nas mensagens e *cache* nos serviços.	
	Afeta a confidencialidade do sistema	Atacantes interceptam mensagens XML (Espionagem)	Uso de Criptografia (*WS-Security*)	
	Afeta a confidencialidade, integridade do sistema	Atacantes interceptam e alteram o conteúdo das mensagens (*Man-in-the-middle*)	Associação de Criptografia e Assinatura de mensagens (*WS-Security*)	
Interferência nas credenciais de segurança XML	Afeta a confidencialidade do sistema	Um atacante pode roubar ou alterar as credenciais dos clientes e serviços	Uso de criptografia na comunicação e de chaves “fortes” nas credenciais. Proteção de credenciais.	
Interferência nas negociações de chaves e seções	Afeta a confidencialidade, integridade e disponibilidade do sistema	Falhas de implementação de segurança, geração de chaves pobres e uso de algoritmos criptográficos fracos ou customizados.	Uso de padrões criptográficos robustos com chaves “fortes”.	

Tabela A.8: Riscos de segurança combinados

Obj.	Risco	Ameaça	Conseqüências	Como pode ocorrer	Referências
O1	RS1.1	Espionagem	Confidencialidade	Atacantes interceptam as mensagens de alerta trocadas entre os elementos de IDS	[Mell et al 2000] [Dacier 2002]
O1	RS1.2	Espionagem	Confidencialidade	Elementos comprometidos são usados para coletar e enviar informações sigilosas	[Lindqvist and Jonsson 1998]
O1	RS1.3	Negação de Serviço	Disponibilidade	Desativação de Elementos por ataques diretos, explorando vulnerabilidades	[Yegneswaran et al. 2004] [Ptacek and Newsham 1998] [Dacier 2002]
O1	RS1.4	Negação de Serviço	Disponibilidade	Elementos maliciosos enviam grande quantidade de dados falsos	[Yegneswaran et al. 2004] [Ptacek and Newsham 1998] [Dacier 2002]
O1	RS1.5	Mascaramento	Integridade e Confidencialidade	Um elemento malicioso pode se passar por um elemento verdadeiro	[Yegneswaran et al. 2004]
O1	RS1.6	Ataque de Evasão	Disponibilidade	O sistema alvo aceita pacotes que o IDS rejeita. Atacante envia pacotes truncados ou com uma ordem trocada para iludir o sensor	[Ptacek and Newsham 1998]
O1	RS1.7	Ataque de Inserção	Disponibilidade	O IDS aceita pacotes que são rejeitados pelo sistema alvo. Atacante envia pacotes diretamente ao sensor a fim de iludi-lo.	[Ptacek and Newsham 1998]
O1	RS1.8	Ofuscação	Disponibilidade	Elemento malicioso envia grande quantidade de dados falsos para ofuscar o processo de detecção	[Yegneswaran et al. 2004]
O1	RS1.9	Ofuscação	Disponibilidade	Elemento malicioso envia pequena quantidade dados falsos para ofuscar o processo de detecção	[Yegneswaran et al. 2004]
O1	RS1.10	Ofuscação	Disponibilidade	Tentativa de localização (*scanning*) furtiva ou coordenada dos elementos	[Yegneswaran et al. 2004]
O1	RS1.11	Filtragem de Alertas	Disponibilidade, Integridade e Confidencialidade	Atacantes interceptam mensagens com alertas sobre suas atividades, descartando-as, redirecionando-as ou alterando-as seletivamente	[Dacier 2002]
O1	RS1.12	Comprometimento de Elementos de IDS	Disponibilidade, Integridade e Confidencialidade	Atacantes alteram o código ou a configuração do elemento	[Dacier 2002]
O1	RS1.13	Interrupção ou desvio de conexão (*hijacking*)	Disponibilidade, Integridade e Confidencialidade	Atacantes interceptam ou desviam uma conexão entre elementos de IDS	[Dacier 2002]
O2	RS2.1	Uso de Elementos vulneráveis	Disponibilidade, Integridade e Confidencialidade	Falhas de *design* e implementaçã. Atacantes exploram vulnerabilidades conhecidas de um determinado elemento de IDS	[Lindqvist and Jonsson 1998]
O2	RS2.2	Uso de elementos contendo código malicioso	Disponibilidade, Integridade e Confidencialidade	Não realização de processo de validação do software no momento da aquisição ou de entrega do software	[Lindqvist and Jonsson 1998] [Han e Zheng, 2000]
O2	RS2.3	Confiar em elementos não confiáveis	Disponibilidade, Integridade e Confidencialidade	Inclusão de novas Vulnerabilidades. Elemento não foi projetado para funcionar de forma segura	[Lindqvist and Jonsson 1998]
O2	RS2.4	Dificuldade de Gerenciamento	Disponibilidade, Integridade e Confidencialidade	Inclusão de novas Vulnerabilidades. Atualizações inseguras.	[Lindqvist and Jonsson 1998]
O2	RS2.5	Dificuldade de Gerenciamento	Disponibilidade, Integridade e Confidencialidade	Inclusão de novas Vulnerabilidades. Efeitos inesperados	[Lindqvist and Jonsson 1998]
O2	RS2.6	Dificuldade de Gerenciamento	Disponibilidade, Integridade e Confidencialidade	*Inclusão de novas Vulnerabilidades. Backdoors* de gerenciamento	[Lindqvist and Jonsson 1998]
O2	RS2.7	Dificuldade de Interoperabilidade	Disponibilidade, Integridade e Confidencialidade	Inclusão de novas vulnerabilidades. Níveis de segurança diferentes entre os elementos	[Lindqvist and Jonsson 1998]
O3	RS3.1	Exposição de informações confidenciais	Confidencialidade	Informações confidenciais são repassadas a novos elementos que não possuem autorização para recebê-las	[Feiertag et al., 2000a, 2000b] [Frincke, 2000]
O3	RS3.2	Negação de Serviço	Disponibilidade	Um elemento provedor de serviço aceita mais requisições de serviço do que é capaz de atender	[Feiertag et al., 2000b]
O3	RS3.3	Negação de Serviço	Disponibilidade	Um elemento cliente de serviço recebe mais mensagens do que é capaz de processar	[Feiertag et al., 2000b]
O3	RS3.4	Redução da confiabilidade da composição	Disponibilidade, Integridade e Confidencialidade	Novo elemento altera as características de segurança da composição	[Charfi e Mezini, 2005] [Feiertag et al., 2000a, 2000b] [Frincke, 2000]
O3	RS3.5	Elementos indisponíveis	Afeta a integridade e a disponibilidade do sistema	Falha nos mecanismos de busca	[Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]
O3	RS3.6	Elementos indisponíveis	Afeta a integridade e a disponibilidade do sistema	Falha nos componentes	[Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]
O4	RS4.1	Sondagem às interfaces dos Web Services	Confidencialidade	Atacante identifica interfaces, operações e parâmetros dos serviços que são publicados indevidamente em documentos WSDL ou não estão publicados	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.2	Ataque ao analisador gramatical XML	Disponibilidade, Integridade e Confidencialidade	O analisador gramatical XML é iludido para subjugar a capacidade de processamento ou executar códigos móveis maliciosos	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.3	Conteúdo XML malicioso	Disponibilidade, Integridade e Confidencialidade	Textos XML contém códigos maliciosos que exploram vulnerabilidades ou são executados pelas aplicações	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.4	Ataques por referências externas	Disponibilidade, Integridade e Confidencialidade	Documentos XML contêm ponteiros maliciosos para referências externas que são chamadas ou executadas indevidamente	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.5	Ataques aos protocolos SOAP/XML	Disponibilidade	Um atacante tenta sobrecarregar o sistema com mensagens SOAP (SOAP *Flooding*)	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.6	Ataques aos protocolos SOAP/XML	Disponibilidade, Integridade e Confidencialidade	Mensagens são interceptadas e retransmitidas como se fossem mensagens legítimas (*Replay)*	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.7	Ataques aos protocolos SOAP/XML	Confidencialidade	Atacantes interceptam mensagens XML (Espionagem)	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.8	Ataques aos protocolos SOAP/XML	Confidencialidade e Integridade	Atacantes interceptam e alteram o conteúdo das mensagens (*Man-in-the-middle*)	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.9	Interferência nas credenciais de segurança XML	Confidencialidade	Um atacante pode roubar ou alterar as credenciais dos clientes e serviços	[Demchenko et al 2005] [Yu et al., 2005]
O4	RS4.10	Interferência nas negociações de chaves e seções	Disponibilidade, Integridade e Confidencialidade	Falhas de implementação de segurança, geração de chaves pobres e uso de algoritmos criptográficos fracos ou customizados.	[Demchenko et al 2005] [Yu et al., 2005]

Tabela A.9: Riscos operacionais combinados

Obj.	Item	Ameaça	Conseqüências	Como pode ocorrer	Referências
O2	RO2.1	Aumento dos custos operacionais	Dificuldade de Gerenciamento	Altos custos para administrar e manter atualizada uma base de software e hardware muito heterogênea	[Lindqvist and Jonsson 1998]
O2	RO2.2	Falha de comunicação	Dificuldade de Interoperabilidade	Os elementos utilizam protocolos de comunicação ou formatos de mensagens incompatíveis	[Bass 2004]
O3	RO3.1	Falha no processo de descoberta dos componentes	Dificuldade de localização e escolha dos elementos	Incompatibilidade nas descrições dos componentes e falhas na caracterização das interfaces	[Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]
O3	RO3.2	Falha no processo de descoberta dos componentes	Dificuldade de localização e escolha dos elementos	Descrições imprecisas dos componentes	[Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]
O4	RO4.1	O sistema não consegue integrar com novos elementos de IDS ou de fabricantes distintos	Dificuldade de interoperabilidade	A especificação usada não se torna um padrão	[Parastatidis and Webber 2004]
O4	RO4.2	O sistema não consegue integrar com novos elementos de IDS ou de fabricantes distintos	Dificuldade de interoperabilidade	Os padrões adotados não são compatíveis entre si ou o custo de integração é muito alto.	[Parastatidis and Webber 2004]
O4	RO4.3	O sistema não consegue integrar com novos elementos de IDS ou de fabricantes distintos	Dificuldade de interoperabilidade	Os padrões dependem da tecnologia adotada ou são disponibilizados por um único fabricante	[Parastatidis and Webber 2004]
O4	RO4.4	O sistema não consegue integrar com novos elementos de IDS ou de fabricantes distintos	Dificuldade de interoperabilidade	O padrão adotado no projeto pode não ser adotado pelo mercado	[Parastatidis and Webber 2004]
O4	RO4.5	A composição não pode ser implementada em determinados ambientes	Dificuldade de Implementação	Componentes, ferramentas e implementações dos padrões não são completamente compatíveis com a especificação original	[Parastatidis and Webber 2004]
O4	RO4.6	Atrasos na implementação	Dificuldade de Implementação	Falta de pessoal qualificado	[Parastatidis and Webber 2004]

Tabela A.10: Classificação de controles de segurança e seus custos

Item	Controle	Referências	Custo
CS1	Autenticação	[Yegneswaran et al. 2004] [Demchenko et al 2005] [Yu et al., 2005] [Lindqvist and Jonsson 1998] [Han e Zheng, 2000]	BAIXO
CS2	Assinatura	[Demchenko et al 2005] [Yu et al., 2005] [Dacier 2002]	BAIXO
CS3	Controle de Acesso	[Demchenko et al 2005] [Yu et al., 2005] [Yegneswaran et al. 2004]	BAIXO
CS4	Controle de fluxo	[Lindqvist and Jonsson 1998] [Yegneswaran et al. 2004] [Ptacek and Newsham 1998] [Dacier 2002] [Feiertag et al., 2000b] [Frincke, 2000]	BAIXO
CS5	Criptografia	[Mell et al 2000] [Dacier 2002] [Demchenko et al 2005] [Yu et al., 2005]	BAIXO
CS6	Detecção de falhas	[Yegneswaran et al. 2004] [Ptacek and Newsham 1998] [Dacier 2002] [Demchenko et al 2005] [Yu et al., 2005] [Feiertag et al., 2000a, 20 00b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]	BAIXO
CS7	Filtragem de Tráfego	[Yegneswaran et al. 2004] [Ptacek and Newsham 1998] [Dacier 2002] [Feiertag et al., 2000a, 2000b]	BAIXO
CS8	Reativação automática	[Yegneswaran et al. 2004] [Ptacek and Newsham 1998] [Dacier 2002] [Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]	BAIXO
CS9	Sensores baseados em aplicação	[Ptacek and Newsham 1998]	BAIXO
CS10	*Timestamps* e *cache*	[Demchenko et al 2005] [Yu et al., 2005]	BAIXO
CS11	Análise de Conteúdo	[Demchenko et al 2005] [Yu et al., 2005]	MÉDIO
CS12	Correlação de dados	[Yegneswaran et al. 2004] [Dacier 2002]	MÉDIO
CS13	Distribuição de sistemas	[Yegneswaran et al. 2004] [Feiertag et al., 2000a, 2000b] [Esfandiari e T osic, 2004, 2005] [Wang et al., 2004]	MÉDIO
CS14	Diversidade	[Ptacek and Newsham 1998]	MÉDIO
CS15	Política de Segurança	[Lindqvist and Jonsson 1998] [Feiertag et al., 2000a, 2000b]	MÉDIO
CS16	Replicação	[Yegneswaran et al. 2004] [Ptacek and Newsham 199 8] [Dacier 2002] [Dacier 2002] [Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]	MÉDIO
CS17	Seleção dinâmica	[Yegneswaran et al. 2004] [Ptacek and Newsham 1998] [Dacier 2002]	MÉDIO
CS18	Gerenciamento	[Lindqvist and Jons son 1998] [Han e Zheng, 2000] [Charfi e Mezini, 2005] [Feiertag et al., 2000a, 2000b] [Demchenko et al 2005] [Yu et al., 2005] [Frincke, 2000]	MÉDIO
CS19	Uso de recursos exclusivos	[Mell et al 2000] [Dacier 2002]	ALTO

Tabela A.11: Classificação de controles operacionais e seus custos

Item	Controles	Referências	Custos
CO1	Avaliação dos padrões	[Parastatidis and Webber 2004]	BAIXO
CO2	Caracterização padronizada	[Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]	ALTO
CO3	Comunicação padronizada	[Bass 2004]	MÉDIO
CO4	Descrição padronizada	[Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]	MÉDIO
CO5	Disponibilidade de ferramentas	[Parastatidis and Webber 2004]	MÉDIO
CO6	Documentação	[Parastatidis and Webber 2004]	BAIXO
CO7	Gerenciamento padronizado	[Lindqvist and Jonsson 1998]	BAIXO
CO8	Independência de Tecnologia	[Parastatidis and Webber 2004]	BAIXO
CO9	Qualificação de recursos humanos	[Lindqvist and Jonsson 1998] [Parastatidis and Webber 2004]	MÉDIO

A.3.8 Nível de Risco

Após combinar os riscos de segurança identificados anteriormente, aplicamos os critérios de análise definidos na seção A.2.2 para construir a Tabela A.12. A tabela apresenta os riscos, as conseqüências e o impacto da exploração do risco. A tabela também contém o escore de risco de cada um dos itens relacionados à segurança. Esse escore foi obtido com a aplicação da metodologia CVSS, descrita na seção A.2.2. Assumimos nesta análise que não há controles implementados, pois os mesmos serão propostos nas próximas etapas do processo de gestão de riscos.

Para exemplificar o cálculo do escore e a definição do nível de risco, tomemos como exemplo o risco RS1.1 (Espionagem - atacantes interceptam as mensagens de alerta trocadas entre os elementos

de IDS). Para explorar a vulnerabilidade, o atacante pode estar remotamente (valor de acesso = 1), a complexidade de acesso é baixa (valor complexidade de acesso = 1) e é desnecessária a autenticação no sistema alvo (valor autenticação = 1). Como conseqüência da exploração da vulnerabilidade, há uma quebra completa da confidencialidade (valor impacto na confidencialidade = 1), mas não são afetadas a integridade e a disponibilidade do sistema (valores de impacto = 0). Como nesse caso, a confidencialidade é um fator importante, aplica-se uma tendência de impacto maior a este quesito (valor tendência de impacto = impacto na confidencialidade * 0,5 + impacto na integridade * 0,25 + impacto na disponibilidade * 0,25). Concluída a análise, é aplicada a fórmula apresentada na seção A.2.2 para a obtenção do escore:

(10 * 1 * 1 * 1 * ((1 * 0,5) + (0 * 25) + (0 * 0,25))) = 5

Após ser calculado o escore, é atribuído o nível de risco de acordo com a classificação do NIST (ver seção 6.4.2), na qual o escore com valor “5” é considerado como nível de risco médio. Essa operação de cálculo é repetida para todos os riscos listados.

Tabela A.12: Níveis de risco de segurança e a avaliação dos riscos

Risco	Acesso	Comple-xidade de Acesso	Autenticação	Impacto na Confidencialidade	Impacto na Integridade	Impacto na Disponibilidade	Tendência de Impacto	**Escore**	**Nível de Risco**
RS1.1	remoto	baixa	desnecessária	completa	nenhuma	nenhuma	confidencialidade	**5**	**médio**
RS1.2	local	alta	necessária	completa	nenhuma	nenhuma	confidencialidade	**1,7**	**baixo**
RS1.3	remoto	baixa	desnecessária	nenhuma	parcial	completa	disponibilidade	**6,8**	**médio**
RS1.4	remoto	baixa	desnecessária	nenhuma	nenhuma	completa	disponibilidade	**5**	**médio**
RS1.5	local	alta	necessária	completa	parcial	nenhuma	normal	**3,2**	**baixo**
RS1.6	remoto	baixa	desnecessária	nenhuma	nenhuma	parcial	disponibilidade	**3,5**	**baixo**
RS1.7	remoto	baixa	desnecessária	nenhuma	nenhuma	parcial	disponibilidade	**3,5**	**baixo**
RS1.8	remoto	baixa	desnecessária	nenhuma	nenhuma	parcial	disponibilidade	**3,5**	**baixo**
RS1.9	remoto	baixa	desnecessária	nenhuma	nenhuma	parcial	disponibilidade	**3,5**	**baixo**
RS1.10	remoto	baixa	desnecessária	nenhuma	nenhuma	parcial	disponibilidade	**3,5**	**baixo**
RS1.11	remoto	alta	desnecessária	completa	completa	parcial	normal	**7,2**	**alto**
RS1.12	local	alta	necessária	completa	completa	completa	normal	**3,4**	**baixo**
RS1.13	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS2.1	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS2.2	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS2.3	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS2.4	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS2.5	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS2.6	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS2.7	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS3.1	remoto	alta	necessária	completa	nenhuma	nenhuma	confidencialidade	**2,4**	**baixo**
RS3.2	remoto	baixa	desnecessária	nenhuma	nenhuma	completa	disponibilidade	**5**	**médio**
RS3.3	remoto	baixa	desnecessária	nenhuma	nenhuma	completa	disponibilidade	**5**	**médio**
RS3.4	remoto	alta	necessária	completa	completa	completa	normal	**4,8**	**médio**
RS3.5	remoto	baixa	desnecessária	nenhuma	parcial	completa	normal	**5,7**	**médio**
RS3.6	remoto	baixa	desnecessária	nenhuma	parcial	completa	normal	**5,7**	**médio**
RS4.1	remoto	alta	desnecessária	parcial	nenhuma	nenhuma	confidencialidade	**2,8**	**baixo**
RS4.2	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS4.3	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS4.4	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS4.5	remoto	baixa	desnecessária	nenhuma	nenhuma	completa	disponibilidade	**5**	**médio**
RS4.6	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS4.7	remoto	alta	desnecessária	completa	nenhuma	nenhuma	confidencialidade	**4**	**médio**
RS4.8	remoto	alta	desnecessária	completa	completa	nenhuma	normal	**5,3**	**médio**
RS4.9	remoto	alta	desnecessária	completa	completa	completa	normal	**8**	**alto**
RS4.10	remoto	alta	necessária	completa	completa	completa	normal	**4,8**	**médio**

A.4 Avaliar os Riscos

Parte do trabalho de avaliação dos riscos de segurança foi adiantado ao se confeccionar a Tabela A.12, tomando como base os dados obtidos na análise de riscos. Analisando a Tabela A.12, foi construído o gráfico da Figura A.1, no qual verifica-se uma maior quantidade de vulnerabilidades com alto risco de segurança, representando 39% do total de vulnerabilidades, enquanto as vulnerabilidades de baixo e médio risco representam respectivamente 28% e 33% do total. Isso demonstra a necessidade de um tratamento de riscos para reduzir a quantidade de vulnerabilidades de médio e alto risco a um nível mínimo aceitável, de acordo com os recursos disponíveis. Sendo assim, é dada prioridade ao tratamento das vulnerabilidades de médio e alto risco.

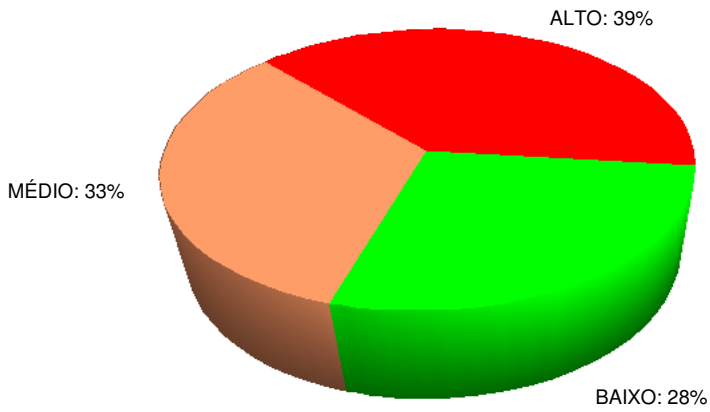

Figura A.1: Distribuição dos níveis de risco iniciais

Os riscos operacionais relacionados ao uso de elementos heterogêneos estão associados a dificuldades de interoperabilidade e gerenciamento, que podem ser tratados com a adoção de padrões (Objetivo 3 – O3). Os riscos operacionais da composição dinâmica estão ligados a problemas de descrição e caracterização de componentes e serviços. Esses tipos de risco necessitam para seu tratamento de áreas de pesquisa incipientes que envolvem o uso de taxonomias, ontologia e semântica.

A.5 Tratar os Riscos

As opções de tratamentos foram identificadas na literatura e agrupadas na Tabela A.10 (segurança) e na Tabela A.11 (operacional), junto com os custos envolvidos. Para algumas vulnerabilidades é necessário associar diversos tratamentos.

Os tratamentos identificados para os riscos operacionais O2 e O3 foram considerados como boas práticas de desenvolvimento, para a seleção dos padrões utilizados no projeto. Tais controles foram adotados, por exemplo, na escolha de padrões de organizações conhecidas, como o IETF, W3C e OASIS. Também foram úteis na escolha de ferramentas de desenvolvimento que são bem documentadas e conhecidas. O controle para a caracterização padronizada (CO2) foi implementado com a aplicação de uma taxonomia de elementos de detecção de intrusão. Já o controle para o uso de linguagens de descrição padrão foi efetivado com a adoção de XML e da especificação BPEL4WS.

Tabela A.13: Análise dos riscos de segurança tratados

Risco	Controles	Acesso	Complexidade de Acesso	Autenticação	Impacto na Confidencialidade	Impacto na Integridade	Impacto na Disponibilidade	Escore	Nível de Risco
RS1.1	C19	LOCAL	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	**0,0**	**BAIXO**
RS1.1	C5	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	**0,0**	**BAIXO**
RS1.2	C4	LOCAL	ALTA	NECESSÁRIA	PARCIAL	NENHUMA	NENHUMA	**1,2**	**BAIXO**
RS1.3	C6 + C8	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**3,5**	**BAIXO**
RS1.3	C16	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**3,5**	**BAIXO**
RS1.3	C7	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**3,5**	**BAIXO**
RS1.3	C17	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**3,5**	**BAIXO**
RS1.4	C4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**3,5**	**BAIXO**
RS1.5	C1	LOCAL	ALTA	NECESSÁRIA	COMPLETA	PARCIAL	NENHUMA	**1,9**	**BAIXO**
RS1.6	C9	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**2,8**	**BAIXO**
RS1.6	C14	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**2,8**	**BAIXO**
RS1.7	C9	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**2,8**	**BAIXO**
RS1.7	C14	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**2,8**	**BAIXO**
RS1.8	C4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**3,5**	**BAIXO**
RS1.9	C12	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**2,8**	**BAIXO**
RS1.10	C3	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**1,7**	**BAIXO**
RS1.10	C13	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**2,8**	**BAIXO**
RS1.11	C2 + C5	REMOTO	ALTA	NECESSÁRIA	PARCIAL	NENHUMA	NENHUMA	**1,1**	**BAIXO**
RS1.12	C16	LOCAL	ALTA	NECESSÁRIA	PARCIAL	NENHUMA	PARCIAL	**1,6**	**BAIXO**
RS1.13	C5	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	PARCIAL	PARCIAL	**3,7**	**BAIXO**
RS2.1	C6+C13+C14+C16	REMOTO	ALTA	DESNECESSÁRIA	PARCIAL	NENHUMA	PARCIAL	**3,7**	**BAIXO**
RS2.1	C18	REMOTO	ALTA	DESNECESSÁRIA	PARCIAL	PARCIAL	PARCIAL	**5,6**	**MÉDIO**
RS2.2	C18	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	**0,0**	**BAIXO**
RS2.2	C1	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	**0,0**	**BAIXO**
RS2.3	C18	LOCAL	ALTA	DESNECESSÁRIA	PARCIAL	PARCIAL	PARCIAL	**3,9**	**BAIXO**
RS2.3	C19	LOCAL	ALTA	DESNECESSÁRIA	PARCIAL	PARCIAL	PARCIAL	**3,9**	**BAIXO**
RS2.4	C18	LOCAL	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**5,6**	**MÉDIO**
RS2.5	C18	LOCAL	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**5,6**	**MÉDIO**
RS2.6	C18	LOCAL	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**5,6**	**MÉDIO**
RS2.7	C15 + C3	REMOTO	ALTA	NECESSÁRIA	NENHUMA	PARCIAL	PARCIAL	**2,2**	**BAIXO**
RS3.1	C7 + C15	REMOTO	ALTA	NECESSÁRIA	PARCIAL	NENHUMA	NENHUMA	**1,7**	**BAIXO**
RS3.2	C3 + C4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**3,5**	**BAIXO**
RS3.3	C4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**3,5**	**BAIXO**
RS3.4	C18	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**1,1**	**BAIXO**
RS3.5	C13 + C16	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**1,9**	**BAIXO**
RS3.6	C6+C8+C16	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**1,9**	**BAIXO**
RS4.1	C18	REMOTO	ALTA	DESNECESSÁRIA	PARCIAL	NENHUMA	NENHUMA	**2,8**	**BAIXO**
RS4.1	C3	REMOTO	ALTA	NECESSÁRIA	PARCIAL	NENHUMA	NENHUMA	**1,7**	**BAIXO**
RS4.2	C1+C2+C3+C11	REMOTO	ALTA	NECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**4,8**	**MÉDIO**
RS4.3	C1+C2+C3+C11	REMOTO	ALTA	NECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**4,8**	**MÉDIO**
RS4.4	C1+C2+C3+C11	REMOTO	ALTA	NECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**4,8**	**MÉDIO**
RS4.5	C1 + C3	REMOTO	BAIXA	NECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**2,1**	**BAIXO**
RS4.6	C10	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**1,9**	**BAIXO**
RS4.6	C2 + C5	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**1,1**	**BAIXO**
RS4.7	C5	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	**0,0**	**BAIXO**
RS4.8	C2 + C5	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	**0,0**	**BAIXO**
RS4.9	C5 + C18	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	**0,0**	**BAIXO**
RS4.10	C5 + C18	REMOTO	ALTA	NECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	**0,0**	**BAIXO**

As tabelas de identificação dos riscos de segurança (ver seção 6.5) e as tabelas com os controles para cada vulnerabilidade foram combinadas para uma nova análise de riscos, conforme ilustrado na Tabela A.13. Para simplificar a tabela, não foi incluída a tendência de impacto, que permanece a mesma da análise dos riscos não tratados.

Para mensurar os custos desses tratamentos combinados, será adotado o custo do tratamento mais oneroso, independentemente da quantidade de tratamentos associados.

Comparando os níveis de risco originais, os níveis de risco tratados e os custos de tratamento, identificamos quais tratamentos são mais eficientes e quais deverão ser implementados. A Tabela A.14 apresenta essa comparação e os tratamentos selecionados. Foram 36 tratamentos selecionados e 12 tratamentos rejeitados. Apenas em um caso, no item RS1.3, foram combinadas as alternativas de tratamento, devido ao baixo custo de ambas.

Após a seleção dos tratamentos, verificamos a eliminação das vulnerabilidades de alto risco e a redução significativa dos riscos médios, conforme pode ser observado no gráfico da Figura A.2.

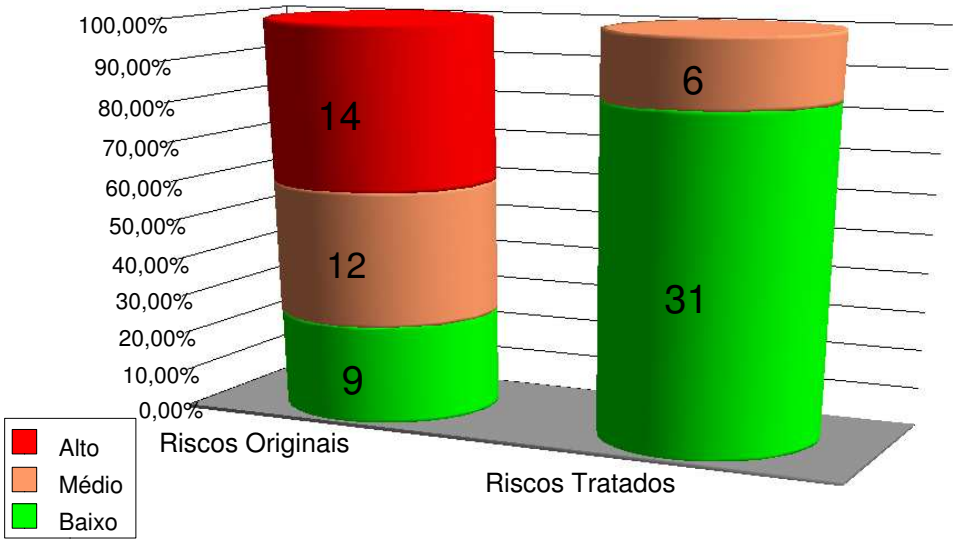

Figura A.2: Comparativo dos níveis de risco de segurança, antes e após o tratamento

A.5.1 Monitorar e Rever

Nos projetos científicos, as revisões da gestão de riscos podem ser motivadas pela inclusão de novos elementos ao projeto, como, por exemplo, mudanças de paradigmas, novas bibliografias ou revisões de trabalhos submetidos para publicação. O processo de comunicação e consulta (seção 6.3) é de extrema importância na revisão de projetos científicos.

Um método de revisão é o refinamento contínuo da análise. Parte-se de um escopo mais global no qual novos elementos do projeto são identificados. Depois, tais elementos são analisados separadamente para verificar se novos riscos foram descobertos e se novos tratamentos são necessários.

Tabela A.14: Avaliação e seleção dos tratamentos de segurança

Risco	Controles	Escore Original	Escore Tratado	Nível de Risco Original	Nível de Risco Tratado	Custo	Aplicação
RS1.1	C19	4	**0,0**	MÉDIO	**BAIXO**	**ALTO**	**Não**
RS1.1	C5	4	**0,0**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS1.2	C4	1,7	**1,2**	BAIXO	**BAIXO**	**BAIXO**	**Sim**
RS1.3	C6 + C8	6,8	**3,5**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS1.3	C16	6,8	**3,5**	MÉDIO	**BAIXO**	**MÉDIO**	**Não**
RS1.3	C7	6,8	**3,5**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS1.3	C17	6,8	**3,5**	MÉDIO	**BAIXO**	**MÉDIO**	**Não**
RS1.4	C4	5	**3,5**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS1.5	C1	3,2	**1,9**	BAIXO	**BAIXO**	**BAIXO**	**Sim**
RS1.6	C9	3,5	**2,8**	BAIXO	**BAIXO**	**BAIXO**	**Sim**
RS1.6	C14	3,5	**2,8**	BAIXO	**BAIXO**	**MÉDIO**	**Não**
RS1.7	C9	3,5	**2,8**	BAIXO	**BAIXO**	**BAIXO**	**Sim**
RS1.7	C14	3,5	**2,8**	BAIXO	**BAIXO**	**MÉDIO**	**Não**
RS1.8	C4	3,5	**3,5**	BAIXO	**BAIXO**	**BAIXO**	**Sim**
RS1.9	C12	3,5	**2,8**	BAIXO	**BAIXO**	**MÉDIO**	**Não**
RS1.10	C3	3,5	**1,7**	BAIXO	**BAIXO**	**BAIXO**	**Sim**
RS1.10	C13	3,5	**2,8**	BAIXO	**BAIXO**	**MÉDIO**	**Não**
RS1.11	C2 + C5	7,2	**1,1**	ALTO	**BAIXO**	**BAIXO**	**Sim**
RS1.12	C16	3,4	**1,6**	BAIXO	**BAIXO**	**MÉDIO**	**Sim**
RS1.13	C5	8	**3,7**	ALTO	**BAIXO**	**BAIXO**	**Sim**
RS2.1	C6+C13+C14+C16	8	**3,7**	ALTO	**BAIXO**	**MÉDIO**	**Sim**
RS2.1	C18	8	**5,6**	ALTO	**MÉDIO**	**MÉDIO**	**Não**
RS2.2	C18	8	**0,0**	ALTO	**BAIXO**	**MÉDIO**	**Não**
RS2.2	C1	8	**0,0**	ALTO	**BAIXO**	**BAIXO**	**Sim**
RS2.3	C18	8	**3,9**	ALTO	**BAIXO**	**MÉDIO**	**Sim**
RS2.3	C19	8	**3,9**	ALTO	**BAIXO**	**ALTO**	**Não**
RS2.4	C18	8	**5,6**	ALTO	**MÉDIO**	**MÉDIO**	**Sim**
RS2.5	C18	8	**5,6**	ALTO	**MÉDIO**	**MÉDIO**	**Sim**
RS2.6	C18	8	**5,6**	ALTO	**MÉDIO**	**MÉDIO**	**Sim**
RS2.7	C15 + C3	8	**2,2**	ALTO	**BAIXO**	**MÉDIO**	**Sim**
RS3.1	C7 + C15	2,4	**1,7**	BAIXO	**BAIXO**	**MÉDIO**	**Sim**
RS3.2	C3 + C4	5	**3,5**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS3.3	C4	5	**3,5**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS3.4	C18	4,8	**1,1**	MÉDIO	**BAIXO**	**MÉDIO**	**Sim**
RS3.5	C13 + C16	5,7	**1,9**	MÉDIO	**BAIXO**	**MÉDIO**	**Sim**
RS3.6	C6+C8+C16	5,7	**1,9**	MÉDIO	**BAIXO**	**MÉDIO**	**Sim**
RS4.1	C18	2,8	**2,8**	BAIXO	**BAIXO**	**MÉDIO**	**Não**
RS4.1	C3	2,8	**1,7**	BAIXO	**BAIXO**	**BAIXO**	**Sim**
RS4.2	C1+C2+C3+ C11	8	**4,8**	ALTO	**MÉDIO**	**BAIXO**	**Sim**
RS4.3	C1+C2+C3+ C11	8	**4,8**	ALTO	**MÉDIO**	**BAIXO**	**Sim**
RS4.4	C1+C2+C3+ C11	8	**4,8**	ALTO	**MÉDIO**	**BAIXO**	**Sim**
RS4.5	C1 + C3	5	**2,1**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS4.6	C10	8	**1,9**	ALTO	**BAIXO**	**BAIXO**	**Não**
RS4.6	C2 + C5	8	**1,1**	ALTO	**BAIXO**	**BAIXO**	**Sim**
RS4.7	C5	4	**0,0**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS4.8	C2 + C5	5,3	**0,0**	MÉDIO	**BAIXO**	**BAIXO**	**Sim**
RS4.9	C5 + C18	8	**0,0**	ALTO	**BAIXO**	**MÉDIO**	**Sim**
RS4.10	C5 + C18	4,8	**0,0**	MÉDIO	**BAIXO**	**MÉDIO**	**Sim**

Durante a elaboração desta Tese, ocorreram várias revisões do processo de gestão de riscos. Muitas dessas revisões não seguiram o ciclo completo, envolvendo principalmente as fases de determinação dos riscos. Nesses ciclos foram incluídos novos riscos e os critérios de análise e avaliação precisaram ser revistos até chegar ao resultado final apresentado neste texto.

A.6 Registro do Processo de Gestão de Riscos

Ao longo das seções anteriores foi apresentada a documentação pertinente ao estudo de caso analisado. Na seção 6.5, mantivemos as tabelas de identificação dos riscos originais para ilustrar como os dados fluem ao longo do processo e como depois eles são agrupados para análise (seção 6.6). Várias versões das tabelas e planilhas produzidas no estudo foram armazenadas.

A.7 Considerações sobre a Gestão de Riscos

Com a metodologia aplicada foi possível acompanhar o desenvolvimento das soluções propostas e implementadas nesta Tese e trabalhar os riscos envolvidos, melhorando significativamente a compreensão dos problemas e suas soluções. Tal acompanhamento ocasionou melhorias evidentes nos resultados do projeto.

Durante o processo foi preciso revisar e agrupar várias referências científicas dentro do escopo de cada objetivo. Como conseqüência, além da metodologia de gestão de riscos aplicada a cada área do escopo do projeto, temos como contribuições adicionais:

1. Identificação de riscos nas composições de IDSs;
2. Análise e avaliação dos riscos de segurança;
3. Tratamentos para riscos operacionais discutidos na literatura científica;
4. Identificação de riscos e tratamentos associados a IDSs distribuídos;
5. Identificação de riscos e tratamentos associados ao uso de COTS; e
6. Identificação de riscos e tratamentos associados à composição dinâmica de IDSs.

A gestão de riscos também pode ser aplicada em outras etapas do projeto, como na avaliação dos produtos utilizados no desenvolvimento do protótipo. Nesse caso, as etapas de identificação, análise, avaliação e tratamento assemelham-se às da norma SP800-30 [Stoneburner et al., 2002]. O uso do CVSS aplicado pelo NVD possibilita localizar todas as vulnerabilidades associadas a ambientes de desenvolvimento, linguagens de programação, servidores Web, sistemas operacionais e outras ferramentas usadas no protótipo, de acordo com o produto, a versão e o vendedor, permitindo calcular com precisão os níveis de risco. Também estão disponíveis todas as opções de tratamento para as

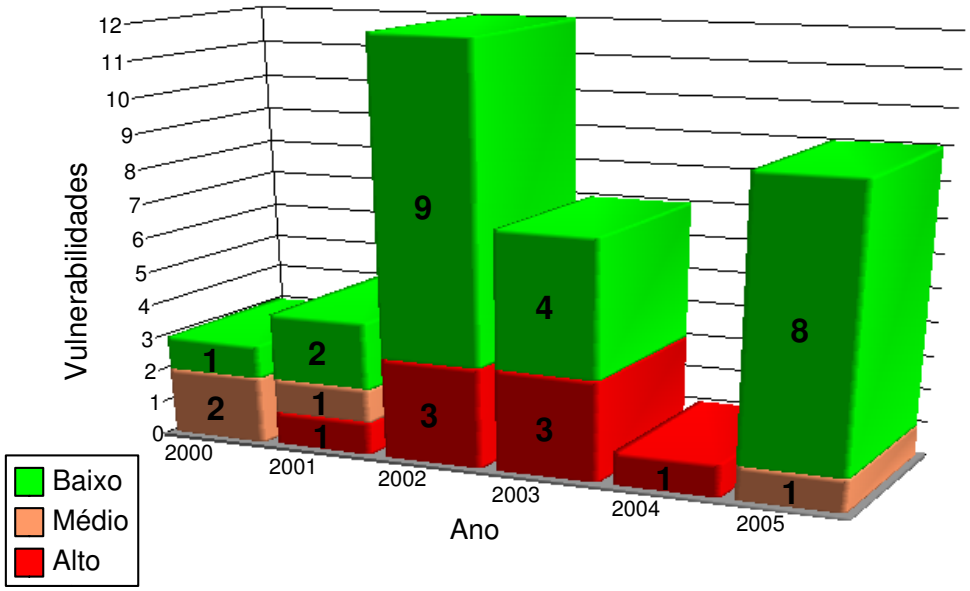

Figura A.3: Níveis de risco das vulnerabilidades do servidor Tomcat

vulnerabilidades listadas. Para a determinação do nível de risco basta pesquisar pelo produto no CVSS.

Tomando como exemplo a quantidade de vulnerabilidades registradas no NVD para o servidor Web *Tomcat*, foi formulado o gráfico da Figura A.3, refletindo as características temporais de quando foram classificadas pelo NIST. No nível de risco não está incluído o impacto ambiental. As trinta e cinco vulnerabilidades reportadas são registradas para a análise de risco. Para cada vulnerabilidade é verificada a pertinência ao projeto e vulnerabilidades relacionadas às versões do software usadas no projeto são reavaliadas e tratadas. Por ser extremamente longa, a documentação dos riscos dos produtos utilizados foi suprimida deste documento.

Com a aplicação da metodologia de gestão de riscos, esperamos ter identificado e tratado a maioria das vulnerabilidades conhecidas e pertinentes às soluções tecnológicas adotadas nesta Tese. Isso sem dúvida auxiliou no entendimento dos problemas que seriam enfrentados, tendo como conseqüência a melhoria da segurança do projeto. Porém, não é possível garantir que o projeto seja totalmente seguro. Contudo, podemos afirmar que foram realizadas boas práticas de gestão e que foram tomadas todas as medidas preventivas necessárias à atenuação do impacto negativo que possíveis vulnerabilidades infringiriam ao projeto.

A.8 Conclusões do Capítulo

Neste capítulo adotamos a metodologia de gestão de riscos para o acompanhamento e a avaliação dos requisitos de segurança envolvidos nesta Tese. A metodologia adotada é baseada na norma AS/NZ4360, que vem sendo aplicada em diversas áreas do conhecimento. Foram apresentados to-

dos os passos da metodologia, ilustrados com a documentação contendo o resultado da aplicação da mesma no desenvolvimento da Tese.

Como resultado do uso da gestão de riscos no acompanhamento do desenvolvimento dos mecanismos de segurança usados na Tese, foram verificados ganhos significativos, tanto na segurança, quanto no processo de desenvolvimento do projeto.