VIII Simpósio Brasileiro em Segurança da Informação
e de Sistemas Computacionais
1 a 5 de setembro de 2008
Gramado, RS

Minicursos
Livro Texto

Editora
Sociedade Brasileira de Computação (SBC)

Organizadores
Carlos Alberto Maziero (PUCPR)
Luciano Paschoal Gaspary (UFRGS)
Raul Fernando Weber (UFRGS)

Realização
Instituto de Informática
Universidade Federal do Rio Grande do Sul (UFRGS)

Promoção
Sociedade Brasileira de Computação (SBC)

Apoio Técnico
International Federation for Information Processing – Technical Committee on Security and Protection in Information Systems (IFIP/TC11)

Capa: Josué Klafke Sperb
Produção Editorial: Alan Diego dos Santos, Carlos Alberto Maziero, Fabrício Girardi Andreis, Juliano Araújo Wickboldt, Luciano Paschoal Gaspary e Roben Castagna Lunardi.

Cópias Adicionais:
Sociedade Brasileira de Computação (SBC)
Av. Bento Gonçalves, 9500 - Setor 4 - Prédio 43.412 - Sala 219
Bairro Agronomia - CEP 91.509-900 - Porto Alegre - RS
Fone: (51) 3308-6835
E-mail: sbc@sbc.org.br

Dados Internacionais de Catalogação na Publicação (CIP)

Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (8. : 2008 : Gramado, RS).

Minicursos / VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais ; organizadores Carlos Alberto Maziero, Luciano Paschoal Gaspary, Raul Fernando Weber. − Porto Alegre : SBC, c2008.

x, 187 p.

ISBN 978-85-7669-191-4

1. Segurança da informação. 2. Segurança de sistemas. I. Maziero, Carlos Alberto. II. Gaspary, Luciano Paschoal. III. Weber, Raul Fernando. IV. Título.

Promoção

Sociedade Brasileira de Computação (SBC)

Diretoria

Presidente
José Carlos Maldonado (USP)

Vice-Presidente
Virgílio Augusto Fernandes Almeida (UFMG)

Diretora Administrativa
Carla Maria Dal Sasso Freitas (UFRGS)

Diretor de Finanças
Paulo Cesar Masiero (USP)

Diretor de Eventos e Comissões Especiais
Marcelo Walter (UFPE)

Diretor de Educação
Edson Norberto Cáceres (UFMS)

Diretora de Publicações
Karin Breitman (PUC-Rio)

Diretor de Planejamento e Programas Especiais
Augusto Cezar Alves Sampaio (UFPE)

Diretora de Secretarias Regionais
Aline Maria Santos Andrade (UFBA)

Diretor de Divulgação e Marketing
Altigran Soares da Silva (UFAM)

Diretor de Regulamentação da Profissão
Ricardo de Oliveira Anido (UNICAMP)

Diretor de Eventos Especiais
Carlos Eduardo Ferreira (USP)

Diretora de Cooperação com Sociedades Científicas
Taisy Silva Weber (UFRGS)

Realização

Comitê de Organização

Coordenação Geral
Luciano Paschoal Gaspary (UFRGS)
Raul Fernando Weber (UFRGS)

Coordenação do Comitê de Programa
André Luiz Moura dos Santos (UECE)
Marinho Pilla Barcellos (UFRGS/PUCRS)

Coordenação de Palestras e Tutoriais
Célio Vinicius Neves de Albuquerque (UFF)

Coordenação de Minicursos
Carlos Alberto Maziero (PUCPR)

Coordenador do Workshop de Trabalhos de Iniciação Científica e de Graduação
Raul Ceretta Nunes (UFSM)

Comitê Consultivo
André Luiz Moura dos Santos (UECE)
Carlos Alberto Maziero (PUCPR)
Joni da Silva Fraga (UFSC)
Jorge Nakahara Jr (MACKENZIE)
Luciano Paschoal Gaspary (UFRGS)
Luiz Fernando Rust da Costa Carmo (UFRJ)
Paulo Sérgio Licciardi Messeder Barreto (USP)
Ricardo Dahab (UNICAMP)

Organização Local
Alan Diego dos Santos (UFRGS)
Carlos Raniery Paula dos Santos (UFRGS)
Cristiano Bonato Both (UFRGS)
Cristina Melchiors (UFRGS)
Fabrício Girardi Andreis (UFRGS)
Flávio Roberto Santos (UFRGS)
Giovane César Moreira Moura (UFRGS)
Guilherme Sperb Machado (UFRGS)
Jéferson Campos Nobre (UFRGS)
Josué Klafke Sperb
Juliano Araújo Wickboldt (UFRGS)
Lourdes Tassinari (UFRGS)
Paulo Eduardo de Castro Teles Barbosa (UFRGS)
Rafael Kunst (UFRGS)
Rafael Santos Bezerra (UFRGS)
Rafael Vieira Coelho (UFRGS)
Roben Castagna Lunardi (UFRGS)
Vânia Regina Sávio Rodenas (UFRGS)
Weverton Luis da Costa Cordeiro (UFRGS)

Mensagem dos Coordenadores Gerais

O Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg) é um evento científico promovido anualmente pela Sociedade Brasileira de Computação (SBC). A partir de 2005, concomitantemente à criação da Comissão Especial em Segurança da Informação e de Sistemas Computacionais, o SBSeg deixou de ser organizado como um *workshop* e passou a ser um simpósio completo. Isso permitiu que, imediatamente, passasse a atender às demandas crescentes da comunidade brasileira de pesquisadores e profissionais atuantes na área e assumisse a posição de principal fórum no País para a apresentação de pesquisas e atividades relevantes ligadas à segurança da informação e de sistemas.

Desde que se estabeleceu como simpósio em 2005, o evento foi organizado, com grande sucesso, em Florianópolis, em Santos e na cidade do Rio de Janeiro. Nesta edição, o SBSeg ocorre de 1º a 5 de setembro na cidade de Gramado, RS, sob a organização do Instituto de Informática da Universidade Federal do Rio Grande do Sul (UFRGS). O simpósio contará com uma rica variedade de atividades, a saber: 6 sessões técnicas de artigos completos e resumos estendidos, 4 minicursos, 3 tutoriais e 4 palestras proferidos por especialistas brasileiros e estrangeiros, painel e Workshop de Trabalhos de Iniciação Científica e de Graduação.

Um aspecto fundamental do SBSeg é o comprometimento com a *qualidade*. Tem operado seguindo, rigorosamente, indicadores visando ao atendimento do padrão Qualis A, conforme critérios da CAPES. Entre esses critérios, destacamos a taxa de aceitação de artigos completos inferior a 35% e a composição de Comitês de Programa por pesquisadores brasileiros e estrangeiros com grande renome e inserção na área. Coroando esse trabalho de anos, em 2008 o SBSeg recebeu, pela primeira vez, o apoio técnico da *International Federation for Information Processing* (IFIP), por meio do *Technical Committee on Security and Protection in Information Systems* (TC11).

Para a realização do SBSeg´08, o envolvimento e a colaboração de várias pessoas e entidades foram imprescindíveis. Em especial, gostaríamos de agradecer aos membros do comitê de organização geral e local que, por conta de seu trabalho voluntário e incansável, ajudaram a proporcionar à comunidade de segurança um evento que julgamos de ótimo nível técnico. Gostaríamos de agradecer, também, à SBC, pelo apoio prestado ao longo das muitas etapas da organização, e aos patrocinadores, pelo incentivo à divulgação de atividades de pesquisa conduzidas no País e pela confiança depositada neste fórum. Por fim, nossos agradecimentos ao Instituto de Informática da UFRGS, por viabilizar a realização de um evento do porte do SBSeg.

Durante a semana de 1º a 5 de setembro estarão reunidos em Gramado estudantes, professores, pesquisadores e profissionais da indústria, todos com o objetivo de trocar idéias, compartilhar experiências e estabelecer laços pessoais. Gramado será, portanto, o centro da discussão sobre avanços realizados e desafios a serem superados na área de segurança da informação e de sistemas. Sejam bem-vindos à Serra Gaúcha e desfrutem de uma semana agradável e proveitosa!

Prof. Luciano Paschoal Gaspary
Prof. Raul Fernando Weber
Coordenadores Gerais do SBSeg´08

Mensagem do Coordenador de Minicursos

Os minicursos vêm sendo oferecidos no SBSeg desde 2005, como contribuição importante do evento para a formação de estudantes em temas relevantes em segurança da informação e de sistemas, mas que normalmente não são cobertos pelas grades curriculares de graduação. Para esta quarta edição da chamada de minicursos, foram submetidas 15 propostas, a grande maioria de boa qualidade técnico-científica e versando sobre temas atuais de interesse da comunidade. Cada proposta submetida foi detalhadamente avaliada por, pelo menos, três membros do comitê de avaliação. Por fim, foram selecionadas as quatro propostas que melhor satisfizeram os requisitos de qualidade técnica, interesse da comunidade e experiência dos proponentes.

Em nome do comitê de avaliação de minicursos, expresso nossos agradecimentos a todos os autores que submeteram propostas de minicursos, pelo interesse demonstrado no evento. É o interesse contínuo e comprometido da comunidade que mantém acesa a chama deste evento. O trabalho de organização de um evento do porte do SBSeg é considerável. Por isso, agradecemos aos profs. Luciano Paschoal Gaspary e Raul Fernando Weber, coordenadores gerais do evento, pelo convite para organizar a chamada de minicursos e pela confiança em nós depositada. Nosso reconhecimento ao primoroso trabalho desenvolvido pelos membros do comitê de avaliação de minicursos, cuja percepção de qualidade foi fundamental para a concretização deste livro. Nosso agradecimento, também, à comissão de organização, pelo excelente trabalho realizado. Por fim, é fundamental agradecer a todas as agências de fomento e demais patrocinadores que tornam este evento possível.

A todos os participantes, um ótimo e proveitoso evento!

Prof. Carlos Alberto Maziero
Coordenador de Minicuros

Comitê de Avaliação de Minicursos

Altair Olivo Santin (PUCPR)
Carlos Alberto Maziero (PUCPR)
Flávia Coimbra Delicato (UFRN)
Jeroen van de Graaf (UFMG)
Jorge Nakahara Jr (MACKENZIE)
Lau Cheuk Lung (UFSC)
Luci Pirmez (UFRJ)
Luiz Fernando Rust da Costa Carmo (UFRJ)
Michelle Silva Wangham (UNIVALI)
Ricardo Dahab (UNICAMP)
Rossana Maria de Castro Andrade (UFC)
Routo Terada (USP)
Thais Vasconcelos Batista (UFRN)

Sumário

Capítulo 1

Gestão de Riscos

José Eduardo Malta de Sá Brandão[1], Joni da Silva Fraga[2]

[1]Instituto de Pesquisa Econômica Aplicada (IPEA)
Brasília DF

[2]Departamento de Automação e Sistemas
Universidade Federal de Santa Catarina
Florianópolis SC

Abstract

The risks management process is based on the best practices principles of management and security, to assist in strategic decisions. It involves an organized and recursive process of documentation, evaluation and decision at all project life cycle steps. The objective of this course is to present the security risks management process, with its advantages and limitations. It includes the theoretical and practical aspects of the risks management, to create a knowledge base for the reader's own projects. We expects in the present course to contribute to the knowledge of the security risk management methodology.

Resumo

A gestão de riscos baseia-se em princípios e boas práticas de gerenciamento e segurança, para auxiliar na tomada de decisões estratégicas. Ela envolve um processo organizado e recursivo de documentação, avaliação e decisão durante todas as fases do ciclo de vida do projeto. O objetivo desse curso é apresentar a disciplina de gestão de riscos de segurança, bem como suas vantagens e limitações. Nesse curso são apresentados aspectos teóricos e práticos da gestão de riscos, fornecendo aos alunos uma base para desenvolverem seus próprios projetos. Espera-se, portanto, com esse curso, contribuir para um melhor entendimento da metodologia de gestão de riscos.

1.1. Introdução

1.1.1. Motivação

Quando um sistema computacional apresenta vulnerabilidades que podem ser exploradas, produzindo algum impacto negativo, afirmamos que tal sistema está em risco. Contudo, quantificar ou até mesmo identificar os riscos não é uma tarefa trivial.

A noção correta dos riscos permite que se definam caminhos e ferramentas para mitigá-los. Infelizmente, Os riscos podem ser identificados e reduzidos, mas nunca totalmente eliminados [Garfinkel et al. 2003].

É comum a aplicação de ferramentas de análise de risco em protótipos desenvolvidos em projetos de software científicos ou comerciais. Isso ocorre após a conclusão de uma versão do desenvolvimento do protótipo, com a finalidade de identificar vulnerabilidades.

Em alguns sistemas computacionais, as vulnerabilidades encontradas podem ser inerentes à tecnologia adotada. Sendo assim, a mitigação dos riscos pode incluir a troca da tecnologia, com a inevitável onerosidade do projeto, ou a aceitação de um risco maior do que o desejado. Quando um projeto de software envolve a integração de múltiplas tecnologias, a identificação e a minimização dos riscos são ainda mais complexas. Tratar os riscos deste tipo de projeto apenas no ponto de protótipo pode ser extremamente dispendioso e, em alguns casos, os resultados podem inviabilizar o próprio projeto. Muitas vezes as vulnerabilidades encontradas poderiam ter sido facilmente identificadas na etapa de planejamento do projeto.

Os resultados esperados, com este minicurso, são um melhor entendimento destas metodologias de gestão de riscos e a difusão da idéia da necessidade da avaliação dos riscos associados aos sistemas computacionais, via estes testes padronizados.

1.1.2. Conceitos

Diversos conceitos tratados nesse curso são descritos nessa seção. Tais conceitos são essenciais para a compreensão dos modelos e padrões associados à gestão de riscos. Entre esses conceitos podemos destacar aqueles relacionados à segurança em tecnologia da informação (TI) e as definições de riscos de segurança.

1.1.2.1. Propriedades de Segurança

Segurança em tecnologia da informação 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 and 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.

Nesse curso adotamos os objetivos da segurança definidos pelo NIST[1] (*National Institute of Standards and Technology*), que preconizam a preservação da integridade, da disponibilidade e da confidencialidade dos recursos dos sistemas de informações, incluindo: *hardware*, *software*, *firmware*, informação/dados e telecomunicações [Ross et al. 2005].

1.1.2.2. Violações de Segurança

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).

1.1.2.3. 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].

1.1.2.4. 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 and 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**.

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

1.1.2.5. 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 impacto do mesmo.

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.

1.1.2.6. Gestão de Riscos

A gestão de riscos ultrapassa a análise de vulnerabilidades e riscos de um produto ou protótipo. A gestão de riscos baseia-se em atividades coordenadas para direcionar e controlar uma organização no que se refere a riscos [ISO 2002]. A mesma envolve um processo criterioso e recursivo de documentação, avaliação e decisão durante todas as fases do ciclo de vida do projeto

1.1.3. Foco do Curso

O objetivo desse curso é apresentar a disciplina de gestão de riscos de segurança, bem como suas vantagens e limitações. O enfoque do curso visa elucidar aspectos conceituais e sistemáticos nestas metodologias, porém, um estudo de caso também enfatizará a aplicabilidade das técnicas apresentadas. Esse curso deverá fornecer aos alunos uma base para desenvolverem seus próprios projetos segundo a ótica da gestão de riscos.

A seção seguinte está focada nos principais conceitos, na descrição de modelos e na comparação dos principais padrões relacionados à gestão de riscos na segurança. Associada a estes modelos e padrões, na terceira seção é apresentada uma metodologia para auxiliar na análise quantitativa dos riscos. Um estudo de caso que visa reforçar a utilidade e necessidade do uso destas metodologias é apresentado na quarta seção. Este estudo de caso envolve um projeto na área de segurança de sistemas, guiado por estas metodologias de gestão de riscos. Finalmente, na última seção são apresentadas algumas considerações finais sobre o emprego da metodologia de gestão de riscos.

1.2. Principais Padrões Relacionados à Gestão de Riscos

Os governos e a sociedade estão cada vez mais preocupados com o que pode acontecer com eventuais perdas de dados, furto de informações e até mesmo com a perda de vidas ocasionadas por possíveis falhas em sistemas computacionais. Por isso, a segurança de sistemas de TI vem sendo foco de diversas organizações voltadas à recomendação de padrões e metodologias. Entre estas organizações podemos destacar a ISO[2](*International Organization for Standardization*), o NIST[3](*National Institute of Standards and Techno-*

[2] http://www.iso.org
[3] http://www.nist.gov

logy), a BSi[4](British Standards) e a AS/NZS[5](*Australian/New Zealand Standard*). No Brasil, a ABNT[6](Associação Brasileira de Normas Técnicas) é a responsável pela recomendação dos padrões técnicos.

As principais recomendações de segurança reforçam a adoção de boas práticas de gerenciamento de sistemas de TI. As recomendações de segurança mais conhecidas são as da série BS 7799 do BSi[BSi 1999][BSi 2002]. Desenvolvidas pelo governo Britânico, essas recomendações foram referências na definição de boas práticas de gestão de segurança em sistemas de informação. Posteriormente, os dois primeiros documentos da série BS 7799 foram revistos e reorganizados na série ISO/IEC 17799.

Atualmente, há um novo esforço de revisão dos documentos de segurança da informação, que estão sendo reclassificados na série ISO/IEC 27000. O objetivo deste esforço é o alinhamento das normas de gestão da segurança da informação às normas das famílias 9000 e 14000 da ISO. A Figura 1.1 mostra a relação e a evolução das normas de segurança.

A norma ISO 27000 está em desenvolvimento e irá definir os conceitos fundamentais e o vocabulário de segurança da informação adotado nesta série de documentos. A terminologia adotada na maioria das normas de segurança da informação é baseada no guia 73[ISO 2002], que está sendo revisto.

As normas ISO 27001[ABNT 2006b] e 27002[ABNT 2006a] foram baseadas nas normas BS 17799-2 [BSi 2002] e BS/ISO 17799-1[BS/ISO 2005], respectivamente. A recomendação ISO 27001 foi preparada para prover um modelo para estabelecer, implementar, operar, monitorar, analisar criticamente, manter e melhorar um Sistema de Gestão de Segurança da Informação.

A norma ISO 27002 introduz os conceitos de segurança da informação e faz uma discussão inicial a respeito das motivações para o estabelecimento da gestão de segurança. Na maior parte do documento são detalhadas as práticas de segurança, que são associadas aos os objetivos de controles, e os controles de segurança citados na norma ISO 27001.

A norma ISO 27003, em desenvolvimento, é baseada no anexo B da norma BS 7799-2, sendo basicamente um guia para a implantação do Sistema de Gestão de Segurança da Informação (SGSI) apresentado nas duas normas anteriores.

Em geral, as técnicas de análise de riscos são focadas em testes ou são analíticas. 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 seja 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.

[4]http://www.bsi-global.com
[5]http://www.standards.org.au
[6]http://www.abnt.org.br

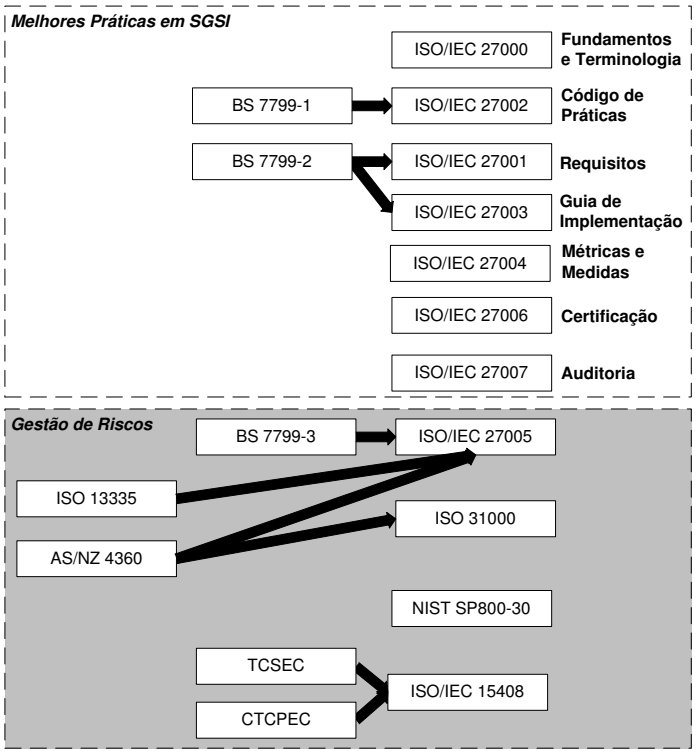

Figura 1.1. Relação entre as normas

Alguns esforços para padronização de metodologias para avaliação de sistemas de tecnologia da informação vêm sendo realizados. 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.

A gestão de riscos baseia-se em princípios e boas práticas de gerenciamento e segurança [Swanson and Guttman 1996], para auxiliar na tomada de decisões. Entre as ferramentas metodológicas disponíveis para o desenvolvimento da gestão de riscos, destacamos a especificação SP800-30 [Stoneburner et al. 2002] desenvolvida pelo NIST, a especificação AS/NZ4360 [AS/NZS 2004a] desenvolvida pelos governos da Austrália e Nova Zelândia, a norma ISO/IEC 27005[ABNT 2008] e a proposta de norma ISO 31000[ISO 2007].

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

1.2.1. Melhores Práticas

A norma ISO 27001 adota o modelo conhecido como PDCA (*Plan-Do-Check-Act*), ilustrado na figura 1.2, que é aplicado para estruturar todos os processos do Sistema de Gestão de Segurança da Informação (SGSI).

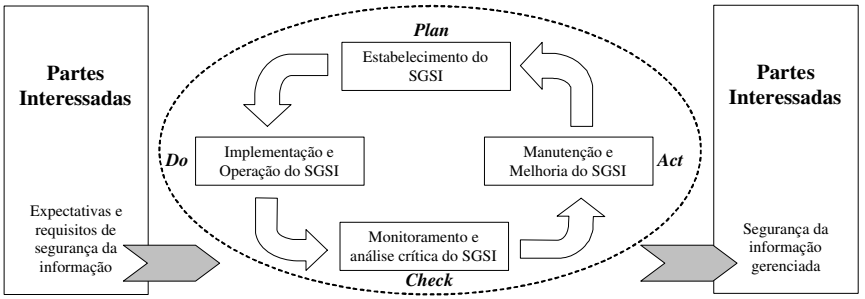

Figura 1.2. Modelo PCDA

O processo do PDCA inicia quando as partes interessadas definem os requisitos e as expectativas de segurança. Em seguida, é iniciado um procedimento cíclico de gestão, composto de quatro etapas complementares.

O ciclo começa com o estabelecimento da política, dos objetivos, dos processos e dos procedimentos do SGSI (Plan), que sejam relevantes para a gestão de riscos e a melhoria da segurança da informação e que produzam resultados de acordo com as políticas e objetivos globais de uma organização.

A segunda etapa (Do) envolve a implantação e a operação da política, dos controles, dos processos e dos procedimentos estabelecidos na primeira etapa.

Na terceira fase (Check) é feita a avaliação e, quando aplicável, a medição do desempenho de um processo frente à política, aos objetivos e à experiência prática do SGSI, apresentando os resultados para a análise crítica pela direção.

No quarto passo (Act) cabe a execução das ações corretivas e preventivas, com base nos resultados da auditoria interna do SGSI e da análise crítica pela direção ou outra informação pertinente, para alcançar a melhoria contínua do SGSI. Após esta etapa, o ciclo é reiniciado, tomando como base o aprendizado do ciclo anterior. O resultado esperado da adoção do PDCA é a segurança da informação devidamente gerenciada.

A norma estabelece ainda as diretrizes para uma auditoria no SGSI, definindo os objetivos de controles e os controles necessários para o gerenciamento de segurança de diversos aspectos de um ambiente de TI.

1.2.2. 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, criado pelo go-

verno canadense; e o ITSEC, desenvolvido pelos países europeus. Lançado 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 por meio 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.

1.2.3. 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 duas etapas: avaliação de riscos (ou determinação dos riscos) e atenuação de riscos. Além destas duas etapas, a revisão periódica de todo o processo é recomendada. O processo de avaliação de riscos segue um fluxo de atividades, conforme ilustrado na Figura 1.3.

O primeiro passo da metodologia consiste em caracterizar o sistema implementado, descrevendo o ambiente ao qual está vinculado, a sua missão, os seus requisitos

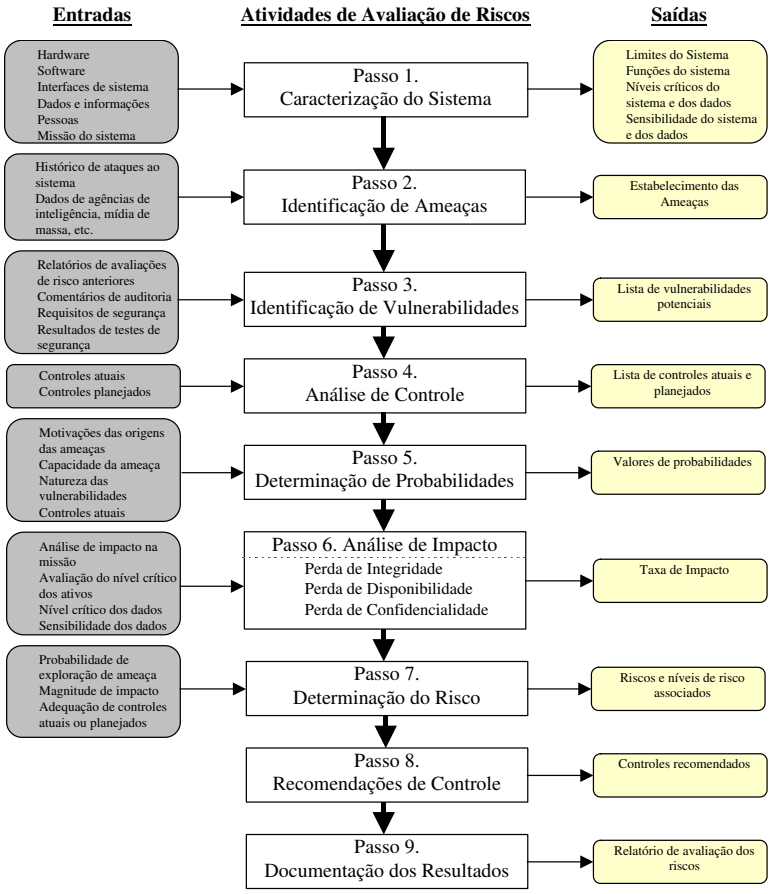

Figura 1.3. Metodologia de gestão de riscos da especificação SP800-30 [Stoneburner et al. 2002]

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 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.

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 inexeqüível 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.

1.2.4. AS-NZ4360, ISO 27005 e ISO 31000

A norma AS/NZ4360 [AS/NZS 2004a] serviu de referência para o desenvolvimento de todas as demais normas de gestão de riscos que a sucederam. A norma ISO 3100, por

exemplo, segue o mesmo processo e possui os mesmos elementos. Já a norma ISO 27005 traz pequenas alterações no processo.

As fases de identificação, análise e avaliação dos riscos da especificação AS/NZ 4360 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 normas de gestão de risco que a antecederam 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. Seguindo essa mesma linha abrangente, está a norma ISO 3100. Por isso, a similaridade entre as duas é inevitável.

As normas AS/NZ4360 e ISO 3100 definem o processo de gestão de riscos por meio de 7 elementos (ou fases) principais, conforme ilustrado na Figura 1.4: comunicar e consultar; estabelecer o contexto; identificar os riscos; analisar os riscos; avaliar os riscos; tratar os riscos; e monitorar e rever.

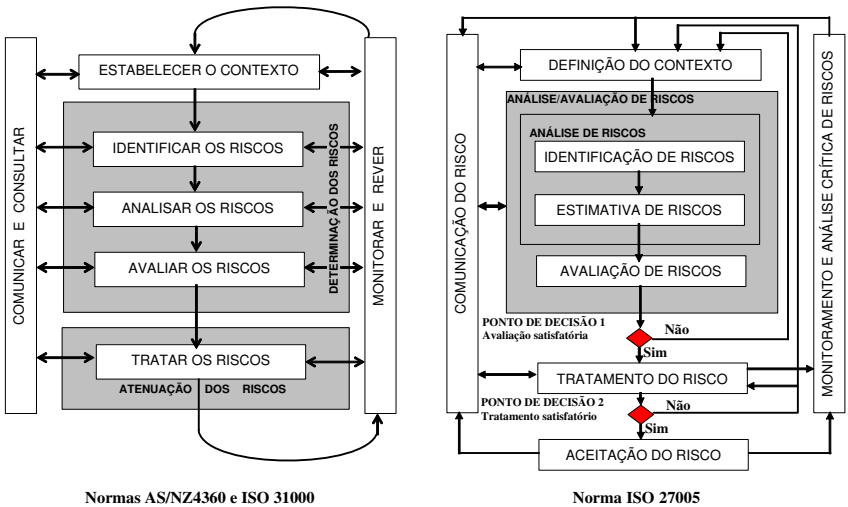

Figura 1.4. Visão Geral do Processo de Gestão de Riscos

Como pode ser observado na Figura 1.4, o processo da norma ISO 27005 traz uma pequena variação dos termos. São indicados dois pontos de decisão nos quais o processo pode ser revisto e uma nova fase de aceitação do risco. Uma contribuição significativa ao processo de gestão de riscos foi feita com a inclusão nos anexos de exemplos e métodos para a identificação de ameaças, além de modelos de análise e avaliação dos riscos.

Outra diferença é o alinhamento com a norma ISO 27001, principalmente em relação ao modelo PDCA. Dentro do modelo PDCA, as etapas da gestão de riscos são divididas nas quatro fases, conforme ilustrado na Figura 1.5. Na fase de planejamento (*Plan*) são agrupadas as etapas: de Definição do Contexto, de Análise/Avaliação de Ris-

cos, de Definição do Plano de Tratamento do Risco e de Aceitação do Risco. Na fase de Execução (*Do*) é realizada a implantação do plano de Tratamento do Risco. Na verificação (*Check*) é feito o Monitoramento Contínuo e Análise Crítica do Risco. Finalmente, a Ação (*Act*) envolve manter e melhorar o processo de Gestão de Riscos de Segurança da Informação.

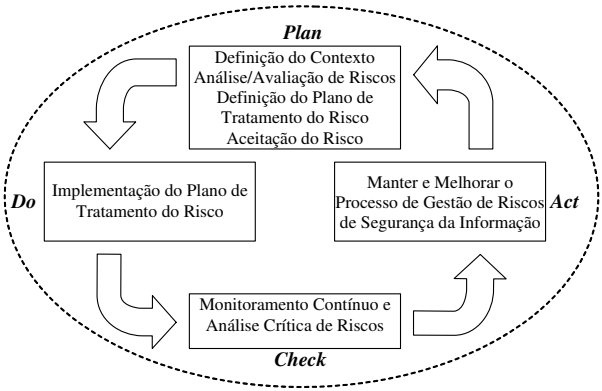

Figura 1.5. Alinhamento do processo do SGSI com o processo de Gestão de Riscos

Cada uma das fases da norma ISO 27005 será apresentada a seguir. Cada fase segue o modelo de procedimento da norma ISO 27001.

1.2.4.1. Comunicação do Risco

A gestão de riscos pode ter diversas partes interessadas. Estas partes devem ser identificadas e seus papéis e responsabilidades delimitados na fase de **Comunicação do Risco**. 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.

1.2.4.2. Definição do Contexto

A etapa de **Definição do Contexto** define os parâmetros básicos, por meio 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, avaliação, impacto e aceitação dos riscos.

A definição do contexto tem como entrada todas as informações relevantes sobre a organização, que sejam relevantes para a definição do contexto da gestão de riscos de segurança.

O passo principal 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.

1.2.4.3. Identificação de Riscos

A **Identificação de Riscos** é uma das etapas mais críticas, pois os riscos não identificados não serão analisados nem tratados. O objetivo dessa etapa é determinar os eventos que possam causar perdas potenciais e deixar claro como, onde e por que a perda pode ocorrer. 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.

A identificação dos riscos envolve a identificação das ameaças, dos controles existentes, das vulnerabilidades e das conseqüências. Para realizar estas tarefas é desejável uma ampla revisão bibliográfica sobre o assunto, incluindo os objetivos do projeto. 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.

Em projetos científicos, as revisões obtidas e a apresentação pública do projeto em congressos científicos também auxiliam na identificação e revisão dos riscos.

1.2.4.4. Estimativa de Riscos

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

Uma metodologia de estimativa pode ser qualitativa ou quantitativa ou uma combinação de ambas, dependendo das circunstâncias. A estimativa qualitativa utiliza uma escala com atributos qualificadores que descrevem a magnitude das potenciais conseqüências e a probabilidade destas conseqüências ocorrerem. A estimativa quantitativa adota uma escala de valores numéricos tanto para conseqüências, quanto para a probabilidade. Na próxima seção será apresentada uma metodologia que combina os dois tipos de estimativas.

1.2.4.5. Avaliação de Riscos

O objetivo da **Avaliação de Riscos** é tomar decisões, baseadas nos resultados da análise de risco (Identificação e Estimativa de Riscos). 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.

Convém que sejam consideradas as propriedades da segurança da informação e a importância do processo de negócio ou a atividade suportada por um determinado ativo ou conjunto de ativos. No caso de projetos científicos, a avaliação dos riscos deve estar alinhada com os objetivos do projeto.

Ao término da avaliação é verificado se seu resultado é satisfatório. Caso não seja, uma nova rodada da Análise/Avaliação dos riscos deve ser empreendida, a partir da revisão e possível redefinição do contexto.

1.2.4.6. Tratamento do Risco

O **Tratamento do Risco** 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 tratamento inicia com uma lista de riscos ordenados por prioridade, conforme os critérios de avaliação dos riscos, associados aos possíveis cenários de incidentes que os provocam.

A cada um dos riscos são relacionadas as opções de tratamento. Há quatro opções possíveis para o tratamento do risco: redução, retenção, evitação e transferência.

Ao término do tratamento são identificados os riscos residuais. Se tais riscos não forem aceitáveis, os tratamentos podem ser refeitos ou, ainda, todo o processo de gestão de riscos pode ser revisto.

1.2.4.7. Aceitação do Risco

A aceitação do risco é feita a partir da análise do risco residual. É conveniente que a decisão de aceitar os riscos seja formalmente registrada, junto com a responsabilidade pela decisão. Infelizmente, nem sempre os riscos residuais estão de acordo com o idealizado no início do processo. Porém, fatores como tempo e custos podem justificar a aceitação dos riscos.

1.2.4.8. Monitoramento e Análise Crítica dos Riscos

O **Monitoramento e Análise Crítica dos Riscos** são partes essenciais da gestão de riscos. Os riscos não são estáticos e 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.

1.3. Estimando os Riscos

Todo risco tem um custo e este custo pode ser quantificado de forma mais ou menos precisa [Blakley et al. 2001]. A mensuração de riscos é bastante comum na área econômica. Porém, pouco adotada na segurança de sistemas de tecnologia da informação.

Conforme definido na primeira seção, o risco pode ser representado matematicamente como uma função da probabilidade de uma origem de ameaça explorar uma vulnerabilidade potencial e o impacto resultante deste evento adverso.

A probabilidade de um evento ocorrer durante um período de tempo determinado é expressa por um número entre zero e um. Quanto maior for o período de tempo considerado, maior será a probabilidade.

Em geral, em um sistema com múltiplas vulnerabilidades, para fins de cálculo, cada vulnerabilidade é considerada como um evento discreto. Ou seja, a probabilidade de um evento ocorrer para determinada vulnerabilidade independe da ocorrência ou não de outro evento.

As probabilidades normalmente são calculadas por meio de análises de dados de ataques ou ameaças. Tais dados podem ser obtidos por experiências na própria empresa ou por coletâneas adquiridas de organizações especializadas.

A norma ISO 27005 traz de forma didática algumas abordagens para a estimativa dos riscos de segurança da informação. Estas abordagens se assemelham às que são sugeridas pelas normas AS/NZ4360 e NIST SP800-30, apresentadas na seção anterior. Apesar das excelentes recomendações contidas nestes documentos, não é apresentada uma metodologia objetiva que possa, de fato, auxiliar o gestor de segurança na tarefa de identificar e mensurar os riscos em um ambiente real.

Para o cálculo do impacto de possíveis vulnerabilidades, é aconselhável a adoção de uma metodologia padronizada e conhecida, como o *Common Vulnerability Scoring System* (CVSS)[7]. Essa metodologia permite ao gestor de segurança da informação calcular os riscos que uma vulnerabilidade inflige no ambiente real da empresa, sem que sejam necessários dados estatísticos precisos sobre ataques anteriores ou análises financeiras complexas. Basta para isso haver um inventário atualizado dos ativos, sistemas e serviços de TI. Tal metodologia será detalhada a seguir.

[7] http://www.first.org/cvss/

1.3.1. Common Vulnerability Scoring System (CVSS)

O CVSS é adotado pelo NIST para a classificação de vulnerabilidades no *National Vulnerability Database* (NVD)[8]. A base de dados de vulnerabilidades NVD é integrada *ao Common Vulnerabilities and Exposures* (CVE)[9] [Mell and 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: métricas básicas, métricas temporais e métricas ambientais. As métricas básicas contêm todas as características que são intrínsecas e fundamentais para determinada vulnerabilidade e que são invariáveis ao longo do tempo ou em ambientes diferentes. As métricas temporais contêm as características que podem mudar ao longo do tempo. No grupo da métricas ambientais estão as características que são atreladas a implementações e ao ambiente. As características são valoradas e processadas 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[10] das vulnerabilidades baseado nesta pontuação: Baixo (valor entre 0 e 3), Médio (valor entre 4 e 7) e Alto (valor acima de 7).

1.3.1.1. Métricas Básicas

O grupo base de características é composto de seis critérios, conforme ilustrado na Figura 1.6. Estes critérios estão divididos em dois grupos: Impacto e Complexidade. O grupo do Impacto é composto pelo impacto na confidencialidade (CI), impacto na integridade (II) e impacto na disponibilidade (AI).

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.

O Impacto é calculado de acordo com a seguinte fórmula:

$$\textbf{Impacto} = 10{,}41 * (1 - (1 - CI) * (1 - II) * (1 - AI))$$

No grupo da Complexidade, há outros três critérios. O **vetor de acesso (AV)** identifica se a vulnerabilidade pode ser explorada tanto localmente, quanto por redes remotas (classificação Rede Remota); se pode ser explorada localmente e por uma rede adjacente, mas não de uma rede remota (Rede Adjacente); ou ainda se requer acesso físico ou *login* autenticado no sistema alvo (classificação Local). A métrica de **autenticação (AU)**

[8] http://nvd.nist.gov/
[9] http://cve.mitre.org/
[10] http://nvd.nist.gov/cvss.cfm

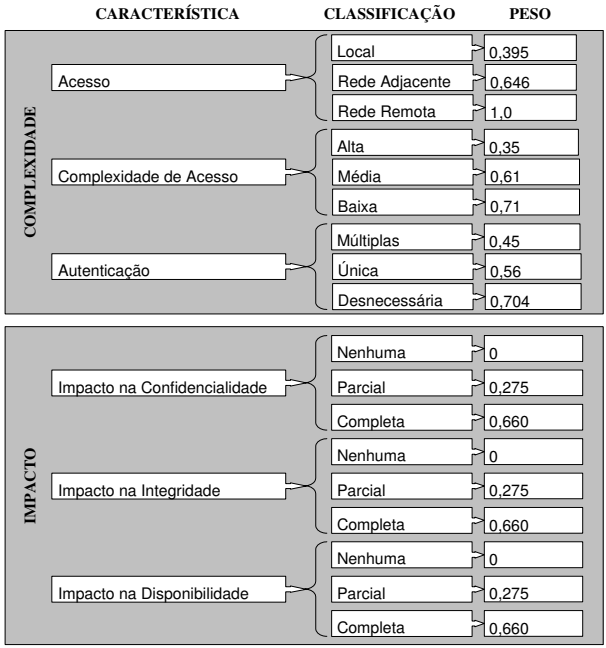

Figura 1.6. Métricas Básicas, conforme o CVSS

verifica o tipo de autenticação necessária pelo atacante no sistema alvo para que a vulnerabilidade seja explorada. A métrica pode ser classificada como: múltipla, se o atacante precisa se autenticar mais de uma vez; autenticação única; ou autenticação desnecessária.

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 somente algumas circustâncias são necessárias para a exploração da vulnerabilidade, a complexidade é média. Se não existem circunstâncias especiais, a complexidade é baixa.

A Complexidade é calculada de acordo com a seguinte fórmula:

$$\textbf{Complexidade} = 20 * AC * AU * AV$$

Para calcular a pontuação, é utilizada a seguinte fórmula sobre os valores atribuídos a cada vulnerabilidade, de acordo com suas características:

$$\textbf{Risco Básico} = (0{,}6 * \text{Impacto} + 0{,}4 * \text{Complexidade} - 1{,}5) * f(\text{Impacto})$$

O Fator de Impacto, **f(Impacto)**, terá o valor 0 (zero) se o Impacto for igual a zero. Caso contrário, receberá o valor 1,176.

1.3.1.2. Métricas Temporais

O grupo temporal é formado por três variáveis: **Explorabilidade (EX)**, o **Nível de Remediação (RL)** e o **Grau de Confiança (RC)**. A Figura 1.7 ilustra estas variáveis e seus pesos.

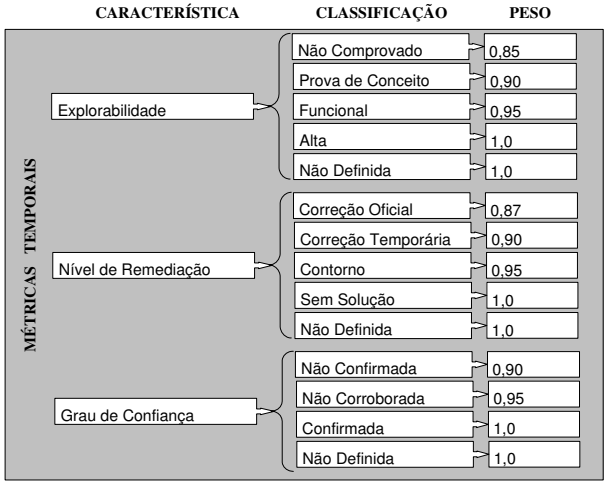

Figura 1.7. Métricas Temporais, conforme o CVSS

A Explorabilidade indica se é ou não possível explorar a vulnerabilidade. Essa variável pode ser classificada como: Não Comprovada, se não há uma ferramenta de exploração (*exploit*) conhecida; Prova de Conceito, se foi criada uma prova da vulnerabilidade; Funcional, quando há um *exploit* desenvolvido; ou Alta, se há relatos de exploração.

No Nível de Remediação é informado se há tratamentos conhecidos para a vulnerabilidade. Tais tratamentos podem ser: uma Correção Oficial do fabricante; uma Correção Temporária fornecida pelo fabricante; um Contorno ao problema; ou Sem Solução disponível.

A credibilidade da divulgação da vulnerabilidade é refletida no Grau de Confiança da existência da mesma. Essa confiança pode ser representada como Não Confirmada, quando há um único relato; Não Corroborada, quando há vários relatos não oficiais; ou Confirmada, quando é reconhecida pelo fabricante.

Caso alguma métrica temporal não seja adotada no cálculo, deve ser classificada como Não Definida.

O cálculo do risco temporal é medido pela seguinte fórmula:

Risco Temporal = Risco Básico * EX * RL * RC

1.3.1.3. Métricas Ambientais

Nem todas as vulnerabilidades são potencialmente perigosas para uma empresa. Para identificar possíveis riscos, é necessário verificar se a tecnologia vulnerável é adotada e quais os possíveis danos que a exploração da vulnerabilidade pode causar. As métricas ambientais têm por objetivo calcular o impacto da vulnerabilidade no ambiente da empresa. A Figura 1.8 ilustra as variáveis ambientais e seus pesos.

Figura 1.8. Métricas Ambientais, conforme o CVSS

A primeira métrica ambiental mede o **Potencial Dano Colateral (CD)** da exploração da vulnerabilidade, representando o risco de danos físicos a ativos ou a perda de vidas. Os danos são classificados em cinco graus diferentes, podendo ser: Nenhum; Baixo, se há danos físicos, perda de lucros ou de produtividade leves; de Baixo a Médio, com danos físicos, perda de lucros ou de produtividade moderados; de Médio a Alto, se houver danos físicos, perda de lucros ou de produtividade significativos; Alto, quando há a possibilidade de danos físicos, perda de lucros ou de produtividade catastróficos.

A segunda métrica indica a **Distribuição dos Alvos (TD)** que podem ser afetados, em termos percentuais, no ambiente empresarial, de acordo com a seguinte classificação: Baixo, entre 1% e 25%; Média, entre 26% e 75%; Alta, acima de 75%; e Nenhum, se não houver sistemas suscetíveis à vulnerabilidade ou estão restritos a ambientes de laboratório muito específicos.

Há ainda outras três métricas, que customizam os cálculos, de acordo com a importância, para a empresa, dos ativos afetados, em relação aos **Requisitos de Segurança**. Essa importância é medida em termos do **Requisito de Confidencialidade (CR)**, do **Requisito de Integridade (IR)** e do **Requisito de Disponibilidade (AR)**. O impacto da perda de cada um dos requisitos pode ser classificada como Baixa, Média ou Alta.

Caso alguma métrica ambiental não seja adotada no cálculo, deve ser classificada como Não Definida.

Se aplicada, a métrica ambiental irá ser combinada com a métrica temporal para calcular o Risco Ambiental. Para isso, o Ajuste Temporal deve ser buscado. O Ajuste Temporal é obtido, conciliando o impacto de cada requisito de segurança ao ambiente, usando a seguintes fórmulas:

Ajuste Temporal = Risco Temporal recalculado, substituindo o Risco Básico pelo Impacto Ajustado

Impacto Ajustado = min (10, 10,41 * (1 - (1 - CI*CR) * (1 - II*IR) * (1 - AI*AR)))

Finalmente, o Risco Ambiental será medido pela seguinte fórmula:

Risco Ambiental = ((Ajuste Temporal + (10 - Ajuste Temporal) * CD) * TD)

1.3.2. Exemplo de Aplicação dos Cálculos

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 vulnerabilidades listadas. Para a determinação do nível de risco basta pesquisar pelo produto no CVSS.

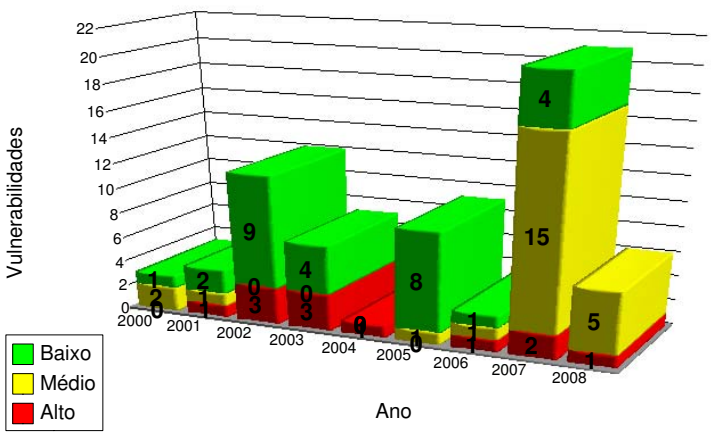

Figura 1.9. Níveis de risco das vulnerabilidades do servidor Tomcat

Tomando como exemplo a quantidade de vulnerabilidades registradas no NVD para o servidor Web *Tomcat*, foi formulado o gráfico da Figura 1.9, refletindo as características temporais de quando foram classificadas pelo NIST. No nível de risco fornecido

pelo NVD, as métricas temporais são consideradas como "Não Definidas"e não é computado o impacto ambiental.

Usando ainda como exemplo o servidor Web *Tomcat*, localizamos a vulnerabilidade *CVE-2008-1947*[11], reportada em 04/06/2008. Essa vulnerabilidade, se explorada, permite a injeção de código malicioso por meio de um parâmetro.

A vulnerabilidade em questão, possui as seguintes características de risco:

- Vetor de Acesso (AV) = Remota = 1,0
- Complexidade de Acesso (AC) = Média = 0,61
- Autenticação (AU) = Desnecessária = 0,704
- Impacto na Confidencialidade (CI) = Nenhuma = 0
- Impacto na Integridade (II) = Parcial = 0,275
- Impacto na Disponibilidade (AI) = Nenhuma = 0
- Métricas Temporais = Não Definidas = 1
- Potencial Dano Colateral (CD) = Baixo = 0,1
- Distribuição dos Alvos (TD) = 5% = 0,25
- Requisito de Disponibilidade (AR) = Alta = 1,51
- Requisito de Integridade (IR) = Alta = 1,51
- Requisito de Confidencialidade (CR) = Alta = 1,51

Calculando o Impacto:

$$\textbf{Impacto} = 10{,}41 * (1 - (1 - CI) * (1 - II) * (1 - AI)) =$$
$$10{,}41 * (1 - (1 - 0) * (1 - 0{,}275) * (1 - 0)) = 2{,}9$$

Calculando a Complexidade:

$$\textbf{Complexidade} = 20 * AC * AU * AV =$$
$$20 * 1 * 0{,}61 * 0{,}704 = 8{,}6$$

Finalmente pode ser calculado o nível de Risco Básico, usando o fator de impacto de valor 1,176:

$$\textbf{Risco Básico} = (0{,}6 * \text{Impacto} + 0{,}4 * \text{Complexidade} - 1{,}5) * f(\text{Impacto}) =$$
$$(0{,}6 * 2{,}86 + 0{,}4 * 8{,}59 - 1{,}5) * 1{,}176 = 4{,}3$$

O valor **4,3** corresponde ao Risco Básico **Médio**.

Para ajustar os riscos ao ambiente da empresa, calcula-se então o Impacto Ajustado e o Risco Básico Ajustado:

[11] http://nvd.nist.gov/nvd.cfm?cvename=CVE-2008-1947

Impacto Ajustado = min(10, 10,41 * (1 - (1 - CI*CR) * (1 - II*IR) * (1 - AI*AR))) =
min(10, 10,41 * (1 - (1 - 0 * 1,51) * (1 - 0,275 * 1,51) * (1 - 0 * 1,51))) = 4,3

Risco Básico Ajustado = (0,6 * 4,3 + 0,4 * 8,59 - 1,5) * 1,176 = 5,3

Com as variáveis da métrica temporal Não Definidas, o Risco Temporal e o Risco Temporal Ajustado são iguais:

Risco Temporal = Risco Básico * EX * RL * RC =
5,3 * 1 * 1 * 1 = 5,3

Resta agora calcular o Risco Ambiental:

Risco Ambiental = ((Ajuste Temporal + (10 - Ajuste Temporal) * CD) * TD) =
((5,3 + (10 - 5,3) * 0,1) * 0,25) = 1,4

Como se pode perceber, apesar do Risco Básico ser alto, quando a vulnerabilidade é avaliada no ambiente da empresa, a mesma recebe uma pontuação baixa. Contudo, o inverso também pode ocorrer. Uma vulnerabilidade com Risco Básico baixo pode afetar elementos críticos da empresa e receber uma pontuação de Risco Ambiental alta.

1.3.3. Considerações Sobre o CVSS

O CVSS sofre atualizações periódicas, tanto na classificação das métricas, quanto no seu peso e cálculo. A versão atual do CVSS é a 2.0.

Entre a versão 1.0 e a 2.0 há diferenças consideráveis, principalmente na Métrica Básica. Na primeira versão havia uma variável de tendência de impacto, que permitia atribuir um peso maior a determinado requisito de segurança. O Acesso, a Complexidade de Acesso e a Autenticação possuíam apenas duas classificações. A fórmula de cálculo e os pesos das variáveis também eram bastante diferentes da versão atual.

1.4. Estudo de caso: Gestão de Riscos no Projeto de Composições de IDSs

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 neste curso, 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.

Nesta seção adotamos a metodologia de gestão de riscos para o acompanhamento e a avaliação dos requisitos de segurança envolvidos em um projeto de Composições de Sistemas de Detecção de Intrusão em ambientes de larga-escala [Brandão 2007]. A metodologia apresentada nessa seção foi inicialmente baseada na norma AS/NZ4360 e posteriormente adaptada para o padrão ISO 27005. Na avaliação original foi adotada a versão 1.0 do CVSS, que também foi atualizada nessa seção, para a versão 2.0.

Nessa seção são apresentados todos os passos da metodologia, ilustrados com a documentação contendo o resultado da aplicação da mesma no desenvolvimento do projeto.

1.4.1. Comunicação do Risco

No desenvolvimento de pesquisas acadêmicas, podem ser identificados pelo menos cinco tipos de papéis associados aos interessados no projeto de pesquisa:

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 pesquisa foram considerados apenas os três primeiros papéis, conforme a Tabela 1.1. 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.

1.4.2. Definição do Contexto

Nesta etapa cabe inicialmente definir o projeto, seu escopo e seus objetivos.

As composições de IDSs [Brandão et al. 2006a, Brandão et al. 2006b, Brandão 2007] 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

Tabela 1.1. 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.

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].

1.4.2.1. Definição dos Objetivos

Tomando como base os requisitos do projeto de pesquisa, 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.

1.4.2.2. Definição dos Critérios Básicos

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

- **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.

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)[12], usada pelo NIST para a classificação de vulnerabilidades no *National Vulnerability Database* (NVD)[13]. A base de dados de vulnerabilidades NVD é integrada *ao Common Vulnerabilities and Exposures* (CVE)[14] [Mell and 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.

Serão aceitos como riscos residuais os riscos Baixos. Os riscos Médios, cujos controles para sua redução sejam Altos, também poderão ser aceitos.

1.4.3. Identificação de Riscos

Para identificar as vulnerabilidades associadas ao projeto foi efetuada uma ampla revisão bibliográfica sobre o assunto. 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.

1.4.3.1. Sistemas de Detecção de Intrusão Distribuídos

Na primeira identificação representada na Tabela 1.2, 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 [Lindqvist and Jonsson 1998], [Ptacek and Newsham 1998], [Mell et al. 2000], [Dacier 2002], [Yegneswaran et al. 2004] e [Yu and Frincke 2004].

[12] http://www.first.org/cvss/
[13] http://nvd.nist.gov/
[14] http://cve.mitre.org/

Tabela 1.2. 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] [Yu e Frincke, 2004]
			Uso de diversidade de sensores	
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] [Yu e Frincke, 2004]
			Uso de diversidade de sensores	
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 1.3. 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]

1.4.3.2. 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 1.3. 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 and 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 and Jonsson 1998]. Também foram identificadas ameaças na comunicação entre elementos independentes de detecção de intrusão [Bass 2004].

Tabela 1.4. 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	

1.4.3.3. Composição Dinâmica de IDSs

A maioria das ameaças pertinentes à composição dinâmica estão relacionadas à composição de serviços [Esfandiari and Tosic 2004][Esfandiari and Tosic 2005] [Wang et al. 2004] [Charfi and Mezini 2005]. Os trabalhos [Feiertag et al. 2000] 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 1.4.

1.4.3.4. 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.

Tabela 1.5. 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	

Como pretendemos adotar padrões de interoperabilidade emergentes, a análise de [Parastatidis and 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 1.5 apresenta tais riscos.

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 1.6.

1.4.3.5. Agrupamento dos Riscos

Os riscos similares levantados foram agrupados e os riscos de menor impacto foram excluídos. Após analisar os resultados da fase anterior do processo de gestão de riscos,

Tabela 1.6. 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".	

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.

Tabela 1.7. Riscos de segurança combinados

Risco	Ameaça	Conseqüências	Como pode ocorrer
RS1.1	Espionagem	Confidencialidade	Atacantes interceptam as mensagens de alerta trocadas entre os elementos de IDS
RS1.2	Espionagem	Confidencialidade	Elementos comprometidos são usados para coletar e enviar informações sigilosas
RS1.3	Negação de Serviço	Disponibilidade	Desativação de Elementos por ataques diretos, explorando vulnerabilidades
RS1.4	Negação de Serviço	Disponibilidade	Elementos maliciosos enviam grande quantidade de dados falsos
RS1.5	Mascaramento	Integridade e Confidencialidade	Um elemento malicioso pode se passar por um elemento verdadeiro
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
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.
RS1.8	Ofuscação	Disponibilidade	Elemento malicioso envia grande quantidade de dados falsos para ofuscar o processo de detecção
RS1.9	Ofuscação	Disponibilidade	Elemento malicioso envia pequena quantidade dados falsos para ofuscar o processo de detecção
RS1.10	Ofuscação	Disponibilidade	Tentativa de localização (*scanning*) furtiva ou coordenada dos elementos
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
RS1.12	Comprometimento de Elementos de IDS	Disponibilidade, Integridade e Confidencialidade	Atacantes alteram o código ou a configuração do elemento
RS1.13	Interrupção ou desvio de conexão (*hijacking*)	Disponibilidade, Integridade e Confidencialidade	Atacantes interceptam ou desviam uma conexão entre elementos de IDS
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
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
RS2.3	Confiar em elementos não confiáveis	Disponibilidade, Integridade e Confidencialidade	Inclusão de novas Vulnerabilidades. Elemento não foi projetado para funcionar de forma segura
RS2.4	Dificuldade de Gerenciamento	Disponibilidade, Integridade e Confidencialidade	Inclusão de novas Vulnerabilidades. Atualizações inseguras.
RS2.5	Dificuldade de Gerenciamento	Disponibilidade, Integridade e Confidencialidade	Inclusão de novas Vulnerabilidades. Efeitos inesperados
RS2.6	Dificuldade de Gerenciamento	Disponibilidade, Integridade e Confidencialidade	*Inclusão de novas Vulnerabilidades. Backdoors* de gerenciamento
RS2.7	Dificuldade de Interoperabilidade	Disponibilidade, Integridade e Confidencialidade	Inclusão de novas vulnerabilidades. Níveis de segurança diferentes entre os elementos
RS3.1	Exposição de informações confidenciais	Confidencialidade	Informações confidenciais são repassadas a novos elementos que não possuem autorização para recebê-las
RS3.2	Negação de Serviço	Disponibilidade	Um elemento provedor de serviço aceita mais requisições de serviço do que é capaz de atender
RS3.3	Negação de Serviço	Disponibilidade	Um elemento cliente de serviço recebe mais mensagens do que é capaz de processar
RS3.4	Redução da confiabilidade da composição	Disponibilidade, Integridade e Confidencialidade	Novo elemento altera as características de segurança da composição
RS3.5	Elementos indisponíveis	Afeta a integridade e a disponibilidade do sistema	Falha nos mecanismos de busca
RS3.6	Elementos indisponíveis	Afeta a integridade e a disponibilidade do sistema	Falha nos componentes
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
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
RS4.3	Conteúdo XML malicioso	Disponibilidade, Integridade e Confidencialidade	Textos XML contém códigos maliciosos que exploram vulnerabilidades ou são executados pelas aplicações
RS4.4	Ataques por referências externas	Disponibilidade, Integridade e Confidencialidade	Documentos XML contêm ponteiros maliciosos para referências externas que são chamadas ou executadas indevidamente
RS4.5	Ataques aos protocolos SOAP/XML	Disponibilidade	Um atacante tenta sobrecarregar o sistema com mensagens SOAP (SOAP *Flooding*)
RS4.6	Ataques aos protocolos SOAP/XML	Disponibilidade, Integridade e Confidencialidade	Mensagens são interceptadas e retransmitidas como se fossem mensagens legítimas (*Replay)*
RS4.7	Ataques aos protocolos SOAP/XML	Confidencialidade	Atacantes interceptam mensagens XML (Espionagem)
RS4.8	Ataques aos protocolos SOAP/XML	Confidencialidade e Integridade	Atacantes interceptam e alteram o conteúdo das mensagens (*Man-in-the-middle*)
RS4.9	Interferência nas credenciais de segurança XML	Confidencialidade	Um atacante pode roubar ou alterar as credenciais dos clientes e serviços
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.

A Tabela 1.7 apresenta os riscos de segurança combinados. A Tabela 1.8 apresenta os riscos operacionais combinados.

Tabela 1.8. Riscos operacionais combinados

Item	Ameaça	Conseqüências	Como pode ocorrer
RO2.1	Aumento dos custos operaci onais	Dificuldade de Gerenciamento	Altos custos para administrar e manter atualizada uma base de software e hardware muito heterogênea
RO2.2	Falha de comunicação	Dificuldade de Interoperabilidade	Os elementos utilizam protocolos de comunicação ou formatos de mensagens incompatíveis
RO3.1	Falha no processo de desc oberta dos componentes	Dificuldade de localiz ação e escolha dos el ementos	Incompatibilidade nas descrições dos componen tes e falhas na caracterização das interfaces
RO3.2	Falha no processo de desc oberta dos componentes	Dificuldade de localiz ação e escolha dos el ementos	Descrições imprecisas dos componentes
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
RO4.2	O sistema não consegue integrar com novos elementos de IDS ou de fabricantes distintos	Dificuldade de interoperabilidade	Os padrões adotados n ão são compat íveis entre si ou o custo de integração é muito alto.
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 disponibiliz ados por um único fabricante
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
RO4.5	A composição não pode ser implementada em determin ados ambientes	Dificuldade de Implementação	Componentes, ferramentas e impleme ntações dos padrões não são complet amente compatíveis com a especificação original
RO4.6	Atrasos na implementação	Dificuldade de Implementação	Falta de pes soal qualificado

1.4.3.6. 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 1.9 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 1.10 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 1.9. 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, 2000 b] [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 an d Newsham 1998] [Dacier 2002] [Feiertag et al., 2000a, 2000b] [Esfandiari e Tosic, 2004, 2005] [Wang et al., 2004]	BAIXO
CS9	Sensores baseados em aplic ação	[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 Tos ic, 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 1998] [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 Jonsso n 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 1.10. Classificação de controles operacionais e seus custos

Item	Controle	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 2 004]	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

1.4.4. Estimativa de Riscos

Após combinar os riscos de segurança identificados anteriormente, aplicamos os critérios de análise definidos na seção 1.4.2 para construir a Tabela 1.11. 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 1.3.1. Assumimos nesta análise que

não há controles implementados, pois os mesmos serão propostos nas próximas etapas do processo de gestão de riscos.

Tabela 1.11. Níveis de risco de segurança e a avaliação dos riscos

Risco	Acesso	Comp. Ac.	Autenticação	Imp. Conf.	Imp. Integr.	Imp. Disp.	Escore	Avaliação
1.1	REMOTO	BAIXA	DESNECESSÁRIA	COMPLETA	NENHUMA	NENHUMA	**7,8**	**ALTO**
1.2	LOCAL	ALTA	ÚNICA	COMPLETA	NENHUMA	NENHUMA	**3,8**	**BAIXO**
1.3	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	PARCIAL	COMPLETA	**8,5**	**ALTO**
1.4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	COMPLETA	**7,8**	**ALTO**
1.5	LOCAL	ALTA	ÚNICA	COMPLETA	PARCIAL	NENHUMA	**4,5**	**MÉDIO**
1.6	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**5,0**	**MÉDIO**
1.7	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**5,0**	**MÉDIO**
1.8	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**5,0**	**MÉDIO**
1.9	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**5,0**	**MÉDIO**
1.10	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	**5,0**	**MÉDIO**
1.11	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	PARCIAL	**7,3**	**ALTO**
1.12	LOCAL	ALTA	ÚNICA	COMPLETA	COMPLETA	COMPLETA	**6,0**	**MÉDIO**
1.13	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
2.1	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
2.2	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
2.3	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
2.4	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
2.5	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
2.6	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
2.7	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
3.1	REMOTO	ALTA	ÚNICA	COMPLETA	NENHUMA	NENHUMA	**4,9**	**MÉDIO**
3.2	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	COMPLETA	**7,8**	**ALTO**
3.3	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	COMPLETA	**7,8**	**ALTO**
3.4	REMOTO	ALTA	ÚNICA	COMPLETA	COMPLETA	COMPLETA	**7,1**	**ALTO**
3.5	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	PARCIAL	COMPLETA	**8,5**	**ALTO**
3.6	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	PARCIAL	COMPLETA	**8,5**	**ALTO**
4.1	REMOTO	ALTA	DESNECESSÁRIA	PARCIAL	NENHUMA	NENHUMA	**2,6**	**BAIXO**
4.2	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
4.3	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
4.4	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
4.5	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	COMPLETA	**7,8**	**ALTO**
4.6	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
4.7	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	NENHUMA	NENHUMA	**5,4**	**MÉDIO**
4.8	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	NENHUMA	**7,1**	**ALTO**
4.9	REMOTO	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	**7,6**	**ALTO**
4.10	REMOTO	ALTA	ÚNICA	COMPLETA	COMPLETA	COMPLETA	**7,1**	**ALTO**

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 (AV = 1), a complexidade de acesso é baixa (AC = 0,71) e é desnecessária a autenticação no sistema alvo (AU = 0,704). Como conseqüência da exploração da vulnerabilidade, há uma quebra completa da confidencialidade (CI = 0,660), mas não são afetadas a integridade e a disponibilidade do sistema (II = AI = 0). Concluída a análise, é aplicada a fórmula apresentada na seção 1.3.1 para a obtenção do escore:

$$\textbf{Impacto} = 10{,}41 * (1 - (1 - 0{,}66) * (1 - 0) * (1 - 0)) = 6{,}9$$

$$\textbf{Complexidade} = 20 * 0{,}71 * 0{,}704 * 1 = 10$$

$$\textbf{Risco Básico} = (0{,}6 * 6{,}9 + 0{,}4 * 10 - 1{,}5) * 1{,}176 = 7{,}8$$

Após ser calculado o escore, é atribuído o nível de risco de acordo com a classificação do NIST, na qual o escore com valor "7,8" é considerado como nível de risco Alto. Essa operação de cálculo é repetida para todos os riscos listados, formando a Tabela 1.11.

1.4.5. Avaliação de Riscos

Parte do trabalho de avaliação dos riscos de segurança foi adiantado ao se confeccionar a Tabela 1.11, tomando como base os dados obtidos na análise de riscos. A partir da observação da Tabela 1.11, foi construído o gráfico da Figura 1.10, no qual verifica-se uma maior quantidade de vulnerabilidades com alto risco de segurança, representando 39% do total, enquanto as 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.

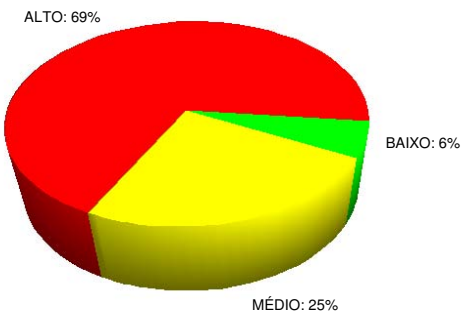

Figura 1.10. Distribuição dos níveis de risco iniciais

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.

1.4.6. Tratamento do Risco

As opções de tratamentos foram identificadas na literatura e agrupadas na Tabela 1.9 (segurança) e na Tabela 1.10 (operacional), junto com os custos envolvidos. Para algumas vulnerabilidades é necessário associar diversos tratamentos.

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 1.12.

Os tratamentos identificados para os riscos operacionais O2 e O3 foram considerados como boas práticas de desenvolvimento, para a seleção dos padrões utilizados

Tabela 1.12. Análise dos riscos de segurança tratados

Risco	Controles	Acesso	Comp. Ac.	Autenticação	Imp. Conf.	Imp. Integr.	Imp. Disp.	Escore	Avaliação
1.1	C19	LOCAL	ALTA	ÚNICA	NENHUMA	NENHUMA	NENHUMA	0,0	BAIXO
1.1	C5	REMOTO	ALTA	ÚNICA	NENHUMA	NENHUMA	NENHUMA	0,0	BAIXO
1.2	C4	LOCAL	ALTA	ÚNICA	PARCIAL	NENHUMA	NENHUMA	1,0	BAIXO
1.3	C6 + C8	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	5,0	MÉDIO
1.3	C16	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	5,0	MÉDIO
1.3	C7	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	5,0	MÉDIO
1.3	C17	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	5,0	MÉDIO
1.4	C4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	5,0	MÉDIO
1.5	C1	LOCAL	ALTA	ÚNICA	COMPLETA	PARCIAL	NENHUMA	4,5	MÉDIO
1.6	C9	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
1.6	C14	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
1.7	C9	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
1.7	C14	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
1.8	C4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	5,0	MÉDIO
1.9	C12	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
1.10	C3	REMOTO	ALTA	ÚNICA	NENHUMA	NENHUMA	PARCIAL	2,1	BAIXO
1.10	C13	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
1.11	C2 + C5	REMOTO	ALTA	ÚNICA	PARCIAL	NENHUMA	NENHUMA	2,1	BAIXO
1.12	C16	LOCAL	ALTA	ÚNICA	PARCIAL	NENHUMA	PARCIAL	2,4	BAIXO
1.13	C5	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	PARCIAL	PARCIAL	4,0	MÉDIO
2.1	C6+C13 + C14+C16	REMOTO	ALTA	DESNECESSÁRIA	PARCIAL	NENHUMA	PARCIAL	4,0	MÉDIO
2.1	C18	REMOTO	ALTA	DESNECESSÁRIA	PARCIAL	PARCIAL	PARCIAL	5,1	MÉDIO
2.2	C18	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	NENHUMA	0,0	BAIXO
2.2	C1	REMOTO	ALTA	ÚNICA	NENHUMA	NENHUMA	NENHUMA	0,0	BAIXO
2.3	C18	LOCAL	ALTA	DESNECESSÁRIA	PARCIAL	PARCIAL	PARCIAL	3,7	BAIXO
2.3	C19	LOCAL	ALTA	DESNECESSÁRIA	PARCIAL	PARCIAL	PARCIAL	3,7	BAIXO
2.4	C18	LOCAL	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	6,2	MÉDIO
2.5	C18	LOCAL	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	6,2	MÉDIO
2.6	C18	LOCAL	ALTA	DESNECESSÁRIA	COMPLETA	COMPLETA	COMPLETA	6,2	MÉDIO
2.7	C15 + C3	REMOTO	ALTA	UNICA	NENHUMA	PARCIAL	PARCIAL	3,6	BAIXO
3.1	C7 + C15	REMOTO	ALTA	ÚNICA	PARCIAL	NENHUMA	NENHUMA	2,1	BAIXO
3.2	C3 + C4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	5,0	MÉDIO
3.3	C4	REMOTO	BAIXA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	5,0	MÉDIO
3.4	C18	REMOTO	ALTA	ÚNICA	NENHUMA	NENHUMA	PARCIAL	2,1	BAIXO
3.5	C13 + C16	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
3.6	C6+C8+C16	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
4.1	C18	REMOTO	ALTA	DESNECESSÁRIA	PARCIAL	NENHUMA	NENHUMA	2,6	BAIXO
4.1	C3	REMOTO	ALTA	ÚNICA	PARCIAL	NENHUMA	NENHUMA	2,1	BAIXO
4.2	C1 + C2 + C3 + C11	REMOTO	ALTA	MÚLTIPLAS	COMPLETA	COMPLETA	COMPLETA	6,8	MÉDIO
4.3	C1 + C2 + C3 + C11	REMOTO	ALTA	MÚLTIPLAS	COMPLETA	COMPLETA	COMPLETA	6,8	MÉDIO
4.4	C1 + C2 + C3 + C11	REMOTO	ALTA	MÚLTIPLAS	COMPLETA	COMPLETA	COMPLETA	6,8	MÉDIO
4.5	C1 + C3	REMOTO	BAIXA	MÚLTIPLAS	NENHUMA	NENHUMA	PARCIAL	3,3	BAIXO
4.6	C10	REMOTO	ALTA	DESNECESSÁRIA	NENHUMA	NENHUMA	PARCIAL	2,6	BAIXO
4.6	C2 + C5	REMOTO	ALTA	MÚLTIPLAS	NENHUMA	NENHUMA	PARCIAL	1,7	BAIXO
4.7	C5	REMOTO	ALTA	ÚNICA	NENHUMA	NENHUMA	NENHUMA	0,0	BAIXO
4.8	C2 + C5	REMOTO	ALTA	MÚLTIPLAS	NENHUMA	NENHUMA	NENHUMA	0,0	BAIXO
4.9	C5 + C18	REMOTO	ALTA	ÚNICA	NENHUMA	NENHUMA	NENHUMA	0,0	BAIXO
4.10	C5 + C18	REMOTO	ALTA	ÚNICA	NENHUMA	NENHUMA	NENHUMA	0,0	BAIXO

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.

Para mensurar os custos desses tratamentos combinados, será adotado o custo do tratamento mais oneroso, independentemente da quantidade de tratamentos associados.

Tabela 1.13. Avaliação e seleção dos tratamentos de segurança

Risco	Controles	Escore Original	Escore Tratado	Nível de Risco Original	Nível de Risco Tratado	Custo	Aplicação
1.1	C19	7,8	0,0	ALTO	BAIXO	ALTO	Não
1.1	C5	7,8	0,0	ALTO	BAIXO	BAIXO	**Sim**
1.2	C4	3,8	1,0	BAIXO	BAIXO	BAIXO	**Sim**
1.3	C6 + C8	8,5	5,0	ALTO	MÉDIO	BAIXO	**Sim**
1.3	C16	8,5	5,0	ALTO	MÉDIO	MÉDIO	Não
1.3	C7	8,5	5,0	ALTO	MÉDIO	BAIXO	**Sim**
1.3	C17	8,5	5,0	ALTO	MÉDIO	MÉDIO	Não
1.4	C4	7,8	5,0	ALTO	MÉDIO	BAIXO	**Sim**
1.5	C1	4,5	4,5	MÉDIO	MÉDIO	BAIXO	**Sim**
1.6	C9	5	2,6	MÉDIO	BAIXO	BAIXO	**Sim**
1.6	C14	5	2,6	MÉDIO	BAIXO	MÉDIO	Não
1.7	C9	5	2,6	MÉDIO	BAIXO	BAIXO	**Sim**
1.7	C14	5	2,6	MÉDIO	BAIXO	MÉDIO	Não
1.8	C4	5	5,0	MÉDIO	MÉDIO	BAIXO	**Sim**
1.9	C12	5	2,6	MÉDIO	BAIXO	MÉDIO	Não
1.10	C3	5	2,1	MÉDIO	BAIXO	BAIXO	**Sim**
1.10	C13	5	2,6	MÉDIO	BAIXO	MÉDIO	Não
1.11	C2 + C5	7,3	2,1	ALTO	BAIXO	BAIXO	**Sim**
1.12	C16	6	2,4	MÉDIO	BAIXO	MÉDIO	**Sim**
1.13	C5	7,6	4,0	ALTO	MÉDIO	BAIXO	**Sim**
2.1	C6+C13 + C14+C16	7,6	4,0	ALTO	MÉDIO	MÉDIO	**Sim**
2.1	C18	7,6	5,1	ALTO	MÉDIO	MÉDIO	Não
2.2	C18	7,6	0,0	ALTO	BAIXO	MÉDIO	Não
2.2	C1	7,6	0,0	ALTO	BAIXO	BAIXO	**Sim**
2.3	C18	7,6	3,7	ALTO	BAIXO	MÉDIO	**Sim**
2.3	C19	7,6	3,7	ALTO	BAIXO	ALTO	Não
2.4	C18	7,6	6,2	ALTO	MÉDIO	MÉDIO	**Sim**
2.5	C18	7,6	6,2	ALTO	MÉDIO	MÉDIO	**Sim**
2.6	C18	7,6	6,2	ALTO	MÉDIO	MÉDIO	**Sim**
2.7	C15 + C3	7,6	3,6	ALTO	BAIXO	MÉDIO	**Sim**
3.1	C7 + C15	4,9	2,1	MÉDIO	BAIXO	MÉDIO	**Sim**
3.2	C3 + C4	7,8	5,0	ALTO	MÉDIO	BAIXO	**Sim**
3.3	C4	7,8	5,0	ALTO	MÉDIO	BAIXO	**Sim**
3.4	C18	7,1	2,1	ALTO	BAIXO	MÉDIO	**Sim**
3.5	C13 + C16	8,5	2,6	ALTO	BAIXO	MÉDIO	**Sim**
3.6	C6+C8+C16	8,5	2,6	ALTO	BAIXO	MÉDIO	**Sim**
4.1	C18	2,6	2,6	BAIXO	BAIXO	MÉDIO	Não
4.1	C3	2,6	2,1	BAIXO	BAIXO	BAIXO	**Sim**
4.2	C1 + C2 + C3 + C11	7,6	6,8	ALTO	MÉDIO	BAIXO	**Sim**
4.3	C1 + C2 + C3 + C11	7,6	6,8	ALTO	MÉDIO	BAIXO	**Sim**
4.4	C1 + C2 + C3 + C11	7,6	6,8	ALTO	MÉDIO	BAIXO	**Sim**
4.5	C1 + C3	7,8	3,3	ALTO	BAIXO	BAIXO	**Sim**
4.6	C10	7,6	2,6	ALTO	BAIXO	BAIXO	Não
4.6	C2 + C5	7,6	1,7	ALTO	BAIXO	BAIXO	**Sim**
4.7	C5	5,4	0,0	MÉDIO	BAIXO	BAIXO	**Sim**
4.8	C2 + C5	7,1	0,0	ALTO	BAIXO	BAIXO	**Sim**
4.9	C5 + C18	7,6	0,0	ALTO	BAIXO	MÉDIO	**Sim**
4.10	C5 + C18	7,1	0,0	ALTO	BAIXO	MÉDIO	**Sim**

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 1.13 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.

1.4.7. Aceitação do Risco

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 1.11.

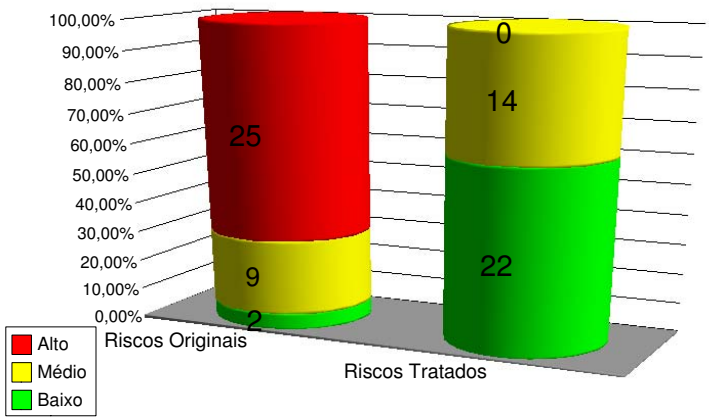

Figura 1.11. Comparativo dos níveis de risco de segurança, antes e após o tratamento

1.4.8. Monitoramento e Análise Crítica de Riscos

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 do Risco 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 os 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.

Outra questão diz respeito a ajustes do processo de gestão de riscos de segurança, em decorrência de mudanças metodológicas, como o lançamento de uma nova norma ou a alteração de versão do CVSS, por exemplo. A primeira análise realizada neste projeto foi elaborada usando exclusivamente a norma AS/NZ4360 e a versão 1,0 do CVSS. A

presente análise reviu estes procedimentos e elaborou novos resultados de acordo com os novos critérios.

1.4.9. Considerações sobre o Estudo de Caso

Com a metodologia aplicada, foi possível acompanhar o desenvolvimento das soluções propostas e implementadas no projeto de pesquisa 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.

1.5. Conclusões

Esse curso apresentou a disciplina de gestão de riscos de segurança, tratando, principalmente, dos principais padrões relacionados ao assunto. Para ilustrar o tema, foi detalhado um estudo de caso, no qual uma metodologia de Gestão de Riscos foi aplicada no acompanhamento de um projeto científico.

Com a utilização da metodologia de gestão de riscos, espera-se a identificação e o tratamento da maioria das vulnerabilidades conhecidas e pertinentes às soluções adotadas em um projeto de Tecnologia da Informações. Isso sem dúvida auxilia no entendimento dos problemas de segurança que seriam enfrentados, tendo como conseqüência a melhoria do projeto como um todo. Infelizmente, não é possível garantir que o projeto seja totalmente seguro. Contudo, podemos afirmar que, com a realização de boas práticas de gestão de risco, são tomadas todas as medidas preventivas necessárias à atenuação do impacto negativo que possíveis vulnerabilidades infringiriam ao projeto.

Esperamos com esse curso ter contribuído para um melhor entendimento destas metodologias de gestão de riscos e a difusão das idéias da necessidade da avaliação via estes testes padronizados dos sistemas de segurança.

Referências

[ABNT 2006a] ABNT (2006a). Código de Prática para a Gestão da Segurança da Informação. ABNT NBR ISO/IEC 27002:2005.

[ABNT 2006b] ABNT (2006b). Sistemas de Gestão de Segurança da Informação - Requisitos. ABNT NBR ISO/IEC 27001:2006.

[ABNT 2008] ABNT (2008). Tecnologia da informação - Técnicas de segurança - Gestão de riscos de segurança da informação. ABNT NBR ISO/IEC 27005:2008.

[AS/NZS 2004a] AS/NZS (2004a). *Risk Management - AS/NZS4360:2004*. Australian/New Zealand Standard, third edition.

[AS/NZS 2004b] AS/NZS (2004b). *Risk Management Guidelines Companion to AS/NZS 4360:2004 - HB 436:2004*. Australian/New Zealand Standard, third edition.

[Athanasiades et al. 2003] Athanasiades, N., Abler, R., Levine, J., Owen, H., and Riley, G. (2003). 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. IEEE Computer Society.

[Barker and Lee 2004] Barker, W. C. and Lee, A. (2004). Information security - volume ii: Appendices to guide for mapping types of information and information systems to security categories. NIST Special Publication 800-60.

[Bass 2004] Bass, T. (2004). Service-oriented horizontal fusion in distributed coordination-based systems. In *IEEE MILCOM 2004*, volume 2, pages 615– 621, Monterey, CA, USA.

[Bishop 2003] Bishop, M. (2003). *Computer Security: Art and Science*. Addison Wesley, Bonston, MA.

[Blakley et al. 2001] Blakley, B., McDermott, E., and Geer, D. (2001). Information security is information risk management. In *NSPW '01: Proceedings of the 2001 workshop on New security paradigms*, pages 97–104, New York, NY, USA. ACM Press.

[Brandão 2007] Brandão, J. E. M. S. (2007). *COMPOSIÇÕES DE IDSs: VIABILIZANDO O MONITORAMENTO DE SEGURANÇA EM AMBIENTES DE LARGA ESCALA*. PhD thesis, Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina, Florianópolis, Santa Catarina.

[Brandão et al. 2006a] Brandão, J. E. M. S., Fraga, J. S., and Mafra, P. M. (2006a). A New Approach for IDS Composition. In *Proceedings of ICC 2006: IEEE International Conference on Communications*, Istanbul, Turkey.

[Brandão et al. 2006b] Brandão, J. E. M. S., Fraga, J. S., Mafra, P. M., and Obelheiro, R. R. (2006b). 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. Springer Berlin / Heidelberg.

[Bray et al. 2004] Bray, T., Paoli, J., and Sperberg-McQueen, C. M. (2004). Extensible Markup Language (XML) 1.0 (third edition)". W3C Recommendation.

[BSi 1999] BSi (1999). Information Security Management, Part 1: Code of Practice for Information Security Management. BS 7799-1:1999.

[BSi 2002] BSi (2002). Information Security Management, Part 2: Specification with guidance for use. BS 7799-2:2002.

[BS/ISO 2005] BS/ISO (2005). BS/ISO17799 Information technology – Code of practice for information security managementt. BS/ISO 17799:2005.

[Charfi and Mezini 2005] Charfi, A. and Mezini, M. (2005). 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. IEEE Computer Society.

[Dacier 2002] Dacier, M. (2002). Design of an intrusion-tolerant intrusion detection system. maftia project, deliverable 10. Technical report, IBM Zurich Research Laboratory.

[Debar et al. 1998] Debar, H., Dacier, M., Wespi, A., and Lampart, S. (1998). An Experimentation Workbench For Intrusion Detection Systems. Technical report, IBM, IBM Research, Zurich Research Laboratory.

[Demchenko et al. 2005] Demchenko, Y., Gommans, L., de Laat, C., and Oudenaarde, B. (2005). 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. IEEE Cat. No. 05EX1210C.

[Durst et al. 1999] Durst, R., Champion, T., Witten, B., Miller, E., and Spagnuolo, L. (1999). Testing and evaluating computer intrusion detection systems. *Commun. ACM*, 42(7):53–61.

[Esfandiari and Tosic 2004] Esfandiari, B. and Tosic, V. (2004). Requirements for web service composition management. In *Proc. of the 11th Hewlett-Packard Open View University Association (HP-OVUA) Workshop*, Paris, France. Hewlett-Packard.

[Esfandiari and Tosic 2005] Esfandiari, B. and Tosic, V. (2005). Towards a web service composition management framework. In *ICWS*, pages 419–426. IEEE Computer Society.

[Feiertag et al. 2000] Feiertag, R., Redmond, T., and Rho, S. (2000). A framework for building composable replaceable security services. In *DARPA Information Survivability Conference & Exposition*, volume 2, pages 391–402.

[Frincke 2000] Frincke, D. (2000). Balancing cooperation and risk in intrusion detection. *ACM Trans. Inf. Syst. Secur.*, 3(1):1–29.

[Garfinkel et al. 2003] Garfinkel, S., Spafford, G., and Schwartz, A. (2003). *Practical Unix and Internet security (3rd ed.)*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, third edition.

[Han and Zheng 2000] Han, J. and Zheng, Y. (2000). 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.

[ISO 1989] ISO (1989). Information processing systems - open systems interconnection - basic reference model - part 2: Security architecture. ISO 7498-2.

[ISO 2002] ISO (2002). Risk management - Vocabulary - Guidelines for use in standards. ISO/IEC Guide 73:2002.

[ISO 2007] ISO (2007). Risk management - guidelines on principles and implementation of risk management. ISO/TMB WG on Risk management N 047.

[ISO/IEC 2005a] ISO/IEC (2005a). Common criteria for information technology security evaluation - part 1: Introduction and general model. ISO/IEC 15408:2005.

[ISO/IEC 2005b] ISO/IEC (2005b). Common criteria for information technology security evaluation - part 2: Security functional requirements. ISO/IEC 15408:2005.

[ISO/IEC 2005c] ISO/IEC (2005c). Common criteria for information technology security evaluation - part 3: Security assurance requirements. ISO/IEC 15408:2005.

[Lindqvist and Jonsson 1998] Lindqvist, U. and Jonsson, E. (1998). A map of security risks associated with using cots. *Computer*, 31(6):60–66.

[Lippmann et al. 2000a] Lippmann, R., Haines, J. W., Fried, D. J., Korba, J., and Das, K. (2000a). The 1999 darpa off-line intrusion detection evaluation. *Comput. Networks*, 34(4):579–595.

[Lippmann et al. 2000b] 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., and Zissman, M. A. (2000b). Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation. *discex*, 02:1012.

[McHugh 2000] McHugh, J. (2000). 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.

[Mell and Grance 2002] Mell, P. and Grance, T. (2002). Use of the common vulnerabilities and exposures (cve) vulnerability naming scheme. NIST Special Publication 800-51.

[Mell et al. 2000] Mell, P., Marks, D., and McLarnon, M. (2000). A denial-of-service resistant intrusion detection architecture. *Comput. Networks*, 34(4):641–658.

[Parastatidis and Webber 2004] Parastatidis, S. and Webber, J. (2004). 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. IEEE Computer Society.

[Ptacek and Newsham 1998] Ptacek, T. H. and Newsham, T. N. (1998). Insertion, evasion, and denial of service: Eluding network intrusion detection. Technical report, Secure Networks,Inc.

[Puketza et al. 1996] Puketza, N. J., Zhang, K., Chung, M., Mukherjee, B., and Olsson, R. A. (1996). A methodology for testing intrusion detection systems. *IEEE Trans. Softw. Eng.*, 22(10):719–729.

[Ross et al. 2005] Ross, R., Katzke, S., Johnson, A., Swanson, M., Stoneburner, G., Rogers, G., and Lee, A. (2005). Recommended security controls for federal information systems. NIST Special Publication 800-53.

[Stoneburner et al. 2002] Stoneburner, G., Goguen, A., and Feringa, A. (2002). Risk management guide for information technology systems. NIST Special Publication 800-30.

[Swanson and Guttman 1996] Swanson, M. and Guttman, B. (1996). Generally accepted principles and practices for securing information technology systems. NIST Special Publication 800-14.

[US Department of Homeland Security 2002] US Department of Homeland Security (2002). Federal Information Security Management Act of 2002, H.R. 2458-48. (Public Law 107-347).

[W3C 2004] W3C (2004). Web Services Architecture. W3C Working Group Note 11.

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

[Yegneswaran et al. 2004] Yegneswaran, V., Barford, P., and Jha, S. (2004). Global intrusion detection in the DOMINO overlay system. In *NDSS*, San Diego, California, USA. The Internet Society.

[Yu and Frincke 2004] Yu, D. and Frincke, D. (2004). Towards survivable intrusion detection system. *hicss*, 09:90299a.

[Yu et al. 2005] Yu, W. D., Supthaweesuk, P., and Aravind, D. (2005). Trustworthy web services based on testing. *sose*, 0:167–177.

Capítulo 2

Segurança em Redes *Mesh*: Tendências, Desafios e Aplicações

Elisangela Santana Aguiar[1], Rafael Lopes Gomes[2], Antônio Jorge Gomes Abelém[2], Douglas Brito Damalio[2] e Billy Anderson Pinheiro[3].

[1] Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal do Pará (PPGEE/UFPA)
[2] Faculdade de Computação, Universidade Federal do Pará (FC/UFPA)
[3] Programa de Pós-Graduação em Ciência da Computação, Universidade Federal do Pará (PPGCC/UFPA)

Abstract

Mesh networks are multi-hop wireless networks emerging as a low cost infrastructure for community access networks and digital cities. In this context, support for killer applications such as cooperative services and mobile multimedia applications are the in great demand. This mini-course aims at presenting, in a theoretical way, main problems, solutions and challenges for providing security in wireless mesh networks. The course has three main focuses: the mesh networks contextualization; Security issues in wireless mesh networks, with your challenges and deficiency and; the main proposals found in the literature.

Resumo

Redes mesh são redes em malha sem fio autoconfiguráveis e de crescimento orgânico. Recentemente vêm sendo consideradas como infra-estrutura de baixo custo para a construção de redes de acesso comunitárias e de cidades digitais. Neste contexto, é grande o interesse em suportar aplicações multimídia como telefonia IP móvel, e aplicações cooperativas. Este mini-curso tem como objetivo apresentar, de maneira teórica, os principais problemas de segurança em redes mesh e a discussão de soluções propostas para solucioná-los. O curso tem três focos principais: a contextualização das redes mesh; o estado da arte e as questões de segurança em redes em malha sem fio, com seus desafios e deficiência e; as principais propostas encontradas na literatura.

2.1. Introdução

Desde a apresentação, em 1997, do padrão IEEE (*Institute of Electrical and Electronic Engineers*) 802.11, muitas aplicações vêm sendo criadas para esta tecnologia. Seu principal uso é em redes locais e públicas, através de pontos de acesso interligados diretamente a uma rede fixa cabeada tradicional (*wired*), utilizando redes 802.11 infra-estruturadas [GT-Mesh 2006].

Com o avanço das tecnologias sem fio (*wireless*) e o baixo custo destes produtos, o uso de dispositivos móveis que se comunicam através de ondas de rádio está se tornando cada vez maior. Por esta razão, mais estabelecimentos comerciais como *shoppings* e aeroportos estão procurando meios, através da tecnologia sem fio, para oferecer aos seus clientes acesso à Internet banda larga.

Uma dessas novas aplicações são as redes em malha sem fio, mais conhecidas como redes *mesh* (*WMN - Wireless Mesh Networks*). Este novo tipo de rede dispensa o uso da rede fixa entre os pontos de acesso utilizados para realizar o roteamento do tráfego entre si dinamicamente.

As redes em malha sem fio, também conhecidas como redes comunitárias de acesso sem fio, surgiram da constatação de que as redes sem fio poderiam ser aproveitadas para reduzir o custo da "última milha" no acesso à Internet. Através da colaboração entre os nós, um enlace com a rede fixa poderia ser compartilhado, permitindo um uso mais eficiente da banda, evitando o custo da passagem de fios até os usuários finais [Breuel 2004].

Uma rede *mesh* possibilita a comunicação entre diferentes dispositivos. Alguns participantes comporão a estrutura principal da rede, ou seja, formarão o *backbone*, trabalhando apenas como roteadores, e comunicando-se via interface sem fio. Outros nós podem se conectar a estes roteadores por cabos e trabalharem apenas como clientes [GT-Mesh 2006].

As redes m*esh* assemelham-se em muito às redes móveis *Ad-hoc* (*Mobile Ad-hoc networks*, ou MANETs), já que ambas utilizam transmissão sem fio e têm topologia dinâmica variável e de crescimento orgânico. A principal diferença entre as duas tecnologias, no entanto, reside no fato de que nas redes *mesh* os nós clientes não precisam obrigatoriamente ser roteador, possuindo, portanto, menor complexidade nas pontas da rede.

O conceito de redes *mesh* traz consigo uma serie de vantagens que tornam cada vez mais interessantes a sua implantação, como por exemplo:

- Redes de baixo custo: O compartilhamento de recursos faz com que o custo total da rede caia, viabilizando a criação de redes comunitárias [Luiz e Júnior 2005].
- Fácil implantação: Como as redes *mesh* possuem a característica de serem autoconfiguráveis, a sua implantação se torna fácil, pois não são necessárias configurações complexas, nem necessidade de mudança caso algum nó venha a entrar na rede.
- Tolerante a falhas: A capacidade de roteamento dinâmico aliado à existência de múltiplas rotas de acesso a um nó faz com que a rede consiga se recuperar de falhas como a perda de um enlace de comunicação.

- Escalável: Uma das melhores características das redes *mesh* é que sua capacidade de roteamento cresce conforme os nós são adicionados, logo o crescimento das redes, diferente da arquitetura tradicional não é um problema [Harada 2006].

Entretanto, existem algumas desvantagens, a maioria presente ainda pela recente atenção dada as redes *mesh* por parte tanto do mercado quanto da academia, como por exemplo:

- Falta de padronização: Este problema impossibilita até então a adoção da tecnologia em larga escala, espera-se que em 2009 tal problema já esteja solucionado [802.11s 2008].
- Alto preço dos dispositivos: Atualmente o preço dos dispositivos torna o acesso a estes muito restritivos, espera-se que com a padronização da tecnologia os preços tornem-se mais acessíveis.
- Interferência: O uso da faixa, não regulamentada de 2.4 GHz, possibilita a interferência de equipamentos externos à rede que degradam a qualidade desta como um todo [802.11s 2008].
- Baixo *throughput*: Os valores atuais ainda devem ser aprimorados, tendo em mente a possibilidade de crescimento de uma rede *mesh*. Uma alternativa para maximizar o *throughput* é utilizar um canal exclusivo para o tráfego de *backbone*, o que gera um desempenho bastante superior [Kyasanur 2007].
- Falta de segurança: A segurança ainda é um campo aberto no que diz respeito às redes *mesh*, além dos problemas normais de segurança em redes sem fio tradicionais, ainda existe o problema de garantir a privacidade dos dados que estão trafegando entre os nós [Breuel 2004].
- Ausência de qualidade de serviço: Assim como os problemas de segurança, a falta de qualidade de serviço em redes *mesh* é uma linha de pesquisa pouco explorada, tendo algumas propostas para estes problemas, como o algoritmo de roteamento denominado WMR (*Wireless Mesh Routing*) [Xue e Ganz 2005].

Dois tipos de nós podem ser encontrados nas redes *mesh*, no entanto elas aceitam a comunicação com outros tipos de redes e seus respectivos equipamentos [Akyildiz et al. 2005]:

a) Roteadores *mesh* (*MR - Mesh Routers*)

Possuem as mesmas funcionalidades de roteadores convencionais, como *gateway* e *bridge*, porém com o suporte a *mesh*, que provê maior flexibilidade a rede, pois permite a comunicação com outros tipos de redes, como a cabeada, através do uso de múltiplas interfaces. Para que todos estes recursos possam ser executados de maneira satisfatória é necessário um maior poder computacional, necessitando normalmente de um computador para realizar o papel de nó central ou simplesmente fazer uso de um sistema embarcado.

b) Clientes *mesh* (MC - *Mesh Clients*)

Podem realizar o processo de encaminhamento de pacotes entre os demais elementos *mesh* da rede, no entanto não podem exercer as funções de *bridge* ou *gateway*. Em contrapartida, este tipo de nó apresenta apenas uma interface de rede e um *hardware*

bem mais simples, podendo variar desde um *laptop* até um telefone IP (*Internet Protocol*).

Através da combinação desses dois tipos de nós *mesh* juntamente com a utilização de interfaces de redes não *mesh*, podem ser formadas três tipos de arquiteturas [Akyildiz et al. 2005]:

a) Arquitetura Cliente

Apenas nós clientes (*Mesh Clients*) são usados nesta arquitetura. Cada nó faz tanto o papel de cliente como o de roteador, como mostra a Figura 2.1. Eles se comunicam como em uma rede *peer-to-peer* formando uma estrutura muito próxima a de uma rede *ad hoc*, diferindo apenas na utilização de uma única tecnologia de transmissão.

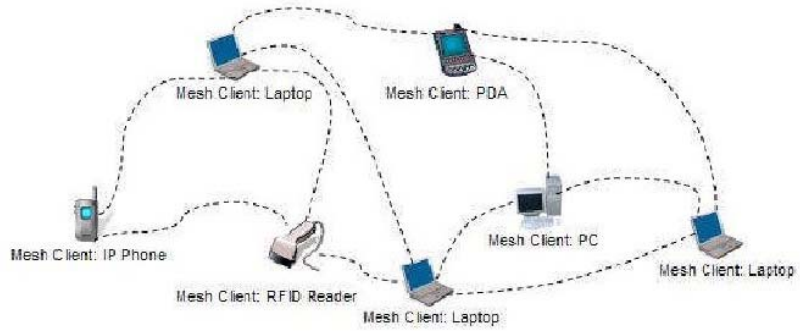

Figura 2.1. Arquitetura cliente (Fonte [Akyildiz et al. 2005])

b) Arquitetura Infra-Estruturada

O *backbone* da rede é composto de roteadores *mesh* que fornecem a infra-estrutura básica para a conexão de clientes não *mesh*, vista na Figura 2.2. Através deste *backbone* formado é possível interligar diferentes redes com diferentes tecnologias de transmissão. Esse é o tipo de rede *mesh* mais usada, pois necessita de modificações apenas nos seus roteadores que normalmente utilizam duas antenas com canais distintos, uma para o *backbone* e outra para atender os clientes.

c) Arquitetura Híbrida

Esta arquitetura faz o uso tanto dos roteadores como dos clientes *mesh*. Ela é a configuração mais completa, fazendo o uso de todas as possibilidades de comunicações que as redes *mesh* oferecem, possibilitando que clientes *mesh* e convencionais tenham acesso ao *backbone mesh* que oferece uma série de interligações com outras redes. A Figura 2.3 apresenta este tipo de arquitetura.

Das arquiteturas expostas cada uma tem seu grau de utilização e aplicação, cabe lembrar que as redes *mesh*, não estão restritas a tecnologia *Wi-Fi (Wireless Fidelity)*, sendo possível a utilização, por exemplo, de dispositivos *Bluetooth* em uma arquitetura cliente *mesh*, assim como vários outros tipos de configurações.

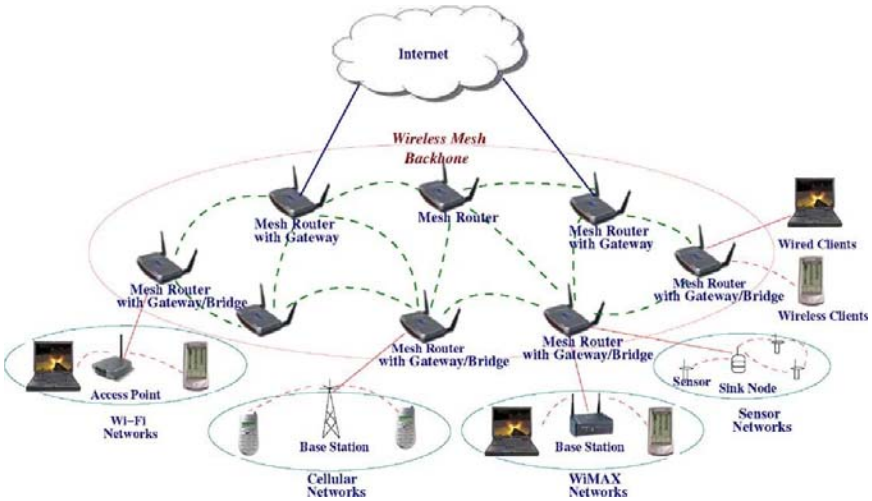

Figura 2.2. Arquitetura Infra-estruturada (Fonte [Akyildiz et al. 2005])

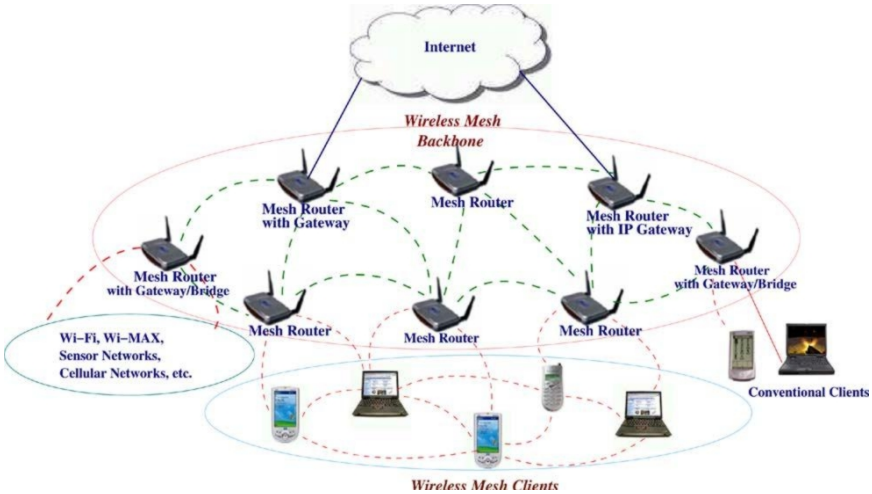

Figura 2.3. Arquitetura Híbrida (Fonte [Akyildiz et al. 2005])

As principais linhas de pesquisas em redes *mesh* estão diretamente ligadas aos principais desafios da tecnologia, sendo eles: a busca pelo melhor roteamento na rede, garantir a segurança e integridade da rede e garantir a qualidade de serviço perante uma topologia dinâmica. Essas abordagens podem ser encontradas em [Faccin et al. 2006].

Considerando a existência de mobilidade em maior ou menor grau e a configuração dinâmica da rede, mecanismos de autenticação serão necessários para garantir o acesso à rede de forma segura e íntegra. Recomenda-se que o mínimo possível seja alterado nas especificações do padrão IEEE 802.11i.

Os mesmos tópicos de segurança identificados nas redes sem fio tradicionais valem para as redes *mesh*, entretanto ganham um nível de dificuldade a mais pela necessidade de serem obtidos salto-a-salto. São três os pontos fundamentais a serem alcançados [Salem e Hubaux 2006]:

- A descoberta de pontos *mesh* corrompidos. Como exemplo dessas corrupções tem-se a possibilidade dos pontos serem removidos, acessados para roubo de informações, acessados com alterações de informações e por último, clonados.
- A definição e o uso de um protocolo de roteamento seguro.
- A definição e a utilização de uma métrica apropriada e justa para redes *mesh*.

Considerando vantagens como baixo custo, fácil implantação e tolerância à falhas, a tecnologia de redes *mesh* é extremamente promissora para implementação de acesso de última milha e capilarização de *backbones*. Entretanto, todo esse potencial não deve ser estudado sem a consideração dos aspectos de segurança envolvidos.

Uma rede *mesh* em seu escopo é semelhante a uma WLAN (*Wireless Local Area Network*), onde a transmissão via rádio pode ser interceptada. Desta forma, as mensagens podem ser interceptadas e alteradas, ou seja, a rede pode ser acessada indevidamente ou sofrer ataques de negação de serviços, DoS (*Denial of Service*).

Em virtude da sua descentralização através de múltiplos saltos, pode-se levar um maior tempo para detectar e tratar um ataque indevido em uma rede *mesh*, permitindo assim ao atacante uma vantagem indesejada para a administração da rede. Conseqüentemente o roteamento efetuado em uma rede *mesh* deve ser seguro.

Os serviços visualizados em redes *mesh* envolvem a privacidade do usuário e a confiabilidade dos fluxos da comunicação. De forma generalizada, assim como nos demais tipos de redes, é importante garantir confidencialidade, integridade, autenticação, controle de acesso e disponibilidade.

O tráfego de qualquer rede pode ser protegido em diferentes camadas (física, rede, transporte e aplicação), entretanto especificamente no caso das redes *mesh*, significa proteger o enlace sem fio, através do uso de diferentes esquemas de encapsulamento de *frames*, diferentes protocolos de autenticação e algoritmos de criptografia.

Os requisitos de segurança a serem atendidos em uma rede *mesh* estão diretamente associados ao seu cenário de utilização, como por exemplo: Domínios administrativos; Tipos de nós envolvidos na rede; Classes dos usuários; Integração a outras redes; etc.

Este capítulo tem como principal objetivo apresentar, de maneira teórica, os principais problemas de segurança em redes *mesh* e a discussão de soluções propostas para solucioná-los, bem como as suas tendências e aplicações.

O restante do texto está estruturado da seguinte maneira. A Seção 2.2 aborda o estado da arte e as questões de segurança em redes em malha sem fio, com seus desafios e metas. A Seção 2.3 aborda os aspectos de roteamento. A Seção 2.4 aborda os aspectos de gerenciamento, apresentando a sua relação com segurança. A Seção 2.5 versa sobre as principais propostas encontradas na literatura. Para finalizar, a Seção 2.6 apresenta as

conclusões, apontando os principais desafios atuais que demandam novas pesquisas na área, bem como suas tendências futuras.

2.2. Segurança em Redes *Mesh*

Para contextualizar o cenário das redes *mesh*, usaremos como base a Figura 2.4 abaixo, representando uma comunicação simples e típica, envolvendo um cliente *mesh* (MC - *Mesh Client*) associado ao ponto de acesso 3 (AP_3 – *Access Point* 3) com saída Internet através do WHS (*Wireless Hot Spot*), passando pelos pontos de acesso intermediários AP_2 e AP_1. Ou seja, as informações de/para o cliente passam por 4 saltos até a Internet.

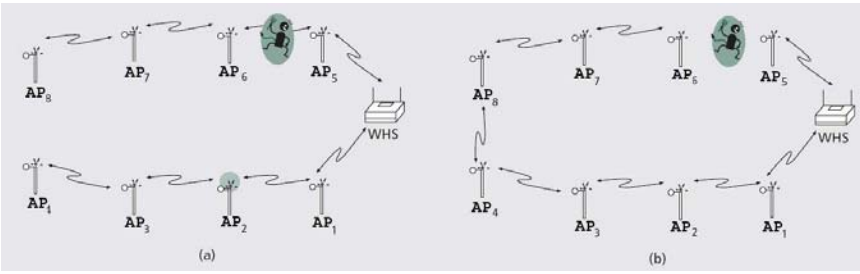

Figura 2.4. Exemplo de comunicação em redes *mesh* (Fonte [Salem e Hubaux 2006])

Baseado nisso, algumas análises podem ser realizadas antes de uma mensagem chegar à rede infra-estruturada:

a) Autenticação

Geralmente acesso à Internet é um serviço pago pelo cliente, conseqüentemente o AP_3 precisa autenticar o MC para executar o faturamento corretamente. Esta autenticação pode ser realizada de modos diferentes, como por exemplo:

- Usando uma conta de bilhetagem temporária (cartão de crédito).
- Através do uso de uma senha pré-definida e compartilhada (se o MC for um cliente da rede associada ao AP_3).
- Usando um serviço similar ao *roaming* da rede celular (se o MC não for um cliente da rede associada ao AP_3).

A autenticação deverá evitar o uso da troca de chaves de criptografia assimétricas pelo MC, uma vez que este faz uso de bateria e necessitaria de um consumo maior para o processamento computacional, sendo propenso a ataques de DoS. Isso porque se o protocolo de autenticação necessitar do processamento ou validação de uma chave, poderá ser usado por um invasor que continuamente poderá fazer solicitações de serviços ao MC, levando ao consumo total da sua bateria.

b) Autenticação mútua entre os nós da rede (APs e WHS)

Existe diferença entre a autenticação necessária durante a inicialização ou re-inicialização dos nós e a durante uma sessão estabelecida pelo MC.

A fase de inicialização ou re-inicialização ocorre quando a rede é descoberta pela primeira vez ou quando a rede necessita de uma reconfiguração (por exemplo, um

AP foi desligado). No geral, os APs e o WHS são energizados de forma cabeada, sem a restrição do uso de baterias, podendo desta forma, processar autenticação através de chaves criptográficas para autenticação mútua.

A autenticação mútua dos nós durante uma sessão é diferente, pois as mensagens geradas de/para o MC são enviadas através de múltiplos saltos e o uso de chaves criptográficas para autenticar o receptor/emissor de cada e todo pacote é um processo pesado que introduz atrasos e pode ocasionar alta utilização de recursos de rede. Nesse caso a autenticação ocorreria a cada AP ou no WHS, dependendo do sentido da comunicação.

c) Integridade das mensagens trocadas

Uma vez que o MC e os nós de rede tenham sido autenticados, é necessário verificar a integridade das mensagens trocadas. Esta análise pode ser feita fim-a-fim, a cada AP ou de ambas as formas.

A partir disso, pode-se dizer que os desafios de segurança em redes *mesh* além de estarem associados ao seu cenário de utilização, referem-se ainda às suas características de topologia/arquitetura de rede, como por exemplo:

- O dinamismo da rede pode provocar alterações na topologia, desta forma qualquer esquema de segurança estático não será suficiente.
- A diferença entre os componentes da rede (clientes e roteadores/APs) representa característica e funcionamento diferente relacionada à mobilidade e consumo de energia, portanto a solução de segurança pode não ser aplicável a ambos.

2.2.1. Tipos de Ataques

Quanto aos possíveis ataques em uma rede *mesh*, pode-se caracterizá-los em dois tipos: Ataques externos (nos quais os atacantes que não pertencem à rede *mesh* podem sobrecarregar a rede ou injetar informações erradas) e Ataques internos (os quais possibilitam ameaças mais severas a partir de um dos nós da rede, o que é mais difícil de ser prevenir).

É válido ressaltar que esses ataques podem atingir diferentes camadas de protocolos. Além disso, pode-se ter ainda ataques passivos, os quais pretendem roubar informação e escutar às escondidas a comunicação dentro da rede, e ataques ativos, os quais modificam as informações. Os principais ataques são:

a) Easvesdropping

É um ataque passivo caracterizado pela escuta do tráfego sem modificação dos dados, sendo que o atacante aproveita-se do meio inseguro com o objetivo de roubar informações, podendo descobrir pontos críticos da rede e executar ataques ativos.

Geralmente a proteção contra ataques de espionagem é uma responsabilidade das camadas superiores do modelo OSI (*Open Systems Interconnection*), no entanto, caso não haja criptografia em nível de roteamento, a topologia da rede pode ser facilmente descoberta

b) Ataque bizantino ou alteração de mensagens de roteamento

É um ataque ativo onde, um ou mais nós maliciosos trabalham conjuntamente para gerar problemas como *loops* e falsos pacotes de roteamento, além da escolha de caminhos (rotas) não-ótimos. Este ataque é de difícil detecção, pois para os reais nós da rede, o funcionamento está correto, embora de fato esteja apresentando anomalias do tipo pacotes falsos, alterados e descartados.

Redes baseadas em protocolos de roteamento como OLSR (*Optimized Link State Routing Protocol*) e AODV (*Ad hoc On-demand Distance Vector*) sempre confiam em informações passadas pelos nós vizinhos a rede, ficando suscetível a tomar decisões de roteamento incorretas caso existam nós enviando informações incorretas para a rede.

c) Estouro da tabela de roteamento

É um ataque ativo que se baseia no fato de protocolos pró-ativos armazenarem todas as rotas anunciadas pelos nós vizinhos. O objetivo deste ataque é realizar o anúncio de diversas rotas para nós inexistentes, de modo que a tabela de roteamento aumente progressivamente a tal ponto que ela estoure e os nós não consigam mais armazenar as rotas reais. Este ataque pode ser grave em redes em que os nós possuem recursos escassos, onde a recepção de um número excessivo de mensagens e o estouro de *buffer* são cruciais.

d) Replicação de pacotes

É um ataque ativo que possui dois objetivos principais: ocupar o meio de transmissão e aumentar o desperdício de recursos. Para alcançar este objetivo um nó envia réplicas de pacotes antigos ou atuais para a rede, impedindo a transmissão de outros nós nos momentos em que está enviando as réplicas.

e) “Envenenamento” de c*ache*

Similar ao ataque de estouro de tabela de roteamento, esse ataque aproveita-se de protocolos de roteamento reativos como o AODV, que mantém rotas para nós em *cache*. O objetivo é “envenenar” a *cache* de roteamento realizando anúncios falsos de rotas para nós reais.

f) Inundação de mensagens *HELLO*

É um ataque ativo onde o nó malicioso envia mensagens *hello*, informando que o nó possui enlaces de boa qualidade com determinados destinos, atingindo assim um grande número de nós, que por terem recebido a mensagem, o colocam na lista de vizinhos e podem escolhê-lo para encaminhamento de dados, podendo fazer com que vários nós da rede apontem suas rotas de encaminhamento para um nó inalcançável.

A prevenção para estes ataques e muitos outros que não são tão difundidos é por exemplo, a limitação do número máximo de rotas nas tabelas de roteamento, autorização, assinaturas digitais, uso de múltiplos caminhos e utilização de pacotes de investigação.

Algumas destas técnicas de prevenção já são utilizadas em protocolos de roteamento mais robustos que propõem implementações de segurança. Estes protocolos e as técnicas utilizadas serão descritos no item 2.5.

Baseado na Figura 2.5a abaixo, vislumbra-se dois exemplos de ataques realizados por um mesmo atacante. No primeiro, o AP_2 é invadido e no segundo, acontece um ataque DoS, baseado em inundação (*jamming),* entre o AP_5 e o AP_6. O atacante corrompendo o AP_2 pode obter seus dados e comprometer a integridade e confidencialidade de todo o tráfego que passe por ele, assim como de todos os clientes associados aos AP_2, AP_3 e AP_4. Por outro lado, um ataque DoS é um modo muito simples e eficiente para particionar uma rede sem fio, forçando a sua reconfiguração.

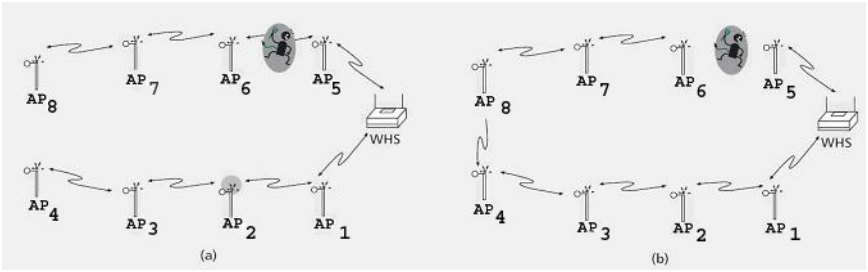

Figura 2.5. Exemplo de ataques em redes *mesh* (Fonte [Salem e Hubaux 2006])

É imperativa a descoberta desses ataques em busca de uma reação de acordo com a sua necessidade. Uma possível reação para o ataque de um AP invadido (Figura 2.5a) pode ser a substituição, pelo operador de rede, por outro íntegro (Figura 2.5b), entretanto, nem sempre isso é possível.

A descoberta e anulação do atacante que esteja inundando a rede podem ser ações trabalhosas, dependendo da capilaridade da rede. Além da dificuldade na localização física e lógica do atacante, ainda que ele seja encontrado, o operador de rede pode não ter permissão para acessar a máquina que está sendo utilizada por ele. Neste caso, a reconfiguração da rede é inevitável.

As mudanças de conectividade podem afetar o desempenho do roteamento na rede e aumentar o número de saltos a partir de um AP de/para o WHS. Por exemplo, na Figura 2.5a, AP_6 estava há dois saltos de distância do WHS e após a reconfiguração da rede, Figura 2.5b, passou a estar há sete saltos.

2.2.2. Desafios

De forma generalizada, são quatro as principais limitações em uma rede sem fio ou em qualquer sistema que tenha dispositivo móvel (*notebooks*, *PDAs*, aparelhos celulares etc):

- Capacidade do processador: costuma ser menor nos dispositivos móveis, tornando o processamento computacional mais lento e demorado.
- Capacidade de bateria: costuma ser limitada, não sendo recomendável o uso dos dispositivos móveis para a realização de processamento de aplicações que necessitem de grande poder computacional.
- Mobilidade: os dispositivos podem, durante sua utilização, estarem em movimento, o que pode produzir uma maior latência na convergência da rede.
- Largura de banda: no geral, costuma ser limitada.

Através da relação dessas limitações com as redes *mesh*, surgem metas a serem alcançadas para superarem os desafios criados:

- Os enlaces na rede *mesh* serão propensos a ataques ativos, passivos e distorções de mensagens. Desta forma, os ataques ativos podem resultar na violação da disponibilidade, integridade e autenticação na/da rede. Os ataques passivos podem comprometer a confidencialidade na/da rede.

- Existe a possibilidade de um nó da rede ser comprometido, em função da falta de proteção física. Conseqüentemente, a rede fica suscetível a ataques maliciosos de fora e de dentro da rede.

- Em função do dinamismo que mudanças freqüentes na topologia da rede podem ocasionar, essa natureza *ad hoc* pode prejudicar a relação de confiança entre os nós da rede.

- Os dispositivos de baixo custo possuem limitações das suas capacidades de armazenamento/memória e de processamento computacional, desta forma, os esquemas tradicionais de segurança não são aplicáveis nas redes *mesh*.

Não se tem como esgotar todos os desafios de segurança em redes *mesh*, entretanto, optou-se pela abordagem de três focos [Salem e Hubaux 2006], conforme mencionado no item 2.1.

Além disso, é válido ressaltar que o assunto segurança em um nível de abstração maior nas redes *mesh* possuem semelhanças com qualquer outro sistema de comunicação e também será abordado abaixo [Siddiqui e Hong 2007].

2.2.2.1. Descoberta de APs corrompidos

As redes *mesh* fazem uso, no geral, de dispositivos de baixo custo que não podem ser protegidos contra remoção e alterações/modificações. Um atacante pode capturar um AP e modificá-lo de diversas formas, ainda que remotamente. Por outro lado, um WHS é assumido como protegido fisicamente. Desta forma, pelo menos quatro tipos de ataques podem ser visualizados:

- A simples remoção ou substituição de um AP da rede. A proteção física de um AP é crucial. Este ataque pode ser descoberto pelo WHS ou pelos APs vizinhos quando uma mudança incomum na topologia da rede for vista, bem como se a topologia da rede for permanentemente monitorada.

- Um ataque passivo. A obtenção de acesso ao AP capturado sem modificá-lo é um ataque passivo de difícil descoberta, uma vez que nenhuma mudança é realizada no AP. Entretanto, o atacante terá total controle do AP corrompido e poderá ter acesso a todo o tráfego que passar por ele.

- A obtenção de acesso ao AP corrompido com a realização de modificações na sua configuração.

- A clonagem do AP corrompido e a instalação de réplicas em alguns locais estrategicamente escolhidos na rede *mesh*, que permitam ao atacante injetar falsos dados ou desconectar partes da rede, podendo comprometer seriamente o mecanismo de roteamento utilizado na rede.

2.2.2.2. Roteamento seguro através de múltiplos saltos

Através de um mecanismo de roteamento inseguro, o atacante pode alterar a topologia da rede, conseqüentemente afetar ao seu funcionamento. Um ataque desse tipo pode ter origem “racional” ou “malicioso”.

Por exemplo, um atacante malicioso pode querer dividir a rede, ou isolar um AP ou uma determinada região geográfica coberta pela rede. No caso de um atacante racional, este pode querer forçar o tráfego por um AP específico na rede para monitorar o tráfego de um determinado cliente móvel ou região.

Um ataque ao mecanismo de roteamento inseguro pode ser realizado pelo atacante através de modificações nas mensagens trocadas, inclusive as de controle, alterando assim os seus estados ou ainda o estado de um AP na rede, em especial através de ataques de DoS, os quais representam um modo simples e eficiente para atacar APs corrompidos.

Para prevenir ataques contra as mensagens, pode-se usar protocolos de roteamento seguros, conforme será abordado no item 2.5.2. Se o atacante escolher modificar o estado de um ou mais APs na rede, o ataque poderá ser descoberto e a rede reconfigurada.

2.2.2.3. Justiça

Nas redes *mesh*, todos os APs usam o mesmo WHS como saída de/para a rede infra-estruturada, comumente sendo a Internet. Desta forma, o processamento realizado pelos APs pode variar, dependendo significativamente da sua posição dentro da rede e da própria topologia da rede. Os APs que estejam há mais de dois saltos do WHS podem ser penalizados em seu acesso, por exemplo os clientes podem não conseguir enviar/receber tráfego, caso não haja nessa rede um mecanismo de qualidade de serviço implementado.

Entretanto, simplesmente garantir banda para um AP, não é a melhor solução para as redes *mesh*. Considere a Figura 2.6 abaixo, onde uma política justa por AP conduziria os fluxos 1, 2, e 3 cada, tendo a mesma banda compartilhada, sem levar em consideração o número de clientes associados a cada um do APs.

O conceito de justiça é relacionado ao número de saltos entre os APs e o WHS. Isto significa que se o atacante conseguir aumentar o número de saltos entre um determinado AP e o WHS, poderá diminuir a banda compartilhada por esse AP.

Uma possível solução contra este ataque poderia ser a reconfiguração periódica da rede, uma vez que alguns APs são fixos (os que compõem o *backcbone* da rede *mesh*) e possuem um poder computacional maior que os dispositivos dos clientes/usuários, o operador da rede pode definir, baseado no tráfego da rede, a melhor configuração para esta, definindo as rotas ótimas.

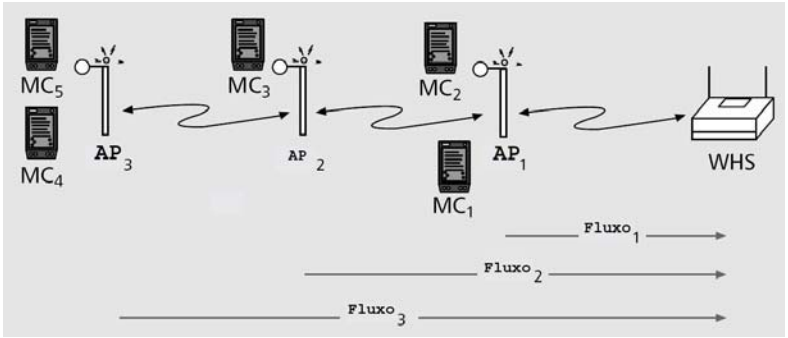

Figura 2.6. Problema de injustiça (Fonte [Salem e Hubaux 2006])

Uma vez que a rede tenha uma configuração ótima, é possível usar algum tipo de mecanismo para assegurar justiça de/por clientes/usuários e otimizar a utilização da largura de banda na rede.

2.2.2.4. Disponibilidade

Disponibilidade assegura o acesso aos serviços de rede ainda que esta esteja sob ataque, não sendo considerado como um problema da importância da confidencialidade e integridade, porém a sua garantia é um assunto relacionado com segurança. Além disso, os processos exigidos para prevenir os efeitos de perda de disponibilidade estão relacionados aos mecanismos de segurança, uma vez que o conceito básico de disponibilidade é assegurar que os usuários autorizados tenham acesso contínuo à informação do sistema, sem interrupções.

A disponibilidade em uma rede *mesh* pode ser atingida através de: Inundação de sinal - *Signal Jamming* (Nas camadas física e de acesso ao meio, um atacante pode prejudicar a disponibilidade da rede inundando um canal de comunicação utilizado); Ataque DoS/DdoS - *Denial of Service /Distributed Denial of Service (*Podem ser lançados em qualquer camada de uma rede *mesh* ou ainda através de um atacante externo à rede ou até mesmo a partir de um nó interno; e Ataque de esgotamento de bateria (A duração de uma bateria é um dos parâmetros mais críticos para a maioria dos nós de uma rede sem fio).

Ataque de esgotamento de bateria também conhecido como *"sleep deprivation attack"* é uma ameaça real e em algumas situações, mas prejudiciais que um ataque DoS.

2.2.2.5. Autenticidade

Autenticidade permite a um nó da rede assegurar a identidade do nó com o qual se esteja estabelecendo uma comunicação. Sem autenticidade, um atacante poderia mascarar um nó da rede, ganhando acesso sem autorização a recursos e informações, interferindo no funcionamento da rede.

Com a implementação de conceitos como sistemas onipresentes, a quantidade de nós em uma rede *mesh* pode ser significativa. Entretanto os mecanismos de autenticação

habituais envolvem sistemas centralizados que administram através de restrições (listas de acessos) ou certificados, o acesso na/da rede.

Desta forma, em uma rede *mesh*, a presença de um servidor centralizado, nem sempre é viável ou possível, em função da característica de dinamismo e crescimento orgânico, porém existem algumas formas, como por exemplo: Associação Temporária Segura (*Secure Transient Association*) e *Imprinting*.

2.2.2.6. Integridade

O conceito de integridade assegura que os conteúdos dos dados são preservados durante uma transferência entre emissor/receptor, garantindo desta forma que uma mensagem recebida foi a mesma enviada originalmente, ou seja, não foi alterada de forma intencional ou não, durante a transmissão. Os mecanismos comuns para assegurar a integridade de dados geralmente baseiam-se no uso de funções *hash* e de mensagens, criptografia ou uma combinação entre eles.

a) Criptografia e assinaturas digitais

Quando os nós suportam (possuem poder computacional e bateria suficiente) o uso de assinaturas digitais, chaves públicas de criptografia podem ser utilizadas para que os nós gerem a chave privada e através dela troquem mensagens seguras e protegidas.

b) Compartilhando pares de chaves (*Pair-Wise Key Sharing*)

Se o nó tiver limitações de poder computacional e/ou capacidade de bateria, mecanismos com criptografia simétrica também podem ser utilizados.

2.2.2.7. Confidencialidade

O conceito de confidencialidade é a garantia de que dados estejam sendo acessados pelos usuários/clientes que são autorizados a isso. Para garantir confidencialidade, primeiramente é necessário que a autenticidade da informação seja válida. Uma vez que a autenticidade seja garantida, a confidencialidade estará associada a mecanismo que use criptografia.

2.2.3. Metas

Objetivando a segurança em redes *mesh*, alguns atributos são considerados e desejados, como por exemplo: roteamento seguro, sistemas de detecção de intrusão, gerenciamento confiável e de chaves [Siddiqui e Hong 2007].

2.2.3.1. Roteamento seguro

Para garantir disponibilidade da rede, os protocolos de roteamento devem ser robustos contra ataques maliciosos e suportar a topologia dinâmica da rede *mesh*. A maioria desses protocolos assume uma colaboração confiável entre os dispositivos envolvidos na comunicação, mas existem várias ameaças de segurança que surgem a partir de falhas nos protocolos e/ou da falta de mecanismos de autenticação. Alguns ataques em protocolos de roteamento são apresentados na Tabela 2.1 abaixo.

Há duas fontes de ameaças aos protocolos de roteamento. A primeira a partir de atacantes externos, injetando na rede informações erradas de roteamento, podendo dividir uma rede ou introduzir carga excessiva de tráfego causando retransmissões e um

roteamento ineficiente. A segunda é o tipo mais severo de ameaças, pois é originada a partir de nós comprometidos na própria rede, anunciando informação incorreta aos outros nós.

Tabela 2.1. Ataques durante o roteamento (Fonte [Siddiqui e Hong 2007])

Fase do roteamento	Ataques
Descoberta de rotas	Estouro das tabelas de rotas.
Manutenção das rotas	Mensagens de controle com falsas rotas.
Encaminhamento dos dados	Descarte de dados; Consumo de recursos; *Worms* etc.

Para proteger a rede do primeiro tipo de ameaça, os nós podem proteger as informações de roteamento da mesma maneira que os dados são protegidos, porém essa defesa é ineficaz contra ataques a partir de servidores corrompidos. A descoberta de um nó comprometido através de informações de roteamento é difícil em uma rede *mesh* em virtude da característica da possibilidade de mudanças dinâmicas na sua topologia. Uma idéia básica seria transmitir informação redundante por rotas adicionais para a descoberta e correção de erros.

2.2.3.2. Sistemas de Detecção de Intrusão

Algumas das características das redes *mesh*, como por exemplo a ausência de um ponto centralizado para gerenciamento e monitoração, não possibilitam que as técnicas tradicionais de detecção de intrusão (*IDS - Intrusion Detection System*) sejam utilizadas.

Sistemas de detecção de intrusão são sistemas de defesa contra atividades hostis em uma rede de computadores e procuram prevenir contra ataques que visam comprometer sistemas de segurança [Mishra 2004]. As prevenções envolvem tantos alertas das mais variadas formas como ações executadas pelo próprio sistema de detecção de intrusão como bloqueio de portas ou conexões suspeitas. Nas redes *mesh* a cooperação é muito importante para suportar as funções básicas da rede. Desta forma mecanismos baseados em *tokens* podem ser utilizados para garantir esta cooperação.

Um sistema de detecção de intrusão coleta informações das atividades de todos os nós e a partir disso, analisa-as para determinar se há qualquer atividade que esteja violando as regras de segurança. Uma vez que seja detectada uma atividade incomum ou que seja conhecida como um ataque, algum tipo de alarme é gerado para alertar ao administrador de segurança. Além disso, o sistema de detecção de intrusão pode iniciar uma resposta à atividade maliciosa.

Para as redes *mesh*, uma solução de detecção de intrusão pode depender da própria infra-estrutura de rede, podendo ainda ser classificada como: *Stand-alone (*O sistema de detecção de intrusão roda de forma independente e isolada em cada nó para a detecção de intrusões); Distribuído e cooperativo (Todo nó participa da detecção de intrusão e através de um agente que é executado em todos eles, identifica possíveis intrusões e inicializa respostas a elas); e Hierárquico (Nós *"Clusterheads"* atuam como pontos de controle para prover a funcionalidade aos seus nós associados).

As formas de respostas a intrusões para redes *mesh* dependem do tipo de intrusão, dos protocolos de rede utilizados e aplicações em uso na rede. Algumas respostas são: reinicialização dos canais de comunicação entre nós, identificação de nós

comprometidos da rede e reorganização da mesma, notificação de usuários para que os mesmos possam realizar suas próprias investigações e tomar as devidas ações e como última ação do IDS, pode-se reiniciar a requisição de autenticação de todos os nós da rede. Um sistema de detecção de intrusão pode ainda ser dividido em três categorias:

a) Detecção de Anomalias

Em um sistema de detecção de anomalias é criado um perfil de uma atividade normal do sistema. Qualquer atividade do sistema que estiver fora do perfil é tratada como uma possível intrusão. A grande desvantagem para redes sem fio é que o perfil de atividades deverá ser constantemente atualizado e as discordâncias do perfil criado, terão de ser registradas, isto pode causar uma sobrecarga em dispositivos com baixo poder de processamento.

b) Detecção de abusos

Na detecção de abusos, as decisões são tomadas baseadas em uma base de conhecimento de um modelo de processo intrusivo e vestígios de prováveis intrusões.

c) Detecção baseada em especificidades

Sistemas de detecção de intrusão baseados em especificidades definem uma série de diretivas que descrevem o correto funcionamento de um determinado programa ou protocolo e os monitora de acordo com as diretivas. Esta técnica é a que provê maior capacidade de detecção prévia de ataques e o menor número de falsos positivos.

2.2.3.3. Gerenciamento Confiável e de Chaves

Confidencialidade e segurança são dois conceitos mutuamente dependentes que não devem ser separados. As redes *mesh* baseiam-se em relações de confiança entre seus nós vizinhos. Como o ambiente global é cooperativo, estas relações de confiança são extremamente suscetíveis a ataques. Além disso, a ausência de infra-estrutura de confiança fixa, recursos limitados, meio sem fio compartilhado e vulnerabilidades físicas, torna o estabelecimento de uma confiança inviável.

Desta forma, nas redes *mesh* as características de gerenciamento confiável em oposição aos mecanismos tradicionais e centralizados, são: confiança em evidências incertas e incompletas, confiança na troca de informações das localidades, computação distribuída e avaliação de confiança empregada individualmente.

Para superar estes problemas, confidencialidade está sendo estabelecida em redes *mesh* usando por exemplo a pré-configuração dos nós com chaves criptográficas ou a presença de uma autoridade certificadora. Todos os esquemas baseados em chaves de criptografia demandam um serviço de gerenciamento de chaves, sendo responsável por manter o histórico das relações entre as chaves e os nós, estabelecendo uma confiança mútua e comunicação segura entre os nós.

Gerenciamento de chaves através de um canal de comunicação inseguro possui um risco alto a ataques. Existem três tipos de gerenciamento de chaves que podem ser aplicados em redes *mesh*: autenticação via certificado, árvores de certificação e a combinação de ambos.

2.3. Aspectos de Roteamento

Os protocolos de roteamento para redes *mesh* têm papel importante no que diz respeito à comunicação entre as partes envolvidas. Este tipo de rede é caracterizado pela presença de nós moveis e fixos interconectados através de enlaces sem fio formando uma rede de múltiplos saltos.

Alguns pontos importantes devem ser considerados para que estes protocolos sejam desenvolvidos de uma forma a aproveitar as características da topologia em questão. Estes pontos incluem: topologias dinâmicas (nós podem se mover livre e randomicamente dentro de uma mesma rede ou entre redes); recursos limitados como banda (na realidade, as redes sem fio ainda estão trabalhando nas faixas de MHz) e energia (equipamentos portáteis são operados a bateria); segurança (nós não participantes da rede podem "escutar" as transmissões); e escalabilidade (devido ao alto *overhead* advindo do grande número de nós presente na rede) [Held 2005].

Observa-se que, para que os protocolos de roteamento ofereçam um serviço de qualidade, eles devem considerar métricas multidimensionais [Faccin et al. 2006] para que a melhor rota seja selecionada de acordo com a aplicação e cenário onde os mesmos estão atuando. É importante frisar que as redes *mesh* tiveram suas origens nas redes *ad hoc* e devem ser capazes de se auto administrar, configurar e restabelecer no caso de perda de enlaces [Bruno et al. 2005]. Portanto, este tipo de rede demanda protocolos de roteamento que, também, tenham estas características.

De acordo com [Murthy e Manoj 2004], os protocolos podem ser classificados considerando vários aspectos que podem ser baseados: Nos mecanismos de atualização de informação; No uso de informação temporal; Na topologia; E na utilização de rotas específicas. A classificação utilizada neste mini-curso é a baseada nos mecanismos de atualização de informação. Neste tipo de classificação os protocolos são divididos em roteamento:

- Pró-ativo: Onde os nós trocam tabelas de roteamento periodicamente mantendo informações sobre toda a topologia com cada nó conhecendo o menor caminho para cada outro nó da rede. Os protocolos mais comumente utilizados são o DSDV (*Destination Sequenced Distance Vector*) [Perkins 1994] e o OLSR (*Optimized Link State Routing*) especifidaco pela [RFC3626 2003].
- Reativo: Com as rotas sendo estabelecidas sob demanda, ou seja, rotas são criadas quando solicitadas pelo transmissor sendo o DSR (*Dynamic Source Routing*) [Song 2005] e o AODV (*Ad Hoc On-Demand Vector*) definido pela [RFC3561 2003] os mais utilizados.
- Híbridos: Que combinam as melhores características dos protocolos pró-ativos (roteamento dentro da mesma zona) e reativos (roteamento para fora da zona). Como exemplo deste tipo de roteamento tem-se o ZRP (*Zone Routing Protocol*) [Haas 1997] [Haas e Pearlman 1998] e o CEDAR (*Core-Extraction Distributed Ad hoc Routing*) [Sivakumar et al. 1998].

A maioria dos protocolos de roteamento nas redes *mesh* funciona de forma cooperativa onde os nós conhecem seus vizinhos. Com isso, aplicações maliciosas podem parar a rede facilmente inserindo atualizações erradas nas tabelas de roteamento.

As soluções de protocolos de roteamento seguros são específicas para combater ataques ou grupo de ataques, classificando-se em soluções baseadas em criptografia e/ou em adaptações de protocolos existentes.

Para aumentar a segurança das redes *mesh*, duas estratégias precisam ser adotadas: embutir mecanismo de segurança nos protocolos de roteamento; e desenvolver sistemas de monitoramento e resposta perante descobertas de ataques e quedas de serviços.

2.3.1. Protocolos de Roteamento não seguros

Para cada necessidade pode-se empregar um protocolo de roteamento e esta característica deve-se ao fato de que um único protocolo não consegue ajustar-se a cenários diferenciados e muito menos a vários padrões de tráfego.

Protocolos de roteamento pró-ativos adequam-se muito bem em redes sem fio de banda larga, baixa escalabilidade e de considerável mobilidade dos nós, enquanto que protocolos reativos adequam-se melhor em redes de banda estreita, alta escalabilidade com baixa mobilidade de nós. Já os protocolo híbridos procuram adequar-se a redes em ambas as situações.

Dentre as soluções de protocolos de roteamento existentes para redes *ad hoc* e redes *mesh* encontram-se como propostas os protocolos OLSR, DSR, DSDV, AODV, ZRP e ZHLS (*Zone-Based Hierarchical Link State*) [Joa et al. 1999].

2.3.1.1. OLSR

Este protocolo é uma versão melhorada do clássico algoritmo de estado de enlace para atender as necessidades de redes sem fio. Assim, o protocolo herda a estabilidade do algoritmo clássico com a vantagem de ter as rotas disponíveis imediatamente devido sua natureza pró-ativa.

O ponto principal deste protocolo está na utilização dos MPRs (*Multipoint Relays*) que é um subconjunto de nós vizinhos selecionados responsáveis pelo encaminhamento de tráfego de controle. Com isso, o *overhead* proveniente do tráfego de controle é reduzido se comparado com o mecanismo de inundação clássico onde cada nó retransmite cada mensagem quando o mesmo as recebe.

Os nós MPRs devem anunciar para a rede, em suas mensagens de controle, que são deste tipo com o intuito de informar sua alcançabilidade aos nós que o elegeram como MPR. Neste protocolo, somente os nós MPR podem gerar informação de estado de enlace, minimizando, assim, o número de mensagens de controle na rede.

Como um nó MPR deve reportar sobre somente os enlaces entre ele mesmo e os nós que o selecionaram como nó MPR, isso leva a outra vantagem deste protocolo se comparado ao algoritmo de estado de enlace clássico, já que as informações de estado de enlace parciais encontram-se distribuídas na rede. Esta informação é usada para os cálculos de melhor rota em termos de números de saltos.

O OLSR define três tipos básicos de mensagens: *Hello (*transmitidas entre os vizinhos e são utilizadas para obter informações de enlace, detecção de vizinhos e sinalização MPR); *Topology control* (são sinalizações, anúncios de estado de enlace, feitas pelo OLSR, sendo que a transmissão destas mensagens é otimizada de várias

formas usando MPRs); e *Multiple interface declaration* (responsável em informar sobre a presença de mais de uma interface em um nó, sendo que essas mensagens listam todos os endereços IP usados por este nó).

2.3.1.2. DSR

É um protocolo de roteamento pró-ativo, onde um remetente determina a seqüência completa das estações responsáveis pelo transporte de pacotes até um destinatário. Diferente dos protocolos existentes em redes *ad hoc*, o DSR não realiza freqüentemente os anúncios de rota. Ao invés disso, quando uma estação necessita de uma rota para um destino, ela faz uma consulta nas informações em *cache* e nos resultados obtidos com o protocolo de descoberta de rotas.

Assim, para enviar um pacote para outra estação, o remetente constrói uma "rota de origem" no cabeçalho do pacote, informando o endereço de cada estação, através do qual o pacote será encaminhado ordenadamente até alcançar o destino.

O remetente então transmite o pacote no enlace sem fio para a estação que estiver a distância de um salto e identificada na "rota de origem". Quando esta estação recebe o pacote, se ela não for o destinatário do pacote, ela simplesmente retransmite para a próxima estação identificada.

Cada nó móvel da rede armazena em *cache* as rotas que foram descobertas. Se a rota não for encontrada em *cache*, o remetente então usa o protocolo de descoberta de rotas, que permite que o nó remetente envie uma solicitação de rota e fique recebendo pacotes de outros nós até que a informação de rota retorne. O nó poderá armazenar o pacote original em ordem para transmitir enquanto a rota estiver em processo de descoberta, ou descartar o pacote.

2.3.1.3. DSDV

É um protocolo de roteamento pró-ativo que tenta manter a simplicidade do algoritmo de vetor distância e ao mesmo tempo corrigir dois problemas que impediam seu uso nas redes *ad hoc*: a possível criação de *loops* pelo algoritmo de vetor de distância e a necessidade de um ajuste mais rápido da topologia da rede a medida que seus nós se movem.

O DSDV possui o atributo de número de seqüência, para cada entrada na tabela de roteamento do protocolo RIP (*Routing Information Protocol*) convencional. Através da utilização deste novo atributo, os nós móveis podem identificar que a informação de uma rota está desatualizada evitando a formação de *loops* no roteamento.

Cada nó em uma rede *ad hoc* mantém uma tabela que lista todos os possíveis destinos dentro da rede. Cada entrada nesta tabela contém o endereço de um possível destino, a menor distância conhecida (normalmente em número de saltos) para aquele destino e o endereço do nó vizinho que será o primeiro salto neste caminho mais curto para aquele destino. A distância para o destino é conhecida como *metric* nas entradas da tabela. Quando um nó estiver realizando o roteamento de um pacote para algum destino, o nó transmite o pacote para o roteador (nó) vizinho indicado, e cada roteador usa sua própria tabela para realizar o próximo salto do pacote em busca do seu destino.

2.3.1.4. AODV

Este protocolo combina características do DSR (mecanismos sob demanda de descobrimento e manutenção de rotas) e do DSDV (roteamento salto-a-salto, números de seqüência e atualizações periódicas) [Broch et al. 1998].

Quando um nó deseja enviar dados a um destino e ainda não tem rotas para o mesmo, o nó fonte inicia o processo de descoberta de rota. Via *broadcast*, o pacote RREQ (*Route Request*) é enviado a todos os vizinhos do nó fonte e cada vizinho, por sua vez, encaminha aos seus vizinhos até que o destino ou nó intermediário, que contenha esta rota, seja alcançado.

Para evitar *loops* e ter rotas recentes, o AODV utiliza números de seqüência. Cada nó tem seu próprio número de seqüência e uma identificação de *broadcast,* incrementado para cada RREQ gerado pelo nó, que vão juntos com o número de seqüência mais recente do destino, distinguindo, assim, cada RREQ.

Desta forma, os nós intermediários somente respondem ao RREQ se tiverem uma rota com número de seqüência maior ou igual ao presente no RREQ. Durante o encaminhamento do RREQ, cada nó intermediário armazena em sua tabela de rotas o endereço do vizinho do qual ele recebeu o RREQ primeiramente com o objetivo de descartar cópias do mesmo RREQ e estabelecer um caminho reverso até a fonte.

Uma vez que o RREQ chega ao destino/nó intermediário, este envia um pacote RREP (*Route Reply*) ao vizinho do qual ele recebeu o primeiro RREQ e este processo continua até que RREP chegue à fonte.

O protocolo AODV periodicamente envia mensagens *hello* para saber se há novos nós na sua vizinhança e estas mensagens podem ser utilizadas com o objetivo de manter a conectividade local.

2.3.1.5. ZRP

É um protocolo de roteamento híbrido, que faz uso das vantagens de um protocolo pró-ativo e das vantagens de um protocolo reativo.

A especificação do protocolo não define se há um protocolo pró-ativo ao qual o ZRP herda peculiaridades, mas é possível identificar algumas características próprias como o fato de necessitar que sejam enviadas mensagens periódicas para manter o roteamento local, suporte a dispositivos de fabricantes diferentes, desde que cada um tenha um IP único associado a cada interface de rede.

O protocolo é naturalmente distribuído e não depende de uma unidade central e ainda existe a prevenção de *loops* no roteamento através de número de seqüência e de endereço de origem.

2.3.1.6. ZHLS

É um protocolo de roteamento híbrido onde a rede é dividida em zonas sem sobreposição. Ao contrário de outros protocolos hierárquicos, não existe um representante da zona (*zone-head*). Define dois níveis de topologia: nível do nó e nível da zona.

A topologia de nível do nó informa como os nós de uma zona estão conectados fisicamente entre si. Um enlace virtual entre duas zonas existe se pelo menos um nó de uma zona está fisicamente conectado a um nó de outra zona.

A topologia de nível da zona informa como as zonas estão conectadas. Existem dois tipos de pacotes de estado de enlace (LSP – *Link State Packet*), que são o "*node LSP*" e "*zone LSP*". O "*node LSP"* de um nó contém a informação dos seus nós vizinhos e é propagado dentro da sua zona, enquanto que o "*zone LSP"* contém a informação da zona e é propagado globalmente.

Desta forma cada nó possui um conhecimento total sobre a conectividade dos nós da sua zona e somente a informação da conectividade zonas do resto da rede. Então dado a identificação da zona e do nó de um destino, o pacote é roteado baseado na identificação da zona até que este chegue na zona correta, quando então é roteado baseado na identificação do nó. O par <id. zona, id. nó> do destino é suficiente para o roteamento, portanto é adaptável para mudanças de topologia.

2.4. Aspectos de Gerenciamento

As facilidades proporcionadas pelas redes tendem a estimular o crescimento das mesmas, provendo necessidades de manutenção e planejamento. Em um ambiente com poucas máquinas conectadas em rede, uma única pessoa é capaz de gerenciá-las. Mas, considerando um ambiente onde a rede esteja distribuída entre várias salas, ou até mesmo prédios distintos, a manutenção torna-se mais difícil consumindo tempo e recursos.

Em redes de longa distância, a tarefa de gerenciamento é inerentemente mais complexa e indispensável, uma vez que cobre uma grande área geográfica e envolve um grande número de equipamentos e usuários dependentes de seus serviços. Além disso, com a grande utilização e a heterogeneidade das redes, tem havido a necessidade de um esquema que ofereça soluções de gerenciamento de redes coerentes e estruturadas, permitindo o monitoramento e o controle de equipamentos [Kurose 2006].

O gerenciamento de redes *mesh* é uma tarefa significativamente mais complexa do que controlar redes com infra-estrutura cabeada, também é diferente de gerenciar redes puramente do tipo *ad hoc*. As redes *mesh* por serem um tipo híbrido de rede, com características de redes infra-estruturadas e *ad hoc*, não são atendidas por completo pelas ferramentas existentes, pois estas são usualmente desenvolvidas frente a um conjunto de requisitos pertencentes ao tipo de rede infra-estruturadas ou do tipo *ad hoc* [Karpijoki 2001].

Como breves exemplos destas diferenças nas características, as redes infra-estruturadas possuem topologias totalmente fixa, e com enlaces que sofrem pouca variação de desempenho, por outro lado, as redes *ad hoc* possuem uma forte limitação no fornecimento de energia por não terem nenhuma ligação com alguma infra-estrutura fixa e também devido a grande mobilidade dos nós.

Diferentemente das redes cabeadas, as redes *mesh* ainda não têm um padrão público definido pelos órgãos reguladores. Como conseqüência, não existe uma grande quantidade de ferramentas de gerência voltadas para redes *mesh*. As ferramentas encontradas são desenvolvidas normalmente por fornecedores que buscam atender as suas soluções *mesh* especificamente, como [MotoMesh 2008]. Outro ponto negativo, é

que essas ferramentas específicas desses fornecedores, não possuem código aberto, ou até mesmo uma licença GPL (*General Public License*) que permita a realização de adaptações necessárias a outras redes.

Um importante problema em redes *mesh*, está relacionado ao processo de coleta de dados, este processo causa sobrecarga (*overhead*) [Badonnel 2005]. Redes que utilizam enlaces segundo o padrão IEEE 802.11 [802.11 2007], possuem uma limitada largura de banda, que pode sofrer grandes variações e, portanto, as mensagens de gerência não devem consumir uma porção significativa do meio de comunicação.

A solução mais simplista para extrair as informações da rede, considerando que as ferramentas de gerência serão implementadas ao nível de aplicação, acessar individualmente cada ponto da rede e, assim, recolher os dados desejados.

Esta técnica pode resultar em uma utilização ineficiente dos recursos de comunicação da rede, levando a uma grande sobrecarga. A quantificação deste impacto negativo gerado pela troca de mensagens torna-se difícil de predizer ou controlar, pois a qualidade dos enlaces de comunicação pode variar de uma maneira muito dinâmica. Simplesmente impor um limite de vazão no tamanho das mensagens pode não ser uma solução viável.

Uma rede *mesh* tem como vantagem possuir na sua topologia pontos fixos, que formam um *backbone* (infra-estrutura de comunicação utilizada pelos clientes). Ou seja, a maior parte da comunicação da rede passa, ou deve passar, por estes equipamentos. Outras características adicionais destes pontos são, em sua maioria, alimentação irrestrita de energia, posicionamentos fixo, e acesso de gerência por um terminal remoto através de múltiplos saltos.

Uma questão que deve ser levantada para o gerenciamento de redes *mesh* é relativa a tarefa de monitoramento, o que deve ser observado é a propriedade temporal da informação [Duarte 2008]. Esta propriedade é determinada pelos requisitos de gerência. Por exemplo, a tarefa de visualizar a topologia da rede, em tempo real, necessita que as informações sejam processadas em tempo, o mais real possível, senão a representação não será precisa o suficiente. De outra forma, um outro exemplo é a tarefa de obter o histórico de estatísticas de algum ponto fixo, que possui uma restrição de tempo muito baixa.

Para proteção dos mecanismos básicos de operação da rede a solução mais intuitiva é proteger suas trocas de mensagens, assim como deve ser feito com as informações dos usuários da rede. Para tanto devem ser adotados esquemas de criptografia adaptados para estes ambientes.

Sendo assim, chega-se ao ponto mais vulnerável do sistema de segurança que é o gerenciamento das chaves do esquema de criptografia. Por razões de eficiência um bom esquema faria uso de chaves simétricas para a autenticação, e dentro desse contexto, o estabelecimento seguro de chaves simétricas que seriam posteriormente utilizadas para a comunicação entre os nós.

Este esquema de chaves deve levar em conta características como: as propriedades da autoridade da rede, acessibilidade de um nó em relação a rede, o comportamento da fase de inicialização do esquema, o tipo de relação entre os nós, o tipo de relação entre os nós e a(s) autoridade(s) da rede e a distribuição da confiança na rede.

Dentro desse contexto são empregados esquemas de criptografia, como assinaturas digitais, com infra-estrutura de chave pública, onde cada nó possui um par de chaves (uma pública e a outra privada), e uma entidade confiável, denominada autoridade de certificação.

Esta entidade fica responsável pela associação confiável entre os nós da rede e suas chaves públicas. Enquanto um nó tem confidencialmente sua chave privada, a sua chave pública deve ser anunciada na rede. Cabe ressaltar que o próprio serviço de certificação também possui um par de chaves pública/privada. A diferença nesta abordagem para redes *mesh* é que a responsabilidade de gerenciamento de chaves será distribuída numa comunidade de nós, os quais também precisam ser gerenciados [Tamashiro 2007].

A chave pública do serviço de certificação é conhecida por todos os nós integrantes da rede, mas a chave privada do serviço e dividida e compartilhada entre *n* nós que passam a ser denominados de servidores, que serão as autoridades da rede. Todos os nós da rede podem submeter pedidos de chaves públicas da rede e renovações da própria chave para este serviço de certificação distribuído.

A partir das idéias tratadas acima, serão abordados a seguir os gerenciamentos específicos necessários para se prover uma maior segurança em uma rede *mesh*.

2.4.1. Gerenciamento de Segurança

Tanto redes *mesh*, quanto *ad hoc*, são muito vulneráveis a ameaças de segurança, devido aos nós serem facilmente manipulados, e sinais interceptados, falsificados e etc. Atualmente protocolos, tais como SNMPv3 (*Simple Network Management Protocol v3*), implementam alguns mecanismos de proteção contra escutas e alguns ataques usando transmissões *unicast* seguras [Stallings 1998]. No entanto, há algumas formas de comunicações adicionais que precisam ser consideradas, para assim desenvolver uma maior segurança.

Um protocolo de gerenciamento deve acompanhar constantemente os nós da rede visando determinar se eles são confiáveis ou não. Esta informação deve ser utilizada para determinar se os dados coletados são confiáveis ou sem valor. Da mesma maneira, cada gestor deve ser capaz de determinar se alguma rede esta trocando informações confiáveis ou não.

Outra consideração em relação ao gerenciamento de segurança, diz respeito ao processo de coleta de dados, a fim de se fazer uma coleta de dados eficaz, usando uma árvore de abrangência [Stallings 1998]. Isso permite que um nó intermediário possa combinar dados de diferentes nós (de um nível mais abaixo) antes que repasse as informações para um nó de nível superior, assim diminuindo o tráfego de informações de gerenciamento da rede. O problema deste modelo é a possibilidade de uma violação de segurança.

A formação de *cluster* é a forma mais lógica de divisão de uma rede, a fim de simplificar a tarefa de gerenciamento [Chen 1999]. Essa divisão facilita na descentralização do gerenciamento que torna a rede mais tolerante a falhas e com mensagens eficientes.

Para se manter o baixo *overhead* de mensagens, é mais vantajoso não se tentar manter a par de todos os movimentos de todos os nós da rede. Acredita-se que o

overhead na coleta de tais informações não se justifica, e na maioria dos casos, não é totalmente necessária. Em vez disso, responder aos movimentos apenas quando há mudanças significativas na topologia, torna-se a maneira mais viável de se fazer a coleta de dados necessária.

2.4.2.Gerenciamento de Grupos e Membros

A segurança das redes mais tradicionais baseia-se na existência de uma estrutura permanente, especializada na administração da rede, que define a política de segurança e fornece a infra-estrutura necessária para as suas aplicações. Em redes *mesh*, todos os serviços, incluindo uma infra-estrutura de segurança, devem ser criados e administrados pelos nós que decidiram formar a rede [Maki 2000].

Um conceito básico em muitas redes *mesh* e *ad hoc*, é um grupo de usuários ou nós da rede. Um grupo é um conjunto de entidades que querem se comunicar uns com os outros e de cooperar para algum efeito, ou objetivo. A necessidade de formar um grupo poderia ser uma partilha de aplicação, localização física, ou tarefas administrativas que alguns nós associam uns com os outros e etc.

O gerenciamento de membros dos grupos envolve adicionar e remover nós, no grupo, assim como fornecer um método para autenticar os membros do grupo. Um nó pode revelar a sua filiação em um grupo para nós que não são membros do grupo também. As mais importantes funções de segurança para um grupo são os seguintes [Maki 2000]:

- Autenticação mútua dos membros do grupo.
- Autenticação dos membros do grupo de forasteiros.
- Autenticação dos papéis, em especial com os membros da filiação de gestão.

Grupos em redes *mesh* devem lidar com características especiais de redes *mesh*, e *ad hoc* também. Como conseqüência, os grupos devem ser geridos tão atenciosamente quanto possíveis, independentemente da rede fixa e de serviços e uns com os outros. A gestão deve ser distribuída e tolerante a falhas de segurança física. Os grupos devem ser fáceis de construir e se modificarem.

Os nós de uma rede *mesh* são muitas vezes expostos a perigos físicos, e, independentemente de qualquer garantia, é provável que algumas chaves privadas ou equipamentos que as contenham, possam cair nas mãos de atacantes (indivíduos mal intencionados com relação à integridade da rede).

A reconstituição do grupo é uma maneira segura e, muitas vezes a mais recomendável para continuar apenas com os membros considerados confiáveis na rede [Maki 2000]. No entanto, pode-se adquirir a reconstituição em pleno tempo em que alguns dos nós confiáveis possam estar ocasionalmente fora de alcance. Os membros de um grupo ainda necessitam de alguns mecanismos de cancelamento imediato da filiação, sem sacrificar a adesão dos demais, ainda membros confiáveis.

Infelizmente, cancelar uma adesão que já tenha sido concedida não é uma tarefa simples. Os certificados dos membros podem ser criados, armazenados e verificados simultaneamente em diferentes partes do sistema, característica que torna esse cancelamento um trabalho de certa forma complexo e árduo. Existem algumas formas de se eliminar os membros não confiáveis:

a) Reconstituição de grupo

Para substituir um grupo por um novo, uma nova chave de grupo deve ser gerada e novos certificados de membros e liderança devem ser emitidos para os antigos membros.

A reconstituição pode ser feita periodicamente, ou quando tenham ocorrido mudanças suficientes na composição do grupo. Devido à sua simplicidade conceitual, a reconstituição de grupos deve ser usada como a principal proteção contra membros comprometidos.

É possível a criação do novo grupo, enquanto o antigo ainda esta funcional e, progressivamente, se promover a passagem do antigo para o novo. Dessa forma, a nova chave de grupo e os novos certificados podem ser propagados para todos os nós necessários antes que a antiga chave de grupo seja abandonada.

b) Expiração de membros

A expiração de certificados pode ter um período de validade que é decidido pelas autoridades da rede. Ao optarem por curtos períodos de validade dos certificados, os líderes do grupo podem rever periodicamente o *status* dos membros.

O mecanismo de expiração tem a grande vantagem de uma vez que um membro com certificado expirado, pode ser simplesmente tirado do grupo. Por outro lado, os membros que querem manter-se membros do grupo têm que periodicamente obter novos certificados. Depende ainda se os *clocks* de cada nó da rede estão sincronizados. É significativo lembrar que o nó que emite os certificados de curta duração tem que examinar periodicamente a confiabilidade dos membros.

Assim, com o tempo de vida limitado mantém-se uma consciência dos riscos de segurança e protege-se contra a acumulação de longo prazo do comprometimento de membros. Os nós que geram os certificados podem adaptá-los em função do custo de distribuição dos certificados e renovando a probabilidade de uma solução de compromisso, para cada membro.

c) Extinção de membros

A extinção de membros permite que a rede possa reagir imediatamente contra a possibilidade de uma falha de segurança. No entanto, quando é feita uma decisão de revogação de uma adesão, as informações sobre a revogação devem ser propagadas para todas as partes do sistema onde as certificações podem estar sendo utilizadas.

Uma distribuição pouco eficaz ao longo das conexões pode causar que a informação sobre uma revogação de adesão não chegue a todas as entidades que dela necessitam. Atrasos na propagação da revogação de dados também podem ser consideráveis. Pode-se dar a todos os líderes de um grupo o direito de revogar si próprio e de qualquer outro líder e membro do mesmo grupo. Eles fazem isto através da assinatura de listas de revogação de chaves de grupo pares. As listas que lhes são propagadas no esforço de uma melhor forma a todos os membros do grupo e para outras partes que possam verificar o grupo adesão.

Os membros deverão ser revogados apenas quando existe uma razão para suspeitar que a chave privada foi descoberta por um nó intruso a rede. O custo da

distribuição de listas da revogação é bastante elevado para algumas coisas, mas há situações de emergência em que se torna um mecanismo viável.

Líderes podem ser revogados, mas a operação afetará também todos os membros certificados pela chave revogada. Se a chave privada de um líder foi comprometida, o intruso na posse da chave pode revogar outros membros e lideres. Portanto, num caso em que dois líderes revoguem uns aos outros, é impossível saber qual líder é que teve sua chave comprometida. Por conseguinte, ambas as chaves deve ser revogada, para se precaver com relação a intrusos com "permissões" de líder na rede.

2.4.3.Gerenciamento de Confiança

Confiança e segurança são dois conceitos mutuamente dependentes, que não podem ser separados. Por exemplo, confiança não pode ser garantida sem o controle de comunicações seguras, da mesma forma como atributos de segurança requerem uma criptografia confiável para se realizar as tarefas desempenhadas da rede.

A medida que o ambiente da rede é cooperativo, as relações de confiança entre os nós são extremamente vulneráveis aos ataques. Além disso, a ausência de uma infra-estrutura fixa, os recursos limitados, disponibilidade e conectividade passageira, compartilhamento do meio sem fio e vulnerabilidade física, faz com que o estabelecimento de confiança se torne praticamente uma tarefa complexa [Siddiqui et al. 2008].

Sendo assim, as propriedades únicas de confiança na gestão de uma rede *mesh*, em oposição à maneira tradicional são: provas de confiança não completas, computação distribuída e a confiança na avaliação é empregada individualmente.

Para superar estas características a confiança em redes *mesh* deve ser estabelecida usando uma série de pressupostos incluindo a pré-configuração dos nós com chaves secretas, ou da presença de uma autoridade central, ou um conjunto destas. Uma confiança direta pode ser estabelecida entre as duas partes usando técnicas de autenticação.

A terceirização da confidencialidade pode ser implementada utilizando certificado de autoridade, que é computacionalmente uma solução mais cara e difícil de aplicar, devido à natureza das redes m*esh* ser um tipo híbrido de rede.

Em redes m*esh* a confiança e a distribuição são restritas apenas as interações locais. Cada nó deve agir como um agente autônomo, o que torna a decisão sobre a confiança uma avaliação individual. As decisões são baseadas em informações que se tenham obtido, por si só ou de seus vizinhos.

Ainda não é confiável um único nó em uma rede m*esh*, devido a fraca segurança física e de disponibilidade, com isso se deve distribuir confiança para um conjunto de nós. Partindo do princípio que qualquer n + 1 nós improvavelmente estarão todos comprometidos, se tem o consenso de que, pelo menos, n + 1 nós são confiáveis.

2.4.4.Gerenciamento de Chaves

Muitos objetivos da segurança podem ser obtidos utilizando mecanismos criptográficos. Por outro lado, os mecanismos criptográficos desenvolvidos para redes *mesh* e *ad hoc*,

bem como para as redes tradicionais, confiam que o gerenciamento das chaves criptográficas está sendo realizado de forma apropriada. Em redes *mesh* o gerenciamento de chaves é um grande desafio. Os mecanismos tradicionais de gerenciamento de chaves não são adequados para as redes *mesh* e *ad hoc*, uma vez que necessitam de uma entidade confiável central, conhecida da como AC (Autoridade Certificadora) [Capkun 2003].

O principal problema de qualquer sistema de segurança baseado em chaves públicas é fazer com que a chave pública de cada usuário da rede seja disponibilizada para os demais usuários de forma que sua autenticidade seja verificada. Esse problema é ainda maior nas redes *mesh* e *ad hoc*, pois, como já foi dito, não existe o papel de uma autoridade central na rede [Asokan 2000]. Outra característica é que eles podem ser particionados devido o dinamismo em sua topologia.

Uma abordagem amplamente utilizada para a solução dos problemas de gerenciamento de chaves públicas é o uso de certificado de chave pública, o qual é uma estrutura de dados onde a chave pública é associada a uma identidade por meio da assinatura digital do emissor do certificado.

De certa forma é uma questão problemática utilizar uma única AC em uma rede *mesh*, uma vez que a AC é responsável pela segurança da rede inteira, esta se torna um ponto vulnerável da rede. Caso a AC fique indisponível, os nós da rede não podem obter as chaves públicas atuais dos outros nós, e como conseqüência, não podem estabelecer uma comunicação segura com os demais nós. Além disso, se uma AC fica comprometida, um atacante pode assinar certificados falsos usando uma chave privada obtida da AC comprometida e, personificar qualquer outro nó, ou ainda, revogar os certificados válidos. Muitos *frameworks* de gestão de redes usam uma AC virtual.

No entanto, é importante notar que nenhuma destas abordagens tem sido apresentada para fornecer soluções eficazes em diversos ambientes. Estas limitações vêm principalmente do fato de que a maior parte das abordagens tenta adaptar soluções de ambientes cabeados com adequações para enfrentar os desafios específicos em redes *mesh*. A princípio, a participação do nó estabelece que uma chave no *framework* de gestão deve basear-se em um grande número de nós para a disponibilidade e um pequeno grupo de nós para segurança [Becker 1998]. Dada a vulnerabilidade física dos nós móveis em redes *mesh*, não é eficaz a carga em único nó com a responsabilidade de proporcionar um serviço de segurança como a gestão das chaves.

Uma forma natural de resolver este problema é a distribuição dos serviços de segurança sobre vários nós. No entanto, igualdade de distribuição de funcionalidades de segurança ao longo dos demais nós conduz a um sistema vulnerável. Esta observação leva a duas questões importantes quanto à participação dos nós no gerenciamento de chave:

- Quantos nós devem participar? A participação de uma fração maior dos nós na rede pode melhorar a disponibilidade e tolerância à falhas. No entanto, sem uma reflexão cuidadosa, uma maior participação pode levar a uma maior vulnerabilidade.
- Como os nós devem participar? Quando um serviço de segurança está dividido entre um grande número de nós a igualdade de responsabilidades, e a

disponibilidade do serviço também aumenta, uma vez que há mais nós que um usuário final possa entrar em seu contato.

No entanto, isso também ajuda os nós "adversários" a localizar esses nós e comprometer a segurança do serviço. Por isso, a igualdade de distribuição de funcionalidade para vários nós pode degradar a segurança global da rede. Em vez disso, o núcleo das funcionalidades do serviço de segurança deve ser distribuído a um conjunto restrito e seguro de nós, proporcionando uma maior segurança e um nível aceitável de disponibilidade.

Os demais nós, que fazem parte de um nível menor, tem a funcionalidade de melhorar a disponibilidade dos principais nós. Comprometendo qualquer destes nós de baixo nível, não devem comprometer a segurança global da rede, e sim afetarão a capacidade do núcleo do serviço.

A utilização de uma terceirização TTP (*Trusted Third Party*) faz com que tenha um princípio fundamental de gestão do *framework*, que deverá melhorar a qualidade de autenticação das chaves. Uma vez que não há garantias sobre o comportamento dos nós participantes, qualquer autenticação baseada em tais relações casuais pode não ser confiável para a segurança de aplicações. A TTP proporciona uma confiança que pode ser usada como base para as relações ainda mais confiáveis. Uma vez que cada nó que confia no TTP, a autenticação fornecida pela TTP é confiável com um elevado nível de confiança.

Essencialmente, sem uma autenticação confiável, não há mais nenhuma garantia de serviço que possa ser construído de modo a garantir um nível elevado de confiabilidade. Por conseguinte, a utilização de uma terceirização, é fundamental em qualquer rede *mesh* com fortes requisitos de segurança. Dentro deste contexto de gerenciamento de chaves serão mostrados a seguir tópicos com pontos relevantes em relação ao gerenciamento de chaves, visando um aumento na segurança de uma rede *mesh*.

a) Infra-estrutura de chave pública

A utilização de chaves de criptografia pública exige que as autenticidades das chaves públicas possam ser estabelecidas. Uma simples abordagem exige que qualquer um dos dois usuários que desejam se comunicar, devem trocar suas chaves públicas e, desta forma, exige que a distribuição inicial de n (n-1) chaves públicas. No entanto, se há uma terceirização para emitir os certificados a cada um dos usuários, apenas a chave pública do TTP precisa ser distribuída a cada um dos usuários.

b) Distribuição de chaves

O principal objetivo do gerenciamento de chaves é compartilhar uma chave com um grupo de participantes. Para tanto, quatro operações podem ser necessárias: a pré-distribuição, o transporte, a arbitração e o acordo de chaves.

A pré-distribuição de chaves consiste da distribuição das chaves pelos nós interessados antes do início da comunicação. Isto exige que todos os nós da rede sejam previamente conhecidos, embora não seja exigido que todos participem sempre da rede [IEEE 2000].

No transporte de chaves, as entidades trocam chaves para se comunicar. O método mais simples para essa fase se chama KEK (*Key Encription Key*) [Bird 1995], e

consiste em criptografar a nova chave com o segredo compartilhado, e apenas os nós que possuírem esse segredo podem obter a nova chave. No caso de não existir uma chave previamente conhecida por um grupo, mas existir uma infra-estrutura de chave pública, essa nova chave pode ser trocada criptografando-a com a chave pública do nó que irá recebê-la.

A distribuição de chaves utiliza um arbitrador (distribuidor) central para criar e distribuir chaves entre os participantes, o que a torna uma especialização da fase de transporte. Em sistema infra-estruturado, um ponto central é escolhido para exercer a função de arbitrador. No entanto, em redes *mesh* e *ad hoc*, esta função centralizada de arbitrador é proibitiva por causa da ausência de infra-estrutura e restrições de recursos. Entre esses está a necessidade do arbitrador estar sempre ativo e acessível, sob pena de negação de serviço, caso o nó se mova ou saia da rede.

A utilização de réplicas da base de dados para resolver o problema da negação de serviço aumentaria o número de nós guardando os segredos da rede, gerando mais pontos de vulnerabilidade, além de ser uma solução que acarretaria uma maior saturação à rede.

Finalmente, o acordo de chaves corresponde à troca de chaves posterior ao início da rede. Serão estabelecidos segredos entre nós através de chaves assimétricas, se elas estiverem disponíveis. Isto é necessário para realizar uma comunicação segura dentro da rede, embora seja uma operação muito custosa, devido a sua complexidade de manutenção destas chaves assimétricas.

c) Gerenciamento dinâmico de chave

Uma alternativa que pode enriquecer o gerenciamento de chaves é o mecanismo de gerenciamento dinâmico, os quais necessitam, em sua maioria, de um sistema de distribuição ordenado para iniciar o processo pré-existente [Lin 1997].

O mecanismo de pré-distribuição de chaves deve distribuir um conjunto de chaves secretas para cada nó antes que os nós fiquem ativos na rede. Depois que os nós são implantados, o regime de pré-distribuição pode fornecer garantias de que dois nós partilham uma chave secreta com uma determinada fração de seus vizinhos, esta fração é um dos parâmetros fundamentais do regime de pré-distribuição, pois garante que há uma certa confiabilidade entre os nós vizinhos, ou no mínimo uma fração deles.

Dessa forma, um nó pode criar, ou receber, uma chave secreta compartilhada com todos, ou alguns, nós vizinhos. A fim de tornar o sistema dinâmico, os nós desencadeiam este processo para cada conjunto de vizinhos. Enquanto dois conjuntos de vizinhos consecutivos de um nó se sobrepõem, os nós, nesta sobreposição podem ser usados para estabelecer chaves com todos os novos nós do novo conjunto de vizinhos, assim tornando o processo mais dinâmico.

d) Gerenciamento de chave de grupo

O gerenciamento de chaves de grupo inclui atividades para criação e manutenção da chave do grupo. Manutenção das atividades consiste em mudar a chave do grupo devido a adição, exclusão ou devido ao uso de chave do grupo durante longos períodos de tempo [Anton 2002].

O estabelecimento de uma chave de grupo pode ser centralizado, também chamado de distributivo, isto ocorre quando uma entidade é responsável por gerar as

chaves de grupo e distribuí-las para os outros membros do grupo. Esta abordagem tem a vantagem de ser simples. Já em um estabelecimento distribuído, também chamado de contributivo, todos os membros do grupo contribuem para o grupo de geração de chaves. Esta abordagem é tolerante a falhas e diminui os riscos de geração viciosa de chave por uma única entidade. Uma variação desta abordagem consiste em ser parcialmente contributivo o que permite a um subgrupo de ser responsável por gerar a chave do grupo.

Um grupo pode ser estático ou dinâmico. Um grupo dinâmico permite a exclusão de membros, bem como a adição de novos membros. Chave para a gestão dinâmica de grupos pode proporcionar sigilo, quando os membros que deixam o grupo são incapazes de calcular o futuro grupo chaves.

2.5. Soluções Propostas

Muitas técnicas de segurança em redes *mesh* foram adaptadas ou trazidas das redes sem fio convencionais. No entanto, várias destas técnicas de segurança, ou são específicas para a camada de aplicação como por exemplo, utilizar protocolos como HTTPS (*HyperText Transfer Protocol Secure*), SSH (*Secure Shell*) e SFTP *(Secure File Transfer Protocol)* ou são políticas de segurança, onde o usuário é quem tem a obrigação ou necessidade de seguir.

As técnicas de segurança voltadas para a camada de transporte existem em menor número, porém, mais eficientes, ou pelo menos impõem a responsabilidade de segurança da rede para a tecnologia, passando mais confiabilidade aos usuários da rede.

Na camada de rede do modelo TCP/IP, as técnicas de segurança focam nos protocolos de roteamento, onde os mais usados e conhecidos são: OSLR, WRP (*Wavelength Routing Protocol*) [Murthy e Garcia 1996], CGSR (*Cluster-Head Gateway Switch Routing*) [Chiang et al. 1997], FSR (*Fisheye State Routing*) [Pei et al. 2000], DSR, AODV e TORA (*Temporally Ordered Routing Algorithm*) [Royer e Toh 1999].

Esses protocolos possuem grande aceitação por suas funcionalidades vitais, como rápida alocação da tabela de rotas em casos de mudança de disposição da rede, escalabilidade e controle reduzido de *overhead*. Porém, estes protocolos não têm um tratamento adequado contra técnicas MITM (*Man In The Middle*), DoS (*Denial of Service*), *eavesdropping* (conhecido como escuta passiva) e ataques de inserção de informações de rota errôneas na rede.

2.5.1. Arquiteturas

O anonimato vem recebendo uma crescente atenção pela literatura, como por exemplo [Rahman et al. 2006] e [Seys e Preneel 2006], devido ao fato do usuário querer garantir sua privacidade ao usar a rede. Por outro lado, a autoridade de rede requer um anonimato condicional que permita um rastreamento.

[Sun et. al 2008] propõe uma arquitetura de segurança com o objetivo de assegurar nas redes *mesh*, um anonimato incondicional para os usuários confiáveis e o rastreamento de usuários não confiáveis. Essa arquitetura proposta busca solucionar os conflitos entre a demanda pelo anonimato e a necessidade de rastreamento, além de garantir requisitos fundamentais de segurança, incluindo autenticação, confidencialidade, integridade de dados, e não-repúdio. Especificamente, as

contribuições principais são: A modelagem do sistema; O projeto de um sistema de anonimato baseado em *tickets* com a propriedade para garantir rastreamento; Controle de acesso de anônimos através da relação dos *tickets* com pseudônimos, sendo o processo de revogação simplificado e; Adoção de um sistema de detecção de intrusão hierárquico baseado em criptografia para autenticação intra-domínio.

Em [Li 2007], a arquitetura ISA *(Identity-based Security Architecture*) é proposta, eliminando a necessidade por distribuição de chave públicas baseadas em certificados, introduzindo um mecanismo de descoberta de vizinhança. Habilita ainda a possibilidade de atualização eficiente de chaves através de mensagens *broadcast*.

[Zhang e Fang 2006] identifica as necessidades de segurança em redes *mesh* e em seguida, propõe a arquitetura ARSA (*Attack-Resilient Security Architecture*), a qual elimina a necessidade do estabelecimento de acordos bilaterais e interações em tempo real entre as operadoras.

Em [Lin et al. 2008], é proposto um paradigma de projeto para uma arquitetura de autenticação e resiliência em redes *mesh* denominada TUA (*Threshold User Authentication*).

2.5.2. Protocolos de Roteamento Seguros

Os protocolos mais conhecidos que propõem técnicas de segurança em nível de transporte são: SOLSR (*Secure Optimized Link State Routing Protocol*) [Hong e Fu 2005], SAODV (*Secure Ad Hoc On-Demand Distance Vector*) [Zapata 2002], SRP (*Secure Routing Protocol*) [Hu e Perrig 2004], Ariadne [Hu et al. 2005], SEAD (*Secure Eficient Ad Hoc Distance Vector Routing Protocol*) [Hu et al. 2003], ARAN (*Authenticated Routing for Ad Hoc Networks*) [Mahmoud et al. 2005], SLSP (*Secure Link State Protocol*) [Papadimitratos 2003] e o RM-AODV (*Radio Metric Ad Hoc On-Demand Distance Vector*) [Takeda et al. 2005]. Abaixo, os principais protocolos seguros serão abordados.

2.5.2.1. SOLSR

É uma extensão segura do protocolo de roteamento OLSR, onde a idéia é assinar cada pacote de controle do OLSR, com chaves simétricas, com o objetivo de garantir a autenticidade das mensagens. Um diferencial do SOLSR é que a autenticação é realizada salto-a-salto. Isso garante a segurança, inclusive de campos que são atualizados por nós intermediários, como o número de saltos e o campo TTL (*Time To Live*). É necessária somente uma assinatura por salto, levando-se em conta que muitas mensagens de roteamento são encapsuladas em um único pacote do OLSR. Em contrapartida, a abordagem salto-a-salto não garante assinaturas fim-a-fim, já que um pacote recebido por um nó terá sido assinado pelo nó anterior e não pelo nó de origem.

Todavia, o protocolo determina que os nós devem somente encaminhar pacotes originados de nós confiáveis. Por conseqüência os nós de uma determinada rota serão confiáveis, dois a dois. O processo de assinar digitalmente utiliza uma função *hash* com chave, de forma que um nó que não tenha acesso à chave secreta não poderá reproduzir a assinatura do nó emissor.

O SOLSR possui mensagens determinadas para acomodar as assinaturas, de forma que se garanta a compatibilidade com nós que não estejam operando a versão

segura do OLSR. O protocolo, para evitar ataques de replicação, utiliza também a técnica de *timestamp* que nada mais é que uma seqüência de caracteres que denotam data e hora que um determinado evento ocorreu, utilizado exclusivamente para realizar *logging* de eventos.

2.5.2.2. SAODV

É uma extensão do protocolo AODV que garante segurança no processo de descobrimento de rotas. A idéia básica neste protocolo é usar assinaturas para autenticar a maioria dos campos dos pacotes RREQ e RREP, e usar cadeias de *hash* para autenticar a contagem de saltos. A assinatura digital garante autenticação fim-a-fim dos campos que mantém informações imutáveis da mensagem e que devem ser assinados pelo nó de origem antes do envio da mensagem. Porém, o campo número de saltos (*Hop Count*) deve ser decrementado a cada salto pelas estações intermediárias. Nesse caso, a cadeia de *hash* autentica o campo número de saltos a cada salto, garantindo a integridade do algoritmo de vetor de distâncias.

No processo de enviar mensagens, o nó de origem inicializa o campo *hash* com uma variável aleatória, o campo número máximo de saltos com o valor desejado e o campo *top hash* com o resultado da aplicação da função *hash* sobre a variável um número de vezes igual ao número máximo de saltos.

Quando um nó intermediário receber a mensagem, poderá autenticar o campo número de saltos, aplicando a mesma função *hash* um número de vezes igual a diferença entre número máximo de saltos e número de saltos sobre o campo *hash* e comparando o resultado com o campo *top hash*. Os campos sendo iguais, pode-se assegurar que o campo número de saltos não foi modificado, então, o nó intermediário deverá incrementar o campo número de saltos em 1 e aplicar a função *hash* no campo *hash* antes de reencaminhar a mensagem. De outra forma, a mensagem será descartada.

Dessa maneira, o mecanismo de cadeias de *hash* impossibilita que um nó malicioso altere o número de saltos da mensagem de roteamento, uma vez que não se pode obter os valores anteriores da seqüência, levando em conta a característica unidirecional da função *hash*.

2.5.2.3. SRP

Foi projetado para manter e normalizar o roteamento correto em redes *ad hoc* onde ocorrem mudanças freqüentes e pode haver nós maliciosos, mas que não agem em conjunto para realizar ataques DDoS.

O protocolo foi desenvolvido como uma extensão, que pode ser aplicada em alguns protocolos de roteamento reativos existentes, especialmente o DSR. No SRP, o nó inicia o procedimento de descoberta de rota, identifica e descarta respostas contendo informações de roteamento falsas e ainda evita recebê-las, garantindo a obtenção de informações da topologia da rede corretamente.

Para isso considera-se a existência de uma AS (*Security Association*) entre os nós comunicantes, como uma chave simétrica compartilhada. Além disso, supõe-se que os nós possuem uma única interface de rede, com uma correspondência biunívoca entre os endereços IP e o da interface.

Ao iniciar o procedimento de descoberta de rota, o nó de origem deve gerar um MAC (*Message Authentication Code*), usando uma função *hash* com chave que recebe como argumentos de entrada o cabeçalho IP, os campos básicos da mensagem de roteamento e a chave secreta compartilhada entre os nós de origem e destino. As estações intermediárias são responsáveis por encaminhar a mensagem até o seu destino final.

Quando o nó de destino recebe a mensagem de roteamento, é verificada a integridade da mensagem e é assegurada a autenticidade da origem, uma vez que o MAC só pode ser calculado pelos nós que possuem a chave secreta, garantindo que somente a origem poderia ter computado o MAC recebido. Caso a mensagem recebida seja autêntica e integra, o nó destino envia uma mensagem de resposta ao nó de origem realizando o mesmo procedimento feito pelo nó de origem. O nó de origem recebendo a mensagem de resposta, verifica sua integridade usando o MAC computado pelo nó de destino e descarta a resposta se ela não tiver o mesmo identificador da mensagem inicial.

O protocolo possui um mecanismo que regula as requisições de descoberta de rota. Cada nó mede as freqüências de requisições realizadas pelos seus vizinhos e mantém uma fila na qual a prioridade de atendimento às requisições é inversamente proporcional à freqüência com que elas são feitas.

Caracterizando assim, um mecanismo de *feedback* negativo que controla a freqüência de requisições realizadas pelos nós vizinhos, impedindo ataques nos quais o nó malicioso inunda a rede com requisições de descoberta de rota, já que o atacante será o último a ser atendido ou será ignorado, devido sua baixa prioridade de atendimento.

Uma das desvantagens do SRP é a ausência de autenticação das mensagens de erro, embora o nó malicioso só consiga prejudicar rotas às quais ele pertence. Outra desvantagem é que, como o protocolo não previne ações maliciosas em conjunto, ele não está imune aos ataques de atração e descarte de pacotes.

O protocolo possui o diferencial da imunidade aos ataques que modificam a origem do pacote ou simulam identidades. Isso se deve ao protocolo NLP (*Neighbor Lookup Protocol*) de descoberta de vizinhos, que integra o SRP, mantendo um mapeamento dos endereços da subcamada de acesso ao meio MAC e da camada de rede dos nós da rede.

2.5.2.4. Ariadne

É um protocolo de roteamento reativo seguro para redes móveis *ad hoc*, baseado no protocolo DSR. Para ser utilizado, o protocolo necessita de uma sincronização fraca de tempo entre os nós da rede, de modo que um nó possa estimar o tempo de transmissão fim-a-fim para qualquer nó da rede.

Faz-se necessário um mecanismo para o estabelecimento de chaves secretas entre os nós comunicantes e um meio de fazer a distribuição de uma chave pública TESLA (um mecanismo eficiente de autenticação *broadcast* que requer uma sincronização fraca de tempo entre os nós) autêntica para cada nó.

O protocolo provê autenticação ponto-a-ponto das mensagens de roteamento usando um MAC e uma chave secreta compartilhada pelas duas entidades.

Para garantir a autenticação fim-a-fim do esquema de descoberta de rota, o nó de origem calcula um MAC da mensagem usando a chave secreta conhecida exclusivamente pelos nós de origem e destino.

Dessa maneira, assegura-se que a mensagem foi transmitida do nó de origem e que as informações de roteamento não foram alteradas. Antes de enviar a mensagem, o nó de origem estipula um tempo máximo para o atraso fim-a-fim, inserindo esta informação na mensagem, em conjunto com uma lista de nós e uma lista de MACs, ambas inicialmente vazias.

Após o término do tempo estimado o nó de origem irá divulgar sua chave TESLA. Assim, quando uma estação intermediária recebe a mensagem, ela verifica se o tempo de divulgação da chave já esgotou. Sendo positivo o resultado, descarta-se a mensagem, caso contrário, insere-se o endereço na lista de nós. A integridade da lista de endereços é obtida através do mecanismo de cadeias de *hash*, usando um esquema similar utilizado pelo SAODV.

Neste instante é calculado um novo *hash* sobre o campo *hash chain*. A estação intermediária ainda utiliza sua chave TESLA atual para computar o MAC da mensagem, que é inserido na lista de MACs.

A mensagem modificada é reenviada para os vizinhos recursivamente pelas estações intermediárias até chegar ao nó de destino. Ao receber a mensagem, o nó de destino, verifica se o valor final da cadeia de *hash* está correto e se as chaves TESLA já foram divulgadas, caso a mensagem recebida seja válida, o nó de destino calcula o MAC da resposta usando a chave secreta compartilhada com o nó de origem e envia a mensagem de resposta para a origem. Ao final da rota reversa, a origem autentica a resposta antes de aceitá-la.

2.5.2.5. SEAD

Diferentemente do Ariadne, o SEAD é um protocolo pró-ativo, que é baseado no protocolo de vetor distância DSDV. Foi desenvolvido para suportar a existência de nós com baixa capacidade de processamento e possui defesas contra ataques de negação de serviço onde o atacante visa esgotar recursos da vítima, como banda passante e processamento e até mesmo bateria.

Utiliza-se uma técnica de cadeia de *hash* para autenticar os campos número de saltos e número de seqüência. A cadeia é criada aplicando-se repetidas vezes uma função *hash* a um valor aleatório inicial e assume-se a existência de algum mecanismo que permita que um nó distribua um elemento autenticado da cadeia para seus vizinhos.

Um elemento autenticado da cadeia de *hash* é utilizado para garantir uma atualização segura das mensagens de roteamento. Os elementos seguintes da cadeia podem ser autenticados aplicando-se sobre eles a função *hash* um número determinado de vezes.

2.5.2.6. ARAN

É um protocolo de roteamento reativo que utiliza certificado digital para garantir autenticação, integridade e não-repúdio de mensagens de roteamento em uma estrutura de rede *ad hoc*.

Baseado em métricas lógicas de roteamento e certificados, este protocolo é imune a alteração de mensagens por *airspoofing* e mensagens de roteamento geradas por ferramentas como o *ettercap* [Ornaghi e Valleri 2008], que é um *sniffer* para realizar ataques do tipo *Man In The Middle*.

Existem dois estágios que podem ser utilizados pelo protocolo ARAN. No primeiro a rota encontrada entre a origem e o destino, não tem garantia de ser a menor rota possível, levando em consideração a quantidade de nós intermediários na rota. Já o segundo estágio, exige maior processamento e consumo de banda, garantindo que a rota utilizada é a menor possível. Entretanto, o segundo estágio é opcional, o que é aceitável, levando em conta que a rota encontrada no primeiro estágio, mesmo podendo não ser a menor, é a com menor tempo de retardo entre a origem e o destino.

Quando um nó origem necessita de uma rota para um destino qualquer, inicia-se o processo de descoberta de rota. Cria-se um pacote contendo seu certificado digital assinando-o, e encaminhando-o através da rede por *broadcast*, sendo que apenas o nó destino da requisição poderá responder a mesma.

Os nós vizinhos a origem da requisição, receberão o pacote e validarão sua assinatura. Tratando-se de um pacote assinado devidamente, esse vizinho, por sua vez, assina o pacote e inclui seu certificado no mesmo, sem remover nenhum dado do pacote original.

Os próximos vizinhos recebem o pacote de requisição, antes que ele chegue ao destino, e seguirão o mesmo procedimento, validar o certificado do nó vizinho que enviou o pacote, remover o certificado do nó vizinho que enviou o pacote, incluir o próprio certificado no pacote e encaminhar o pacote através da rede.

Alcançando o nó destino, a requisição é mais uma vez validada com seu certificado, sendo que dessa vez, o certificado do nó origem também é validado. Sendo considerados válidos, o destino criará um pacote de resposta para a requisição e o encaminhará de volta à origem da requisição. No momento em que a origem da requisição recebe a resposta enviada pelo destino, verifica-se a autenticidade do pacote, tendo em vista que apenas o destino poderia ter respondido sua requisição.

Essa limitação tem por objetivo garantir maior segurança, evitando a formação de *loops* em uma rota, além de uma possível perda na eficiência no processo de requisição de rotas. O protocolo exige a criação de uma entrada na tabela de roteamento, para cada par origem-destino.

A grande diferença no funcionamento do primeiro estágio do protocolo ARAN, para seu segundo estágio, é a necessidade de que todos os nós intermediários, entre a origem e o destino da requisição, assinem o pacote antes de encaminhá-lo através da rede, porém sem remover a assinatura do pacote vizinho que o encaminhou anteriormente. Dessa forma, alcançando o nó destino, o pacote de requisição irá conter o caminho completo desde a origem.

2.5.2.7. SLSP

É um protocolo de roteamento reativo seguro baseado em estado de enlace que foi proposto pelos criadores do SRP. Leva-se em consideração a existência de um par de chaves assimétricas para cada interface de rede em um nó e de um sistema de gerenciamento das chaves públicas dos nós. O SLSP compõe-se de três procedimentos:

distribuição de chaves públicas, descoberta de vizinhos e atualização dos estados dos enlaces.

O procedimento de distribuição das chaves públicas é feito de forma distribuída, a fim de evitar o uso de um servidor central de gerenciamento de chaves. Cada nó envia por difusão um pacote de distribuição de chaves públicas. A garantia de segurança no nível de roteamento agregada a uma garantia de segurança em nível de aplicação garante um mínimo de confidencialidade em uma rede *mesh*.

2.5.2.8. Resumo comparativo entre os protocolos seguros

As tabelas abaixo, ver tab. 2.2 e 2.3, apresentam um comparativo dos protocolos seguros apresentados.

Tabela 2.2. Resumo comparativo dos protocolos seguros

Parâmetros	ARAN	Ariadne	SAODV
Tipo	Reativo	Reativo	Reativo
Algoritmo de Criptografia	Assimétrico	Simétrico	Assimétrico
Protocolo Origem	AODV/DSR	DSR	AODV
Sincronização	Não	Sim	Não
Autoridade Certificadora	Necessita CA	Necessita KDC	Necessita CA
Autenticação	Sim	Sim	Sim
Confidencialidade	Sim	Não	Não
Integridade	Sim	Sim	Sim
Não-repudiação	Sim	Não	Sim
Anti-spoofing	Sim	Sim	Sim
Ataques DoS	Não	Sim	Não

Tabela 2.3. Resumo comparativo dos protocolos seguros

Parâmetros	SEAD	SLSP	SRP	SOLSR
Tipo	Pró-Ativo	Pró-Ativo	Pró-Ativo	Pró-Ativo
Algoritmo de Criptografia	Simétrico	Assimétrico	Simétrico	Simétrico/Assimétrica
Protocolo Origem	DSDV	ZHLS	DSR/ZRP	OLSR
Sincronização	Sim	Não	Não	Sim
Autoridade Certificadora	Necessita CA	Necessita CA/ KDC	Necessita CA	Necessita KDC
Autenticação	Sim	Sim	Sim	Sim
Confidencialidade	Não	Não	Não	Não
Integridade	Não	Sim	Sim	Sim
Não-repudiação	Não	Sim	Não	Não
Anti-spoofing	Não	Sim	Sim	Sim
Ataques DoS	Sim	Sim	Sim	Sim

2.5.3. Mecanismos de Detecção de Intrusão, Autenticação e Contabilização

Como soluções de segurança envolvendo autenticação, cita-se os protocolos WADP (*Wireless Dual Authentication Protocol*) [Zheng et al. 2004] e SUMP (*Secure Unicast Messaging Protocol*) [Janies et al. 2006].

Contabilização ou bilhetagem é tratada em [Xiao 2008] e [Zhu et al. 2007]. O primeiro proporciona um estudo do estado da arte a respeito de contabilização. Descreve e analisa aplicações, propondo uma arquitetura denominada *A-NET*, com uma análise voltada para redes *ad hoc* e rede *mesh.* O segundo apresenta o esquema SLAB *(Secure Localized Authentication and Billing),* o qual visa manter a segurança e o desempenho do sistema em termos de resiliência, capacidade, *handoff* entre domínios, autenticação e latência. É demonstrado matematicamente.

[Zhang et al. 2008] apresenta um esquema de descoberta de anomalias chamado RADAR (ReputAtion-based Scheme for Detecting Anomalous Nodes in WiReless Mesh Networks). Devido à infra-estrutura especial e o modo de comunicação, descoberta de intrusão em redes *mesh* é especialmente desafiadora, pois requer considerações particulares de projeto. Inicialmente apresenta conceitos gerais, caracterizando e quantificando *status*/perfis dos nós em termos de métricas de desempenho, em seguida apresenta o esquema proposto.

Em [Li et al. 2006], é proposto uma solução para a autenticação de pontos de acesso em redes *mesh* no modo infra-estruturado, combinando 802.1X/EAP e gráfos dos nós vizinhos. Realiza simulações para avaliar a proposta.

[Kim e Bahk 2008] apresenta a arquitetura Meca *(Mesh Certification Authority)*, a qual é uma arquitetura, para autoridades certificadoras distribuídas, desenvolvida para redes *mesh.* Utiliza o esquema FVSR (*Fast Verifiable Secret Redistribution*) para atualizar compartilhamentos secretos e para a mudança de participantes. A arquitetura proposta reduziu o *overhead* existente utilizando *multicast.* Consegue prover

atualização de certificações, exclusões e verificação de *status* de forma mais segura e eficiente. Ao basear-se no fato de que roteadores *mesh* possuem potência suficiente e são fixos, faz com que todos estes sejam capazes de trabalhar como nós autoridade certificadora distribuída e o uso de *multicast* traz uma significante melhora no desempenho da rede.

Em [Islam et al. 2008] é proposto um esquema de autenticação anônima entre o cliente/usuário e o roteador *mesh* para preservar a privacidade, identidade e segurança na comunicação em redes *mesh*. Direta ou indiretamente relacionados ao tema, citam-se ainda [Krebs et al. 2008], [Kim e Bahk 2008], [Graffi 2006] e [Krebs et al. 2008].

2.6. Comentários Finais e Tendências Futuras

Atualmente, as redes *mesh* são vistas como uma tecnologia promissora para próxima geração das redes sem fio [Ju et al. 2007]. Algumas aplicações estão estimulando o seu rápido desenvolvimento, porém é necessário que sua inserção no mercado seja fortalecida e que pesquisas científicas sejam realizadas para sua melhoria. O principal atrativo das redes *mesh* é o de prover uma rede comunitária de acesso banda larga com infra-estrutura sem fio, oferecendo acesso à Internet a baixo custo para comunidades que, por exemplo, possuem baixas condições econômicas.

Nas redes *ad hoc*, questões de segurança já vêm sendo estudadas há certo tempo, como por exemplo em [Zhou e Haas 1999], [Lou e Fang 2004], [Rahman et al. 2006] e [Raya e Hubaux 2007]. Entretanto, nas redes *mesh*, segurança ainda é um tema muito novo e pouco abordado. A principal diferença entre as redes *ad hoc* e as redes *mesh* reside no fato de que os nós das redes em malha sem fios têm localização fixa, embora suas localizações não sejam predeterminadas. Os algoritmos de roteamento, portanto, têm muita semelhança entre si.

Observa-se que, cada protocolo tem características próprias que atendem as mais variadas exigências. Entretanto a utilização do protocolo de roteamento correto depende do cenário o qual ele será utilizado. Portanto, a escolha correta se dá após estudos detalhados sobre a topologia da rede para que o protocolo escolhido atenda as necessidades da rede em questão. E como as redes *mesh* são compostas de nós moveis e fixos, um protocolo híbrido ou a combinação de protocolos reativos e pró-ativos é a solução mais aconselhável [Lee et al. 2006].

O MHRP (Multi-path Hybrid Routing Protocol) é um dos poucos protocolos híbridos propostos para redes *mesh* [Siddiqui et al. 2007]. É válido ressaltar ainda [Hamid e Khant 2006], um dos poucos trabalhos que propõe um protocolo de roteamento para ser utilizado em ambientes 802.16. Atualmente, encontram-se várias linhas de pesquisa dentro do tema segurança em redes *mesh*, além do que foi abordado neste capítulo, como por exemplo [Mogre et al. 2007] e [Hamid et al. 2008]. O problema começa a ser estudado matematicamente, como por exemplo em [Li et al. 2007], onde conceitos de processos estocásticos são abordados.

As redes em malha sem fio podem ser construídas baseadas em tecnologias existentes. Algumas empresas já têm à venda produtos, porém, tentativas em laboratórios e experiências com redes em malha sem fio existentes provam que o desempenho delas ainda é distante quando comparado com o que é esperado.

Pelo fato de não existir um padrão único para a construção de redes *mesh* e sim várias opções incluindo soluções acadêmicas e comerciais, são inúmeras as propostas encontradas na literatura tentando solucionar as questões apresentadas.

Mesmo após a especificação do futuro padrão IEEE 802.11s [Hertz et al. 2008] que permitirá a transmissão em múltiplos saltos no nível de enlace, resolvendo os problemas de mobilidade e comunicação em grupo, as soluções propostas no nível de rede continuarão sendo investigadas, pois são necessárias para permitir o uso de equipamentos já existentes nas redes *mesh* atuais e futuras.

Os desafios nas redes *mesh* existem e estão relacionados a todas as camadas de rede, daí a grande relevância do termo *cross-layer* para a academia, sendo segurança o menos desenvolvido até o momento.

Semelhante as redes *ad hoc*, porém em um maior nível, as redes em malha sem fio ainda possuem deficiências em soluções eficientes e escaláveis de segurança devido a vários fatores: sua arquitetura de rede distribuída, vulnerabilidade de canais e nós no meio sem fio compartilhado e a possibilidade de mudanças dinâmicas na topologia da rede.

Os ataques podem ocorrer na camada de rede através dos protocolos de roteamento, na atualização errada de informações, causando facilmente falhas na rede. O atacante pode se mover furtivamente na rede representando um nó legítimo, sem seguir as especificações necessárias ao protocolo de roteamento. Também podem ocorrer nos protocolos de acesso ao meio.

Uma forma de resposta aos ataques amplamente aceita é autenticação e autorização. Nas redes sem fio tradicionais serviços AAA (*Authentication, Authorization, and Accounting*) são utilizados diretamente nos pontos de acesso ou através de *gateways*, sendo executados por um servidor centralizado como RADIUS (*Remote Authentication Dial-In User Service*), o qual por ser um esquema centralizado não é aplicável nas redes em malha sem fio.

Além disso, o gerenciamento seguro de chaves nas redes em malha sem fio é muito mais difícil que nas redes sem fio tradicionais porque não há nenhuma autoridade central ou servidor para gerenciar as chaves de segurança. Desta forma, gerenciamento de chaves nas redes em malha sem fio precisa ser executado dentro de um ambiente distribuído, porém seguro. Portanto um esquema distribuído de autenticação e autorização, com gerenciamento de chaves precisa ser validado para as redes em malha sem fio.

Adicionalmente, para garantir segurança nas redes em malha sem fio, duas estratégias precisam ser consideradas: mecanismos de segurança embutidos nos protocolos de roteamento e de acesso ao meio e mecanismos de segurança que monitorem a rede, detectem ataques e/ou quebra de serviços, bem como possam responder aos mesmos de forma ágil e segura.

Nas redes em malha sem fio, para um protocolo de rede seguro, um esquema seguro através de múltiplas camadas é desejado, uma vez que um ataque pode ocorrer simultaneamente em diferentes camadas de protocolos. Neste caso, um *framework* baseado no conceito de *cross-layer* para um sistema de monitoração precisa ser desenvolvido, sendo uma linha de pesquisa desafiadora uma vez que envolve o projeto e

a implementação, com o uso de informações cruzadas e relacionadas, bem como através de algoritmos diversos de detecção de intrusão.

Um sistema de detecção de intrusão para redes *ad hoc* deve atender dois requisitos básicos: Eficácia (que define como fazer o IDS realizar uma classificação correta de atividades malignas e benignas) e Eficiência (como fazer o IDS funcionar de tal forma que haja o menor custo para a rede e opere todas as funcionalidades esperadas).

A ausência de uma infra-estrutura física facilita o sucesso de um ataque de negação de serviços em uma infra-estrutura de rede sem fio. De tal maneira o desenvolvimento de uma arquitetura de um IDS é um fator desafiador uma vez que sem um ponto de auditoria centralizado como roteadores, um IDS para redes *mesh* torna-se limitado utilizando somente o tráfego de entrada e saída dos nós da rede como fonte de informações de auditoria.

Um requisito fundamental é quanto a questão dos algoritmos utilizados pelos IDSs, pois estes são naturalmente distribuídos o que dificulta uma auditoria correta da rede pois, considera-se que um nó da rede possa visualizar somente uma porção do tráfego da rede. Levando em consideração que as redes *mesh* possuem roteamento dinâmico e os nós da rede podem mover-se livremente, existe a possibilidade de nós serem capturados como nós maliciosos. Se o algoritmo do IDS for cooperativo, torna-se necessário identificar quais nós são confiáveis.

Em redes sem fio não é viável trafegar dados para IDSs como em redes cabeadas, pois é necessário conservar largura de banda. Questões como largura de banda e tempo de bateria podem influenciar a eficiência e a eficácia dos IDSs, pois a disponibilidade de dados de auditoria parcial torna difícil a tarefa de distinguir um ataque de um comportamento real. As técnicas de detecção de ataques, conhecidas para redes estruturadas, são ineficientes em redes em malha sem fio. Como sinal de amadurecimento das redes *mesh*, áreas que antes tinham uma abordagem simplista passaram a receber propostas mais rebuscadas. Comercialmente, existem soluções ofertadas, como por exemplo Tropos [Tropos 2008] e Meshdynamics [Meshdynamics 2001].

A principal aplicação que vem sendo avaliada e estudada nas redes *mesh* é VoIP (*Voice over IP*), como pode ser visto em [Xian e Huang 2007], onde já trata inclusive a questão de segurança em redes *mesh.*

A grande tendência dos estudos em redes *mesh*, independente de ser em segurança ou não, diz respeito a alocação e utilização de múltiplos rádios e canais, como por exemplo em [Haq et al. 2007] e [Naveed e Kanhere 2006]. Além disso, assim como gerenciamento é um assunto associado a segurança [Duarte et al. 2007] e [Siddiqui et al. 2008], gerenciamento de mobilidade também começa a ser estudado, como por exemplo em [Huang et al. 2007].

Referências Bibliográficas

802.11, IEEE (2007). IEEE Standard for Information Technology Telecommunications and Information Exchange between Systems-local and Metropolitan Area Networks Specific Requirements parte 11: Wireless lan medium access control (mac) and

physical layer (phy) specifications. IEEE Std 802.11-2007 (Revisão de IEEE Std 802.11-1999).

802.11s, IEEE (2008) "Task Group S. Status of Project IEEE 802.11s". Disponível em <http://www.ieee802.org/11/Reports/tgs_update.htm>. Acessado em Junho de 2008.

Akyildiz, I. F.; Wang, X. e Wang, W. (2005) Wireless Mesh Networks: a Survey. Comput. Netw. ISDN Syst. Janeiro.

Anton, E.R. Duart, O.C. (2002) "Group key establishment in wireless ad hoc networks". In E.R. Workshop em Qualidade de Serviço e Mobilidade. Novembro.

Asokan N., Ginzborg P., (2000) "Key Agreement in Ad Hoc Networks", Computer Communications Volume 23.

Badonnel, R., State, R., and Festor, O. (2005) "Management of mobile ad hoc networks: information model and probe-based architecture". Int. J. Netw. Manag.

Becker K., Willie U., (1998) "Communication complexity of group key distribution", Proceedings of 5th ACM Conference on Computer and Communications Security, ACM Press.

Bird R., Gopal I., Herzberg A., Janson P., Kutten S., Molva R. , Yung M., (1995) "The KryptoKnight family of light-weight protocols for authentication and key distribution". IEEE/ACM Transactions on Networking, Volume 3 , Issue 1, Fevereiro.

Breuel, Cristiano Malanga. (2004) "Redes em Malha sem Fios". Instituto de Matemática e Estatística - Universidade de São Paulo (USP), Dezembro. Disponível em: <http://grenoble.ime.usp.br/movel/Wireless_Mesh_Networks.pdf>. Acessado em Janeiro de 2007.

Broch, J., Maltz, D., Johnson, D., Hu, Y. E Jetcheva, J. (1998) "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols". Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking. Outubro.

Bruno, R., Conti, M. e Gregori, E. (2005) "Mesh Networks: Commodity Multihop Ad Hoc Networks". IEEE Communications Magazine. Março.

Capkun, S., Buttyan, L., and Hubaux, J.-P. (2003). "Self-organized public-key management for mobile ad hoc networks". IEEE Transactions on Mobile Computing,

Chen, W., Jain N., Singh S., (1999) "ANMP: Ad Hoc Network Management Protocol", IEEE Journal on selected areas in communications, Vol. 17, No. 8, Agosto.

Chiang, C., Wu, H., Liu, W., Gerla, M. (1997) "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel". IEEE Singapore International Conference on Networks (SICON), pp. 197-211, Abril.

Duarte, J. (2008) "Escalabilidade, Gerência e Mobilidade para Redes Mesh de Acesso à Internet". Dissertação de Mestrado, instituição Universidade Federal Fluminense – UFF.

Duarte, J.L.; Passos, D.; Valle, R.L.; Oliveira, E.; Muchaluat-Saade, D.; Albuquerque, C.V. (2007). "Management Issues on Wireless Mesh Networks". Latin American

Network Operations and Management Symposium (LANOMS), pp. 8 – 19. Setembro.

Faccin, S., Wijting, C., Kneckt, J. e Damle, A. (2006) “Mesh WLAN Networks: Concept and System Design”. IEEE Wireless Communications. Abril.

G. Pei, M. Gerla, Tsu-Wei Chen (2000) "Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks," IEEE ICC, vol. 1, pp. 70 -74.

GT-Mesh (2006) “RT1 – Termo de referência e estado da arte”. Disponível em <http:// mesh.ic.uff.br>. Acessado em Junho de 2008.

Haas, Z. e Pearlman, M. (1998) "The zone routing protocol (ZRP) for ad hoc networks". Internet-Draft, draft-ietf-manet-zone-zrp-04.txt. Agosto.

Haas, Z.J. (1997) “A new routing protocol for the reconfigurable wireless networks”, IEEE Computer Society. Vol 2. pp. 562-566.

Hamid, Md. Abdul; Islam, Md. Shariful; Hong, Choong Seon. (2008). “Misbehavior Detection in Wireless Mesh Networks”. International Conference on Advanced Communication Technology (ICACT), Vol. 2, pp. 1167 – 1169, Fevereiro.

Hamid, Md. Abdul; Islam, Md. Shariful; Hong, Choong Seon. (2008). “Developing Security Solutions for Wireless Mesh Enterprise Networks”. IEEE Wireless Communications and Networking Conference (WCNC) 2008, pp. 2549 – 2554, Abril.

Hamid, Z., Khant, S. (2006) “An Augmented Security Protocol for WirelessMAN Mesh Networks”. IEEE.

Haq, A., Naveed, A., Kanhere, S. (2007) “Securing Channel Assignment in Multi-Radio Multi-Channel Wireless Mesh Networks”. IEEE Communications Society subject matter experts for publication in the WCNC 2007 proceedings.

Harada, Eduardo. (2006) “Wireless Mesh Networks: Uma tecnologia que promete”, Disponível em <http://sisnema.com.br/Materias/idmat017459.htm>. Acessado em Junho de 2008.

Held, G. (2005) “Wireless Mesh Networks”. 1ª Edição. USA, Auerbach Publications – Taylor & Francis Group.

Hiertz, G., Zang, Y., Max, S., Junge, T., Weiss, E., Wolz, B. (2008) “IEEE 802.11s: WLAN Mesh Standardization and High Performance Extensions”. IEEE Network, Junho.

Hong, F.; Fu, L. H. C. “Advanced Information Networking and Applications”, IEEE Computer Society, vol. 1. 2005, pp. 713-718.

Hu, Y. C., Perrig, A. (2004) “A Survey of Secure Wireless Ad Hoc Routing”, IEEE Computer Society, vol. 2, pp. 28-29.

Hu, Y. C., Perrig, A., Johnson, D. B. (2005) “Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks”. Portal ACM. vol. 11, pp. 21-38.

Hu, Y. C., Perrig, A., Johnson, D. B., (2003) “SEAD: Secure Efficient Distance Vector Routing for Mobile wireless ad hoc networks”. IEEE Computer Society. vol. 1. pp. 3-13.

Huang, R., Zhang, C., Fang, Y., (2007) "A Mobility Management Scheme for Wireless Mesh Networks". IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2007 proceedings.

IEEE (2000). IEEE Standard Specifications for Public-Key Cryptography. IEEE Std 1363-2000.

Islam, Md. Shariful; Hamid, Md. Abdul; Hong, Choong Seon; Chang, Beom-Hwan . (2008). "Preserving Identity Privacy in Wireless Mesh Networks". International Conference on Information Networking (ICOIN) pp. 1 – 5, Janeiro.

Janies, J.; Chin-Tser Huang; Johnson, N.L. (2006) "SUMP: A Secure Unicast Messaging Protocol for Wireless Ad Hoc Sensor Networks". IEEE International Conference on Communications (ICC), Vol. 8, Issue , pp. 3663 – 3669, Junho.

Joa-Ng, M., Lu, I. T. (1999) "A Peer-to-Peer zone-based two-level link state routing for mobile Ad Hoc Networks". IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks, pp.1415-25, Agosto.

Jongtack, Kim; Saewoong, Bahk. (2008) "Distributed Certification Authority in Wireless Mesh Networks". IEEE Consumer Communications and Networking Conference (CCNC), pp. 267 – 271, Janeiro.

Ju, N., Ju,D., Santhanam, L., He, B., Wang, J., Agrawal, D., (2007) "Wireless Mesh Networks: Current Challenges and Future Directions of Web-in-The-Sky". IEEE Wireless Communications, Agosto.

K. Graffi (2006) "A Security Framework for Organic Mesh Networks," Master's thesis, TU-Darmstadt, Maio.

Karpijoki ,V. (2001), "Security in ad hoc networks", no "Seminar on Network Security.

Kim, J., e Bahk, S. (2008) "MeCA: Distributed Certi.cation Authority in Wireless Mesh Networks". IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2008 proceedings.

Krebs, M., Krempels, K., Kucay, M. (2008) "Service Discovery in Wireless Mesh Networks". IEEE Communications Society subject matter experts for publication in the WCNC 2008 proceedings.

Krebs, Martin; Krempels, Karl-Heinz; Kucay, Markus (2008) "Service Discovery in Wireless Mesh Networks". IEEE Wireless Communications and Networking Conference (WCNC), pp. 3093 – 3098, Abril.

Kurose, James F, Ross, Keith W. (2006) "Redes de Computadores e a Internet – Uma abordagem Top-Down". Addison Wesley. 3ª Edição.

Kyasanur, P., So, J., Chereddi, C., Vaidya Nitin H., (2007) "Multi-Channel Mesh Networks: Challenges and Protocols", University of Illinois at Urbana-Champaign. Disponível em <http://www.hserus.net/~cck/pubs/wcom.pdf>. Acessado em Junho de 2008.

Lee, M., Zheng, J., Ko, Y. e Shrestha, D. (2006) "Emerging Standards For Wireless Mesh Technology". IEEE Wireless Communications. Abril.

Li, Guangsong (2007) “An Identity-Based Security Architecture for Wireless Mesh Networks”. NPC Workshops. IFIP International Conference on Network and Parallel Computing Workshops, pp. 223 - 226, Setembro.

Li, X., Yang, W., Moon, S., Mal, J. (2006) “Authentication Method for 802.11s Infrastructure Mode”. IEEE.

Li, Xiang-Yang; Wu, Yanwei; Wang, WeiZhao. (2007) “Stochastic Security in Wireless Mesh Networks via Saddle Routing Policy”. International Conference on Wireless Algorithms, Systems and Applications, pp. 121 – 128.

Lin C., (1997) “Dynamic key management schemes for access control in a hierarchy”, Computer Communications, Vol 20, No 15, Dezembro.

Lin, X., Lu, R., Ho, P., Shen, X., Cao, Z. (2008 “TUA: A Novel Compromise-Resilient Authentication Architecture for Wireless Mesh Networks”. IEEE Transactions on Wireless Communications, Vol. 7, No. 4, Abril.

Lou, W., Fang, Y. (2004) “A Survey on Wireless Security in Mobile Ad Hoc Networks: Challenges and Possible Solutions”, edited by X. Chen, X. Huang and D.-Z. Du, Kluwer Academic Publishers/Springer.

Luiz, A., Júnior, O. (2005) “Infra-estrutura e Roteamento em Redes Wireless Mesh”, Pontifícia Universidade Católica do Paraná – PUC-PR.

Mahmoud, A., Sameh, A.,El-Kassas, S. (2005) “Reputed authenticated routing for ad hoc networks protocol (reputed-ARAN)”. IEEE Computer Society, pp. 258-259.

Maki S., Aura T., Hietalahti M. (2000) ”Robust membership management for ad-hoc groups.” In Proc. 5th Nordic Workshop on Secure IT Systems (NORDSEC 2000), Reykjavik, Iceland.

Meshdynamics Home Page (2001) “High Performance Outdoor Wireless Mesh Networking”. Disponível em <http://www.meshdynamics.com/>. Acessado em Junho de 2008.

Mishra, A., NAadkami K., Patcha, A. (2004) “Intrusion detection in wireless ad hoc networks”. IEEE Wireless Communications, vol. 11. pp 48-60.

Mogre, Parag S., Graffi, K´alm´an, Matthias Hollick, and Ralf Steinmetz (2007) “AntSec, WatchAnt, and AntRep: Innovative Security Mechanisms for Wireless Mesh Networks”. 32nd IEEE Conference on Local Computer Networks (LCN), pp. 539 – 547, Outubro.

MotoMesh (2008). Disponível em <http://www.motorola.com/business/v/index.jsp?vgnextoid=ef98fde3807f6110VgnVCM1000008406b00aRCRD> . Acessado em Junho de 2008.

Murthy, C. e Manoj, B. (2004) “Ad Hoc Wireless Networks: Architectures and Protocols”. 1ª Edição. USA, Prentice Hall Professional Technical Reference.

Naveed, A., Kanhere, S. (2006) “Security Vulnerabilities in Channel Assignment of Multi-Radio Multi-Channel Wireless Mesh Networks”. IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.

Ornaghi, A., Valleri, M. (2008) “Etercap”. Disponível em <http://ettercap.sourceforge.net>. Acessado em julho de 2008.

Papadimitatos, P., Haas, Z. J. (2003) “Secure link state routing for mobile ad hoc networks”. IEEE Computer Society. pp 379-383.

Perkins, C. E. (1994) “Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers”. Portal ACM, vol. 24, pp 234-244.

Rahman, S. M. M., Inomata, A., Okamoto, T., Mambo, M., Okamoto, E. (2006) “Anonymous secure communication in wireless mobile ad-hoc networks”. Proc. 1st Intl. Conf. on Ubiquitous Convergence Technology, pp. 131–140, Dezembro.

Raya, M., Hubaux, J-P. (2007) “Securing vehicular ad hoc networks,” Journal of Computer Security, Special Issue on Security of Ad Hoc and Sensor Networks, vol. 15, no. 1, pp. 39–68.

RFC 3561 (2003), “Ad hoc On-Demand Distance Vector (AODV) Routing”. Disponível em <http://www.ietf.org/rfc/rfc3561.txt>, visitado em julho de 2008.

RFC 3626 (2003), “Optimized Link State Routing Protocol (OLSR)”. Disponível em <http://www.ietf.org/rfc/rfc3626.txt>, visitado em julho de 2008.

Royer, E., Toh, C. (1999) “A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks”. IEEE Personal Communications, pp. 46 – 55, Abril.

S. Murthy and J. J. Garcia-Luna-Aceves (1996) “An Efficient Routing Protocol for Wireless Networks”. ACM Mobile Networks and App. J., Special Issue on Routing in Mobile Communication Networks, pp. 183–97, Outubro.

S. Seys and B. Preneel, (2006) “ARM: Anonymous routing protocol for mobile ad hoc networks”. Proc. 20th Int’l Conf. on Advanced Information Networking and Applications (AINA), pp. 133–137, Abril.

Salem, B. N.; Hubaux, J.-P.; (2006) “Securing Wreless Mesh Networks”. IEEE Wirelesss Communications. Abril.

Salem, N. B. and Hubaux, J-P. (2006) “Securing Wireless Mesh Networks”, in IEEE Wireless Communication, Vol 13, Issue 2, pp. 50 – 55, Abril.

Siddiqui, M. S., Amin, S. O., Hong, C. S., (2008) “An Efficient Mechanism for Network Management in Wireless Mesh Network”. 10th International Conference on Advanced Communication Technology (ICACT), Vol 1, pp. 301 – 305, Fevereiro.

Siddiqui, M. S., Amin, S. O., Kim, J., Hong, C., (2007) “MHRP: A secure multi-path hybrid routing protocol for wireless mesh network”

Siddiqui, M. S., Hong, C. S. (2007) “Security Issues in Wireless Mesh Networks”. International Conference on Multimedia and Ubiquitous Engineering (MUE), pp. 717 – 722, Abril.

Sivakumar, R., Sinha, P. e Bharghavan, V. (1998) "Core extraction distributed ad hoc routing (CEDAR) specification". Internet-Draft, draft-ietf-manetzone-cedar-00.txt. Outubro.

Song, R., Korba, L., Yee, G. (2005) “AnonDSR: efficient anonymous dynamic source routing for mobile ad-hoc networks”. Portal ACM. pp 33-42.

Stallings W., (1998) “Snmp, SnmpV2, Snmpv3, Rmon 1 e 2”. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA

Sun, J., Zhang, C., Fang, Y. (2008) “A Security Architecture Achieving Anonymity and Traceability in Wireless Mesh Networks”. IEEE INFOCOM 2008.

Takeda, S., Yagyu, K., Aoki, H., Matsumoto, Y. (2005) “Multi-Interface Oriented Radio Metric On-Demand Routing Protocol for Layer-2 Mesh Network”. IEICE Technical Report RCS2005-59, Julho.

Tamashiro, C. (2007) “Uma Análise de Protocolos de Roteamento Anônimo para Redes Sem Fio Ad Hoc Móveis”. Dissertação de Mestrado, instituição Universidade Federal de Santa Catarina – UFSC.

Tropos Home Page (2008) “ Wireless Broadband You Control”. Disponível em <http:// www.tropos.com/>. Acessado em Junho de 2008.

Xian, Y., Huang, C. (2007) “Securing VoIP Services in Multi-Hop Wireless Mesh Networks”. IEEE ISWCS.

Xiao, Yang (2008) “Accountability for Wireless LANs, Ad Hoc Networks, and Wireless Mesh Networks”. IEEE Communications Magazine, pp 116 a 126, Abril.

Xue, Q., Ganz, A. (2005) “QoS routing for Mesh-based Wireless LANs”, Department of Electrical and Computer Engineering - University of Massachusetts. Disponível em <http://www-unix.ecs.umass.edu/~qxue/publ/WMR_April(IJWIN).pdf>. Acessado em Junho de 2008.

Zapata, M. G. (2002) “Secure ad hoc on-demand distance vector routing”. Portal ACM. vol. 6. pp. 106-107.

Zhang, Y., Fang, Y. (2006) “ARSA: An attack-resilient security architecture for multihop wireless mesh networks,” IEEE J. Select. Areas Communications, vol. 24, no. 10, pp. 1916–1928, Outubro.

Zhang, Z., Nait-Abdesselam, F., Ho, P., Lin, X. (2008) “RADAR: a ReputAtion-based Scheme for Detecting Anomalous Nodes in WiReless Mesh Networks”. IEEE Communications Society subject matter experts for publication in the WCNC 2008 proceedings.

Zheng, X., Chen, C., Huang, C., Matthews, M., Santhapuri, N. (2004) “A Dual Authentication Protocol for IEEE 802.11 Wireless LANs”. Disponível em <http://www.cse.sc.edu/~huangct/iswcs05cr.pdf >. Acessado em Junho de 2008.

Zhou, L, Haas, Z. J. (1999) “Securing ad hoc networks,” IEEE Network Magazine, vol. 13, no. 6, pp. 24–30, Dezembro.

Zhu, H., Lin, X., Lu, R, Ho, P., Shen, X. (2007) “Secure Localized Authentication and Billing for Wireless Mesh Networks”. IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2007 proceedings.

Capítulo

3

Tráfego Internet não Desejado: Conceitos, Caracterização e Soluções

Eduardo Feitosa[1,2], Eduardo Souto[2], Djamel Sadok[1].

[1]Centro de Informática – Grupo de Pesquisa em Redes e Telecomunicações (GPRT)
Universidade Federal de Pernambuco
Caixa Postal 7851 – Recife – PE - Brasil

[2]Departamento de Ciência da Computação (DCC)
Universidade Federal do Amazonas
69077000 - Manaus - AM - Brasil

{elf, jamel}@cin.ufpe.br, esouto@ufam.edu.br

Abstract

In today's Internet architecture, the unwanted traffic generated by technological trends in network, new applications and services, and security breaches has affected a large portion of the Internet. This chapter presents the unwanted traffic universe including definitions, classification, and the reasons that explain its growth. It also points out the shortcomings of the existent solutions for detection of unwanted traffic. Lastly, this chapter presents potential solutions and a list of research topics and open problems on unwanted Internet traffic.

Resumo

Na atual arquitetura da Internet, o tráfego não desejado gerado por tendências tecnológicas em redes, novas aplicações e serviços, e violações de segurança tem afetado uma porção cada vez maior da Internet. Este capítulo apresenta o universo do tráfego não desejado incluindo definições, classificação e os motivos que explicam seu surpreendente crescimento. Além disso, aponta as deficiências das soluções existentes e recentes para detecção de tráfego indesejado. Por fim, este capítulo apresenta potenciais soluções e enumera temas de pesquisa em aberto sobre tráfego não desejado na Internet.

3.1. Introdução

Uma breve análise do tráfego Internet comprova o crescente aumento no transporte do tráfego considerado desconhecido, não solicitado, improdutivo e muitas vezes ilegítimo, em outras palavras, tráfego não desejado. Originado através de atividades como, por exemplo, mensagens eletrônicas não solicitadas (*spam*); atividades fraudulentas como *phishing*[1] e *pharming*[2]; ataques de negação de serviço (do inglês *Distributed Denial of Service* - DDoS); proliferação de vírus e *worms*; *backscatter*[3], entre outros, o tráfego não desejado pode ser considerado uma pandemia cujas conseqüências refletem-se no crescimento dos prejuízos financeiros dos usuários da Internet.

Exemplos de perdas financeiras podem ser encontrados em todo o mundo. Em 2006, os prejuízos ocasionados por *worms* foram de aproximadamente US$ 245 milhões, somente entre provedores de acesso norte-americanos [Morin, 2006]. O instituto de segurança americano CSI (*Computer Security Institute*) [Richardson, 2007], depois de entrevistar 194 empresas nos Estados Unidos, contabilizou perdas superiores a US$ 66 milhões ocasionadas pelo tráfego não produtivo gerado principalmente por fraudes, vírus, *worms*, *spyware* e intrusões em 2007. No Brasil, o CERT.br (Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil) contabilizou o número de incidentes relacionados às tentativas de fraude em 45.298 em 2007 [CERT.br, 2008] enquanto que, no mesmo período, o CAIS (Centro de Atendimento a Incidentes de Segurança) da RNP (Rede Nacional de Pesquisa) registrou cerca de 4000 tentativas de fraudes através de *spam* e *phishing*.

Parte desses prejuízos se deve a ineficiência das atuais soluções em identificar, reduzir e interromper o tráfego não desejado. Tipicamente, a efetividade fornecida pelas soluções existentes só é percebida após a ocorrência de algum dano. Além disso, a alta taxa de alarmes falsos e a falta de cooperação com outras soluções ou mesmo com a infra-estrutura de rede são fatores considerados incentivadores do aumento do tráfego não desejado. Como mencionado em [Oliveira et al., 2007], as soluções usadas para detectar e reduzir os efeitos de ataques DDoS tais como filtragem, limitação de banda, IP *traceback* e esquemas de marcação de pacotes são difíceis de implementar porque necessitam de mudanças na infra-estrutura da Internet. Ao mesmo tempo, soluções tradicionais como firewall e VPN (*Virtual Private Network*) são ineficazes contra códigos maliciosos e *spam*.

Outro ponto relacionado à ineficácia das atuais soluções é a definição do que é tráfego não desejado. Uma vez que normalmente parte deste tipo de tráfego é o resultado de atividades recreativas como o download de músicas e vídeos e jogos on-line. Como exemplos de aplicações utilizadas em atividades recreativas podem-se citar: Emule, Bit Torrent e Kazaa na categoria de aplicações P2P; Skype, MSN e Google Talk como aplicações utilizadas na comunicação instantânea; e Joost e Justin.TV[4] como aplicações de TV via Internet (em tempo real). Muitas das soluções existentes não reconhecem ou são configuradas para não detectar o tráfego gerado por essas atividades.

---

[1] *Phishing* é um tipo de fraude eletrônica caracterizada pela tentativa de obter informações pessoais privilegiadas através de sites falsos ou mensagens eletrônicas forjadas.

[2] *Pharming* refere-se ao ataque de envenenamento de cache DNS cujo objetivo é preparar terreno para atividades de *phishing*.

[3] *Backscatter* é o tráfego recebido de vítimas que estão respondendo a ataques de negação de serviço.

[4] http://www.justin.tv

3.1.1. Definições

O termo "tráfego não desejado" foi introduzido na década de 80 e sempre esteve relacionado com atividades maliciosas como vírus, *worms*, intrusões e ataques em baixa escala. Já a nomenclatura que o tipifica como qualquer tráfego Internet não solicitado, não produtivo, não desejado e ilegítimo é recente.

O trabalho de Pang et al. [Pang et al., 2004] define tráfego não desejado como sendo um tráfego não produtivo composto por uma parte maliciosa (causada pelo tráfego de *backscatter* associado a atividades maliciosas como varreduras por vulnerabilidades, *worms*, mensagens de *spam*, etc.) e uma parte benigna (causada por má configuração de roteadores, *flash crowds*, etc.). Soto [Soto, 2005] complementa essa definição, afirmando que o tráfego não desejado também pode ser gerado pelo tráfego "corrompido" devido a ruído ou interferências em linhas de transmissão da rede.

Em [Xu et al., 2005a], tráfego não desejado é caracterizado como aquele malicioso ou não produtivo cuja finalidade é comprometer computadores (*hosts*) vulneráveis, propagar códigos maliciosos, proliferar mensagens de *spam* e negar serviços. Outras nomenclaturas para definir tráfego não desejado são: tráfego lixo (*junk traffic*), tráfego de fundo (*background*) e tráfego anormal.

Em linhas gerais, a definição mais genérica sobre tráfego não desejado é qualquer tipo de tráfego de rede não requisitado e/ou inesperado, cujo único propósito é consumir recursos computacionais da rede, desperdiçar tempo e dinheiro dos usuários e empresas e que pode gerar algum tipo de vantagem ou benefício (lucro) para seus criadores.

Entretanto, não existe um consenso sobre o que é ou não tráfego indesejado. Comumente, essa definição depende do contexto em que está localizada e/ou da aplicação que está sendo utilizada. Por exemplo, a China trata o tráfego gerado pelo SkypeOut[5] (especificamente chamadas feitas de computadores para telefones comuns) como ilegal porque afeta a receita das operadores de telefonia. Seguindo a mesma linha de raciocínio, provedores de serviço Internet, empresas de telecomunicações, empresas públicas e privadas têm limitado o uso de aplicações *peer-to-peer* (P2P) alegando que este tipo de tráfego não é produtivo e é potencialmente empregado para proliferação de vírus e distribuição de códigos maliciosos, além de ferir e quebrar as leis de direitos autorais. Na linha de aplicações consideradas "proibidas" estão IRC (*Internet Relay Chat*), sites de relacionamento (Orkut, MySpace, Facebook, entre outros), mensagens instantâneas (MSN, Google Talk, ICQ) e jogos on-line.

3.1.2. Discussões

Tráfego não desejado e não solicitado não é nenhuma novidade na Internet. No entanto, nos últimos 10 anos, este tipo de tráfego cresceu de forma surpreendente em volume de informação e sofisticação, atingindo níveis considerados alarmantes. Mas como explicar essa explosão? Fatores como a filosofia aberta da Internet (sem centros de controle), a existência de uma indústria ilícita por trás da produção e a proliferação de tráfego indesejado, a pouca importância atribuídas às questões de segurança, entre outros podem ajudar a explicar o significativo aumento do tráfego não desejado.

[5] http://www.skype.com

A preocupação com este tipo de tráfego é tamanha que congressos e workshops como o SRUTI (*Steps to Reducing Unwanted Traffic on the Internet*) [SRUTI, 2005], que desde 2005 vem sendo realizado para discutir os problemas e apresentar novas soluções. Até o momento, o evento mais importante nesta área foi o workshop do IAB (*Internet Architecture Board*) sobre tráfego não desejado, realizado em Março de 2006. Seu objetivo foi o de promover o intercâmbio de informações e experiências entre operadores, fabricantes, desenvolvedores e pesquisadores. Outro objetivo foi levantar projetos e tópicos de pesquisa que provavelmente serão geridos pelo IAB, IETF (*Internet Engennering Task Force*), IRTF (*Internet Research Tack Force*) e pela comunidade no desenvolvimento de soluções contra o tráfego não desejado. O resultado do workshop foi descrito na RFC (*Request for Comments*) 4948 [Anderson et al., 2007] que relata os tipos de tráfego não desejado encontrados na Internet, suas principais causas, as soluções existentes e as ações a serem tomadas para resolver o problema.

3.1.3. Organização do Capítulo

Diante do exposto, torna-se claro que o tráfego não desejado apresenta-se como um dos principais problemas de segurança e que precisa de soluções que contribuam para redução de seu atual nível, embora que ainda não seja trivial produzi-las. Quais os tipos de tráfego não desejado estão disponíveis na Internet? Qual é o objetivo de cada um? Quais são e onde estão as principais fontes de tráfego? O que precisa ser feito para atenuar os efeitos destes tipos de tráfego? Estas perguntas precisam ser respondidas cuidadosamente se quisermos continuar a utilizar a Internet. Os autores deste minicurso partilham da opinião de que a descoberta e interceptação precoce do tráfego não desejado é o modo mais seguro de garantir danos limitados.

O restante deste capítulo está organizado da seguinte forma. A seção 3.2 caracteriza o tráfego não desejado apresentando as principais vulnerabilidades da Internet que permitem sua geração e proliferação. Além disso, também são descritos os principais tipos de ataques, atividades e aplicações relacionados ao tráfego indesejado. Esta seção finaliza com a apresentação de algumas taxonomias. A seção 3.3 apresenta as soluções existentes contra o tráfego não desejado. Tais soluções são divididas em tradicionais (por exemplo, firewall, *honeypots* e ferramentas de medição de tráfego) e as baseadas na análise do tráfego. Em ambos os casos, as vantagens e desvantagens de cada solução são comentadas. A seção 3.4 discute soluções consideradas potencias e/ou futuras. Esta seção tem o intuito de apresentar ferramentas, aplicações e soluções disponíveis (implementadas) apontadas como promissoras na contenção do tráfego não desejado, mas cuja implantação exige mudanças na infra-estrutura da Internet ou apenas vontade e incentivo para fazê-la. Além disso, os autores deste minicurso apresentam uma nova proposta nesta seção. Por fim, a seção 3.5 apresenta os comentários finais sobre o tema e aponta algumas questões em aberto.

3.2. Caracterização de Tráfego não Desejado

A atual estratégia de pesquisa sobre tráfego não desejado é baseada em três passos:

i) Adquirir conhecimento sobre as origens e os diferentes tipos de tráfego não desejado;

ii) Avaliar o impacto e a efetividade das soluções existentes; e

iii) Desenvolver novas contramedidas eficazes contra o tráfego não desejado.

Esta seção foca no primeiro passo, descrevendo as principais vulnerabilidades da Internet que contribuem para a geração de anomalias de tráfego e ataques a sua infra-estrutura. Além disso, os principais tipos de tráfego não desejado são apresentados, classificados e exemplificados.

3.2.1. Vulnerabilidades e Problemas Conhecidos

O tráfego não desejado está presente na Internet desde seu surgimento na forma de vírus, *worms*, má configuração de serviços, falhas transitórias em dispositivos de rede como roteadores, ataques DoS e intrusões. Esses incidentes eram tipicamente s causados por erros de operação ou por pessoas (jovens em sua maioria) que tentavam chamar atenção ou provar suas habilidades para o mundo. O exemplo mais famoso foi o *worm* da Internet [Eichin e Rochlis, 1989]. Com o passar dos anos, tais incidentes foram sendo adaptados e/ou substituídos por atividades abusivas e maliciosas em larga escala (por exemplo, *spam*, *phishing*, etc.).

Existem diversas explicações para essa "evolução", até certo ponto natural, dos incidentes e a conseqüente massificação do tráfego não desejado. Na visão deste minicurso, os mais importantes são:

1. **A natureza aberta (sem controle) da Internet**

   A Internet é uma das poucas plataformas operacionais que funcionam sem centros de controle. Fato este que acabou contribuindo para seu efetivo sucesso. Contudo, essa característica impõe uma série de limitações técnicas e problemas, especialmente os relacionados com segurança. A identificação de um atacante é um bom exemplo. Em um ambiente distribuído e diversificado como a Internet, a tarefa de descobrir um atacante é extremamente difícil, visto que a comunicação pode ser feita entre computadores localizados em qualquer lugar do mundo. Além disso, a pilha de protocolos TCP/IP que possibilita essa comunicação não disponibiliza qualquer mecanismo de auditoria que permita o acompanhamento das ações realizadas por um atacante. Como resultado, não existe um limite bem definido sobre o que um computador pode fazer e também não existe qualquer tipo de registro "disponível" do que um computador fez após um incidente.

2. **A veracidade dos endereços de origem**

   Muitos dos ataques encontrados na Internet caracterizam-se pela existência de endereços IP de origem forjados (falsificados). Uma vez que a "falta de controle" é característica funcional da Internet, cabe a cada elemento da rede que recebe pacotes com origens questionáveis ou desconhecidas decidir se aceita ou não, sob pena de bloquear tráfego requisitado e legítimo ou permitir ataques de negação de serviço. Atualmente, existem *Botnets* ou redes zumbi que são formadas por um conjunto de computadores infectados por códigos maliciosos que permitem que esses computadores sejam controlados remotamente para a realização de diversos ataques. Portanto, a questão vai além da diferenciação da veracidade dos endereços IP de origem.

3. **O mau uso dos protocolos na Internet**

   Devido ao fato de sistemas de segurança e firewall bloquearem portas não usadas, o protocolo HTTP (*Hyper Text Tranfer Protocol*), usado inicialmente

para acessar sites web, agora é constantemente empregado como protocolo de transporte genérico para aplicações que tem pouca ou nenhuma relação com a web como, por exemplo, comunicação VoIP e compartilhamento de arquivos. A explicação é simples, o HTTP tem caminho liberado em quase todos os firewalls. Assim, é mais fácil reaproveitar a infra-estrutura de comunicação do HTTP em novas aplicações ao invés de projetar aplicações seguras e que negociam passagem pelos filtros de segurança e firewall. O resultado é que a mesma infra-estrutura Internet utilizada para realizar tarefas cotidianas como acessar sites web, ler mensagens de correio eletrônico, conversar com pessoas e até fazer compras, também é utilizada para divulgar o tráfego não desejado.

4. Computadores e sistemas comprometidos

A existência de enormes quantidades de computadores e sistemas comprometidos capazes ou efetivamente usados em atividades maliciosas serve como campo fértil para a proliferação do tráfego não desejado. Basicamente, o que acontece é que existe uma enorme quantidade de usuários (pessoas e empresas) novatos ingressando no ambiente da Internet, onde grande parte desses usuários não é preparada ou não está interessada em questões de segurança. Associado a este fato, diariamente são descobertas vulnerabilidades em sistemas operacionais, plataformas e aplicações. A soma desses dois fatos torna a Internet um local adequado e convidativo para o tráfego não desejado. Exemplos como, o aparecimento do grande número de *worms* que exploram erros de programação e falhas de segurança em sistemas operacionais e aplicativos, sites e mensagens de correio eletrônico falsificados que instalam códigos maliciosos no computador das vítimas e aplicações P2P que ajudam a disseminar vírus são noticiados quase que diariamente no mundo todo. Além disso, já faz um bom tempo que as atividades maliciosas não são realizadas por "iniciantes curiosos" utilizando *script kiddies*[6]. Atualmente, atacantes e criminosos contratam programadores profissionais para desenvolver ferramentas avançadas para comprometer computadores e sistemas. O pior é que muitas dessas ferramentas estão disponíveis na Internet para melhorias (código aberto) e para uso por novos atacantes.

5. Autenticação

Considere o seguinte (e cada vez mais comum) cenário: "um usuário com um *smartphone* está conectado a rede sem fio da empresa onde trabalha. Ao se deslocar para uma reunião fora da sede, o seu aparelho passa a fazer parte da rede corporativa da operadora de telefonia celular GPRS (*General Packet Radio Service*). Ao parar para tomar um café, é interligado a uma rede pública através de um *hotspot*. Ao chegar à reunião, o aparelho é novamente adicionado à rede da empresa". Apesar de existirem soluções mais simplificadas que permitem toda essa mobilidade, o atual processo de autenticação de usuários e dispositivos conectados a redes ainda é demasiadamente complexo para ser considerado viável ou fácil de usar. Geralmente, os mecanismos de autenticação são vinculados ao tipo de meio físico utilizados, onde são empregadas diferentes credenciais, semânticas e bases de dados de autenticação (diversas tecnologias).

[6] Códigos genéricos usados por atacantes iniciantes para realizar ataques e intrusões

> É nesta verdadeira torre de babel de protocolos e serviços que atacantes investem para conseguir se infiltrar em computadores, sistemas e redes e, conseqüentemente, obter lucro.

Seja de forma isolada ou através da combinação, o fato é que todos esses fatores têm contribuído inevitavelmente para a proliferação do tráfego não desejado tanto em diversidade quanto em volume. Contudo, existe um aspecto mais preocupante sobre tudo isso: a existência de um verdadeiro mercado de submundo (um mercado negro) fortemente estabelecido dentro da Internet responsável por financiar grande parte das atividades ilícitas com o único objetivo de tentar obter ganhos financeiros. Davies [Davies, 2007] afirma que esta "economia informal" está enraizada como uma espécie de cultura dentro da Internet, sendo capaz de movimentar bilhões de dólares ao redor do mundo e que sua erradicação é praticamente impossível. O IAB considera este mercado como "*a raiz de todos os males da Internet*".

A base deste mercado negro são os servidores de IRC que muitas vezes são usados para gerenciar e executar atividades ilícitas como o roubo e a venda de número de contas, senhas de bancos e números de cartões de crédito e, a proliferação de códigos maliciosos. Parte do lucro obtido é reinvestido em atividades ilícitas incluindo a contratação de escritores profissionais para redação de mensagens de *spam* bem elaboradas, programadores para o desenvolvimento de novos e mais robustos vírus, *worms* e *spywares*, além de especialistas em web (programadores e projetistas) para a criação de sites mais sofisticados para atividade de *phishing*.

Para completar esse quadro nada animador, devido à própria arquitetura da Internet, não existe um modo simplificado de atribuir responsabilidade ou punição para atividades não intencionais ou maliciosas. Tomando como exemplo um ataque de negação de serviço distribuído, onde uma grande quantidade de computadores previamente comprometidos é utilizada para espalhar o ataque através da rede passando por diferentes caminhos e *backbones* até atingir a vítima. Uma vez que existe um grande número de elementos envolvidos (computadores, redes de acesso, *backbones*, roteadores e vítimas) não fica claro quem tem a responsabilidade pelo problema. Além disso, a ausência de um sistema legal em muitos países, incluindo o Brasil, que forneça proteção contra crimes na Internet ou que regule a conduta dos usuários também tem contribuído para o crescimento do tráfego malicioso, não produtivo e não desejado na Internet. Mesmo quando existe alguma jurisdição sobre crimes, como é o caso dos Estados Unidos e Inglaterra, as leis geralmente penalizam as violações somente quando um crime tiver ocorrido.

3.2.2. Tipos de Tráfego não Desejado

Para melhorar a compreensão dos leitores sobre o assunto, esta seção apresenta os principais tipos ataques, anomalias e aplicativos relacionados com o tráfego não desejado ou não solicitado.

3.2.2.1. Ataques de Negação de Serviço

De acordo com a definição do CERT (*Computer Emergency Response Team*) [CERT, 2008], um ataque de negação de serviço consiste em tentativas de impedir usuários legítimos de utilizarem um determinado serviço de um computador ou rede. Em outras palavras, tentar tornar indisponíveis recursos ou serviços oferecidos por um servidor ou

rede. Não existe a tentativa de roubar ou se apropriar de dados sigilosos de usuários como números de cartões de crédito e senhas de contas, mas sim a tentativa de parar serviços que são oferecidos a usuários legítimos. Geralmente, ataques de negação de serviço consumem recursos como memória, poder de processamento, espaço em disco e, principalmente, largura de banda através do envio de uma quantidade de pacotes (solicitações) maior do que o serviço pode suportar.

Os ataques de negação de serviço geralmente são executados de forma distribuída (DDoS) para aumentar ou superdimensionar sua potência. Um ataque deste tipo é mais elaborado e utiliza uma espécie de arquitetura (classe) social composta por:

- **Atacante**: O responsável por coordenar o ataque.
- **Mestre**: Computador intermediário localizado entre o atacante e os computadores zumbis, que recebe os parâmetros (ordens) para o ataque. Cada mestre controla certo número (centenas ou milhares) de zumbis.
- **Zumbi**: Computador que efetivamente realiza o ataque.

A Figura 3.1 exemplifica a estrutura de um ataque de negação de serviço distribuído (DDoS).

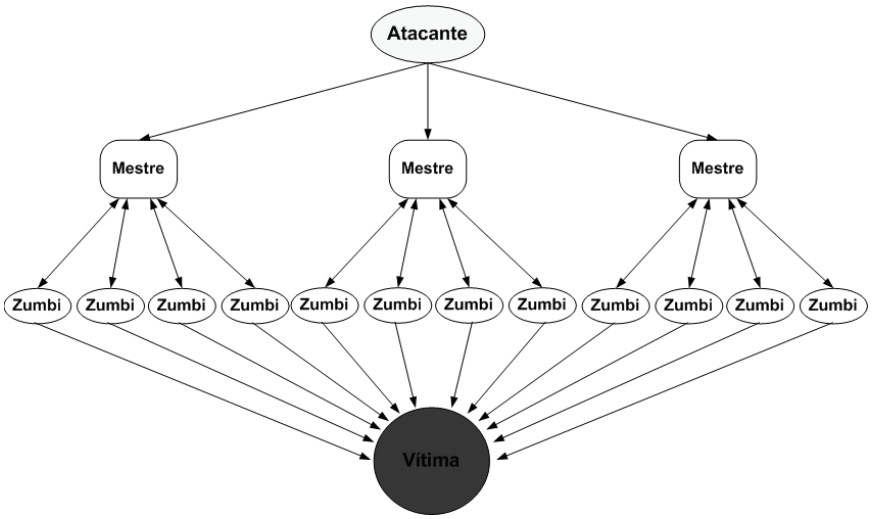

Figura 3.1. Estrutura de um ataque de negação de serviço distribuído.

Os principais tipos de ataque de negação de serviço baseiam-se em algumas características da pilha de protocolos TCP/IP, podendo, assim, afetar praticamente todos os computadores. A forma mais conhecida de ataque é a inundação (*flooding*). Normalmente, utilizam o processo de estabelecimento de conexão do protocolo de transporte TCP (*three-way handshake*), onde um pedido de conexão é enviado ao servidor através de um pacote TCP com a flag SYN (*Synchronize*) ativa. Se o servidor atender ao pedido de conexão, responde com um pacote TCP com a flag ACK (*Acknowledgement*) ativa. Desta forma, um ataque desse tipo (SYN *Flooding*) envia uma grande quantidade de pedidos de conexão até que o servidor não possa mais aceitar

novos pedidos. Existem variantes do ataque de inundação para os protocolos ICMP e UDP como, por exemplo, ICMP *Unreachable*, *Smurf*[7], *Fraggle* e UDP *Packet Storm*[8].

Outra modalidade são os ataques por refletores, chamados de ataques lógicos, que também executam inundações. Neste tipo de ataque, uma estação intermediária (refletor) é colocada entre o atacante e a vítima, para redirecionar o ataque diretamente para a vítima. Para tanto, o atacante envia uma requisição para o refletor onde o endereço de origem enviado é na realidade o endereço da vítima. Um exemplo desse tipo de ataque é o *Smurf*.

O *backscatter*, tráfego recebido de vítimas que estão respondendo a ataques de negação de serviço, também está inserido nesta categoria. Como é caracterizado pelo tráfego de resposta dos ataques, alguns tipos de pacotes envolvidos no *backscatter* são TCP SYN/ACK, TCP RST/ACK e certos tipos de ICMP como *Echo Reply* e *Destination Unreachable*. Normalmente, tráfego *backscatter* tem seu endereço IP de origem forjado para representar espaços de endereçamento não usados.

Informações mais detalhadas sobre ataques de negação de serviço e variantes podem ser encontradas em [Laufer et al., 2005] e [Mirkovic et al., 2004].

3.2.2.2. Ataques ao DNS

O DNS (*Domain Name System*) é uma base de dados hierárquica distribuída que fornece informações fundamentais para operação da Internet como a tradução de nomes dos computadores para endereços IP. Devido a sua importância na estrutura da Internet, qualquer falha tem o potencial de afetar um grande número de usuários. Além da negação de serviço, os ataques ao DNS estão relacionados com a falta de autenticação e integridade dos dados.

O principal tipo de ataque relacionado ao tráfego não desejado é o envenenamento de cache (do inglês *cache poisoning*). Basicamente consiste em corromper a base de informação do serviço DNS, alterando ou adicionando dados sobre os endereços dos servidores. A finalidade é redirecionar conexões legítimas para servidores sobre o domínio dos atacantes (sites e endereços falsos). A principal conseqüência desse tipo de ataque é o *pharming* e seu principal alvo são páginas de instituições financeiras. Segundo [Hyatt, 2006], os atacantes usam *pharming* por quatros razões: identificar dados pessoais para efetuar roubos, distribuição de códigos maliciosos, disseminação de informações falsas e ataques *man-in-the-middle*.

3.2.2.3. Ataques ao Roteamento

O roteamento da Internet é baseado em um sistema distribuído composto por diversos roteadores, agrupados em domínios de gerência chamados sistemas autônomos (do inglês *Autonomous System* - AS). Dessa forma, o roteamento ocorre de duas maneiras: internamente (intra) ou externamente (inter) aos domínios. Em relação aos ataques ocorridos intra-domínios, apesar de relevantes, tendem a não produzir grandes efeitos, uma vez que a quantidade de elementos envolvidos é normalmente reduzida. Ataques ao roteamento entre domínios diferentes são mais preocupantes porque podem afetar todo o tráfego da Internet.

[7] http://www.cert.org/advisories/CA-1998-01.html
[8] http://www.cert.org/advisories/CA-1996-01.html

O grande alvo do tráfego não desejado em relação ao roteamento é o BGP (*Border Gateway Protocol*), um protocolo projetado para o roteamento inter domínios. Segundo [Arbor Networks, 2005], o BGP é um dos cinco pontos mais vulneráveis da Internet. De modo geral, ataques ao BGP interferem no roteamento modificando as rotas do tráfego Internet, mas não afetam a entrega de pacotes normais. Os principais ataques ao BGP são:

- **Redirecionamento**: ocorre quando o tráfego destinado a um determinado endereço, domínio ou rede é forçado a tomar um caminho diferente até um destino forjado. O objetivo deste ataque é personificar o verdadeiro destino para receber dados confidenciais. Comumente, este tipo de ataque é usado em atividades de *phishing* e principalmente fonte de *spam*.
- **Subversão**: é um caso especial de redirecionamento, onde o atacante força o tráfego a passar através de certos enlaces com o objetivo de escutar ou modificar os dados. Em ataques de subversão o tráfego é repassado ao destino correto, tornando o ataque mais difícil de detectar.

3.2.2.4. SPAM

Alguns autores consideram *spam* como sendo toda mensagem comercial não solicitada (do inglês *Unsolicited Commercial E-mail* - UCE). Outros consideram *spam* como sendo mensagens não solicitadas enviadas de forma massiva (do inglês *Unsolicited Bulk E-mail* - UBE). De forma geral, o termo *spam* refere-se ao envio de mensagens não solicitadas de correio eletrônico a um grande número de usuários. Uma definição mais aprofundada sobre *spam* é apresentada em [Taveira et al., 2006].

O *spam* pode ser classificado de acordo com seu conteúdo em:

- Boatos (*hoaxes*) tentam impressionar os usuários através de estórias falsas e assim garantir sua divulgação. Exemplos incluem mensagens do tipo roubo de rins, desaparecimento de crianças, difamação de empresas, a Amazônia como território mundial, etc.
- Correntes (*chain letters*) são mensagens que prometem algum de tipo de lucro financeiro ao leitor se este repassar a mensagem a um determinado número de usuários.
- Propagandas são as mais comuns e mais divulgadas. O melhor exemplo são os comerciais de produtos farmacêuticos para homens.
- Golpes (*scam*) representam mensagens enganosas que afirmar que o leitor foi "agraciado" com algum produto ou que tem a oportunidade de "se dar bem". Exemplos corriqueiros incluem sorteios, novos empregos, chance de ter o próprio negócio, etc.
- Estelionato (*phishing*) são aquelas mensagens escondidas, ou melhor, ocultas em *spams* comerciais que visam obter informações pessoais (contas de banco e senhas, por exemplo) para serem usadas em fraudes ou compras pela Internet. Normalmente, induzem o leitor a acessar a URL indicada na mensagem ou preencher algum tipo de formulário.

- O tipo mais perigoso de *spam* é aquele que contém códigos maliciosos que tentam enganar o leitor a executar um determinado programa enviado junto com a mensagem. O resultado é a instalação de vírus, *worms* e cavalos de tróia, sempre visando alguma tentativa de fraude ou ataques de negação de serviço.

Além desses tipos, existem variações como SPIT e SPIM. SPIT (*Spam via Internet Telephony*) é o envio de mensagens não solicitadas a usuários de telefonia VoIP. SPIM (*Spam via Instant Messages*) representa o envio de mensagens através de aplicativos para troca de mensagens instantâneas.

O fato é que *spam* é ou tornou-se uma verdadeira praga na Internet. O Radicati Group [Radicati Group, 2006] estimou perdas mundiais equivalentes a US $ 198 bilhões relacionadas às mensagens de *spam* em 2007. Além disso, projetou que o número de mensagens de *spam* atingirá 79% do volume mundial de mensagens de correio eletrônico em 2010.

3.2.2.5. Códigos Maliciosos

Códigos maliciosos representam o tráfego empregado para causar danos, inicialmente, em computadores e, por conseguinte, em redes sem o consentimento do usuário. Normalmente, esses códigos maliciosos roubam dados, permitem acesso não autorizado, vasculham sistemas (*exploits*) e utilizam computadores e redes comprometidas (*botnets*) para proliferar mais tráfego não desejado. Os principais exemplares de códigos maliciosos são:

- **Vírus**: são programas que modificam a operação normal de um computador, sem permissão e conhecimento do usuário. Assim como um vírus biológico, um vírus de computador se replica e se espalha introduzindo cópias suas em outros códigos ou programas executáveis. Tipicamente, o ciclo de vida de um vírus tem quatro fases: *(i)* dormente, onde permanece desativado esperando um sinal para acordar como, por exemplo, uma determinada data; *(ii)* propagação ou replicação; *(iii)* ativação (é ativado para executar sua "função"); e *(iv)* execução [Heidari, 2004]. Ao contrário de um *worm*, um vírus não pode infectar outros computadores sem auxílio externo.

- **Worms**: é um programa auto-replicante que é capaz de se auto-propagar através da rede explorando principalmente falhas de segurança em serviços. A taxa de propagação dos *worms* é muito rápida e pode ameaçar a infra-estrutura da Internet uma vez que cada máquina infectada torna-se um potencial atacante. De modo geral, prejudicam a rede consumindo largura de banda. A ativação de um *worm* pode ser tão rápida quanto sua velocidade de propagação. Entretanto, alguns podem esperar dias ou semanas até se tornarem ativos. O processo de ativação pode ser direto, através da execução por um usuário humano, programado ou ainda auto-ativado. Informações mais detalhadas sobre *worms* podem ser encontradas em [Weaver et al., 2003].

- **Cavalo de Tróia (*Trojan Horse*)**: são programas que uma vez ativados, executam funções escondidas e não desejadas como, por exemplo, varreduras de endereços IP e porta (TCP SYN) de alta carga, envio de grande volume de *spam*, ataques DDoS ou até mesmo adicionar o computador em uma *botnet*. Diferente dos *worms*, um cavalo de tróia não se auto-propaga, depende da interferência e

curiosidade humana para se propagar. Atualmente, cavalos de tróia são conhecidos como "*gimme*", uma gíria para "*give me*", em referência as mensagens de *spam* que prometem lucro ou conteúdo picante.

- **Spyware**: são programas espiões que automaticamente recolhem informações sobre o usuário e os transmite para seus "instaladores". Geralmente, os dados monitorados referem-se aos habítos de compras na Internet ou a informações confidenciais como contas bancárias e senhas pessoais. Tais dados são, então, vendidos para terceiros ou usados para roubos e fraudes. Os *spywares* são aperfeiçoados constantemente de forma a dificultar sua detecção e remoção. Além dos próprios *spywares*, na categoria de programas espiões também estão o *adware* e o *keylogger*. *Adware*, também usado para definir *spyware*, é um programa que exibe automaticamente publicidade. *Keylogger* é um programa que captura os dados digitados no teclado e os envia ao atacante.

3.2.2.6. Aplicações Recreativas

O tráfego não desejado gerado por esta classe de aplicações é motivado pelo real crescimento da Internet, ou seja, representa a convergência natural entre os diversos tipos de dados, especialmente os multimídia, associado com a demanda dos novos usuários. Exemplos desse tipo de tráfego incluem rádio e televisão via Internet, compartilhamento de arquivos, mensagens instantâneas, jogos on-line interativos e multimídia.

A relação das aplicações recreativas com o tráfego não desejado não é percebida facilmente. Por exemplo, jogos on-line, IPTV e rádio via Internet não são relacionados diretamente a atividades maliciosas, mas o tráfego gerado pelos seu usuários espalhados pelo mundo é responsável por um grande consumo de largura de banda em determinadas redes, especialmente as de borda. Outro bom exemplo são as redes sociais[9] que também não estão diretamente relacionadas a atividades maliciosas, mas contribuem para o tráfego não desejado proliferando vírus, *worms* e *spywares*. Além disso, afetam a produtividade de empresas, uma vez que os funcionários que participam delas "dedicam" parte de seu tempo de trabalho para manter todos os seus contatos atualizados.

Por outro lado, o compartilhamento de arquivos via aplicações P2P é uma fonte reconhecida de tráfego não desejado responsável pela proliferação de códigos maliciosos e principal incentivador da pirataria, uma vez que fere as leis de direitos autorais ao "divulgar" conteúdo restrito. Um fato que chama atenção é o rápido crescimento do tráfego gerado por este tipo de aplicação desde 2003, o que tem exigido cada vez mais recursos das redes. Um estudo da empresa alemã Ipoque [Schulze e Mochalski, 2007] mostra que em 2007, o tráfego da Internet gerado pelo compartilhamento de arquivos via P2P correspondeu a algo entre 49% e 83% do volume mundial. Em certos períodos do dia, como as madrugadas, esse índice se aproximava dos 95%.

[9] Redes sociais representam a interação entre seres humanos através da formação de grupos ou relacionamentos. MySpace, Facebook e Orkut são grandes expoentes de redes sociais. Apesar de ser classificado como mundo virtual, o Second Life também pode ser enquadrado nessa categoria.

3.2.3. Classificação do Tráfego não Desejado

Para compreender o universo do tráfego não desejado, o modo mais usual é a categorização (classificação) dos tipos comuns. A primeira classificação formal sobre o assunto foi especificada em [Anderson et al., 2007] que define três categorias:

- **Perturbantes** (do inglês *nuisance*): como o próprio nome diz, representa o tráfego de fundo que "atrapalha" o uso da largura de banda e outros recursos como poder de processamento e espaço de armazenamento. Exemplos típicos incluem mensagens de *spam* e o compartilhamento de arquivos P2P. Estes tipos de tráfego normalmente transportam códigos maliciosos ou iludem os usuários a acessar sites não confiáveis e ferem ou quebram as leis de direitos autorais. Nesta categoria também se encaixam os *pop-up spams*, aplicações tipicamente perturbadoras que exibem janelas de mensagens em sistemas operacionais Windows tais como "ocorreu um erro" e "máquina comprometida". Apesar de pouco discutida, segundo [Krishnamurthy, 2006], esse tipo de aplicação é responsável por enviar centenas de milhares de mensagens a cada hora. Ataques DDoS também podem ser incluídos nessa categoria.
- **Maliciosos**: representam o tráfego responsável por divulgar e espalhar códigos maliciosos incluindo vírus, *worms*, *spywares*, etc. Tipicamente, esta classe se categoriza por apresentar um pequeno volume de tráfego, mas também por provocar altas perdas financeiras as vítimas. Normalmente, muitas empresas não "respeitam" esse tipo de tráfego até que algum grave incidente de segurança aconteça. É neste momento, na hora de "colocar a casa em ordem", que operadores e gerentes de rede se deparam com soluções custosas, específicas, que necessitam de especialistas e consomem tempo da equipe de operação e gerenciamento.
- **Desconhecido**: representam todo tráfego que mesmo quando pertencente a uma das categorias acima, por algum motivo não pode ser classificado com tal (tráfego malicioso criptografado ou misturado com tráfego legitimo, por exemplo) ou que ninguém conhece suas intenções ou origens. *Worms* silenciosos (*quiet worms*) são bons exemplos. Tais *malwares* abrem *backdoors* nas vítimas e ficam dormentes por um longo tempo.

Outra forma de caracterização do tráfego não desejado, mencionada por [Soto, 2005], é a classificação de sua origem em: primárias e secundárias. Origens primárias correspondem a toda requisição inicial de comunicação como pacotes TCP SYN, UDP e ICMP *Echo Request*. Nesta categoria se encaixam aplicações P2P, mensagens de *spam*, vírus e *worms*, intrusões e ataques massivos. Já as origens secundárias correspondem ao tráfego de resposta como, por exemplo, pacotes TCP SYN/ACK, TCP RST/ACK e ICMP *Echo Response*. Esta categoria inclui todo o tráfego gerador por *backscatter*, ataques de baixa intensidade ou baixa carga, e tráfego "benigno" como, por exemplo, falhas transitórias, interrupções, má configuração de equipamentos, *flash crowds*[10], entre outros.

[10] O termo *flash crowd* refere-se à situação quando milhares de usuários acessam simultaneamente um site popular. Exemplos comuns incluem liquidações em grandes empresas, divulgação de catástrofes, eventos esportivos, entre outros. O resultado pode ser a interrupção do serviço devido ao grande número de acessos.

3.3. Soluções Existentes

Como mencionado anteriormente, a Internet pode ser vista como uma das poucas plataformas operacionais existentes sem centros de controle. Essa característica serviu tanto como fator de sucesso, o que ajuda a explicar o rápido e exponencial crescimento da Internet, quanto serviu de ponto de fraco e que resultou no tráfego não desejado. Na tentativa de lidar com o lado negativo desse cenário, vários esquemas e soluções têm sido desenvolvidos e usados para identificar e minimizar o tráfego não desejado.

Nesta seção são apresentadas as soluções usadas na detecção e limitação do tráfego não desejado baseado nas vulnerabilidades mencionadas anteriormente. Para facilitar o entendimento, primeiro são discutidas as soluções consideradas tradicionais como firewall, sistemas de detecção de intrusão, *honeypots*, softwares "anti-alguma coisa", ferramentas de medição de tráfego e controle de acesso. Em seguida, serão apresentadas soluções baseadas na análise de tráfego.

3.3.1. Soluções Tradicionais

3.3.1.1. Firewall

O mecanismo para detecção e controle de tráfego indesejado mais empregado no mundo é o firewall. Neste documento, o termo "firewall" refere se a dispositivos (hardware ou software) que aprovam ou negam a troca de tráfego entre redes. Basicamente, firewalls utilizam regras (filtros) que definem o que deve ser feito. Desta forma, todo o tráfego que entra ou sai da rede ou máquina é comparado com as regras e o resultado é uma ação, geralmente permitir ou negar o tráfego.

Não existe um consenso sobre a classificação dos tipos de firewall. A mais usual considera três tipos: filtro de pacotes, filtro de estados e filtros de aplicação (*gateways*). O **filtro de pacote** é um firewall que compara as informações do cabeçalho de cada pacote (endereços IP, portas e protocolo) com as regras definidas para decidir qual ação tomar. Foi o primeiro tipo a ser criado e ainda hoje é bastante utilizado por ser simples e fácil de configurar. Os exemplos mais comuns são as listas de controle de acesso (do inglês *Access Control List* – ACL) e o *ipchain* (integrado ao Kernel 2.2 dos sistemas operacionais Linux). Contudo, são vulneráveis a ataques com endereços IP forjados (bastante usado para geração de tráfego não desejado) e ineficazes contra tráfego criptografado.

Os **filtros de estado** mantêm registros do estado das conexões de rede (TCP e UDP) que estão ativas. A diferença quanto ao filtro de pacotes é que a filtragem pode ser baseada na tabela de estados de conexões estabelecidas e não apenas no cabeçalho. Em outras palavras, o estado das conexões é monitorado a todo o momento, o que permite que a tomada de ação seja definida de acordo com os estados anteriores mantidos em tabela. Existem três tipos de estados: NEW (novas conexões), ESTABLISHED (conexões estabelecidas) e RELATED (conexões relacionadas a outras já existentes). O *iptables*[11] é solução mais conhecida de firewall de estados. Questões de complexidade e custo são apontadas como desvantagem desse tipo de firewall. Atualmente, soluções de filtro de estados incorporaram a inspeção profunda de pacotes (do inglês *Deep Packet Inspection* – DPI) para verificar o tráfego na perspectiva da

[11] http://www.iptables.org

tabela de estado de conexões legítimas. Além disso, técnicas de identificação de tráfego também são utilizadas para procurar possíveis ataques ou anomalias.

Os **gateways de aplicação** (***application-level gateways***) são bastante utilizados no controle de tráfego indesejado uma vez que operam na camada de aplicação vasculhando o conteúdo dos pacotes a procura de indícios de anomalias como, por exemplo, seqüências de caracteres específicos (palavras ou frases) que indicam a presença de ataques, código maliciosos e até mesmo de determinadas aplicações como SMTP (*Simple Mail Transfer Protocol*), FTP (*File Transfer Protocolo*), HTTP, P2P, MSN, etc. Normalmente, funcionam como intermediários (*proxies*) de um determinado serviço, recebendo solicitações de conexão e gerando uma nova requisição para o servidor de destino. A resposta do serviço é recebida pelo firewall e avaliada para checar sua conformidade antes de ser repassada a quem originou a solicitação. Os tipos de filtros de aplicação mais comuns são voltados para correio eletrônico (anti-*spam*) e web (Squid[12], por exemplo). A principal vantagem desse tipo de firewall é a capacidade de avaliar tráfego bem específico e transações criptografadas. Por outro lado, cada novo serviço necessita de um *proxy* específico. Além disso, seu uso geralmente insere mudanças de desempenho no tráfego.

Com a proliferação do tráfego não desejado nos últimos anos, as soluções de firewall, até então destinadas a proteger o perímetro da rede, passaram a ser desenvolvidas para uso pessoal. ZoneAlarm[13] e Sygate[14] são firewall conhecidos voltados para o sistema operacional Windows. Já usuários Linux podem contar com o *IPTables*, inclusive para trabalhar na rede.

3.3.1.2. IDS

Os sistemas de detecção de intrusos ou simplesmente IDS (do inglês *Intrusion Detection System*), como o próprio nome diz, são sistemas (hardware ou software) que tentam descobrir quando um alvo está sob algum tipo de tentativa de acesso não autorizado, ou seja, alguma tentativa de intrusão. Normalmente, os IDS são compostos por sensores que geram eventos e alarmes de segurança que são enviados para uma estação de gerenciamento.

Os IDS podem ser classificados de acordo como a tecnologia de análise (assinaturas e anomalias) ou em relação ao local de aplicação (baseados em rede ou em *host*). Tradicionalmente, as técnicas usadas para analisar os dados coletados buscando detectar intrusões podem ser classificadas em dois grupos: detecção de assinatura e detecção de anomalia. A estratégia baseada em assinatura identifica padrões que correspondem ao tráfego de rede ou dados da aplicação e os compara com uma base de padrões (assinaturas) de ataques conhecidos. Desta forma, ataques conhecidos são detectados com bastante rapidez e com baixa taxa de erro (falsos positivos). Por outro lado, ataques desconhecidos não são detectados. A estratégia baseada na detecção de anomalia funciona com base na construção de perfis de comportamento para aquilo que é considerado como atividade normal. Desvios da normalidade são então tratados como ameaças. Assim, os sistemas de detecção de intrusão baseados em anomalia são capazes de se adaptar a novas classes de anomalias bem com detectar ataques desconhecidos

[12] http://www.squid-cache.org
[13] http://www.zonealarm.com
[14] http://www.symantec.com/norton/sygate/index.jps

(ataques “zero-day”). Uma forma de detectar intrusões é através da análise de seqüências de chamadas de sistema executadas pelos processos, pois estas constituem uma rica fonte de informação sobre a atividade de um sistema.

Em relação ao local de aplicação, existem duas abordagens básicas para a detecção de intrusão: detectores baseados em rede (do inglês *Network-based* IDS - NIDS), que analisam o tráfego de rede dos sistemas monitorados; e detectores baseados em *host* (do inglês *Host-based* IDS - HIDS), que monitoram as atividades locais em um computador, como processos iniciados, conexões de rede estabelecidas e chamadas de sistema executadas. Os NIDS são geralmente empregados em pontos estratégicos da rede para monitorar o tráfego de entrada e saída de todos os dispositivos em uma rede. Podem produzir alarmes para atividades suspeitas e atuar em conjunto com um firewall para automaticamente bloquear o tráfego analisado como malicioso ou anormal. Snort [Snort, 2008] e Bro [Paxson, 1998] são exemplos de sistemas de detecção de intrusão baseados na análise do tráfego de rede. Já os HIDS coletam dados do sistema onde estão instalados. Assim, os sensores ficam instalados na máquina que está sendo monitorada. A maior deficiência dos detectores baseados em *host* é sua relativa fragilidade, uma vez que podem ser desativados ou modificados por um intruso bem sucedido, para esconder sua presença e suas atividades. Esse problema é conhecido como subversão.

O principal problema dos IDS é a precisão. Como existe uma constante mutação seja no tráfego de rede seja nos padrões de assinatura, a detecção de intrusão se torna cada vez mais difícil e sujeita a erros. Segundo Kumar e Stafford [Kumar e Stafford, 1994], uma atividade intrusiva pode ser classificada como verdadeiros positivos, verdadeiros negativos, falsos positivos e falsos negativos. A Figura 3.2 resume os quatro casos possíveis. Os dois primeiros, verdadeiro positivo e verdadeiro negativo correspondem, respectivamente, a correta detecção de uma intrusão e a correta detecção de um evento normal. Já um falso positivo ocorre quando um evento normal é classificado como anômalo. O resultado é que uma atividade maliciosa pode, no futuro, não ser detectada devido a todos os falsos positivos anteriores. O pior caso é falso negativo, onde o tráfego normal é classificado como anômalo.

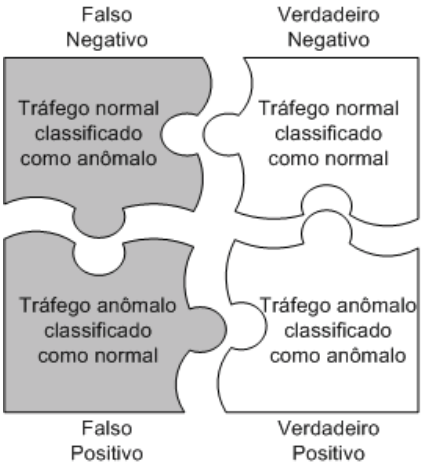

Figura 3.2. Possíveis classificações de um evento.

Por definição, os sistemas de detecção de intrusão são passivos, ou seja, apenas detectam e registram eventos. Contudo, devido à evolução das técnicas de intrusão e a necessidade de tomar decisões de forma mais rápida e precisa, sentiu-se a necessidade de atuar de forma reativa. Surgiram, então, os sistemas de prevenção de intrusão (do inglês *Intrusion Prevention System* - IPS). Um IPS é um sistema com as mesmas funcionalidades de um IDS, mas com a capacidade de automaticamente interagir com outros elementos de segurança para bloquear ou limitar um determinado tráfego malicioso. Além da capacidade de reação, os IPS são projetados para operar on-line e assim permitem prevenção em tempo real. Por fim, algumas implementações de IPS agregam mecanismos DPI para analisar protocolos da camada de aplicação e assim fornecer resultados mais precisos. Fabricantes como Cisco[15] e ISS[16] possuem soluções avançadas de IPS. No âmbito de ferramentas gratuitas, Snort e Untangle IPS[17] são bons exemplos.

Existem também os APS (*Anomaly Prevention Systems*) cuja idéia é considerar cada tipo de anomalia separadamente através da criação de um perfil do comportamento do tráfego [Feroul et al., 2005]. Além disso, APS tem características que permitem que seja aplicado fora do perímetro da rede (*backbones*, por exemplo) enquanto IDS e IPS são voltados à segurança interna da rede.

3.3.1.3. Honeypots

A terceira classe de sistemas tradicionais contra o tráfego não desejado são os *honeypots*. O termo "*honeypot*" refere-se a uma ferramenta de segurança cuja função principal é colher informações sobre ataques e atacantes. Em outras palavras, é um software ou sistema que possui falhas reais ou virtuais de segurança, implementadas propositalmente, com a única finalidade de ser invadido, sondado e atacado para que os mecanismos utilizados na invasão possam ser estudados. Segundo [Spitzner, 2002], um *honeypot* é um recurso de rede cuja função é ser atacado e comprometido (invadido). Significa dizer que poderá ser testado, atacado e invadido. Os *honeypots* não fazem nenhum tipo de prevenção, mas fornecem informações adicionais de valor inestimável. Em relação ao tráfego não desejado, *honeypots* têm sido usados para anunciar espaços de endereçamento não alocados ou não permitidos, além de recolher informações sobre os originadores do tráfego de tais endereços.

Os *honeypots* são classificados de acordo com seu nível de atuação em: de baixa interatividade e de alta interatividade. *Honeypots* de baixa interatividade são ferramentas instaladas para emular sistemas operacionais e serviços com os quais os atacantes irão interagir. Desta forma, o sistema operacional real deste tipo de *honeypot* deve ser instalado e configurado de modo seguro, para minimizar o risco de comprometimento. Também são chamados de *honeypots* de produção porque atuam como elementos de distração até que medidas efetivas possam ser tomadas. São normalmente utilizados para proteger empresas. O exemplo mais conhecido de *honeypot* de baixa interatividade conhecido é o honeyd [Provos, 2004] [Provos, 2008].

Honeypots de alta interatividade são máquinas completas que implementam sistemas operacionais e serviços reais comprometidos e são utilizadas quando se deseja

[15] http://www.cisco.com/
[16] http://www.iss.net/
[17] http://www.untangle.com/

compreender detalhadamente os mecanismos e vulnerabilidades exploradas. Normalmente, ficam localizados no perímetro externo da rede como em uma região de entrada ou zona desmilitarizada. Diferentemente dos *honeypots* de baixa interatividade, esses oferecem um maior risco e demandam pessoal especializado, tempo e dinheiro. São chamados de *honeypots* de pesquisa. As *honeynets* são exemplos de *honeypots* de alta interatividade [Honeynet Project, 2006].

Uma *honeynet* é uma ferramenta de pesquisa, que consiste de uma rede projetada especificamente para ser comprometida e que contém mecanismos de controle para prevenir que seja utilizada como base de ataques contra outras redes [Hoepers et al., 2003]. Uma vez que é projetada para ser atingida e não existem sistemas ou aplicações de produção em uma *honeynet*, todo tráfego que chega é presumido ser malicioso ou resultado de uma configuração incorreta dos serviços da rede. De modo geral, uma *honeynet* é uma rede composta por um ou mais *honeypots* e um *honeywall* (Figura 3.3). O *honeywall* é um gateway que separa os *honeypots* do resto da rede e, por isso é o elemento central de uma *honeynet*.

Em resumo, os *honeypots* podem ser usados para caracterizar o tráfego não desejado com o propósito de advertir previamente operadores e gerentes de rede (além de outros dispositivos tais com firewalls e NIDS) de ataques e anomalias e fornecer tendências que os ajudem a melhorar a segurança da rede. O trabalho de Krishnamurthy [Krishnamurhty, 2004] propõe uma solução usando *honeypots* móveis que permite detectar a origem de ataques o mais perto possível.

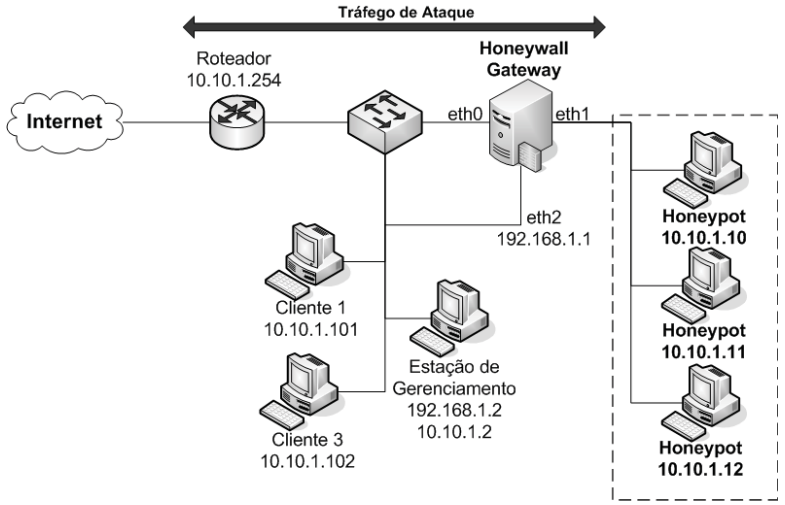

Figura 3.3. Exemplo de arquitetura Honeynet.

3.3.1.4. Software Anti-*

Na categoria de soluções contra o tráfego não desejado estão os software e programas desenvolvidos para detectar e eliminar potenciais ameaças aos computadores e redes como, por exemplo, antivírus, anti-*spyware*, anti-*phishing* e anti-*spam*.

Antivírus são programas que detectam e removem vírus de computador. Uma vez que novos vírus e variantes de vírus conhecidos são "lançados" quase que diariamente, um software antivírus deve manter-se automaticamente ou ser mantido sempre atualizado. Geralmente, os fabricantes de antivírus distribuem atualizações e "vacinas" para vírus específicos gratuitamente. Como existe um grande número de soluções antivírus, o que as diferencia são os métodos de detecção, funcionalidades oferecidas e o preço. Entre os exemplos de antivírus comerciais pode-se citar Norton, McAfee e Trend Micro enquanto que Avast, AVG e Vírus Shield apresentam antivírus gratuitos.

Já os anti-*spywares* são usados no combate a programas e códigos espiões como *spyware*, *adware*, *keyloggers*. Assim como os antivírus, também existem dezenas de soluções anti-*spyware* divididas em comerciais e gratuitas. Ad-Aware SE, Spybot e Windows Defender são exemplos de soluções não comerciais. Spyware Doctor, HijackThis e Spy Sweeper são soluções comerciais. Entretanto, algumas soluções anti-*spyware* são famosas por divulgar *spywares* tais como Spyware Quake, Antivirus Gold, PSGuard, Malware Alarm, entre outros.

Softwares anti-*phishing* visam bloquear possíveis tentativas de fraude através de sites web ou mensagens de correio eletrônico. Normalmente, este tipo de solução é apresentado na forma de barras de tarefas (*toolbar*) integradas ou integráveis com navegadores web (Firefox 2.0, IE 7.0, Opera, por exemplo) ou clientes de correio eletrônico (Mozilla Thunderbird e Microsoft Mail), fornecendo informações como o nome "real" do domínio do site web ou se o SSL (*Secure Sockets Layer*) está ativo. Apesar de auxiliar os usuários, existem críticas as atuais soluções anti-*phishing* [Wu et al., 2006] como, por exemplo, a localização das barras de tarefas e das informações exibidas não ajuda os usuários, e a falta de sugestões sobre o que fazer quando uma tentativa de *phishing* é detectada, uma vez que hoje os indicadores apenas mostram o que está errado.

As soluções anti-*spam* também se enquadram nesta categoria. Basicamente, a detecção de *spam* é baseada na filtragem de mensagens não solicitadas através dos campos do cabeçalho ou do conteúdo da mensagem. A filtragem de cabeçalho verifica o endereço de origem, nome do remetente e assunto de uma mensagem para validá-la ou não. Este tipo de solução anti-*spam* é mais simples e sujeita a erros de configuração, uma vez que é preciso definir regras do que se quer ou não receber, ou seja, quais endereços, remetentes e assuntos são indesejados. As listas negras (*blacklist*) são exemplos de filtragem de cabeçalho. Já a filtragem baseada no conteúdo da mensagem é a mais utilizada. Normalmente esta técnica realiza buscas por palavras chaves tais como "viagra" no conteúdo das mensagens. Quando configurados corretamente, a filtragem baseada em conteúdo é bem eficiente, mas também podem cometer erros (barrar "especialista" porque contém "cialis", por exemplo). Além disso, a inspeção de conteúdo não verifica a origem da mensagem. Atualmente, o uso de filtros com mecanismo de auto-aprendizagem tem sido muito utilizado.

Para finalizar, atualmente existe uma tendência em agregar softwares pessoais como antivírus e anti-*phishing* na forma de pacotes. Por exemplo, a solução gratuita avast! antivírus 4.x Home Edition integra antivírus, anti-*spyware* e anti-*rootkit*[18] para máquinas Windows. O google pack oferece antivírus e anti-*spyware*.

[18] Root Kit são ferramentas que visam obter acesso de administrador em um computador ou sistema.

3.3.1.5. Ferramentas de Medição de Tráfego

O monitoramento de tráfego é uma atividade essencial para o gerenciamento de redes e pode ser realizado através da observação dos pacotes ou fluxos.

3.3.1.5.1. Medição baseada em Pacotes

A análise baseada em pacote consiste da captura e análise do cabeçalho dos pacotes. Exemplos de informações importantes obtidas no cabeçalho do pacote são o endereço IP de origem (srcIP), o endereço IP de destino (dstIP), a porta de origem (srcPrt), a porta de destino (dstPrt) e o número do protocolo.

Existem várias ferramentas de captura de pacotes (*sniffers*) disponíveis na Internet. Por exemplo, o TCPdump[19] é uma famosa ferramenta que permite inspecionar os pacotes da rede e fazer uma análise estatística de arquivos coletados (*traces*). Junto com o Ethereal (Wireshark)[20], que adiciona uma interface amigável para o TCPdump, essas ferramentas fornecem meios para identificação da aplicação baseada no conteúdo do pacote (*payload*).

Além da inspeção do cabeçalho do pacote, outra técnica é a inspeção profunda dos pacotes (DPI) que proporciona a identificação da aplicação permitindo analisar o conteúdo dos pacotes ao longo de uma série de operações. Como resultado, a análise DPI pode ser usada para encontrar protocolos não conformes, vírus, *spam*, invasões e assim por diante. A Figura 3.4 ilustra a inspeção de pacotes e a análise DPI.

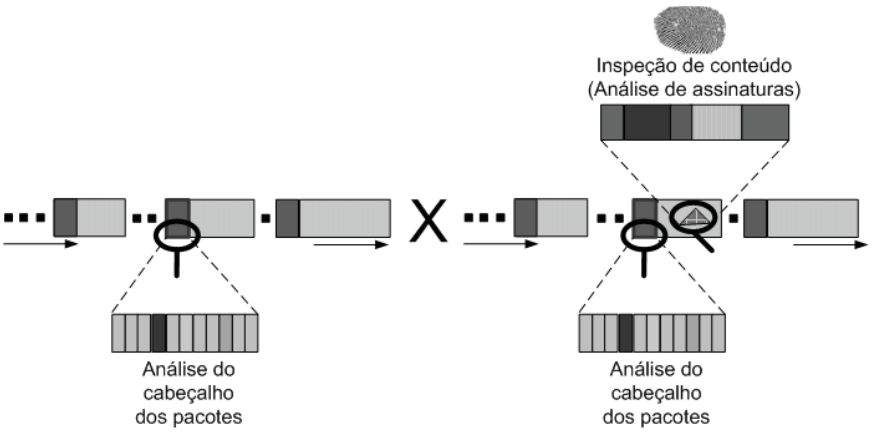

Figura 3.4. Inspeção de pacotes x análise DPI

Apesar de DPI proporcionar uma melhor solução do que filtragens, muitos pesquisadores argumentam que esta técnica fere a neutralidade da Internet, uma vez que pode ser usado para monopolizar o tráfego. Por exemplo, a China desenvolveu o "Grande Firewall da China", uma ferramenta DPI, para acompanhar todas as entradas e saídas de tráfego [OpenNet, 2004]. Este mecanismo é muito sofisticado e eficaz. Prova disso é sua capacidade de bloquear o tráfego Skype e o acesso a diversos sites incluindo o YouTube.

[19] http://www.tcpdump.org/
[20] http://www.wireshark.org/

3.3.1.5.2. Medição baseada em Fluxo

As ferramentas baseadas em fluxo tentam reduzir o volume do tráfego analisado da rede através da agregação de pacotes em fluxos de informação. Nesse caso, regras de agregação são necessárias para combinar pacotes em fluxos. Comumente são utilizados cinco campos do cabeçalho dos pacotes no processo de formação de um fluxo: o endereço IP de origem (srcIP), o endereço IP de destino (dstIP), a porta de origem (srcPrt), a porta de destino (dstPrt) e o protocolo da camada de transporte. A Figura 3.5 mostra um exemplo do processo de agregação de pacotes em fluxos baseado no srcIP e dstPrt. Normalmente, este processo é bastante empregado em ferramentas construídas para modelar o tráfego de rede. Exemplos de ferramentas que lidar com fluxos são Cisco Netflow[21] [Cisco, 2006] (o padrão de facto) e Juniper JFlow[22] [Juniper, 2008]

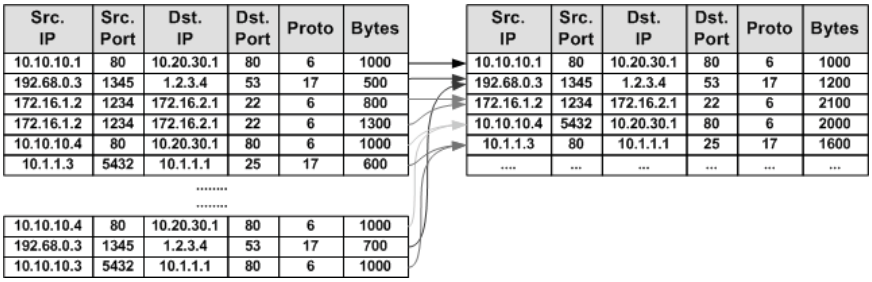

Figura 3.5. Exemplo do processo de agregação de fluxos.

3.3.1.6. Gerenciamento do Controle de Acesso

Atualmente, os produtos de segurança (antivírus, firewalls, IDS, etc.) são projetados como peças independentes de equipamentos ou software. A idéia básica do gerenciamento do controle de acesso é coordenar os mecanismos de segurança sob um computador ou rede. Neste cenário, a rede ou o computador são vistos como um único sistema ao invés de sistemas separados.

A idéia é reforçar a política de segurança regulamentando o acesso a rede pelos computadores. Por exemplo, um computador deve fornecer informações sobre o seu estado de segurança como o atual estado do antivírus, o nível de atualização de seu sistema operacional, etc. Através dessas informações, o sistema de controle de acesso poderá decidir se o computador está seguro e em conformidade com as políticas de segurança antes que seja permitido o acesso aos recursos da rede. Este cuidado no controle de acesso de computadores confiáveis ajuda a garantir que a "saúde" da rede inteira seja preservada.

O controle de acesso de rede pode ser visto como uma automação do processo manual usado pelos administradores do sistema para identificar e isolar as vulnerabilidades dentro de suas redes. O administrador pode manualmente inspecionar a configuração do computador para verificar se um usuário está em conformidade com as políticas de segurança. Os computadores também podem ser testados usando ferramentas (*scanners*) que identificam possíveis fraquezas. Contudo, a maior

[21] http://www.cisco.com/warp/public/732/netflow/
[22] http://www.juniper.net/techpubs/software/erx/junose80/swconfig-ip-services/html/ip-jflow-stats-config2.html

dificuldade encontrada é manter atualizados os diversos computadores da rede e os softwares usados pelos usuários. Por esta razão, a automação deste processo é altamente desejável, especialmente para grandes organizações.

Neste cenário, recentemente têm sido propostas várias abordagens para fornecer controle de acesso. Algumas são descritas a seguir.

Safe Access

O Safe Access (SA) [StillSecure, 2008] é uma solução de controle de acesso que visa proteger a rede através de testes sistemáticos de conformidade dos computadores com as políticas de segurança definidas pela organização. Os computadores que não estão em conformidade são automaticamente isolados da rede. A utilização do SA inicia com a definição das aplicações e serviços que são permitidos aos usuários, bem como as ações que deverão ser tomadas para computadores que não estão conformes.

As políticas de acesso atualmente consistem de testes individuais para avaliar o estado de segurança de cada computador. A ferramenta SA interroga os computadores tentando obter acesso a rede. Diferente das outras abordagens, não é necessário instalar um agente nos computadores. O acesso seguro é verificado através da utilização de programas como antivírus e firewalls. Além disso, SA inclui uma API (*application-programming interface*) que permite a interação com outros programas, por exemplo, um novo antivírus.

Testes específicos avaliam o sistema operacional verificando se as atualizações de *hotfixes* e *patches* foram realizadas, se aplicações de segurança como antivírus e anti-*spyware* estão atualizadas e se existem potenciais aplicações que podem colocar em risco a segurança do sistema como aplicações que compartilham arquivos. Além disso, detectam a presença de códigos maliciosos (vírus, *worms*, cavalos de tróia). Baseados nos resultados dos testes os computadores são permitidos ou negado o acesso à rede

Cisco NAC

A Cisco tem alistado companhias de antivírus como a McAfee, Symantec e a Trend Micro para compor seu sistema de controle de admissão[23] de rede denominado NAC (*Network Admission Control*) [Cisco, 2008] cuja finalidade é validar o acesso a rede somente de dispositivos confiáveis e em conformidade com as políticas de segurança, visando proteger a rede de vulnerabilidades introduzidas por dispositivos que não estão em conformidade.

Como mostrado na Figura 3.6, a arquitetura NAC é formada por agentes denominados CTA (*Cisco Trust Agents*), dispositivos de acesso a rede, o servidor de controle de acesso seguro (ACS – *Access Control Server*) e servidores específicos como de antivírus. O agente Cisco é um pequeno software programado para se comunicar com o ACS e que atua como um IPS baseado em *host* e um firewall distribuído que identifica e bloqueia o tráfego com comportamento malicioso. Além disso, o agente tem a função de manter o sistema operacional atualizado. Os dispositivos de acesso a rede (roteadores, switches e pontos de acesso sem fio) intermediam a comunicação entre os

[23] Apesar de empregado na solução Cisco, o termo *controle de admissão* não é inteiramente apropriado. Controle de admissão tem sido usado no contexto de controle de tráfego referindo-se a aceitação ou rejeição do tráfego visando prevenir o congestionamento na rede.

agentes e o ACS. Nessa comunicação, o protocolo IEEE 802.1x é utilizado para gerenciar o acesso às portas físicas e aplicar as decisões tomadas pelo ACS sobre a admissão de um computador. Exemplos de decisões tomadas por um ACS são desconectar o computador da rede ou mudá-lo para outra rede.

Para o processo de admissão, o agente coleta informações do estado de segurança de múltiplos programas instalados nos clientes como antivírus, firewalls e outras aplicações de segurança. Após a coleta de informações de segurança, o agente envia suas credenciais e essas informações para os dispositivos de acesso a rede que as encaminharão para o ACS. Após o servidor de políticas (ACS) tomar uma decisão sobre a admissão de um computador, os dispositivos de acesso a rede aplicam a decisão de controle.

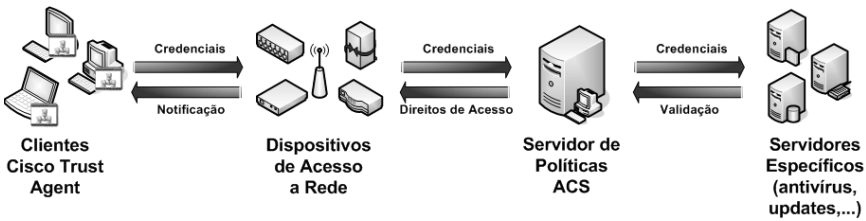

Figura 3.6. Arquitetura NAC.

NAP

A Microsoft tem alistado vários parceiros como companhias de antivírus, vendedores de equipamentos de rede e integradores de sistema para compor uma solução de segurança denominada NAP (*Network Access Protection*) [Microsoft, 2008]. A idéia básica do NAP é detectar o estado de segurança de um computador que está tentando se conectar a rede e restringir o acesso deste até que as políticas de requisitos para conexão da rede sejam atendidas. O NAP realiza este objetivo através de um conjunto de funções:

- **Inspeção:** sempre que um computador tenta conectar-se a rede, o seu estado de segurança é validado em função das políticas de acesso definidas pelo administrador do sistema.
- **Isolamento:** computadores que não estejam em conformidade têm seu acesso negado ou restringido, dependendo das políticas estabelecidas.
- **Remediação:** problemas identificados são resolvidos tornando os computadores em conformidade com as políticas de acesso.

Como mostrado na Figura 3.7, os elementos na arquitetura NAP incluem:

i) Servidores VPN que permitem conexões de acesso remoto baseada em VPN para uma rede privada;

ii) Servidores DHCP (*Dynamic Host Configuration Protocol*) que fornecem configurações de endereço IP para os computadores;

iii) Serviço de diretório usado para armazenar as contas dos usuários e suas credenciais;

iv) Rede de quarentena para computadores que não estão em conformidade e necessitam ser remediados;

v) Servidor NPS (*Network Policy Server*) que contém recursos tais como assinaturas de antivírus e atualizações de software. Estes recursos são usados para manter os computadores em conformidade com as políticas de segurança e fornecer remediação para os computadores não conformes.

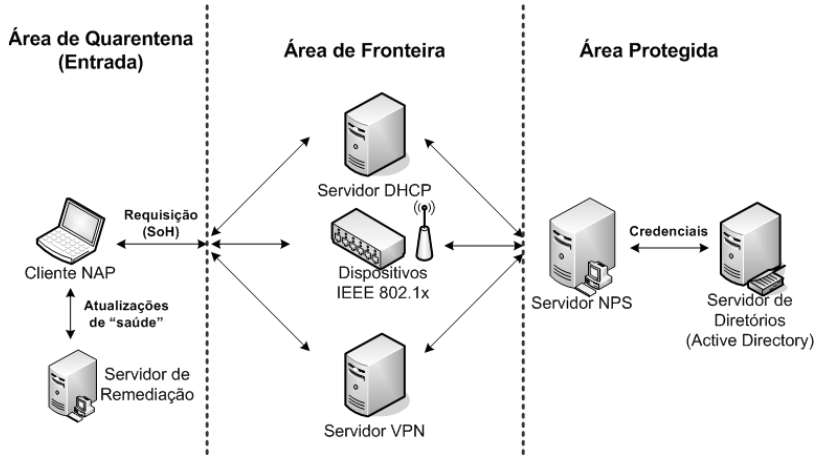

Figura 3.7. Exemplo da arquitetura NAP.

Na rede NAP, um computador age como um cliente DHCP que usa mensagens para requisitar um endereço de IP válido para um servidor DHCP. Neste caso, a requisição deve incluir um indicador do seu estado atual do sistema denominado de SoH (*Statement of Health*). Um computador sem um SoH é automaticamente designado como não conforme. Se o SoH é válido, o servidor DHCP atribui um endereço IP que permite ao computador o acesso a rede. Se o SoH não é válido, o computador é dito não conforme e o servidor DHCP isola-o computador em uma rede de quarentena.

Ao ser designado para a rede de quarentena, o computador reporta ao servidor de políticas requerendo atualizações. O servidor de políticas fornece ao computador as atualizações necessárias (por exemplo, assinaturas de antivírus e atualizações de softwares) para torná-lo em conformidade com as políticas. Após atualizar o seu SoH, o computador pode enviar uma outra requisição para o servidor DHCP que encaminha o SoH atualizado para o servidor NPS, responsável por validar o SoH enviado. Após essa validação, o DHCP permite o acesso normal à rede.

3.3.2. Soluções Baseadas na Análise do Tráfego

3.3.2.1. Técnicas Avançadas para Classificação de Tráfego

Uma das questões desafiadoras resultante das atuais soluções contra o tráfego não desejado é a necessidade prévia de uma análise manual para detectar corretamente o tráfego desconhecido e indesejado. Entretanto, considerando o rápido crescimento do número de novos serviços e aplicações, esse tipo de procedimento é muitas vezes impraticável.

Neste contexto, as abordagens de análise de tráfego têm atraído interesse especial nos últimos anos e apresentado resultados promissores contra este tipo de

tráfego, especialmente para enlaces de alta velocidade. Técnicas para caracterizar o tráfego Internet, métodos para descobrir o tráfego gerado por aplicações, abordagens para desenvolver sistemas de detecção de anomalia mais precisos e soluções específicas para lidar com certos tipos de tráfego não solicitados (por exemplo, *spam* e P2P) surgiram como tentativa de tornar automática, rápida e precisa identificação e redução do tráfego não desejado.

Esta seção apresenta algumas das mais relevantes técnicas e metodologias baseadas em modelos estatísticos, matemáticos e na análise comportamental do tráfego Internet e que podem ser direta ou indiretamente aplicadas na análise do tráfego não desejado.

Modelos Estatísticos e Matemáticos

Modelos estatísticos têm sido empregados para construir modelos de séries temporais do tráfego da Internet e, conseqüentemente, procedimentos capazes de detectar anomalias.

Como argumentado no trabalho proposto por Scherrer et al.[Scherrer et al., 2007], o tráfego de rede consiste de um processo de chegada de pacotes IP que pode ser modelado usando processos pontuais não estacionários ou processos pontuais Markovianos. Entretanto, devido ao tamanho do volume de dados, especialmente em enlaces de alta velocidade, tais modelos geram grandes conjuntos de dados e necessitam de um alto poder computacional para seu processamento. Além disso, muitas das distribuições estatísticas não produzem bons resultados na medição das margens (distribuição marginal). Em linhas gerais, uma distribuição marginal é utilizada quando se tem interesse em informações de uma determinada variável (um dado do tráfego da rede como, por exemplo, endereço IP de origem). Ela sumariza as freqüências obtidas para cada nível. Por exemplo, na Tabela 3.1, as distribuições marginais são observadas nos totais das linhas e colunas (em itálico).

Tabela 3.1. Exemplo de distribuição marginal.

Y\X	**0**	**1**	**2**	**3**	**P(y)**
0	1/8	2/8	1/8	0	*1/2*
1	0	1/8	2/8	1/8	*1/2*
P(x)	*1/8*	*3/8*	*3/8*	*1/8*	*1*

É por este motivo que distribuições estatísticas não Gaussianas, em especial a distribuição Gama, vêm sendo aplicadas na busca de métodos mais rápidos e eficientes.

Por sua vez, os modelos matemáticos, especialmente *wavelets*, têm sido empregados na detecção de anomalias porque capturam correlações temporais complexas através de múltiplas escalas de tempo e encontram variações no comportamento do tráfego de rede. *Wavelets* são funções matemáticas que dividem os dados (sinais) em diferentes componentes de acordo com uma escala de interesse, permitindo realizar análises locais em uma área específica do sinal, como mostra a Figura 3.8.

A seguir são detalhados dois trabalhos baseados nos modelos estatísticos e matemáticos e que podem ser facilmente utilizados para identificação e caracterização de tráfego indesejado.

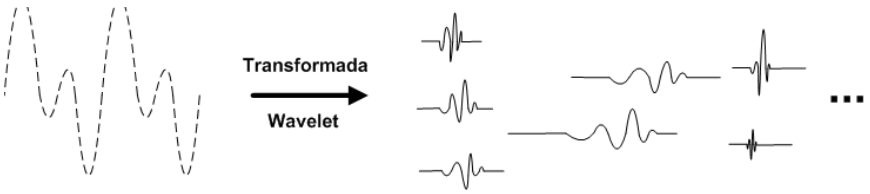

Figura 3.8. Processo de decomposição do sinal usando wavelet.

Extracting Hidden Anomalies using Sketch and Non Gaussian Multiresolution Statistical Detection Procedures

O trabalho de Dewaele et al. [Dewaele et al., 2007] introduz um procedimento especialmente elaborado para detecção de anomalias de baixa intensidade ocultas no tráfego Internet. Essa técnica combina o uso de *Sketches* e um modelo estatístico não gaussiano (que não segue a distribuição normal) para descobrir anomalias no tráfego de dados. *Sketches* são estruturas de dados (geralmente tabelas *hash*) utilizadas para representar (sumarizar) dados massivos de tráfego em vetores bidimensionais que requerem baixo poder de processamento. Isto possibilita a redução do volume de informação e a medição do comportamento do tráfego. A distribuição Gama serve para extrair a função de distribuição marginal do tráfego para cada *sketch*. Desta forma, é possível capturar pequenas correlações entre as estruturas do tráfego. O processo de detecção faz uso da distância de Mahalanobis [Mahalanobis, 1930], uma medida estatística usada para determinar similaridades entre um conjunto de amostras desconhecidas e outro de amostras conhecidas, para executar comparações entre *sketches* e assim determinar comportamentos anômalos.

O procedimento de detecção e análise, exemplificado na Figura 3.9, consiste dos seguintes passos: geração dos sketches, agregação multiresolução, modelagem não gaussiana, referência, distâncias estatísticas e detecção de anomalias.

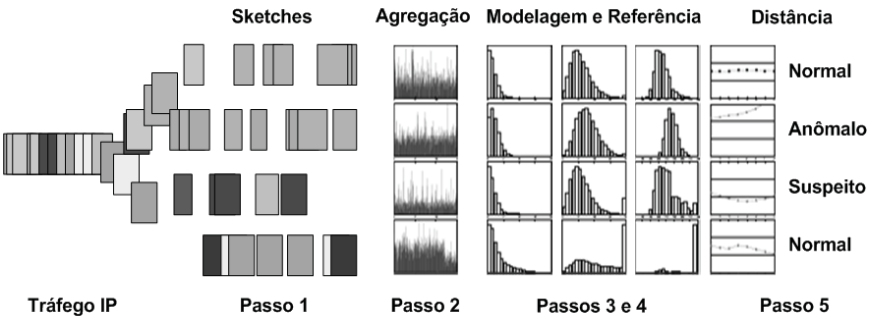

Figura 3.9. Processo de detecção de análise.

1. Geração dos sketches

Os *sketches* são usados para dividir os pacotes dentro de subgrupos de acordo com uma janela de tempo deslizante. Para cada fatia de tempo, somente o tempo de chegada, os endereços IP e as portas são analisados. Como resultado, tabelas *hash* são geradas representando segmentos do tráfego original, onde o endereço IP de origem ou destino é usado como chave da tabela *hash*.

2. Agregação multiresolução

Na agregação multiresolução, os segmentos de tráfego gerado são colocados juntos para formar séries temporais que são agregadas de acordo com uma determinada escala.

3. Modelagem não gaussiana

A distribuição Gama é usada para descrever as distribuições marginais das séries temporais agregadas. Em outras palavras, para cada agregação, os parâmetros α e β são estimados para serem usados para calcular a referência e a distância estatística. A escolha pela distribuição Gama e sua adequação são explicadas em [Abry et al., 2007] [Scherrer et al., 2006] [Scherrer et al., 2007].

4. Referência

A média dos comportamentos e a variabilidade típica são estimadas para cada elemento da tabela *hash* usando estimadores de média e variância. Apesar da simplicidade, a junção dos *sketches* e referências permite a definição dos padrões de comportamento normal e anômalo. Anomalias podem ser encontradas observando mudanças no padrão estatístico através da comparação de *sketches* em um mesmo intervalo de tempo.

5. Distância estatística

A distância de Mahalanobis é usada para medir o comportamento anômalo das referências. Cada distância calculada é comparada com um limiar (*threshold*). Se a distância de referência é menor ou igual ao limiar, o segmento é considerado normal. Caso contrário, é classificado como anômalo.

6. Detecção de anomalias

A detecção de anomalias é realizada comparando *sketches* (chaves da tabela *hash*) com atributos (endereço IP de origem e destino e número das portas) registrados em uma lista durante o processo de detecção.

A validação deste procedimento foi realizada usando o repositório de tráfego do MAWI[24]. Foram investigadas duas anomalias de tráfego: ataques de inundação de baixa intensidade e varreduras curtas. Os resultados demonstram que o procedimento é capaz de descobrir anomalias como pacotes elefante, *flash crowds*, ataques DDoS (inundação TCP SYN e ICMP), varreduras de IP e porta, tráfego P2P, *worms*, entre outros.

Apesar de esse procedimento ser considerado um trabalho em progresso, os resultados iniciais são bastante promissores. Primeiro, nenhum tipo de conhecimento prévio do tráfego e de suas características é necessário. Segundo, é capaz de detectar tanto anomalias com curto tempo de vida (curta duração) quanto as mais demoradas. Terceiro, requer baixo poder computacional e pode ser implementado em tempo real. Por fim, a janela de detecção pode trabalhar tanto em tempo menores (inferior a um minuto) quanto em tempos maiores (superior a dez minutos).

[24] MAWI é um repositório de tráfego, integrante do projeto WIDE, que tem armazenado coleções de pacotes desde 2001 nos enlaces trans-pacíficos entre o Japão e os Estados Unidos. Maiores informações em http://mawi.wide.ad.jp/mawi/

Como limitações, a técnica pode apresentar problemas de identificação (falso positivo). Por exemplo, o serviço DNS pode ser considerado ilegítimo por apresentar um único padrão de tráfego. Uma sugestão para resolver o problema é a adição de filtros para excluir padrões conhecidos durante a fase de análise.

Anomaly Detection of Network Traffic based on Wavelet Packet

Gao et al. [Gao et al, 2006] descrevem um novo método de detecção de anomalias de rede baseado em transformadas *wavelet*. Os autores argumentam que existem algumas questões que precisam ser observadas para aplicação de *wavelet* em métodos de detecção de anomalias. Primeiro, a maioria dos métodos usa análise de multiresolução, que é somente adequada para anomalias de baixa freqüência. Segundo, os resultados podem ser incorretos quando somente uma escala é analisada. Terceiro, as transformadas *wavelet* demandam um alto poder computacional e, conseqüentemente, podem ser consideradas inapropriadas para operações em tempo real.

Para superar estes problemas, os autores propõem o uso da análise de pacotes de *wavelet*, a qual possui a capacidade de decompor o sinal oferecido uma diversa faixa de possibilidades para análise. Em linhas gerais, em análises *wavelet*, um sinal é dividido em aproximação e detalhe. Esta aproximação é então dividida em uma aproximação de segundo nível e detalhe, e o processo se repete. Para uma decomposição de nível *n*, há *n +1* caminhos possíveis para decompor ou codificar o sinal. A Figura 3.10 ilustra esse processo.

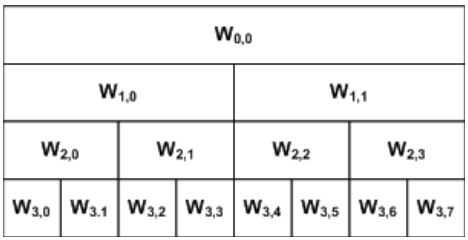

Figura 3.10. Processo de decomposição do sinal.

O método de detecção proposto é capaz de ajustar o processo de decomposição adaptativamente e exibir a mesma capacidade de detecção para anomalias de baixa, média e alta freqüência. Para alcançar este objetivo, o método proposto emprega um estágio de detecção inicial para verificar em cada escala através de um algoritmo de detecção estatístico se existem anomalias indicadas pela análise de pacotes de *wavelet*. No caso onde uma anomalia é percebida, uma nova decomposição de pacotes de *wavelet* é feita e este passo é executado novamente. Os níveis de decomposição são auto-adaptativos. Caso exista uma anomalia, a reconstrução dos pacotes de *wavelet* e a confirmação do estágio de anomalia são usadas para reconstruir os sinais das escalas e checar se a reconstrução do sinal é anômala.

Metodologias baseadas na Análise do Comportamento

Metodologias baseadas no comportamento tentam automaticamente identificar perfis de tráfego em redes de *backbone*. O objetivo por trás dessas abordagens é fornecer um entendimento plausível sobre o padrão de comunicação dos computadores e serviços.

Na detecção de anomalias, a análise do comportamento é caracterizada através do processamento de grandes volumes de tráfego.

A seguir são apresentados recentes trabalhos baseados no padrão de comportamento dos computadores. Tais trabalhos foram projetados para identificar e classificar tráfegos não desejados em redes de *backbones*.

Traffic Classification on the Fly

A técnica proposta em [Bernaille et al., 2006] executa a análise do fluxo através da observação dos cinco primeiros pacotes de uma conexão TCP para identificar a aplicação. Ao contrário de outras técnicas, onde a classificação das aplicações ocorre somente após o final do fluxo TCP, esta técnica utiliza apenas o tamanho dos primeiros pacotes de uma conexão para classificar o tráfego. A idéia da classificação "on the Fly" é identificar com precisão a aplicação associada a um fluxo TCP o mais cedo possível. Essa técnica utiliza o conceito de clusters não supervisionados para descobrir grupos de fluxos que compartilham um comportamento de comunicação comum.

De modo geral, a classificação “on the Fly” é dividida em duas fases: aprendizagem off-line e classificação de tráfego on-line. A primeira fase é utilizada para aprender e detectar comportamentos comuns através de um conjunto de dados de treinamento. No fim desta fase, os fluxos TCP são agrupados de acordo com seus comportamentos. A fase classificação emprega estes fluxos agregados para descobrir qual a aplicação está associada a cada fluxo TCP.

Em resumo, a classificação "on the Fly" obtém mais informação do que a inspeção DPI sem desrespeitar a privacidade do pacote e ferir qualquer restrição legal. Os resultados apontam uma precisão acima de 80% para aplicações TCP bem conhecidas. Entretanto, a técnica também apresenta alguns problemas como pacotes entregues fora de ordem mudarão a representação espacial do fluxo que irá impactar na qualidade da classificação. Além disso, aplicações com comportamento inicial semelhante poderão ser classificadas com o mesmo rótulo.

BLINC: Multilevel Traffic Classification in the Dark

Karagiannis et al. [Karagiannis et al., 2005] apresentaram uma nova metodologia para classificar fluxos de tráfego de acordo com o tipo de aplicação. Diferente de outras propostas que analisam cada fluxo separadamente, o método proposto observa todos os fluxos gerados por computadores específicos. Além disso, o BLINC não examina o conteúdo dos pacotes, não assume que portas bem conhecida representam de maneira confiável as aplicações e somente usa informações obtidas pelos coletores de fluxo, o que explica o termo “in the dark”.

A classificação do BLINC é baseada no padrão de comportamento dos computadores na camada de transporte. Esses padrões são analisados em três níveis de detalhamento: social, funcional e aplicação. O nível social revela a popularidade do computador. Neste nível é investigado o comportamento do computador em relação as suas comunicações com outros computadores. Assim, somente os endereços IP de origem e destino são utilizados como mostrado na Figura 3.11.

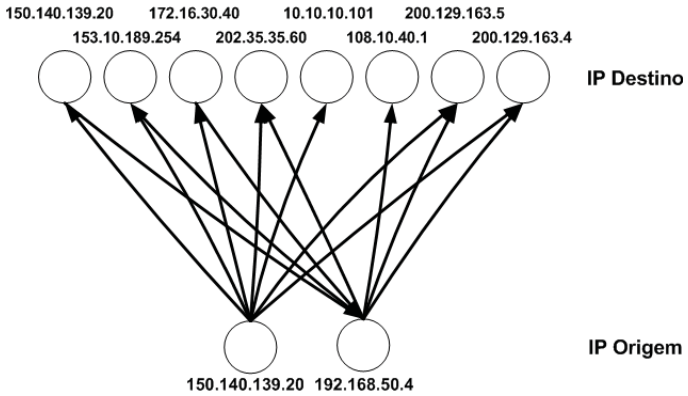

Figura 3.11. Representação visual do nível social através de grafo bipartido.

O nível funcional investiga o que o computador faz. Neste nível é analisado se atua como provedor ou consumidor de um serviço ou se participa de comunicações colaborativas. São avaliados os endereços IP de origem e destino e a porta de origem. Por último, o nível de aplicação captura a interação dos computadores na camada de transporte para identificar aplicações e suas origens. De acordo com Karagiannis [Karagiannis et al., 2005], um *graphlet* pode ser usado para representar as características dos fluxos correspondentes a diferentes aplicações pela captura do relacionamento entre o uso das portas de origem e destino. A Figura 3.12 mostra *graphlets* usados para identificar os fluxos de ataques DDoS a partir de respostas das vítimas.

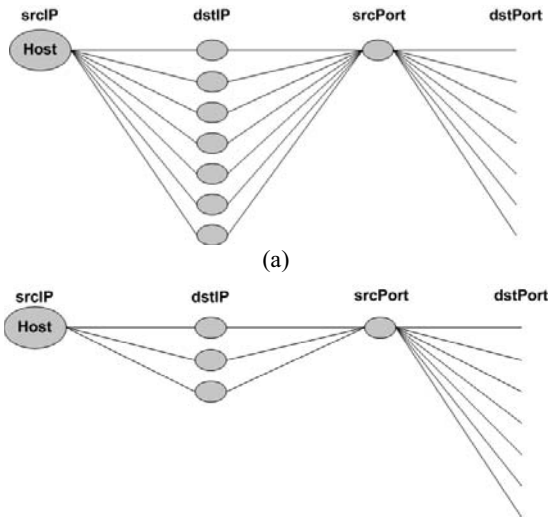

Figura 3.12. Representação visual das interações no nível social através de graphlets.

Profiling Internet backbone Traffic: Behavior Models and Applications

A metodologia proposta por Xu et al. [Xu et al., 2005a] objetiva identificar anomalias de tráfego. O método usa mineração de dados e técnicas de informação teórica para automaticamente descobrir padrões de comportamento significantes no tráfego de dados. A metodologia (aqui denominada de *profiling*) automaticamente descobre comportamentos do tráfego massivo e fornece meios plausíveis para entender e rapidamente reconhecer tráfego anômalo. A metodologia trabalha examinando os padrões de comunicação dos computadores (endereços e portas) que são responsáveis por um significativo número de fluxos em um determinado período de tempo.

O processo do *profiling* basicamente inclui a extração de clusters significativos e a classificação do comportamento deles baseado no relacionamento entre os clusters. Por exemplo, para um dado endereço IP (srcIP) *i*, o processo do *profiling* inclui a extração dos fluxos com srcIP *i* dentro de um cluster (denominado de cluster srcIP) e a caracterização do padrões de comunicação (ou seja, comportamento) usando medições de teoria da informação (entropia) sobre as três dimensões de fluxos restantes, ou seja, endereço IP de destino (dstIP), porta de origem (srcPrt) e porta de destino (dstPrt). A Figura 3.13 ilustra os passos da metodologia.

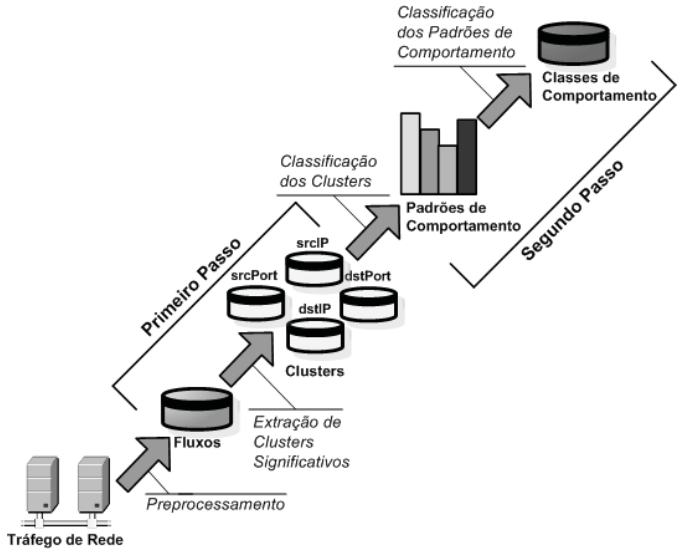

Figura 3.13. Etapas do método Profiling

O primeiro passo da metodologia é analisar um conjunto de fluxos baseado nas tuplas bem conhecidas para decidir sobre um espaço característico de interesse. O objetivo é extrair os clusters significativos de dimensões específicas, isto é, endereços IP de origem e destino e portas de origem e destino. Então, os clusters mais significativos são extraídos de uma dimensão fixa (por exemplo, endereço IP de origem) e o conceito de entropia é usado para medir a quantidade de incerteza relativa (do inglês *Relative Uncertainly* – RU) contida nos dados.

Com os clusters extraídos, o segundo passo é descobrir ligações entre os clusters, ou seja, encontrar modelos de comportamento comuns para o perfil do tráfego. Para alcançar este objetivo, a metodologia propõe uma classificação do comportamento baseado nos padrões de comunicação dos computadores de usuários finais e serviços. Desta forma, para cada cluster, uma RU é computada e usada como métrica para criar classes de comportamento (do inglês *Behavior Classes* – BC). Entre essas classes é possível identificar qual delas representa tráfego anômalo ou indesejado.

Em resumo, esta metodologia é uma ferramenta muito poderosa para detectar anomalias, *exploits* de segurança não conhecidas, criar o perfil do tráfego não desejado, registrar o crescimento de novos serviços e aplicações. Também é flexível e capaz de automaticamente descobrir outros padrões de comportamento significantes. O trabalho em [Xu et al., 2005b] demonstra a habilidade para detectar as mais variadas anomalias massivas tais como varreduras de IP e portas, ataques DoS e DDoS, entre outros. O aspecto negativo é que esta metodologia não é apropriada para o tráfego amplo de rede e sim para enlaces únicos. Além disso, os resultados apresentados não são encorajadores para ataques de baixa intensidade.

Mining Anomalies Using Traffic Feature Distributions

Lahkina et al. [Lakhina et al., 2005] propõem uma metodologia baseada na distribuição características dos pacotes capaz de detectar anomalias de alto e baixo volume. Os autores argumentam que a análise dos fluxos de origem e destino (OD) pode revelar um conjunto geral e diversificado de anomalias, especialmente as maliciosas. O método proposto utiliza o conceito de entropia para fazer observações e extrair informações úteis em relação a dispersões na distribuição do tráfego. A metodologia *mining* está organizada em duas etapas: a distribuição do tráfego e a metodologia de diagnóstico.

Na primeira etapa são extraídos os campos dos cabeçalhos dos pacotes para procurar por eventuais anomalias causadas por mudanças (dispersões) na distribuição endereços ou portas observadas. Por exemplo, durante uma varredura de portas, a distribuição das portas de destino será bem mais dispersa do que durante uma condição normal de tráfego. São analisados quatro campos do cabeçalho IP dos pacotes: os endereços IP origem e destino e as portas de origem e destino (srcIP, dstIP, srcPrt, e dstPrt). Os autores afirmam que é possível capturar o grau de dispersão ou concentração de uma distribuição usando entropia, uma vez que anomalias como, por exemplo, varreduras de portas podem ser vistas claramente em termos de entropia, em comparação com o volume de tráfego.

A segunda etapa, denominada de diagnóstico, usa os resultados da aplicação da entropia sobre a distribuição para fazer um diagnóstico e classificar anomalias. Em seguida, um método chamado *Multiway Subspace* é usado para detectar anomalias, oferecendo uma estratégia de classificação não supervisionada. O método *Multiway Subspace* é derivado do método de subespaço proposto em [Dunia e Qin, 1998], e cujos resultados na análise de tráfego podem ser encontrados em [Lakhina et al., 2004]. A idéia por trás deste método é identificar as variações correlacionadas com múltiplas características de tráfego (no caso os campos do cabeçalho IP), o que provavelmente pode indicar uma anomalia. A classificação não supervisionada utiliza uma abordagem para a construção de grupos (clusters), onde os dados são analisados para detectar anomalias. A operação ocorre em duas fases. Primeiro, as anomalias bem conhecidas são agregadas de forma a adquirir conhecimento sobre suas origens. Assim, os grupos

são rotulados com base em seus tipos. Em seguida, a classificação é feita pela agregação das anomalias desconhecidas.

Como resultado, a metodologia *mining* é capaz de detectar ataques DoS e DDoS, *flash crowds*, varreduras de endereços e porta, *worms*, interrupções e anomalias desconhecidas. Além disso, o método *Multiway Subspace* proposto também demonstra adequação a manipular enormes volumes de fluxo OD e, conseqüentemente, para descobrir anomalias.

Em resumo, o uso da entropia na detecção de variações do tráfego de rede causados pelas mais diversas anomalias é a principal vantagem desta metodologia. A complexidade e o alto poder computacional para implementar a metodologia são a principal desvantagem. Além disso, o tempo de construir séries temporais (fluxos OD) é grande, podendo ser da ordem de alguns minutos.

3.3.3. Considerações

Apesar dos avanços na detecção do tráfego não desejado, especialmente sobre *backbones* de alta velocidade, estas abordagens ou apresentam um caro custo computacional (complexidade) e mudanças na infra-estrutura ou resultados imprecisos.

No entanto, é possível perceber tendências para futuras soluções. Em primeiro lugar, análise gerada através da agregação do tráfego em fluxos permite a exibição dos pontos de penetração (origem) e saída (destino) do tráfego de forma simples, sem a perda de dados. Em segundo lugar, apesar de não apresentar uma precisão superior, técnicas de detecção baseadas no comportamento obtém mais tipos de tráfego indesejado e extraem mais dados correlacionados sobre o tráfego. Por último, esta seção capta a essência do debate e análise sobre técnicas de detecção de tráfego não desejado, mostrando que é possível ver uma "luz no fim do túnel" no que diz respeito a soluções automáticas, rápidas e precisas.

3.4. Potenciais Soluções

O tema tráfego não desejado vem ganhando destaque no mundo todo. A principal prova deste fato é o número de pesquisas e trabalhos publicados nos últimos anos, alguns deles apresentados neste minicurso. No entanto, a própria evolução tecnológica e a filosofia aberta da Internet tem imposto novos desafios a atividade de “controle” do tráfego não desejado.

Nesta seção serão apresentadas algumas abordagens e soluções consideradas pelos autores deste minicurso como potenciais e futuras.

3.4.1. Filtragem avançada

Como visto na seção 3.3.1.1, o uso de filtros de tráfego não é suficiente para lidar com o atual estágio do tráfego indesejado. A fim de resolver este problema, pesquisadores têm proposto mecanismos e sistemas avançados de filtragem. Atualmente, as melhores práticas nesta área estão descritas em dois documentos: BCP 38 (RFC 2827) [Ferguson e Senie, 2000] e BCP 84 (RFC 3704) [Baker e Savola, 2004].

Basicamente, o BCP 38 (*Best Current Pratice*) recomenda que os provedores de Internet filtrem pacotes IP, na entrada de suas redes, provenientes de seus clientes e que descartem aqueles pacotes cujo endereço IP não tenha sido alocado a esses clientes. Na

verdade, muitos provedores de acesso possuem roteadores que suportam as medidas citadas no BCP 38.

O BCP 84 apresenta outras implementações de filtragem na entrada da rede como, por exemplo, *Strict Reverse Path Forwarding* (SRPF), *Feasible Path Reverse Path Forwarding* (Feasible FPR), *Loose Reverse Path Forwarding* (Loose FPR) e *Loose Reverse Path Forwarding Ignoring Default Routes*, que oferecem configuração automática e dinâmica de filtros de entrada ao núcleo e borda das redes.

Pesquisadores e especialistas defendem que a implantação global do BCP 38 e BCP 84 permitirá efetivamente bloquear ataques DDoS baseados em endereços IP de origem forjados. Contudo, a implantação deste tipo de proteção tem sido bastante questionada, especialmente por provedores de acesso Internet. Primeiro, o custo financeiro aumentaria devido à necessidade de pessoal técnico especializado. Segundo, para essa solução ser efetiva, necessita da implantação em todas as redes existentes. Além disso, qualquer eventual bloqueio de tráfego legítimo devido a erros acidentais na configuração desses filtros seria problemático e esse temor é usado com motivo (desculpa) para o BCP 38 e o BPC 84 não serem divulgados hoje em dia. Para finalizar, devido à alta versatilidade de alguns tipos de tráfego não desejado especificamente aqueles que trafegam sobre o protocolo HTTP, essas soluções podem apresentar uma baixa taxa de efetividade na luta contra o tráfego indesejado.

3.4.2. Investigação do Espaço de Endereços IP

Alguns tipos de tráfego indesejado usam espaços de endereço IP não atribuídos (não utilizados) para executar atividades como varredura de vulnerabilidades, ataques DoS e DDoS, divulgação de vírus e *worms*, entre outras. Neste contexto, algumas soluções têm sido apresentadas com o objetivo de monitorar o tráfego considerado não esperado.

Internet Motion Sensor (IMS)[25] [Cooke et al., 2004] [Bailey et al., 2005] é um sistema de vigilância distribuída, de nível mundial, cujo objetivo é identificar e monitorar o tráfego proveniente de espaços de endereçamento IP roteáveis não atribuídos e não anunciados comumente chamados de *darknets*. De acordo com [Anderson et al., 2006], atualmente a IMS controla cerca de 17 milhões de prefixos (/8, /16 e /24), cerca de 1.2% do espaço do endereço IPv4 espalhados ao redor do mundo em provedores de acesso Internet, organizações, empresas e universidades. A Figura 3.14 ilustra a arquitetura da rede IMS. Apesar de não ser considerada uma solução contra o tráfego não desejado, IMS é uma ferramenta eficaz para auxílio que pode ser emprega para combater este tipo de tráfego, seja através da identificação das origens quanto da compreensão dos mecanismos utilizados.

Outra solução semelhante são as Redes Telescópio (*Network Telescopes*) [Moore, 2002] [Moore et al., 2004]. Estas redes são compostas por monitores que observam o tráfego encaminhado para espaços de endereços IP não utilizados e registram eventos como a propagação de vírus na Internet. A idéia é manter os sensores ativos para “escutar” todo o tráfego enviado para estes espaços de endereçamento. Desta forma, é possível ver exatamente todos os eventos em seu "estado bruto", ou seja, sem quaisquer interferências de tráfego.

---

[25] http://ims.eecs.umich.edu.

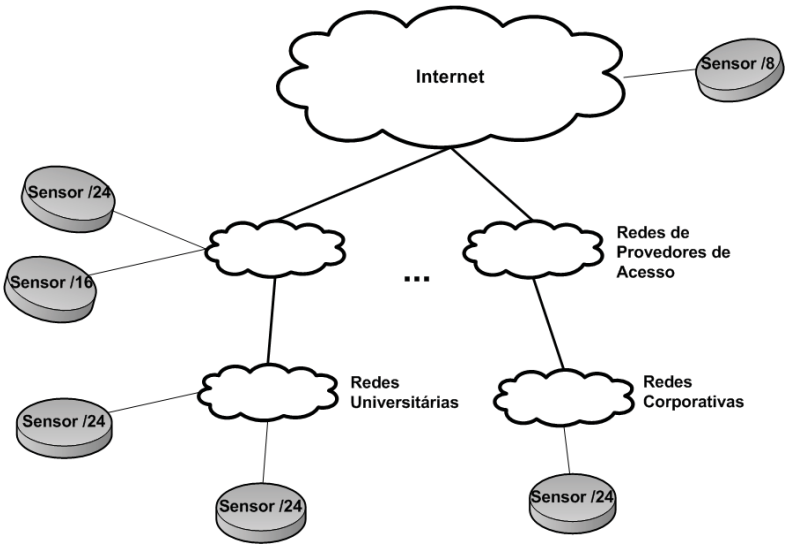

Figura 3.14. Arquitetura da rede IMS.

3.4.3. Lightweight Internet Permit System

Uma questão chave em relação ao tráfego não desejado é falsificação do endereço IP. Soluções como IPSec, SSL, VPN e esquemas de filtragem mais rígidos não apresentam qualquer tratamento para lidar com a falsificação de endereços IP. Soluções mais robustas têm sido propostas, mas dependem de mudanças, algumas profundas, na infra-estrutura da Internet. Como se sabe que tais mudanças não são desejáveis e dificilmente não ocorrerão rapidamente.

É neste contexto o método LIPS (*Lightweight Internet Permit System*) [Choi et al., 2005] se propõe a identificar e bloquear o tráfego não desejado. LIPS fornece um mecanismo que permite a responsabilização do tráfego através da rápida autenticação de pacotes (*fast packet authentication*), ou seja, pacotes de tráfego não desejados são bloqueados e suas origens identificadas.

LIPS funciona como um mecanismo eficiente de filtragem de tráfego. Quando um computador origem quer se comunicar com um computador destino, primeiro deve requisitar uma autorização de acesso ao destino. Se a autorização for concedida, um passaporte é enviado ao computador origem que o utiliza para enviar dados ao computador destino. Apenas pacotes com autorizações de acesso serão aceitos até ao destino. Esta arquitetura simples é escalável e fornece uma possibilidade de estabelecer responsabilização ao tráfego entre redes e computadores, além de garantir o melhor aprovisionamento dos recursos Internet sem sacrificar a sua natureza dinâmica e aberta. Além disso, LIPS também simplifica e facilita à detecção precoce de ataques e intrusões a rede, uma vez que exige autorizações de acesso válidas antes de permitir quaisquer pacotes possam ser aceitos e processados.

LIPS opera de dois modos. A base do LIPS é modo *Host*, onde um computador se comunica diretamente com outro. Esse modo é aplicável para comunicações em

pequena escala, mas também é usado em grande escala (inúmeros computadores) na presença de um *gateway* LIPS. O outro modo é o de *Gateway*, onde os computadores LIPS são organizados em zonas de segurança com base em domínios administrativos. O uso de zonas de autorização permite autenticar o acesso de pacotes entre zonas. Cada zona tem um servidor de licenças (*Permit Server*) para gerenciar as licenças entre zonas e um *gateway* de segurança (*Security Gateway*) para validar os pacotes entre zonas baseado nos pacotes permitidos dentro da zona (internos). Uma vez que uma zona de inter-licenciamento é estabelecida entre duas zonas, as comunicações subseqüentes seguirão a autenticação entre elas. Desta forma, haverá uma redução no overhead de licenciamento.

Os autores argumentam que a solução LIPS é escalável e flexível, visto que pode ser implantada de forma incremental (não é necessário que ambos os lados de uma conexão tenham LIPS). Além disso, nenhuma modificação nos programas ou na arquitetura Internet é necessária e não há necessidade de chaves criptográficas pesadas porque não há troca de chaves.

3.4.4. SHRED

Atualmente, as diversas soluções para lidar com o envio e a recepção de mensagens de *spam* são baseadas no uso de esquemas de filtragem, análise de tráfego e comportamento [Gomes et al., 2005], uso de *honeypots* [Andreolini et al., 2005] e até mesmo em mudanças no protocolo SMTP [Duan et al., 2005]. Uma idéia, não muito inovadora, é o uso de métodos que permitam punir os *spammers* através de cobranças financeiras. Nesta linha de soluções, uma proposta interessante é SHRED.

Spam Harassment Reduction via Economic Disincentives (SHRED) [Krishnamurthy e Blackmond, 2003] é um esquema que visa diminuir o volume do tráfego de *spam* através da punição monetária dos *spammers*. A idéia central é utilizar selos ou marcas (*stamp*) para inserir desincentivos econômicos para usuários que geram ou propagam *spam*.

O esquema proposto utiliza dois conceitos econômicos: passivo contingente com tempo de expiração e limite de crédito. O primeiro se refere ao fato de que uma dívida possa ser gerada baseada em uma ação externa dentro de um tempo limite. O segundo conceito refere-se ao limite de crédito pré-definido para cada usuário. No esquema SHRED, os selos representam o endividamento e possuem um valor monetário associado. Os selos são pré-alocados para os provedores de acesso através de um dos muitas autoridades de marcação eletrônica (do inglês *Eletronic Stamp Authorities* – ESA) que participam do serviço SHRED. Os selos são colados automaticamente a cada mensagem enviada pelo provedor de acesso, de forma transparente e em nome do remetente.

A arquitetura SHRED é formada pelo remetente, seu provedor de acesso, uma ou mais ESA, provedor de acesso do destinatário e o destinatário. A Figura 3.15 ilustra a arquitetura.

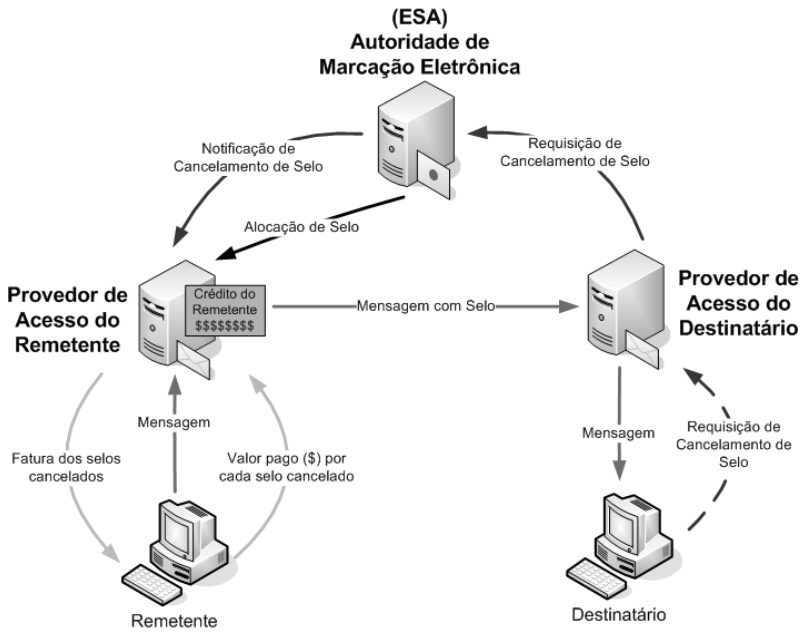

Figura 3.15. Arquitetura SHRED.

O processo de funcionamento é descrito a seguir. Quando o remetente envia um e-mail, seu provedor de acesso adicionará um selo como uma espécie de cabeçalho SMTP e reduzirá o crédito a sua disposição. Os selos têm um tempo de expiração associado a eles e incluem identificações do provedor de acesso de origem e do ESA que emitiu o selo. O provedor de acesso manterá o estado de todos os selos adicionados até que expirem. Quando o provedor de acesso do destinatário recebe o e-mail, verifica se o destinatário recebe mensagens com selo. Em caso negativo, a mensagem é processada e entregue de forma tradicional. Em caso positivo, o provedor de acesso do destinatário valida o selo localmente e o repassa ao destinatário. Podem ser tomadas duas ações:

- Se ao ler a mensagem, o destinatário considerá-la indesejada, o selo pode ser cancelado se ainda estiver dentro do tempo de expiração. O cancelamento do selo pode ser feito através de uma interface de usuário, acessando uma URL anexada à mensagem ou enviando um e-mail para o administrador. Também podem ser utilizados filtros para realizar esta tarefa de maneira automática. Em seguida, o provedor de acesso do destinatário transmite mensagens canceladas para o ESA através de um canal privado e seguro. O provedor de acesso de remetente recebe as mensagens com selo cancelado e cobra do remetente por cada selo.
- Se a mensagem recebida pelo destinatário estiver em conformidade ou o selo não for cancelado antes que o seu tempo de expiração seja atingido, o provedor de acesso do remetente incrementa seu limite de crédito.

Em resumo, SHRED pode ser utilizado para complementar os mais variados tipos de filtros de mensagens de correio eletrônico existentes. Contudo, não existe um resultado sobre a efetividade da solução visto que a mesma não foi colocada em operação.

3.4.5. OADS

Os serviços Internet funcionaram por um longo tempo através de um acordo informal e da boa fé dos usuários. Embora exista a falta de controle centralizado ou dono, a Internet é uma das poucas, se não for a única, infra-estrutura auto governada que funciona razoavelmente bem sobre esse paradigma. Hoje, este modelo de confiança está sobre intenso ataque como resultado da diversidade de comunidades que formam a Internet. Será possível manter esse "estilo de vida"? A qual preço? O que precisa ser feito? A proposta apresentada aqui, pelos autores deste minicurso, é chamada de sistemas de detecção de anomalias orientado a orquestração (*Orchestration oriented Anomaly Detection System* – OADS)

A orquestração não é mais uma metáfora moderna que descreve uma atividade de gerenciamento de segurança bem conhecida. Analogamente a um maestro que mantém o ritmo, conduzindo os diferentes músicos, gerentes de segurança organizam a harmonia e ritmo de vários instrumentos (sistemas) de detecção de anomalia (sistemas de remediação, firewall, antivírus, aplicações de análise de tráfego, etc.) para atingir o efeito desejado, tornando a rede cada vez mais segura.

Esta proposta empresta o conceito introduzido na arquitetura orientada a serviços (do ingles *Service-Oriented Architecture* - SOA) [Erl, 2004], especificamente o de orquestração de serviços, e introduz uma nova metodologia para detecção de anomalias baseada na orquestração dos serviços de segurança.

A idéia por trás da orquestração é o gerenciamento automático da execução de diferentes detectores de anomalia. Em outras palavras, permite a colaboração e harmonização entre as diferentes técnicas e mecanismos contra atividades maliciosas. Colaboração permite que dois ou mais processos trabalhem em conjunto em direção a um objetivo comum sem um líder. Na música, isso ocorre quando músicos trabalham no mesmo álbum ou música. No contexto de segurança, a colaboração é vista como um facilitador de relações entre os diferentes detectores de anomalia. Por exemplo, dois ou mais detectores podem compartilhar a mesma base de tráfego para realizar análises ou o resultado elaborado por um detector pode ser usado como entrada para outro. A harmonia significa que dois ou mais sons diferentes estão próximos. Estendendo o conceito a área de segurança de rede, pode se dizer que a harmonia é permitir que um serviço de qualquer origem, empregando qualquer tecnologia, trabalhe bem como em uma orquestra.

Em comparação com as atuais metodologias, orquestração destaca-se por tornar possíveis interações entre os mais diversificados sistemas de detecção de anomalias. A metodologia OADS harmoniza bases de informação de tráfego, sistemas detectores de anomalia (ADS) e um mecanismo (*engine*) de orquestração. A Figura 3.16 ilustra a organização da metodologia OADS.

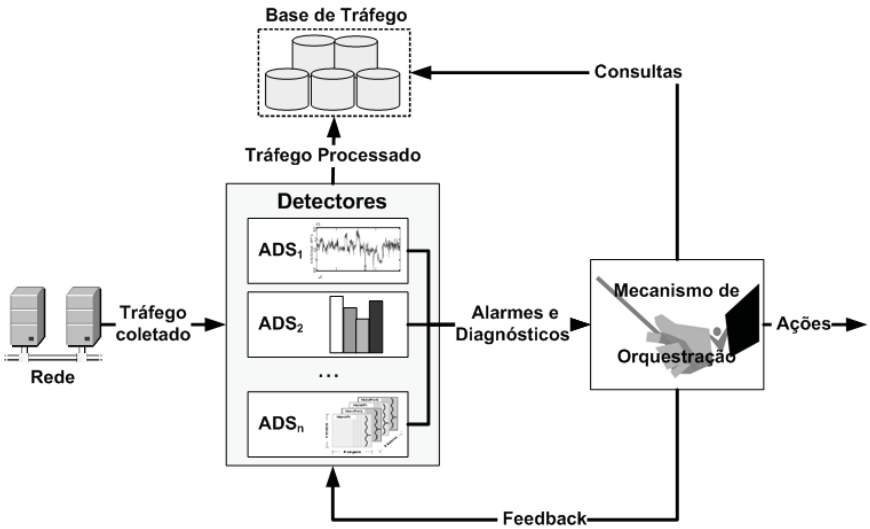

Figura 3.16. Modelo OADS.

- **Bases de tráfego:** armazena o tráfego processado pelos detectores de anomalia. A idéia por trás deste componente é permitir que o mecanismo de orquestração possa obter informações para ajudá-lo a decidir se existe um evento anômalo acontecendo e qual tratamento deve ser dado a este evento. A base de tráfego deve ser maleável para suportar diferentes tipos de tráfego (processado ou não) tais como fluxos OD, pacotes completos, sketches, níveis de agregação, clusters, etc.

- **Sistemas detectores de anomalia (ADS):** atuam como sensores abastecendo o mecanismo de orquestração com informações relevantes (alarmes e diagnósticos), ajudando-o a tomar decisões corretas. Em OADS, os ADS podem e devem colaborar entre si. Por exemplo, a técnica *Profiling* [Xu et al., 2005a] somente reconhece anomalias que geram um grande volume de tráfego. Deste modo, esta técnica pode ser empregada para disponibilizar seus fluxos processados do tráfego para outro detector capaz de encontrar ataques de baixa carga, aumentando assim a probabilidade de descobrir anomalias não percebidas e, conseqüentemente, expandir o nível de segurança da rede.

- **Mecanismo de orquestração**: é responsável por tomar decisões sobre eventos suspeitos ou anômalos baseado principalmente em alarmes disparados pelos diferentes sistemas de detecção de anomalias. Em linhas gerais, a avaliação realizada pelo mecanismo de orquestração leva em consideração as características como a precisão, o tempo de detecção e o grau de sensibilidade de cada elemento. Além dos alarmes, o conhecimento prévio obtido da observação de anomalias anteriores e dos sinais registrados na base de informação do tráfego também pode ser usado para tomar decisões. A implementação do mecanismo de orquestração pode utilizar desde simples esquemas como um algoritmo de votação ou níveis de prioridade até soluções mais elaboradas e

complexas tais como redes neurais e lógica fuzzy [Zhi-tang et al., 2005] [Xiang et al., 2006]. Como benefício, o mecanismo de orquestração auxilia o administrador da rede nas tarefas repetitivas e rotineiras de configuração e avaliação.

Para resumir, a orquestração consiste na captura de regras e seqüências de como e quando as diferentes técnicas irão colaborar entre si para fornecer um serviço mais seguro. Em outras palavras, orquestração consiste em criar um modelo de processos executável que implementa um novo serviço através da harmonização de serviços pré-existentes. Os autores deste minicurso acreditam que o estilo de comunicação colaborativa entre os serviços de segurança representa um significativo passo em direção a desenvolvimento de sistema de autodefesa.

3.5. Conclusões

O tráfego não desejado pode ser considerado a verdadeira "pedra no sapato" da Internet. Embora os tipos e suas conseqüências sejam conhecidos, questões como a definição e as formas de minimizar seus efeitos ainda são motivos de discussão. A existência de uma "indústria do mal" motivada e evoluída, aliada ao surgimento de novos serviços e aplicações, a constante evolução tecnológica e ao boom populacional de novos (e freqüentemente despreparados) usuários impõe novos desafios na atividade de detectar e limitar o tráfego não desejado.

Este capítulo procurou introduzir o leitor no universo do tráfego Internet não desejado. Em primeiro lugar, o problema foi contextualizado e foram apresentadas definições e discussões sobre o assunto. As causas foram explicadas e os principais tipos de tráfego não desejado foram exemplificados. Em seguida foram discutidas as principais soluções desenvolvidas para lidar com o tráfego não desejado. Por último, foram apresenta algumas potencias e futuras soluções que estão à espera de serem implantadas ou ainda não estão em uso. Para estimular ainda mais o leitor, serão discutidas algumas questões em aberto e, por fim, serão feitas as observações finais.

3.5.1. Questões de pesquisa em aberto

Até o momento, a batalha com o tráfego não desejado tem apresentado soluções apenas reativas. Recentemente, o uso de análise de tráfego para identificar anomalias (principalmente ataques) tem recebido grande estímulo. Entretanto, um grande esforço ainda se faz necessário para encontrar alternativas inteligentes que não somente identifiquem, mas que também ajudem a reduzir este tipo de tráfego.

Algumas questões sobre o tráfego não desejado que ainda estão em aberto serão discutidas a seguir:

- Algumas vezes, novas aplicações e serviços impõem mudanças no gerenciamento e na capacidade dos enlaces das redes como, por exemplo, VoIP, IPTV, jogos em redes, entre outros. Contudo, as redes não estão preparadas para atender imediatamente todos os novos requisitos exigidos por estes serviços. A solução atual é "não fazer nada" e deixar que essas novas aplicações sejam penalizadas com o não atendimento de seus requisitos ou mesmo com seu total bloqueio até que seu perfil de tráfego seja entendido e criado. A solução ideal é a construção rápida e precisa dos perfis de tráfego dessas novas aplicações e serviços.

- Muitas das recentes tecnologias de rede proporcionam alta largura de banda. Se por um lado, isso permite o aumento da capacidade de tráfego da rede, por outro impõem uma alta carga sobre o gerenciamento da rede. Soluções como amostragens não atingem um nível de precisão adequado. As atuais pesquisas nesta área têm proposto o uso do tráfego agregado tais como análise fluxos de origem e destino (OD) [Lahkina et al., 2004a], a análise do componente principal *(Principal Component Analysis* – PCA) [Crovella e Krishnamurthy, 2006] e o uso de *sketches* [Li et al., 2006] [Abry et al., 2007]. Entretanto, os resultados ainda podem ser melhorados
- Uma vez que as soluções baseadas na análise do tráfego podem levar minutos ou até mesmo dias para descobrir indícios de tráfego não desejado, o tempo de detecção é muito importante. Tempos curtos indicam uma melhor precisão, mas anomalias de baixa carga tendem a não serem detectados. A solução usual é observar estatísticas de períodos de tempo maiores, mesmo que isso implique em grandes latências. Juntamente com as medidas de tomada de ações (por exemplo, aplicação de novas regras ao firewall), o tempo total resultante pode ser relativamente longo a ponto de permitir danos irreversíveis.
- Embora a diversidade de novas aplicações e serviços só aumente e conseqüentemente estimule o tráfego não desejado, a quantidade de soluções de detecção existentes não acompanha esse crescimento. Primeiro porque existem diferentes tipos ou classes de tráfego não desejado. Segundo, esses tráfegos diferem em termos de duração, objetivo e gravidade. Em resumo, as soluções são obrigadas a recorrer a uma variedade de metodologias e técnicas para resolver todas essas questões.
- Uma vez que os usuários (clientes) de banda larga permanecem longos períodos de tempo conectados, é importante conhecer o perfil do tráfego "normal" para esses usuários. Será que existe uma tendência em direção ao uso maior de tráfego para determinadas aplicações? Como é que estas aplicações irão se comportar? Como se comportam hoje com o número crescente de usuários?
- Com o aumento do número de usuários e computadores ligados na Internet, é importante conhecer quais são os computadores vulneráveis na rede. Será que simplesmente notificar os usuários que seus computadores têm vulnerabilidades e ameaças de segurança é suficiente e eficiente? Porque soluções automatizadas que permitem adequar esses computadores comprometidos tais como NAC [Cisco, 2008] e NAP [Microsoft, 2008] não são muito utilizadas?

3.5.2. Observações Finais

Hoje em dia, a Internet transporta uma grande quantidade de tráfego indesejado. Seja através da deturpação da filosofia aberta da Internet ou mesmo da exploração de diversas vulnerabilidades, o fato é que as conseqüências são altamente prejudiciais. A capacidade de causar danos financeiros é tamanha que esse tipo de tráfego é encontrado até em redes 3G [Ricciato, 2006] e [Ricciato et al., 2006].

A existência de uma economia de submundo financiando e lucrando com esse tipo de tráfego, a falta de normas e leis que responsabilizem e punam os culpados e as

limitadas ferramentas empregadas contribuem para a proliferação e perpetuação do tráfego não desejado.

Atualmente, não existe uma lâmpada mágica que forneça a resposta definitiva contra o tráfego Internet não desejado, mas algumas ações podem e devem ser tomadas:

- Aumentar o número de pesquisas nessa área visando, inicialmente, resolver problemas específicos. O apoio de agências governamentais, órgãos de fomento à pesquisa e a própria iniciativa privada devem servir de fontes financiadoras.
- Estimular a realização de eventos, workshops, conferências sobre o tráfego não desejado. Tais encontros servem como ponto de partida para troca de conhecimentos e o surgimento de novas parcerias.
- Desenvolver um modelo jurídico de alcance global para facilitar a captura, o julgamento e a punição aos criadores e incentivadores do tráfego indesejado. Este trabalho deve ser conduzido por especialistas de modo a garantir da melhor forma a flexibilidade da Internet.
- Aumentar o desenvolvimento e uso de soluções contra o tráfego não desejado, uma vez que mesmo que não encerrem o problema, podem diminuir seu volume.
- Por fim, educar os usuários Internet para torná-los mais conscientes dos riscos existentes para seus computadores e sistemas, e torná-los clientes mais exigentes na questão de segurança junto a suas operadoras.

Referências

[Abry et al., 2007] Abry, P., Borgnat, P., e Dewaele, G. (2007) Sketch based anomaly detection, identification and performance evaluation. *IEEE/IPSJ SAINT Measurement Workshop*, páginas 80-84.

[Anderson et al., 2007] Anderson, L., Davies, E., and Zhang, L. (2007) *Report from the IAB workshop on Unwanted Traffic March 9-10 2006*. RFC 4948. Internet Engineering Task Force.

[Andreolini et al., 2005] Andreolini, M., Bulgarelli, A., Colajanni, M., e Mazzoni, F. (2005) HoneySpam: Honeypots Fighting Spam at the Source. Em *International Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'05)*, páginas 77-83.

[Arbor Networks, 2005] Arbor Networks. (2005) *Worldwide ISP Security Report*. http://www.arbornetworks.com.

[Bailey et al., 2005] Bailey, M., Cooke, E., Jahanian, F., Nazario, J., Watson, D. (2005) The Internet Motion Sensor: A Distributed Blackhole Monitoring System. Em *12th Annual Network and Distributed System Security Symposium*.

[Baker e Savola, 2004] Baker, F., e Savola, P. (2004) *Ingress Filtering for Multihomed Networks*. BCP 84. RFC 3704. Internet Engineering Task Force.

[Bernaille et al., 2006] Bernaille, L., Teixeira, R., Akodjenou, I., Soule, A., e Salamatian, K. (2006) Traffic Classification on the Fly. *ACM SIGCOMM Computer. Communication Review,* 36(2), páginas 23-26, Abril. ACM Press.

[CERT, 2008] CERT – Computer Emergency Response Team. (2008) *Denial of Service Attacks*. http://www.cert.org/tech_tips/denail_of_service.html

[CERT.br, 2008] CERT.br – Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil. (2008) *Estatísticas do CERT.br*. http://www.cert.br/stats/incidentes/

[Choi et al., 2005] Choi, C., Dong, Y., e Zhang, Z. (2005) LIPS: Lightweight Internet Permit System for Stopping Unwanted Packets. *Lecture Notes in Computer Science*, páginas 178-190. Springer.

[Cisco, 2006] Cisco Systems. (2006) *Introduction to Cisco IOS NetFlow - A Technical Overview*. White Paper.http://www.cisco.com/en/US/products/ps6601/products_white_paper0900aecd80406232.shtml.

[Cisco, 2008] Cisco Systems. (2008) *Network Admission Control*. http://www.cisco.com/en/US/netsol/ns466/networking_solutions_package.html

[Cooke et al., 2004] Cooke, E., Bailey, M., Watson, D., Jahanian, F., e Nazario, J. *The Internet Motion Sensor: A Distributed Blackhole Monitoring System*. Technical Report CSE-TR-491-04, University of Michigan, Eletrical Engineering and Computer Science.

[Crovella e Krishnamurthy, 2006] Crovella, M., e Krishnamurthy, B. (2006) *Internet Measurement: Infrastructure, Traffic and Applications*. John Wiley & Sons, Inc., Nova Yorque, NY, USA.

[Davies, 2007] Davies, E. (2007) *Unwanted Traffic*. IETF Journal, volume 3, Dezembro. http://www.isoc.org/tools/blogs/ietfjournal/?p=172

[Dewaele et al., 2007] Dewaele, G., Fukuda, K., Borgnat, P., Abry, P., e Cho, K. (2007) Extracting Hidden Anomalies using Sketch and Non Gaussian Multiresolution Statistical Detection Procedures. *ACM SIGCOMM Workshop on Large-Scale Attack Defense* (LSAD), páginas 145-152.

[Duan et al., 2005] Duan, Z., Gopalan, K., e Dong, Y. (2005) Push vs. Pull: Implications of Protocol Design on Controlling Unwanted Traffic. Em *International Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'05)*.

[Dunia e Qin, 1998] Dunia, R., e Qin, S. J. (1998) A subspace approach to multidimensional fault identification and reconstruction. *American Institute of Chemical Engineers (AIChE) Journal*, páginas 1813–1831.

[Eichin e Rochlis, 1989] Eichin, M. e Rochlis, J. (1988) With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988. Em *IEEE Computer Society Symposium on Security and Privacy*.

[Erl, 2004] Erl, T. (2004) *Service-Oriented Architecture*: A Field Guide to Integrating XML and Web Services. Prentice Hall PTR, Upper Saddle River, NJ, USA.

[Ferguson e Senie, 2000] Ferguson, P., e Senie, D. (2000) *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*. BCP 38. RFC 2827. Internet Engineering Task Force.

[Feroul et al., 2005] Feroul, M., Roth, M., McGibney, J., Scheffler, T., e Waller, A. (2005) *Anomaly Detection System Study*. SEINIT Project. http://www.seinit.org/documents/Deliverables/SEINIT_D2.5_PU.pdf.

[Gao et al., 2006] Gao, J., Hu, G., Yao, X., e Chang, R. (2006) Anomaly Detection of Network Traffic Based on Wavelet Packet. Em *Asia-Pacific Conference on Communications (APPC'06)*.

[Gomes et al., 2005] Gomes, L. H., Castro, F., Almeida, V., Almeida, J., e Almeida, R. (2005) Improving Spam Detection Based on Structural Similarity. Em *International Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI'05),* páginas 89-95.

[Heidari, 2004] Heidari, M. (2004) *Malicious Codes in Depth*. http://www.securitydocs.com.

[Hoepers et al., 2003] Hoepers, C., Stending-Jenssen, K. e Montes, A. (2003) *Honeynets Applied to the CSIRT Scenario*. http://www.honeynet.org.br/papers/hnbr-first2003.pdf

[Honeynet Project, 2006] The Honeynet Project. (2006) *Know Your Enemy: Honeynets*. http://www.honeynet.org/papers/honeynet/index.html.

[Hyatt, 2006] Hyatt, R. (2006). Keeping DNS trustworthy. *The ISSA Journal*, páginas. 37-38.

[Juniper, 2008] Juniper. (2008) *Juniper JFlow*. http://www.juniper.net/techpubs/software/erx/junose80/swconfig-ip-services/html/ip-jflow-stats-config2.html

[Karagiannis et al., 2005] Karagiannis, T., Papagiannaki, K., e Faloutsos, M. (2005) BLINC: Multilevel traffic classification in the dark. *ACM SIGCOMM Computer Communication Review,* 35(4), páginas 229-240. ACM Press.

[Kumar e Spaffor, 1994] Kumar, S., e Spafford, E.H. (1994) *An application of pattern matching in intrusion detection*. The COAST Project, Department of Computer Sciences, Purdue University, West Lafayette, IN, USA, Technical Report CSD-TR-94-013.

[Krishnamurthy, 2006] Krishnamurthy, B. (2006) *Unwanted traffic: Important problems, research approaches*. IAB Workshop. http://www.research.att.com/~bala/papers

[Krishnamurthy e Blackmond, 2003] Krishnamurthy, B., e Blackmond, E. (2003) *SHRED: Spam Harassment Reduction via Economic Disincentives*. White Papers. AT&T. http://www.research.att.com/~bala/papers/

[Lakhina et al., 2004] Lakhina, A., Crovella, M., e Diot, C. (2004) Diagnosing Network-Wide Traffic Anomalies. Em *ACM SIGCOMM*, páginas 219-230.

[Lakhina et al., 2005] Lakhina, A., Crovella, M., e Diot, C. (2005) Mining anomalies using traffic feature distributions. *ACM SIGCOMM Computer Communication Review,* 35(4), páginas 217-228. ACM Press.

[Laufer et al., 2005] Laufer, R., Moraes, I., Velloso, P., Bicudo, M., Campista, M., Cunha, D., Costa, L., e Duarte, O. (2005) Negação de Serviço: Ataques e

Contramedidas. *Livro Texto dos Mini-cursos do V Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais* (SBSeg'2005). Florianópolis, Brasil.

[Li et al., 2006] Li, X., Bian, F., Crovella, M., Diot, C., Govindan, R., Iannaccone, G., e Lahkina, A. (2006) Detection and Identification of Network Anomalies using Sketch Subspaces. Em *6th ACM SIGCOMM on Internet Measurement Conference* (IMC´06), páginas 147-152. Rio de Janeiro, Brasil. ACM Press.

[Mahalanobis, 1930] Mahalanobis, P. C. (1930) On tests and measures of groups divergence. *Journal of the Asiatic Society of Bengal*, volume 26.

[Microsoft, 2008] Microsoft. (2008) *Network Access Protection*. http://technet.microsoft.com/en-us/network/bb545879.aspx

[Mirkovic et al., 2004] Mirkovic, J., Martin, J., e Reiher, P. (2004) A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms. *ACM SIGCOMM Computer Communications Review*, 34(2), páginas 39–53. ACM Press.

[Moore, 2002] Moore, D. (2002) Network Telescopes: Observing Small or Distant Security Events. Em *11th USENIX Security Symposium*.

[Moore et al., 2004] Moore, D., Shannon, C., Voelkery, G. M., e Savagey, S. (2004) *Network Telescopes:* Technical Report. Cooperative Association for Internet Data Analysis.CAIDA. http://www.caida.org/outreach/papers/2004/tr-2004-04

[Morin, 2006] Morin, M. (2006) The Financial Impact of Attack Traffic on Broadband Networks. *IEC Annual Review of Broadband Communications*, páginas 11–14.

[Oliveira et al., 2007] Oliveira, E. L., Aschoff, R., Lins, B., Feitosa, E., Sadok, D. (2007) Avaliação de Proteção contra Ataques de Negação de Serviço Distribuídos (DDoS) utilizando Lista de IPs Confiáveis. *VII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais* (VII SBSEG). Rio de Janeiro, Brasil.

[OpenNet, 2004] OpenNet. (2004) *Internet Filtering in China in 2004-2005: A County Study*.
http://www.opennetinitiative.net/studies/china/ONI_China_Country_Study.pdf

[Pang et al., 2004] Pang, R., Yegneswaran, V., Barford, P., Paxon, V., e Peterson, L. (2004) Characteristics of Internet Background Radiation. Em *4th ACM SIGCOMM on Internet Measurement Conference* (IMC' 04). ACM Press.

[Paxson, 1998] Paxson, V. (1998) Bro: A System for Detecting Network Intruders in Real-Time. Em *8th USENIX Security Symposium*.

[Provos, 2004] Provos, N. (2004) A Virtual Honeypot Framework. Em *13th USENIX Security Symposium*.

[Provos, 2008] Provos, N. (2008) *Honeyd*. http://www.honeyd.org.

[Radicati Group, 2006] Radicati Group. (2006) Corporate Email Market 2006-2010. http://www.radicati.com.

[Ricciato, 2006] Ricciato, F. (2006) Unwanted traffic in 3G networks. *ACM SIGCOMM Computer Communications Review*, 36(2), páginas 53 – 5l. ACM Press.

[Ricciato et al., 2006] Ricciato, F., Hasenleithner, E., Svoboda, P., and Fleischer, W. (2006) On the impact of unwanted traffic onto a 3G network. Em *Second*

International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing (SecPerU 2006).

[Richardson, 2007] Richardson, R. (2007) CSI/FBI Computer Crime Survey. Em *Twelfth Annual Computer Crime and Security*, páginas 1-30.

[Scherrer et al., 2006] Scherrer, A., Larrieu, N., Borgnat, P., Owezarski, P., e Abry, P. (2006) Non Gaussian and Long Memory Statistical Modeling of Internet Traffic. Em *4th International Workshop on Internet Performance, Simulation, Monitoring and Measurement* (IPS-MoMe).

[Scherrer et al., 2007] Scherrer, A., Larrieu, N., Owezarski, P., Borgnat, P., e Abry, P. (2007) Non-Gaussian and Long Memory Statistical Characterizations for Internet Traffic with Anomalies. *IEEE Transactions on Dependable and Secure Computing*, volume 4, páginas 56-70.

[Schulze e Mochalski, 2007] Schulze, H. e Mochalski, K. (2007) *Internet Study 2007*. Ipoque. http://www.ipoque.com/userfiles/file/internet_study_2007.pdf

[Snort, 2008] Snort. (2008) *Snort IDS*. http://www.snort.org.

[Soto, 2005] Soto, P. (2005) *Identifying and Modeling Unwanted Traffic on the Internet*. Dissertação de Mestrador. Departamento de Engenharia Elétrica e Ciência da Computação, Massachusetts Institute of Technology (MIT).

[Spitzner, 2002] Spitzner, L. (2002) *Honeypots: Tracking Hackers*. Addison-Wesley Professional.

[SRUTI, 2005] USENIX. (2005) *International Workshop on Steps to Reducing Unwanted Traffic on the Internet*. http://www.usenix.org/events/sruti05/

[StillSecure, 2008] StillSecure. (2008) *Safe Access – Network access control solution*. http://www.stillsecure.com/safeaccess.

[Taveira et al., 2006] Taveira, D., Moraes, I., Rubinstein, M. e Duarte, O. (2006) Técnicas de Defesa Contra Spam. *Livro Texto dos Mini-cursos do VI Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais* (SBSeg'2006). Santos, Brasil.

[Weaver et al., 2003] Weaver, N., Paxson, V., Staniford, S., e Cunningham. R. (2003) A Taxonomy of Computer Worms. Em *ACM Workshop on Rapid Malcode*.

[Wu et al, 2006] Wu, M., Miller, R.C., Little, G. (2006) Web Wallet: Preventing Phishing Attacks by Revealing User Intentions. Em *Symposium On Usable Privacy and Security* (SOUPS), páginas 102-113.

[Xiang et al., 2006] Xiang, G., Min, W., e Rongchun, Z. (2006) Application of Fuzzy ART for Unsupervised Anomaly Detection System. Em *International Conference on Computational Intelligence and Security*, volume 1, páginas 621-624.

[Xu et al., 2005a] Xu, K., Zhang, Z-L., e Bhattacharrya, S. (2005) Profiling internet backbone traffic: Behavior models and applications. *ACM SIGCOMM Computer Communication Review,* 35(4), páginas 169-180. ACM Press.

[Xu et al., 2005b] Xu, K., Zhang, Z-L., e Bhattacharrya, S. (2005) Reducing unwanted traffic in a backbone Network. Em *International Workshop on Steps of Reducing Unwanted Traffic on the Internet* (SRUTI´05).

[Zhi-tang et al., 2005] Zhi-tang, L., Yao, L., e Li, W. (2005) A Novel Fuzzy Anomaly Detection Algorithm Based on Artificial Immune System. Em *8th International Conference on High-Performance Computing in Asia-Pacific Region* (HPCASIA'05), páginas 479-483.

Capítulo

4

Virtualização: Conceitos e Aplicações em Segurança

Marcos Aurelio Pchek Laureano[1,2], Carlos Alberto Maziero[1]

[1]Programa de Pós-Graduação em Informática
Pontifícia Universidade Católica do Paraná
Curitiba PR

[2]Centro Universitário Franciscano (UNIFAE)
Curitiba PR

Abstract

The recent evolution of the virtual machines' technology allowed their large-scale adoption in production systems. Currently, the main use of virtual machines has been the servers' consolidation, aiming at reducing hardware, software, and management costs of the installed computing infrastructure. However, in the recent years, several research works showed many possibilities of using virtual machines in the systems security area. This work presents the fundamental concepts about virtual machines, discusses advantages and drawbacks of using virtual machines in several application domains, explores the usage possibilities of virtual machines in the systems security area, and shows some examples of popular virtual machine environments.

Resumo

A evolução recente da tecnologia de máquinas virtuais permitiu a sua adoção em larga escala nos sistemas de produção. O principal uso de máquinas virtuais tem sido a consolidação de servidores e, conseqüentemente, a redução de custos em hardware, software e gerência do parque tecnológico. No entanto, várias pesquisas vêm demonstrando possibilidades de aplicação de máquinas virtuais na área de segurança de sistemas. Este trabalho apresenta os fundamentos conceituais sobre máquinas virtuais, discute as vantagens e desvantagens no uso de máquinas virtuais, explora as principais possibilidades de uso de máquinas virtuais na área de segurança de sistemas e expõe alguns exemplos de ambientes de máquinas virtuais de uso corrente.

4.1. Introdução à virtualização

O conceito de máquina virtual não é recente. Os primeiros passos na construção de ambientes de máquinas virtuais começaram na década de 1960, quando a IBM desenvolveu o sistema operacional experimental M44/44X. A partir dele, a IBM desenvolveu vários sistemas comerciais suportando virtualização, entre os quais o famoso OS/370 [Goldberg 1973, Goldberg and Mager 1979]. A tendência dominante nos sistemas naquela época era fornecer a cada usuário um ambiente mono-usuário completo, com seu próprio sistema operacional e aplicações, completamente independente e desvinculado dos ambientes dos demais usuários.

Na década de 1980, com a popularização de plataformas de hardware baratas como o PC, a virtualização perdeu importância. Afinal, era mais barato, simples e versátil fornecer um computador completo a cada usuário, que investir em sistemas de grande porte, caros e complexos. Além disso, o hardware do PC tinha desempenho modesto e não provia suporte adequado à virtualização, o que inibiu o uso de ambientes virtuais nessas plataformas.

Com o aumento de desempenho e funcionalidades do hardware PC e o surgimento da linguagem Java, no início dos anos 90, o interesse pelas tecnologias de virtualização voltou à tona. Apesar da plataforma PC Intel ainda não oferecer um suporte adequado à virtualização, soluções engenhosas como as adotadas pela empresa VMWare permitiram a virtualização nessa plataforma, embora com desempenho relativamente modesto. Atualmente, as soluções de virtualização de linguagens e de plataformas vêm despertando grande interesse do mercado. Várias linguagens são compiladas para máquinas virtuais portáveis e os processadores mais recentes trazem um suporte nativo à virtualização.

4.1.1. Arquitetura dos sistemas computacionais

Uma máquina real é formada por vários componentes físicos que fornecem operações para o sistema operacional e suas aplicações. Iniciando pelo núcleo do sistema real, o processador central (CPU) e o *chipset* da placa-mãe fornecem um conjunto de instruções e outros elementos fundamentais para o processamento de dados, alocação de memória e processamento de entrada/saída. Os sistemas de computadores são projetados com basicamente três componentes: hardware, sistema operacional e aplicações. O papel do hardware é executar as operações solicitadas pelas aplicações através do sistema operacional. O sistema operacional recebe as solicitações das operações (por meio das chamadas de sistema) e controla o acesso ao hardware – principalmente nos casos em que os componentes são compartilhados, como o sistema de memória e os dispositivos de entrada/saída.

Os sistemas de computação convencionais são caracterizados por níveis de abstração crescentes e interfaces bem definidas entre eles. Um dos principais objetivos dos sistemas operacionais é oferecer uma visão abstrata, de alto nível, dos recursos de hardware, que sejam mais simples de usar e menos dependente das tecnologias subjacentes. As abstrações oferecidas pelo sistema às aplicações são construídas de forma incremental, em níveis separados por interfaces bem definidas e relativamente padronizadas. Cada interface encapsula as abstrações dos níveis inferiores, permitindo assim o desenvolvimento independente dos vários níveis, o que simplifica a construção e evolução dos sistemas.

Exemplos típicos dessa estruturação em níveis de abstração crescentes, separados por interfaces bem definidas, são os subsistemas de rede e de disco em um sistema operacional convencional. No sub-sistema de arquivos, cada nível de abstração trata de um problema: interação com o dispositivo físico de armazenamento, escalonamento de acessos ao dispositivo, gerência de *buffers* e *caches*, alocação de arquivos, diretórios, controle de acesso, etc. A figura 4.1 apresenta os níveis de abstração de um subsistema de gerência de disco típico.

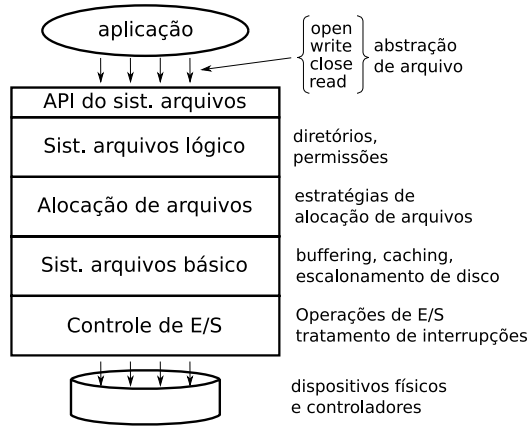

Figura 4.1. Níveis de abstração em um sub-sistema de disco.

As interfaces existentes entre os componentes de um sistema de computação são:

- *Conjunto de instruções* (*ISA – Instruction Set Architecture*): é a interface básica entre o hardware e o software, sendo constituída pelas instruções em código de máquina aceitas pelo processador e todas as operações de acesso aos recursos do hardware (acesso físico à memória, às portas de entrada/saída, ao relógio do sistema, etc.). Essa interface é dividida em duas partes:
  - *Instruções de usuário* (*User ISA*): compreende as instruções do processador e demais itens de hardware acessíveis aos programas do usuário, que executam com o processador operando em modo não-privilegiado;
  - *Instruções de sistema* (*System ISA*): compreende as instruções do processador e demais itens de hardware, unicamente acessíveis ao núcleo do sistema operacional, que executa em modo privilegiado;
- *Chamadas de sistema* (*syscalls*): é o conjunto de operações oferecidas pelo núcleo do sistema operacional aos processos dos usuários. Essas chamadas permitem um acesso controlado das aplicações aos dispositivos periféricos, à memória e às instruções privilegiadas do processador.

- *Chamadas de bibliotecas* (*libcalls*): bibliotecas oferecem um grande número de funções para simplificar a construção de programas; além disso, muitas chamadas de biblioteca encapsulam chamadas do sistema operacional, para tornar seu uso mais simples. Cada biblioteca possui uma interface própria, denominada *Interface de Programação de Aplicações* (*API – Application Programming Interface*). Exemplos típicos de bibliotecas são a *LibC* do UNIX (que oferece funções como `fopen` e `printf`), a GTK+ (*Gimp ToolKit*, que permite a construção de interfaces gráficas) e a SDL (*Simple DirectMedia Layer*, para a manipulação de áudio e vídeo).

A figura 4.2 apresenta essa visão conceitual da arquitetura de um sistema computacional, com seus vários componentes e as respectivas interfaces entre eles.

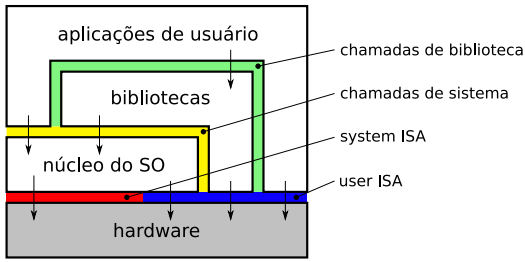

Figura 4.2. Componentes e interfaces de um sistema computacional.

4.1.2. Compatibilidade entre interfaces de sistema

Para que programas e bibliotecas possam executar sobre uma determinada plataforma, é necessário que tenham sido compilados para ela, respeitando o conjunto de instruções do processador em modo usuário (*User ISA*) e o conjunto de chamadas de sistema oferecido pelo sistema operacional. A visão conjunta dessas duas interfaces (*User ISA* + *syscalls*) é denominada *Interface Binária de Aplicação* (*ABI – Application Binary Interface*). Da mesma forma, um sistema operacional só poderá executar sobre uma plataforma de hardware se tiver sido construído e compilado de forma a respeitar sua interface ISA (*User/System ISA*). A figura 4.3 representa essas duas interfaces.

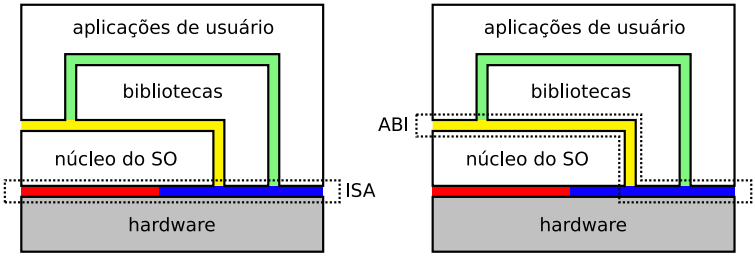

Figura 4.3. Interfaces de sistema ISA e ABI [Smith and Nair 2004].

Nos sistemas computacionais de mercado atuais, as interfaces de baixo nível ISA e ABI são normalmente fixas, ou pouco flexíveis. Geralmente não é possível criar novas instruções de processador ou novas chamadas de sistema operacional, ou mesmo mudar sua semântica para atender às necessidades específicas de uma determinada aplicação. Mesmo se isso fosse possível, teria de ser feito com cautela, para não comprometer o funcionamento de outras aplicações.

Os sistemas operacionais, assim como as aplicações, são projetados para aproveitar o máximo dos recursos que o hardware fornece. Normalmente os projetistas de hardware, sistema operacional e aplicações trabalham de forma independente (em empresas e tempos diferentes). Por isso, esses trabalhos independentes geraram, ao longo dos anos, várias plataformas computacionais diferentes e incompatíveis entre si.

Observa-se então que, embora a definição de interfaces seja útil, por facilitar o desenvolvimento independente dos vários componentes do sistema, torna pouco flexíveis as interações entre eles: um sistema operacional só funciona sobre o hardware (ISA) para o qual foi construído, uma biblioteca só funciona sobre a ABI para a qual foi projetada e uma aplicação tem de obedecer a ABIs/APIs pré-definidas. A figura 4.4, extraída de [Smith and Nair 2004], ilustra esses problemas de compatibilidade entre interfaces.

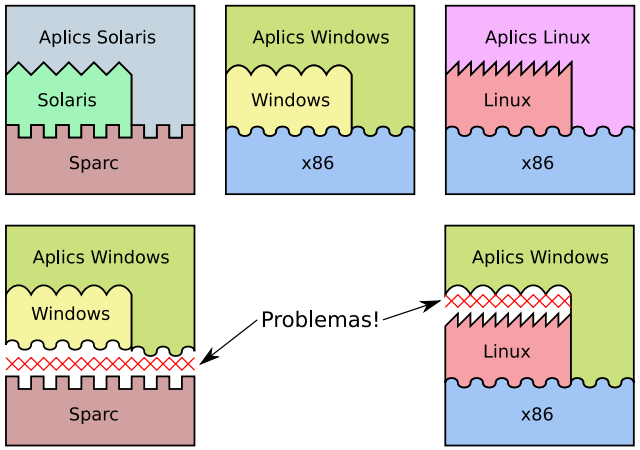

Figura 4.4. Problemas de compatibilidade entre interfaces [Smith and Nair 2004].

A baixa flexibilidade na interação entre as interfaces dos componentes de um sistema computacional traz vários problemas [Smith and Nair 2004]:

- *Baixa portabilidade*: a mobilidade de código e sua interoperabilidade são requisitos importantes dos sistemas atuais, que apresentam grande conectividade de rede e diversidade de plataformas. A rigidez das interfaces de sistema atuais dificulta sua construção, por acoplar excessivamente as aplicações aos sistemas operacionais e aos componentes do hardware.

- *Barreiras de inovação*: a presença de interfaces rígidas dificulta a construção de novas formas de interação entre as aplicações e os dispositivos de hardware (e com os usuários, por conseqüência). Além disso, as interfaces apresentam uma grande inércia à evolução, por conta da necessidade de suporte às aplicações já existentes.

- *Otimizações inter-componentes*: aplicações, bibliotecas, sistemas operacionais e hardware são desenvolvidos por grupos distintos, geralmente com pouca interação entre eles. A presença de interfaces rígidas a respeitar entre os componentes leva cada grupo a trabalhar de forma isolada, o que diminui a possibilidade de otimizações que envolvam mais de um componente.

Essas dificuldades levaram à investigação de outras formas de relacionamento entre os componentes de um sistema computacional. Uma das abordagens mais promissoras nesse sentido é o uso de *máquinas virtuais*, que serão apresentadas na próxima seção.

4.1.3. Máquinas virtuais

Conforme visto, as interfaces padronizadas entre os componentes do sistema de computação permitem o desenvolvimento independente dos mesmos, mas também são fonte de problemas de interoperabilidade, devido à sua pouca flexibilidade. Por isso, não é possível executar diretamente em um processador Intel/AMD uma aplicação compilada para um processador ARM: as instruções em linguagem de máquina do programa não serão compreendidas pelo processador Intel. Da mesma forma, não é possível executar diretamente em Linux uma aplicação escrita para Windows, pois as chamadas de sistema emitidas pelo programa não serão compreendidas pelo sistema operacional subjacente.

Todavia, é possível contornar esses problemas de compatibilidade através de uma *camada de virtualização* construída em software. Usando os serviços oferecidos por uma determinada interface de sistema, é possível construir uma camada de software que ofereça aos demais componentes uma outra interface. Essa camada de software permitirá o acoplamento entre interfaces distintas, de forma que um programa desenvolvido para a plataforma *A* possa executar sobre uma plataforma distinta *B* (figura 4.5).

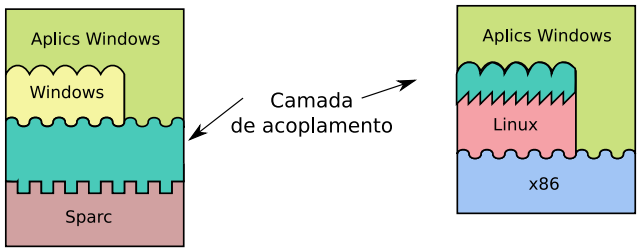

Figura 4.5. Acoplamento entre interfaces distintas

Usando os serviços oferecidos por uma determinada interface de sistema, a camada de virtualização constrói outra interface de mesmo nível, de acordo com as necessidades dos componentes de sistema que farão uso dela. A nova interface de sistema,

vista através dessa camada de virtualização, é denominada *máquina virtual*. A camada de virtualização em si é denominada *hipervisor* ou *monitor de máquina virtual*.

A figura 4.6, extraída de [Smith and Nair 2004], apresenta um exemplo de máquina virtual, onde um hipervisor permite executar um sistema operacional Windows e suas aplicações sobre uma plataforma de hardware Sparc, distinta daquela para a qual esse sistema operacional foi projetado (Intel/AMD).

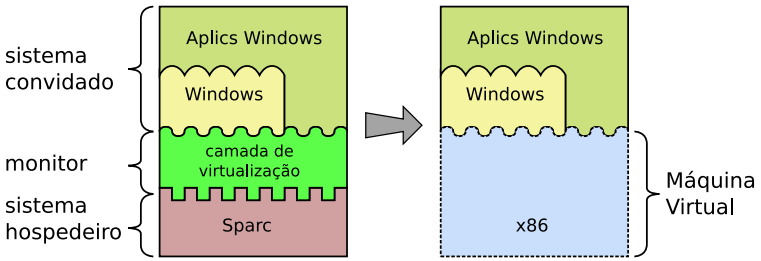

Figura 4.6. Uma máquina virtual [Smith and Nair 2004].

Um ambiente de máquina virtual consiste de três partes básicas, que podem ser observadas na figura 4.6:

- O sistema real, nativo ou hospedeiro (*host system*), que contém os recursos reais de hardware e software do sistema;
- o sistema virtual, também denominado *sistema convidado* (*guest system*), que executa sobre o sistema virtualizado; em alguns casos, vários sistemas virtuais podem coexistir, executando simultaneamente sobre o mesmo sistema real;
- a camada de virtualização, *hipervisor*, ou *monitor* (VMM – *Virtual Machine Monitor*), que constrói as interfaces virtuais a partir da interface real.

Originalmente, uma máquina virtual era definida como uma cópia eficiente, isolada e protegida de uma máquina real [Popek and Goldberg 1974, Goldberg and Mager 1979]. Essa definição, vigente nos anos 1960-70, foi construída a partir da perspectiva de um sistema operacional: uma abstração de software que gerencia um sistema físico (máquina real). Com o passar dos anos, o termo "máquina virtual" evoluiu e englobou um grande número de abstrações, como por exemplo a *Java Virtual Machine* (JVM), que não virtualiza um sistema real [Rosenblum 2004].

A construção de máquinas virtuais é bem mais complexa que possa parecer à primeira vista. Caso os conjuntos de instruções do sistema real e do sistema virtual sejam diferentes, é necessário usar as instruções da máquina real para simular as instruções da máquina virtual. Além disso, é necessário mapear os recursos de hardware virtuais (periféricos oferecidos ao sistema convidado) sobre os recursos existentes na máquina real (os periféricos reais). Por último, pode ser necessário mapear as chamadas de sistema

emitidas pelas aplicações do sistema convidado em chamadas equivalentes no sistema real, quando os sistemas operacionais virtual e real forem distintos.

Existem várias formas de implementar a virtualização, que serão descritas nas próximas seções. Algumas delas são apresentadas sucintamente na figura 4.7, como:

- *Virtualização completa*: um sistema operacional convidado e suas aplicações, desenvolvidas para uma plataforma de hardware *A*, são executadas sobre uma plataforma de hardware distinta *B*.
- *Emulação do sistema operacional*: as aplicações de um sistema operacional *X* são executadas sobre outro sistema operacional *Y*, na mesma plataforma de hardware.
- *Tradução dinâmica*: as instruções de máquina das aplicações são traduzidas durante a execução em outras instruções mais eficientes para a mesma plataforma.
- *Replicação de hardware*: são criadas várias instâncias virtuais de um mesmo hardware real, cada uma executando seu próprio sistema operacional convidado e suas respectivas aplicações.

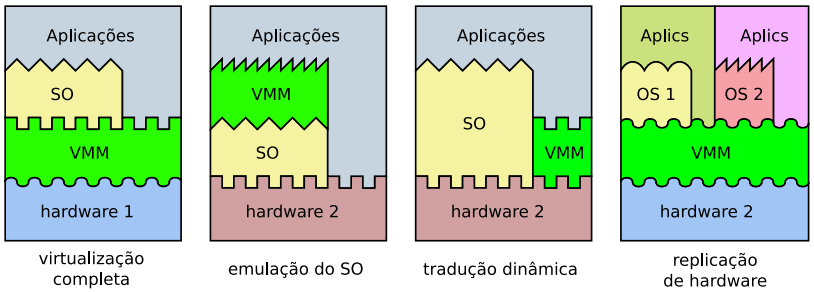

Figura 4.7. Possibilidades de virtualização [Smith and Nair 2004].

4.1.3.1. Abstração versus virtualização

Embora a virtualização possa ser vista como um tipo de abstração, existe uma clara diferença entre os termos "abstração" e "virtualização", no contexto de sistemas [Smith and Nair 2005]. A abstração de recursos consiste em fornecer uma interface de acesso homogênea e simplificada ao recursos do sistema. Por outro lado, a virtualização consiste em criar novas interfaces a partir das interfaces existentes. Na virtualização, os detalhes de baixo nível da plataforma real não são necessariamente ocultos, como ocorre na abstração de recursos. A figura 4.8 ilustra essa diferença: através da virtualização, um processador Sparc pode ser visto como um processador Intel. Da mesma forma, um disco real no padrão SATA pode ser visto como vários discos menores independentes, com a mesma interface (SATA) ou outra interface (IDE).

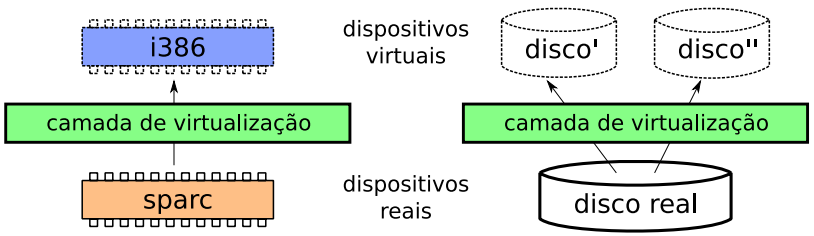

Figura 4.8. Virtualização versus abstração de recursos.

A figura 4.9 ilustra outro exemplo dessa diferença no contexto do armazenamento em disco. A abstração provê às aplicações o conceito de "arquivo", sobre o qual estas podem executar operações simples como `read` ou `write`, por exemplo. Já a virtualização fornece para a camada superior apenas um disco virtual, construído a partir de um arquivo do sistema operacional real subjacente. Esse disco virtual terá de ser particionado e formatado para seu uso, da mesma forma que um disco real.

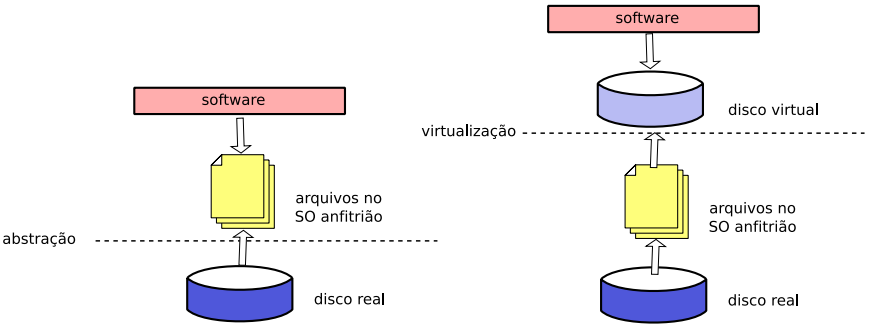

Figura 4.9. Abstração versus virtualização de um disco rígido.

4.1.3.2. Propriedades dos hipervisores

Para funcionar de forma correta e eficiente, um hipervisor deve atender a alguns requisitos básicos: ele deve prover um ambiente de execução aos programas essencialmente idêntico ao da máquina real, do ponto de vista lógico. Programas executando sobre uma máquina virtual devem apresentar, no pior caso, leves degradações de desempenho. Além disso, o hipervisor deve ter controle completo sobre os recursos do sistema real (o sistema hospedeiro). A partir desses requisitos, foram estabelecidas por [Goldberg 1973, Popek and Goldberg 1974] as seguintes propriedades a serem satisfeitas por um hipervisor ideal:

- *Equivalência*: um hipervisor provê um ambiente de execução quase idêntico ao da máquina real original. Todo programa executando em uma máquina virtual deve se

comportar da mesma forma que o faria em uma máquina real; exceções podem resultar somente de diferenças nos recursos disponíveis (memória, disco, etc), dependências de temporização e a existência dos dispositivos de entrada/saída necessários à aplicação.

- *Controle de recursos*: o hipervisor deve possuir o controle completo dos recursos da máquina real: nenhum programa executando na máquina virtual deve possuir acesso a recursos que não tenham sido explicitamente alocados a ele pelo hipervisor, que deve intermediar todos os acessos. Além disso, a qualquer instante o hipervisor pode resgatar recursos previamente alocados.

- *Eficiência*: grande parte das instruções do processador virtual (o processador provido pelo hipervisor) deve ser executada diretamente pelo processador da máquina real, sem intervenção do hipervisor. As instruções da máquina virtual que não puderem ser executadas pelo processador real devem ser interpretadas pelo hipervisor e traduzidas em ações equivalentes no processador real. Instruções simples, que não afetem outras máquinas virtuais ou aplicações, podem ser executadas diretamente no hardware.

Além dessas três propriedades básicas, as propriedades derivadas a seguir são freqüentemente associadas a hipervisores [Popek and Goldberg 1974, Rosenblum 2004]:

- *Isolamento*: esta propriedade garante que um software em execução em uma máquina virtual não possa ver, influenciar ou modificar outro software em execução no hipervisor ou em outra máquina virtual. Esta propriedade pode ser utilizada para garantir que erros de software ou aplicações maliciosas possam ser contidos em um ambiente controlado (uma máquina virtual), sem afetar outras partes do sistema.

- *Inspeção*: o hipervisor tem acesso e controle sobre todas as informações do estado interno da máquina virtual, como registradores do processador, conteúdo de memória, eventos etc.

- *Gerenciabilidade*: como cada máquina virtual é uma entidade independente das demais, a administração de diversas instâncias de máquinas virtuais sobre um mesmo supervisor é simplificada e centralizada. O hipervisor deve ter mecanismos para gerenciar o uso dos recursos existentes entre os sistemas convidados.

- *Encapsulamento*: como o hipervisor tem acesso e controle sobre o estado interno de cada máquina virtual em execução, ele pode salvar *checkpoints* de uma máquina virtual, periodicamente ou em situações especiais (por exemplo, antes de uma atualização de sistema operacional). Esses *checkpoints* são úteis para retornar a máquina virtual a estados anteriores (*rollback*), para análises *post-mortem* em caso de falhas, ou para permitir a migração da máquina virtual entre hipervisores executando em computadores distintos.

- *Recursividade*: alguns sistemas de máquinas virtuais exibem também esta propriedade: deve ser possível executar um hipervisor dentro de uma máquina virtual, produzindo um novo nível de máquinas virtuais. Neste caso, a máquina real é normalmente denominada *máquina de nível 0* (figura 4.10).

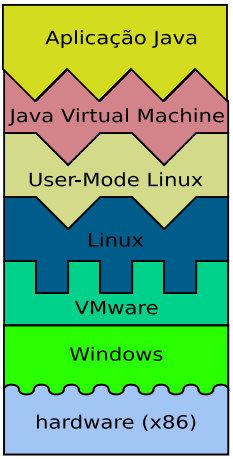

Figura 4.10. Diversos níveis de virtualização [Laureano 2006]

Essas propriedades também podem ser utilizadas na segurança de sistemas [Honeynet Project 2003, Garfinkel and Rosenblum 2003] e outras aplicações. As propriedades básicas caracterizam um hipervisor ideal, que nem sempre pode ser construído sobre as plataformas de hardware existentes. A possibilidade de construção de um hipervisor em uma determinada plataforma é definida através do seguinte teorema, enunciado e provado por Popek e Goldberg em [Popek and Goldberg 1974]:

> *Para qualquer computador convencional de terceira geração, um hipervisor pode ser construído se o conjunto de instruções sensíveis daquele computador for um sub-conjunto de seu conjunto de instruções privilegiadas.*

Para compreender melhor as implicações desse teorema, é necessário definir claramente os seguintes conceitos:

- *Computador convencional de terceira geração*: qualquer sistema de computação convencional seguindo a arquitetura de Von Neumann, que suporte memória virtual e dois modos de operação: modo usuário e modo privilegiado.
- *Instruções sensíveis*: são aquelas que podem consultar ou alterar o status do processador, ou seja, os registradores que armazenam o status atual da execução na máquina real;
- *Instruções privilegiadas*: são acessíveis somente por meio de códigos executando em nível privilegiado (código de núcleo). Caso um código não-privilegiado tente executar uma instrução privilegiada, uma exceção (interrupção) é gerada, ativando uma rotina de tratamento previamente especificada pelo núcleo do sistema real.

De acordo com esse teorema, toda instrução sensível deve ser também privilegiada. Assim, quando uma instrução sensível for executada por um programa não-privilegiado (um núcleo convidado ou uma aplicação convidada), provocará a ocorrência de uma interrupção. Essa interrupção pode ser usada para ativar uma *rotina de interpretação* dentro do hipervisor, que irá simular o efeito da instrução sensível (ou seja, interpretá-la), de acordo com o contexto onde sua execução foi solicitada (máquina virtual ou hipervisor). Obviamente, quanto maior o número de instruções sensíveis, maior o volume de interpretação de código a realizar, e menor o desempenho da máquina virtual.

No caso de processadores que não atendam o teorema de Popek e Goldberg, podem existir instruções sensíveis que executem sem gerar interrupções, o que impede o hipervisor de interceptá-las e interpretá-las. Uma solução possível para esse problema é a *tradução dinâmica* das instruções sensíveis acessíveis nos programas de usuário: ao carregar um programa na memória, o hipervisor analisa seu código e substitui as instruções problemáticas por chamadas a rotinas que as interpretam dentro do hipervisor. Isso implica em um tempo maior para o lançamento de programas, mas torna possível a virtualização. Outra técnica possível para resolver o problema é a *para-virtualização*. Estas técnicas são discutidas na seção 4.1.5.

4.1.3.3. Suporte de hardware

Na época em que Popek e Goldberg definiram seu principal teorema, o hardware virtualizável dos mainframes IBM suportava parcialmente as condições impostas pelo mesmo. Esses sistemas dispunham de uma funcionalidade chamada *execução direta*, que permitia a uma máquina virtual acessar nativamente o hardware para execução de instruções. Esse mecanismo permitia que aqueles sistemas obtivessem, com a utilização de máquinas virtuais, desempenho similar ao de sistemas convencionais equivalentes [Goldberg 1973, Popek and Goldberg 1974, Goldberg and Mager 1979].

O suporte de hardware necessário para a construção de hipervisores eficientes está presente em sistemas de grande porte, mas é apenas parcial nos micro-processadores de mercado. Por exemplo, a família de processadores *Intel Pentium IV* (e anteriores) possui 17 instruções sensíveis que podem ser executadas em modo usuário sem gerar exceções, o que viola o teorema de Goldberg e dificulta a criação de máquinas virtuais em sistemas que usam esses processadores [Robin and Irvine 2000].

Por volta de 2005, os principais fabricantes de micro-processadores (*Intel* e *AMD*) incorporaram um suporte básico à virtualização em seus processadores, através das tecnologias IVT (*Intel Virtualization Technology*) e AMD-V (*AMD Virtualization*), que são conceitualmente equivalentes [Uhlig et al. 2005]. A idéia central de ambas as tecnologias consiste em definir dois modos possíveis de operação do processador: os modos *root* e *non-root*. O modo *root* equivale ao funcionamento de um processador convencional, e se destina à execução de um hipervisor. Por outro lado, o modo *non-root* se destina à execução de máquinas virtuais. Além de Intel e AMD, outros fabricantes de hardware têm se preocupado com o suporte à virtualização. Em 2005, a *Sun Microsystems* incorporou suporte nativo à virtualização em seus processadores *UltraSPARC* [Yen 2007]. Em 2007, a IBM propôs uma especificação de interface de hardware denominada *IBM Power ISA*

2.04 [IBM 2007], que respeita os requisitos necessários à virtualização do processador e da gestão de memória.

4.1.4. Tipos de máquinas virtuais

Conforme discutido na seção 4.1.3, existem diversas possibilidades de implementação de sistemas de máquinas virtuais, cada uma com suas características próprias de eficiência e equivalência. De acordo com o tipo de sistema convidado suportado, os ambientes de máquinas virtuais podem ser divididos em duas grandes famílias (figura 4.11):

- *Máquinas virtuais de aplicação* (*Process Virtual Machines*): são ambientes de máquinas virtuais destinados a suportar apenas um processo ou aplicação convidada específica. A máquina virtual Java e o ambiente de depuração *Valgrind* são exemplos desse tipo de ambiente.
- *Máquinas virtuais de sistema* (*System Virtual Machines*): são ambientes de máquinas virtuais construídos para suportar sistemas operacionais convidados completos, com aplicações convidadas executando sobre eles. Como exemplos, temos os ambientes *VMware* e *VirtualBox*.

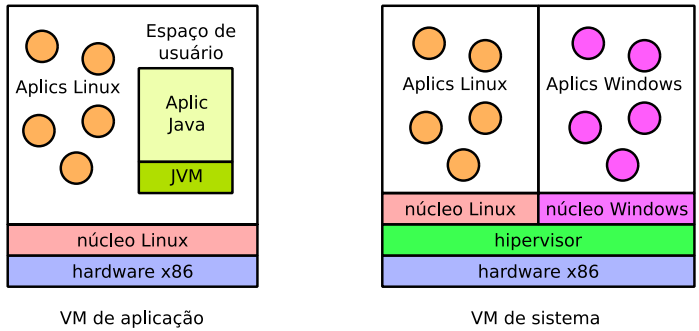

Figura 4.11. Máquinas virtuais de aplicação e de sistema.

Por outro lado, os ambientes de máquinas virtuais também podem ser classificados de acordo com o nível de similaridade entre as interfaces de hardware do sistema convidado e do sistema real (ISA):

- *Interfaces equivalentes*: a interface virtual oferecida ao ambiente convidado reproduz a interface de hardware do sistema real, permitindo a execução de aplicações construídas para o sistema real. Como a maioria das instruções do sistema convidado pode ser executada diretamente pelo processador (com exceção das instruções sensíveis), o desempenho obtido pelas aplicações convidadas pode ser próximo do desempenho de execução no sistema real. Ambientes como *VMware* são exemplos deste tipo de ambiente.

- *Interfaces distintas*: a interface virtual não tem relação com a interface de hardware do sistema real, ou seja, implementa um conjunto de instruções distinto, a ser interpretado pelo hipervisor. A interpretação de instruções impõe um custo de execução significativo ao sistema convidado. O emulador *QEMU*, que provê às aplicações um processador virtual *Intel Pentium II*, é um exemplo desta abordagem.

A virtualização também pode ser classificada como [Rosenblum 2004]:

- *Virtualização do hardware*: a virtualização exporta o sistema físico como hardware abstrato (semelhante ao sistema original). Neste modelo, qualquer software escrito para a arquitetura nativa (*x*86, por exemplo) irá funcionar no sistema convidado. Este foi o modelo adotado na década de 60 para o VM/370 nos mainframes IBM e é a tecnologia de virtualização utilizado pelo *VMware* na plataforma *x*86.
- *Virtualização do sistema operacional*: a virtualização exporta um sistema operacional como abstração de um sistema específico. A máquina virtual executa aplicações – ou um conjunto de aplicações – de um sistema operacional específico. O *FreeBSD Jail* ou o *User-Mode Linux* são exemplos desta tecnologia.
- *Virtualização de linguagens de programação*: a camada de virtualização cria uma aplicação no topo do sistema operacional. Na prática, as máquinas virtuais nesta categoria são desenvolvidas para computadores fictícios projetados para uma finalidade específica. A camada exporta uma abstração para a execução de programas escritos para esta virtualização. *Java JVM* e *Smalltalk* são exemplos deste tipo de máquina virtual.

O trabalho [Nanda and Chiueh 2005] ainda classifica a virtualização como:

- *Abstração da ISA* (*Instruction Set Architecture*): a virtualização é implementada com o uso da emulação completa da ISA. O emulador executa as instruções do sistema convidado (a máquina virtual é obtida por meio da emulação) utilizando a tradução das instruções para o sistema nativo. Esta arquitetura é robusta e simples para implementação, mas a perda de desempenho é significativa. *Bochs*, *Crusoe* e *QEMU* são exemplos da mesma.
- *Hardware Abstraction Layer* (HAL): o hipervisor simula uma arquitetura completa para o sistema convidado. Desta forma, o sistema convidado acredita estar executando sobre um sistema completo de hardware. *VMware*, *Virtual PC*, *Denali* e *Xen* são exemplos desta arquitetura.
- *OS Level* (sistema operacional): este nível de virtualização é obtido utilizando uma chamada de sistema (*system call*) específica. O principal benefício da virtualização neste nível é criar uma camada para obter o isolamento de processos, cada sistema é virtualizado com seu próprio endereço IP e outros recursos de hardware (embora limitados). A virtualização ocorre a partir de um diretório ou sistema de arquivos separadado e previamente preparado para este fim. O *FreeBSD Jail* é um exemplo desta arquitetura.

- *Nível de aplicação* ou virtualização de linguagens de programação: a virtualização é obtida por meio da abstração de uma *camada de execução*. Uma aplicação utiliza esta camada para executar as instruções do programa. Esta solução garante que uma aplicação possa ser executada em qualquer plataforma de software ou hardware, pois a camada é abstraída de forma idêntica em todas as plataformas. Todavia, torna-se necessária uma máquina virtual específica para cada plataforma. *Java JVM*, *Microsoft .NET CLI* e *Parrot* são exemplos desta arquitetura.

- *User level library interface* (biblioteca de interface para usuário): vários sistemas e aplicações são escritos utilizando um conjunto de APIs fornecidos pelo sistema (aplicações sob o sistema Windows são os exemplos mais populares), exportados para o nível do usuário por meio de bibliotecas. A *virtualização* neste nível é obtida com a abstração do topo do sistema operacional, para que as aplicações possam executar em outra plataforma. O *Wine* é um exemplo deste tipo de arquitetura.

4.1.4.1. Máquinas Virtuais de Aplicação

Uma máquina virtual de aplicação (ou *Process Virtual Machine*) suporta a execução de um processo ou aplicação individual. Ela é criada sob demanda, no momento do lançamento da aplicação convidada, e destruída quando a aplicação finaliza sua execução. O conjunto *hipervisor + aplicação* é visto então como um único processo dentro do sistema operacional subjacente, submetido às mesmas condições e restrições que os demais processos nativos.

Os hipervisores que implementam máquinas virtuais de aplicação normalmente permitem a interação entre a aplicação convidada e as demais aplicações do sistema, através dos mecanismos usuais de comunicação e coordenação entre processos, como mensagens, *pipes* e semáforos. Além disso, também permitem o acesso normal ao sistema de arquivos e outros recursos locais do sistema. Ao criar a máquina virtual para uma aplicação, o hipervisor pode implementar a mesma interface de hardware (ISA) da máquina real subjacente, ou implementar uma interface distinta. Quando a interface da máquina real é preservada, boa parte das instruções do processo convidado podem ser executadas diretamente, com exceção das instruções sensíveis, que devem ser interpretadas pelo hipervisor. Os exemplos mais comuns de máquinas virtuais de aplicação que preservam a interface ISA real são os *sistemas operacionais multi-tarefas*, os *tradutores dinâmicos* e alguns *depuradores de memória*:

- *Sistemas operacionais multi-tarefas*: os sistemas operacionais que suportam vários processos simultâneos também podem ser vistos como ambientes de máquinas virtuais. Em um sistema multi-tarefas, cada processo recebe um *processador virtual* (simulado através de fatias de tempo do processador real), uma *memória virtual* (através do espaço de endereços mapeado para aquele processo) e *recursos físicos* (acessíveis através de chamadas de sistema). Este ambiente de máquinas virtuais é tão antigo e tão presente em nosso cotidiano que costumamos ignorá-lo. No entanto, ele simplifica muito a tarefa dos programadores, que não precisam se preocupar com a gestão do compartilhamento desses recursos entre os processos.

- *Tradutores dinâmicos*: um tradutor dinâmico consiste em um hipervisor que analisa e otimiza um código executável, para tornar sua execução mais rápida e eficiente. A otimização não muda o conjunto de instruções da máquina real usado pelo código, apenas reorganiza as instruções de forma a acelerar sua execução. Por ser dinâmica, a otimização do código é feita durante a carga do processo na memória ou durante a execução de suas instruções, de forma transparente. O artigo [Duesterwald 2005] apresenta uma descrição detalhada desse tipo de abordagem.

- *Depuradores de memória*: alguns sistemas de depuração de erros de acesso à memória, como o sistema *Valgrind* [Seward and Nethercote 2005], executam o processo sob depuração em uma máquina virtual. Todas as instruções do programa que manipulam acessos à memória são executadas de forma controlada, a fim de encontrar possíveis erros. Ao depurar um programa, o sistema *Valgrind* inicialmente traduz seu código em um conjunto de instruções interno, manipula esse código para inserir operações de verificação de acessos à memória e traduz o código modificado de volta ao conjunto de instruções da máquina real, para em seguida executá-lo e verificar os acessos à memória realizados.

As máquinas virtuais de aplicação mais populares hoje são aquelas em que a interface binária de aplicação (ABI) requerida pela aplicação é diferente da oferecida pela máquina real. Como a ABI é composta pelas chamadas do sistema operacional e as instruções de máquina disponíveis à aplicação (*user ISA*), as diferenças podem ocorrer em ambos. Nos dois casos, o hipervisor terá de realizar traduções dinâmicas (durante a execução) das ações requeridas pela aplicação em suas equivalentes na máquina real (como visto, um hipervisor com essa função é denominado *tradutor dinâmico*).

Caso as diferenças de interface entre aplicação e máquina real se restrinjam às chamadas do sistema operacional, o hipervisor precisa apenas mapear as chamadas de sistema usadas pela aplicação sobre as chamadas oferecidas pelo sistema operacional da máquina real. Essa é a abordagem usada, por exemplo, pelo ambiente *Wine*, que permite executar aplicações Windows em plataformas Unix. As chamadas de sistema Windows emitidas pela aplicação em execução são interceptadas e transformadas em chamadas Unix, de forma dinâmica e transparente (figura 4.12).

Entretanto, muitas vezes a interface ISA utilizada pela aplicação não corresponde a nenhum hardware existente, mas a um hardware simplificado que representa uma máquina abstrata. Um exemplo típico dessa situação ocorre na linguagem Java: um programa escrito em Java, ao ser compilado, gera um código binário específico para uma máquina abstrata denominada *máquina virtual Java* (*JVM – Java Virtual Machine*). A linguagem de máquina executada pela máquina virtual Java é denominada *bytecode* Java, e não corresponde a instruções de nenhum processador real. A máquina virtual deve então interpretar todas as operações do *bytecode*, utilizando as instruções da máquina real subjacente para executá-las.

A vantagem mais significativa da abordagem adotada por Java é a *portabilidade* do código executável: para que uma aplicação Java possa executar sobre uma determinada plataforma, basta que a máquina virtual Java esteja disponível ali (na forma de um suporte de execução denominado JRE – *Java Runtime Environment*). Assim, a portabilidade dos

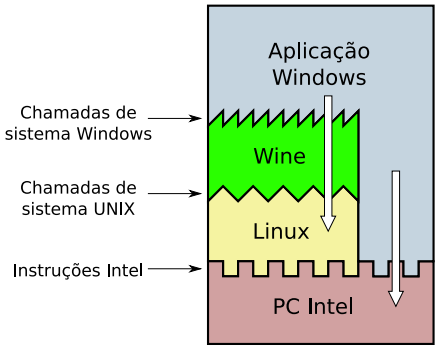

Figura 4.12. Funcionamento do emulador Wine.

programas Java depende unicamente da portabilidade da própria máquina virtual Java. O suporte de execução Java pode estar associado a um navegador Web, o que permite que o código Java seja associado a páginas Web, na forma de pequenas aplicações denominadas *applets*, que são trazidas junto com os demais componentes de página Web e executam localmente no navegador[1].

Em termos de desempenho, um programa compilado para uma máquina virtual executa mais lentamente que seu equivalente compilado sobre uma máquina real, devido ao custo de interpretação do *bytecode*. Todavia, essa abordagem oferece melhor desempenho que linguagens puramente interpretadas. Além disso, técnicas de otimização como a compilação *Just-in-Time* (JIT), na qual blocos de instruções repetidos freqüentemente são compilados pelo hipervisor e armazenados em cache, permitem obter ganhos de desempenho significativos em aplicações executando sobre máquinas virtuais.

4.1.4.2. Máquinas Virtuais de Sistema

Uma máquina virtual de sistema suporta um ou mais sistemas operacionais convidados, com suas respectivas aplicações, que executam de forma isolada e independente. Cada sistema operacional convidado tem a ilusão de executar sozinho sobre uma plataforma de hardware própria.

Em um ambiente virtual, os sistemas operacionais convidados são fortemente isolados uns dos outros, e normalmente só podem interagir através dos mecanismos de rede, como se estivessem em máquinas fisicamente separadas. Todavia, alguns sistemas de máquinas virtuais permitem o compartilhamento controlado de certos recursos. Por exemplo, os sistemas *VMware Workstation* e *VirtualBox* permitem a definição de diretórios compartilhados no sistema de arquivos real, que podem ser acessados pelas máquinas virtuais.

[1]Neste caso, a propriedade de isolamento das máquinas virtuais é aplicada aos *applets* Java, para evitar que códigos maliciosos trazidos da Internet executem ações prejudiciais no sistema local; este mecanismo é conhecido como *Java Sandbox*.

O hipervisor de sistema fornece aos sistemas operacionais convidados uma interface de sistema ISA virtual, que pode ser idêntica ao hardware real, ou distinta. Além disso, ele virtualiza o acesso aos recursos, para que cada sistema operacional convidado tenha um conjunto de recursos virtuais próprio, construído a partir dos recursos físicos existentes na máquina real. Assim, cada máquina virtual terá sua própria interface de rede, seu próprio disco, sua própria memória RAM, etc.

As máquinas virtuais de sistema constituem a primeira abordagem usada para a construção de hipervisores, desenvolvida na década de 1960 e formalizada por Popek e Goldberg [Popek and Goldberg 1974]. Naquela época, a tendência de desenvolvimento de sistemas computacionais buscava fornecer a cada usuário uma máquina virtual com seus recursos virtuais próprios, sobre a qual o usuário executava um sistema operacional mono-tarefa e suas aplicações. Assim, o compartilhamento de recursos não era responsabilidade do sistema operacional convidado, mas do hipervisor subjacente. No entanto, ao longo dos anos 70, como o desenvolvimento de sistemas operacionais multi-tarefas eficientes e seguros como MULTICS e UNIX, as máquinas virtuais de sistema perderam gradativamente seu interesse. Somente no final dos anos 90, com o aumento do poder de processamento dos micro-processadores e o surgimento de novas possibilidades de aplicação, as máquinas virtuais de sistema foram "redescobertas".

Existem vários tipos de ambientes de máquinas virtuais de sistema, que podem ser classificados quanto à sua arquitetura e quanto ao grau de virtualização do hardware. No que diz respeito à arquitetura, existem basicamente dois tipos de hipervisores de sistema, apresentados na figura 4.13:

- *Hipervisores nativos* (ou *de tipo I*): nesta categoria, o hipervisor executa diretamente sobre o hardware da máquina real, sem um sistema operacional subjacente. A função do hipervisor é multiplexar os recursos de hardware (memória, discos, interfaces de rede, etc) de forma que cada máquina virtual veja um conjunto de recursos próprio e independente. Assim, cada máquina virtual se comporta como uma máquina física completa que pode executar o seu próprio sistema operacional. Esta é a forma mais antiga de virtualização, encontrada nos sistemas computacionais de grande porte dos anos 1960-70. Alguns exemplos de sistemas que empregam esta abordagem são o *IBM OS/370*, o *VMware ESX Server* e o ambiente *Xen*.

- *Hipervisores convidados* (ou *de tipo II*): nesta categoria, o hipervisor executa como um processo normal sobre um sistema operacional nativo subjacente. O hipervisor utiliza os recursos oferecidos pelo sistema operacional nativo para oferecer recursos virtuais ao sistema operacional convidado que executa sobre ele. Normalmente, um hipervisor convidado suporta apenas uma máquina virtual com uma instância de sistema operacional convidado. Caso mais máquinas sejam necessárias, mais hipervisores devem ser lançados. Exemplos de sistemas que adotam esta estrutura incluem o *VMware Workstation*, o *QEMU* e o *VirtualBox*.

Pode-se afirmar que os hipervisores convidados são mais flexíveis que os hipervisores nativos, pois podem ser facilmente instalados/removidos em máquinas com sistemas

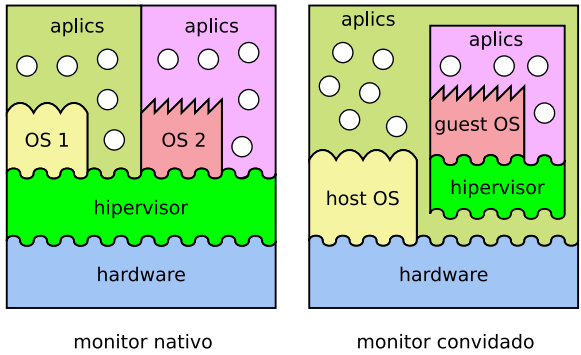

Figura 4.13. Arquiteturas de máquinas virtuais de sistema.

operacionais previamente instalados. Por outro lado, um hipervisor nativo tem melhor desempenho que um hipervisor convidado, pois este último tem de usar os recursos oferecidos pelo sistema operacional subjacente, enquanto o primeiro pode acessar diretamente o hardware real.

No que diz respeito ao nível de virtualização oferecido pelo hipervisor, os ambientes de máquinas virtuais podem ser classificados em duas categorias, conforme apresentado na figura 4.14:

- *Virtualização de recursos*: nesta categoria, a interface ISA de usuário é mantida, apenas as instruções privilegiadas e os recursos (discos, etc) são virtualizados. Dessa forma, o sistema operacional convidado e as aplicações convidadas *vêem* o processador real. Como as quantidades de instruções a virtualizar são pequenas, o desempenho do sistema convidado é próximo daquele obtido, se ele estivesse executando diretamente sobre o hardware real. Os sistemas *VMware Workstation*, *VirtualBox* e *MS VirtualPC* implementam esta estratégia.

- *Virtualização completa*: nesta categoria, toda a interface do hardware é virtualizada, incluindo todas as instruções do processador e os dispositivos de hardware. Isso permite oferecer ao sistema operacional convidado uma interface de hardware distinta daquela fornecida pela máquina real subjacente. O custo de virtualização neste caso é bem maior que na virtualização parcial (de recursos), mas esta abordagem permite executar sistemas operacionais em plataformas distintas daquela para a qual foram inicialmente projetados. Exemplos típicos desta abordagem são os sistemas de máquinas virtuais *QEMU*, que oferece um processador *Intel Pentium II* ao sistema convidado, o *MS VirtualPC for MAC*, que permite executar o sistema Windows sobre uma plataforma de hardware *PowerPC*, e o sistema *Hercules*, que emula um computador *IBM System/390* sobre um PC convencional de plataforma Intel.

Uma categoria especial de hipervisor nativo com virtualização completa consiste nos **hipervisores embutidos no hardware** (*codesigned hypervisors*). Um hipervisor em-

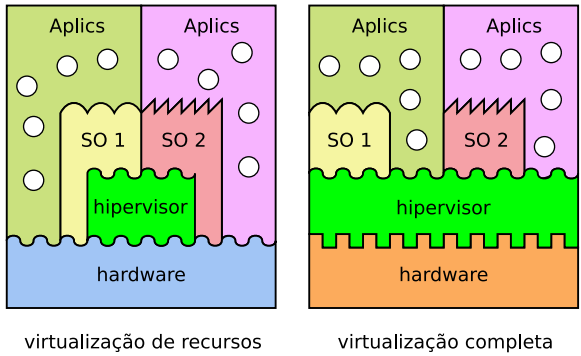

Figura 4.14. Níveis de virtualização: virtualização de recursos e virtualização completa.

butido é visto como parte integrante do hardware da máquina real, e implementa a interface de sistema (ISA) vista pelos sistemas operacionais e aplicações daquela plataforma. O conjunto de instruções do processador real somente está acessível ao hipervisor, que reside em uma área de memória separada da memória principal e usa técnicas de tradução dinâmica para executar as instruções dos sistemas convidados. Um exemplo típico desse tipo de sistema é o processador *Transmeta Crusoe/Efficeon*, que aceita instruções no padrão *Intel 32 bits* e internamente as converte em um conjunto de instruções VLIW (*Very Large Instruction Word*). Como o hipervisor desse processador pode ser reprogramado para criar novas instruções ou modificar as instruções existentes, ele acabou sendo denominado *Code Morphing Software* (figura 4.15).

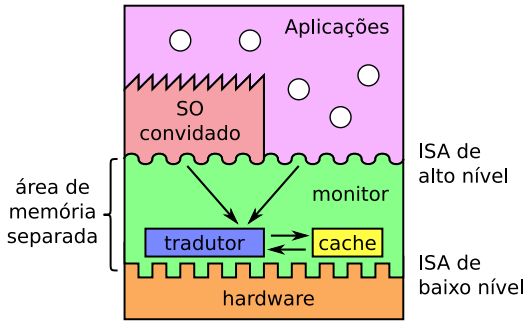

Figura 4.15. Hipervisor embutido no hardware.

A figura 4.16 traz uma classificação dos tipos de máquinas virtuais de sistema, de acordo com os critérios de arquitetura do hipervisor e grau de virtualização oferecido aos sistemas convidados.

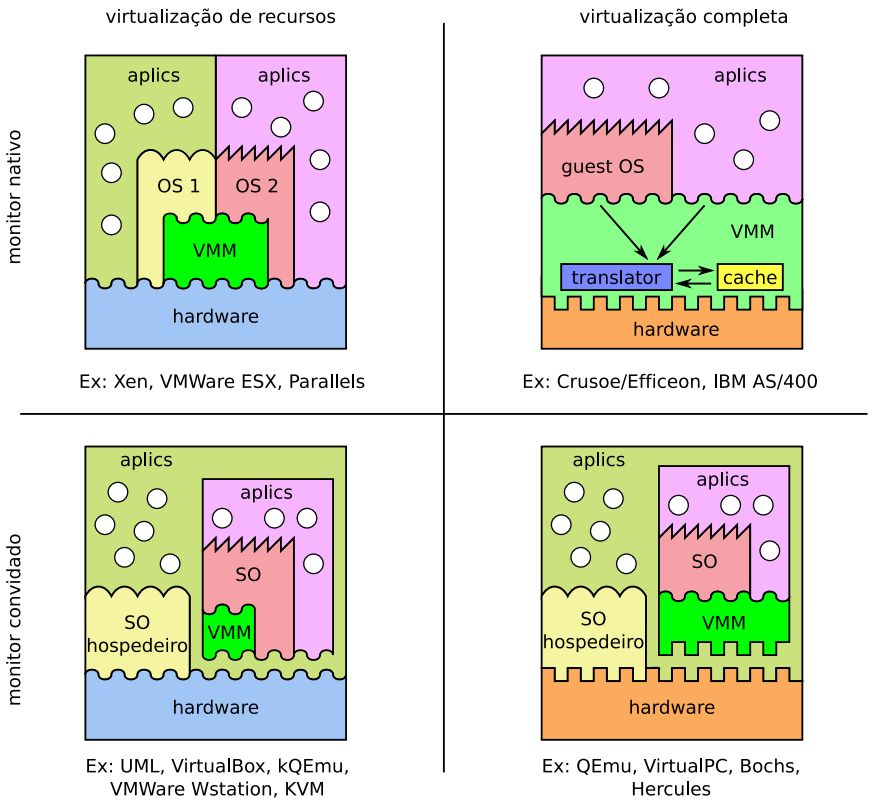

Figura 4.16. Classificação de máquinas virtuais de sistema.

4.1.5. Estratégias de Virtualização

A construção de hipervisores implica na definição de algumas estratégias para a virtualização. As estratégias mais utilizadas atualmente são a virtualização total (*full virtualization*), normalmente associada à tradução dinâmica (*dynamic translation*), e a paravirtualização (*paravirtualization*). Além disso, algumas técnicas complementares são usadas para melhorar o desempenho dos sistemas de máquinas virtuais. Essas técnicas são discutidas nesta seção.

4.1.5.1. Virtualização Total

A virtualização total reconstrói um ambiente virtual no qual o hardware fornecido aos sistemas convidados corresponde a um sistema real existente, Toda a interface de acesso ao hardware é virtualizada, incluindo todas as instruções do processador e os dispositivos de hardware. Desta forma, os sistemas convidados não precisam sofrer nenhum tipo de alteração ou ajuste para executar no ambiente virtual. Esta é a abordagem usada na mai-

oria dos hipervisores de sistema clássicos, como *QEmu* e *VMWare*. Sua maior vantagem consiste em permitir que sistemas operacionais convencionais executem como convidados sem necessidade de modificações. Por outro lado, o sistema convidado executa mais lentamente, uma vez que todos os acessos ao hardware são intermediados pelo hipervisor. Além disso, o hipervisor terá de interceptar e emular todas as instruções sensíveis executadas pelos sistemas convidados, o que tem um custo elevado em plataformas de hardware sem suporte adequado à virtualização.

4.1.5.2. Tradução dinâmica

Uma técnica freqüentemente utilizada na construção de máquinas virtuais é a *tradução dinâmica* (*dynamic translation*) ou recompilação dinâmica (*dynamic recompilation*) de partes do código binário dos sistemas convidados e suas aplicações. Nesta técnica, o hipervisor analisa, reorganiza e traduz as seqüências de instruções emitidas pelo sistema convidado em novas seqüências de instruções, à medida em que a execução do sistema convidado avança.

A tradução binária dinâmica pode ter vários objetivos: (a) adaptar as instruções geradas pelo sistema convidado à interface ISA do sistema real, caso não sejam idênticas; (b) detectar e tratar instruções sensíveis não-privilegiadas (que não geram interrupções ao serem invocadas pelo sistema convidado); ou (c) analisar, reorganizar e otimizar as seqüências de instruções geradas pelo sistema convidado, de forma a melhorar o desempenho de sua execução. Neste último caso, os blocos de instruções muito freqüentes podem ter suas traduções mantidas em cache, para melhorar ainda mais o desempenho.

A tradução dinâmica é usada em vários tipos de hipervisores. Uma aplicação típica é a construção da máquina virtual Java, onde recebe o nome de JIT – *Just-in-Time Bytecode Compiler*. Outro uso corrente é a construção de hipervisores para plataformas sem suporte adequado à virtualização, como os processadores Intel/AMD 32 bits. Neste caso, o código convidado a ser executado é analisado em busca de instruções sensíveis, que são substituídas por chamadas a rotinas apropriadas dentro do supervisor.

No contexto de virtualização, a recompilação dinâmica é composta basicamente dos seguintes passos [Ung and Cifuentes 2006]:

1. *Desmontagem* (*disassembling*): o fluxo de bytes do código convidado a executar é decomposto em blocos de instruções. Cada bloco é normalmente composto de uma seqüência de instruções de tamanho variável, terminando com uma instrução de controle de fluxo de execução;

2. *Geração de código intermediário*: cada bloco de instruções tem sua semântica descrita através de uma representação independente de máquina;

3. *Otimização*: a descrição em alto nível do bloco de instruções é analisada para aplicar eventuais otimizações; como este processo é realizado durante a execução, normalmente somente otimizações com baixo custo computacional são aplicáveis;

4. *Codificação*: o bloco de instruções otimizado é traduzido para instruções da máquina física, que podem ser diferentes das instruções do código original;

5. *Caching*: blocos de instruções com execução muito freqüente têm sua tradução armazenada em cache, para evitar ter de traduzi-los e otimizá-los novamente;

6. *Execução*: o bloco de instruções traduzido é finalmente executado nativamente pelo processador da máquina real.

Esse processo pode ser simplificado caso as instruções de máquina do código convidado sejam as mesmas da máquina real subjacente, o que torna desnecessário traduzir os blocos de instruções em uma representação independente de máquina.

4.1.5.3. Paravirtualização

Em meados dos anos 2000, alguns pesquisadores investigaram a possibilidade de modificar a interface entre o hipervisor e os sistemas convidados, oferecendo a estes um hardware virtual que é similar, mas não idêntico ao hardware real. Essa abordagem, denominada *paravirtualização*, permite um melhor acoplamento entre os sistemas convidados e o hipervisor, o que leva a um desempenho significativamente melhor das máquinas virtuais. As modificações na interface de sistema do hardware virtual (*system ISA*) exigem uma adaptação dos sistemas operacionais convidados, para que estes possam executar sobre a plataforma virtual. Todavia, a interface de usuário (*user ISA*) do hardware é preservada, permitindo que as aplicações convidadas executem sem necessidade de modificações. Esse conceito é ilustrado na figura 4.17, adaptada [Ferre et al. 2006].

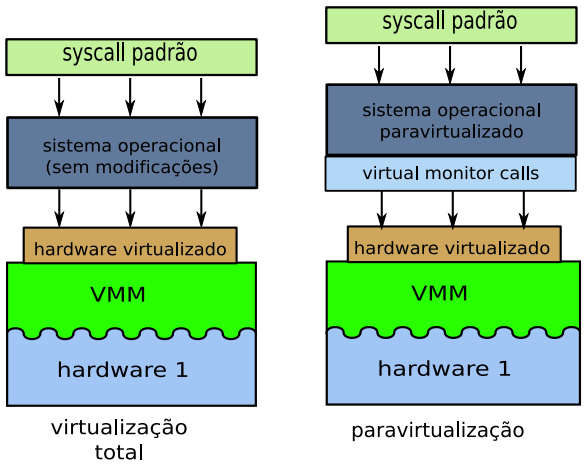

Figura 4.17. Relação entre a virtualização total e a paravirtualização.

Os primeiros ambientes a adotar a paravirtualização foram o *Denali* [Whitaker et al. 2002] e o *Xen* [Barham et al. 2003]. O *Denali* é um ambiente experimental de paravirtualização construído na Universidade de Washington, que pode suportar dezenas de milhares de máquinas virtuais sobre um computador *x86* convencional.

O projeto *Denali* não se preocupa em suportar sistemas operacionais comerciais, sendo voltado à execução maciça de minúsculas máquinas virtuais para serviços de rede. Já o ambiente de máquinas virtuais *Xen*, apresentado com mais detalhes na seção 4.3.3, permite executar sistemas operacionais convencionais como Linux e Windows, modificados para executar sobre seu hipervisor.

Embora exija que o sistema convidado seja adaptado ao hipervisor, o que diminui sua portabilidade, a paravirtualização permite que o sistema convidado acesse alguns recursos do hardware diretamente, sem a intermediação ativa do hipervisor. Nesses casos, o acesso ao hardware é apenas monitorado pelo hipervisor, que informa ao sistema convidado seus limites, como as áreas de memória e de disco disponíveis. O acesso aos demais dispositivos, como mouse e teclado, também é direto: o hipervisor apenas gerencia a ordem de acessos, no caso de múltiplos sistemas convidados em execução simultânea.

A diferença entre a virtualização total e a paravirtualização pode ser melhor compreendida observando-se a virtualização do acesso à memória. Na virtualização total, o hipervisor reserva um espaço de memória separado para cada sistema convidado; no entanto, todos os sistemas convidados "vêem" suas respectivas áreas de memória iniciando no endereço 0000_H. A figura 4.18 ilustra essa situação. Dessa forma, cada vez que um sistema convidado acessa a memória, o hipervisor traduz os endereços gerados por ele para as posições reais da área de memória daquele sistema convidado. Por outro lado, na paravirtualização, o hipervisor informa ao sistema operacional convidado as áreas de memória que este pode utilizar. Dessa forma, o sistema convidado pode acessar e gerenciar diretamente a memória reservada a ele, sem a interferência direta do hipervisor.

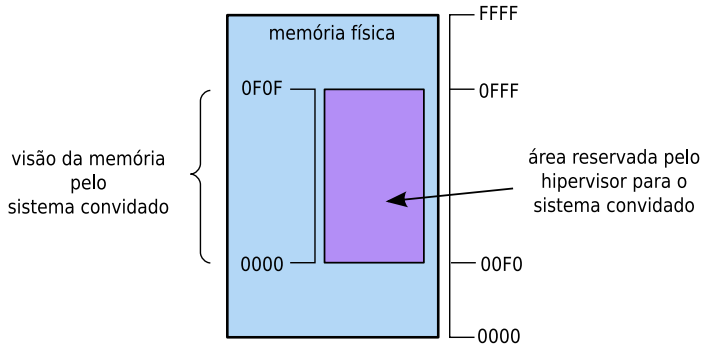

Figura 4.18. Visão da memória por um sistema convidado.

Apesar de exigir modificações nos sistemas operacionais convidados, a paravirtualização tem tido sucesso, por conta do desempenho obtido nos sistemas virtualizados, além de simplificar a interface de baixo nível dos sistemas convidados.

4.1.5.4. Melhoria de desempenho

De acordo com os princípios de Goldberg e Popek, o hipervisor deve permitir que a máquina virtual execute diretamente sobre o hardware sempre que possível, para não prejudicar o desempenho dos sistemas convidados. O hipervisor deve retomar o controle do processador somente quando a máquina virtual tentar executar operações que possam afetar o correto funcionamento do sistema, o conjunto de operações de outras máquinas virtuais ou do próprio hardware. O hipervisor deve então simular com segurança a operação solicitada e devolver o controle à máquina virtual.

Na prática, os hipervisores nativos e convidados raramente são usados em sua forma conceitual. Várias otimizações são inseridas nas arquiteturas apresentadas, com o objetivo principal de melhorar o desempenho das aplicações nos sistemas convidados. Como os pontos cruciais do desempenho dos sistemas de máquinas virtuais são as operações de entrada/saída, as principais otimizações utilizadas em sistemas de produção dizem respeito a essas operações. Quatro formas de otimização são usuais:

- Em hipervisores nativos (figura 4.19):

  1. O sistema convidado (*guest system*) acessa diretamente o hardware. Essa forma de acesso é implementada por modificações no núcleo do sistema convidado e no hipervisor. Essa otimização é implementada, por exemplo, no subsistema de gerência de memória do ambiente Xen [Barham et al. 2003].

- Em hipervisores convidados (figura 4.19):

  1. O sistema convidado (*guest system*) acessa diretamente o sistema nativo (*host system*). Essa otimização é implementada pelo hipervisor, oferecendo partes da API do sistema nativo ao sistema convidado. Um exemplo dessa otimização é a implementação do sistema de arquivos no *VMware* [VMware 2000]: em vez de reconstruir integralmente o sistema de arquivos sobre um dispositivo virtual provido pelo hipervisor, o sistema convidado faz uso da implementação de sistema de arquivos existente no sistema nativo.
  2. O sistema convidado (*guest system*) acessa diretamente o hardware. Essa otimização é implementada parcialmente pelo hipervisor e parcialmente pelo sistema nativo, pelo uso de um *device driver* específico. Um exemplo típico dessa otimização é o acesso direto a dispositivos físicos como leitor de CDs, hardware gráfico e interface de rede provida pelo sistema *VMware* aos sistemas operacionais convidados [VMware 2000].
  3. O hipervisor acessa diretamente o hardware. Neste caso, um *device driver* específico é instalado no sistema nativo, oferecendo ao hipervisor uma interface de baixo nível para acesso ao hardware subjacente. Essa abordagem, ilustrada na figura 4.20, é usada pelo sistema *VMware* [VMware 2000].

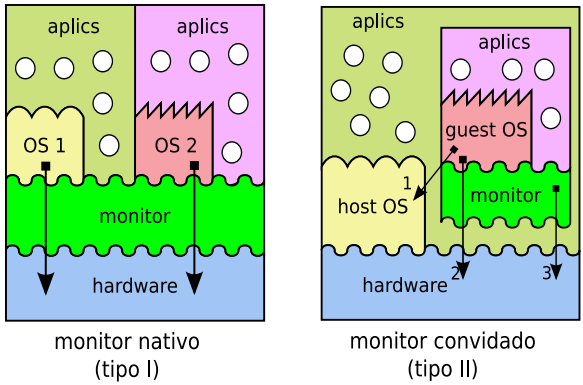

Figura 4.19. Otimizações em sistemas de máquinas virtuais.

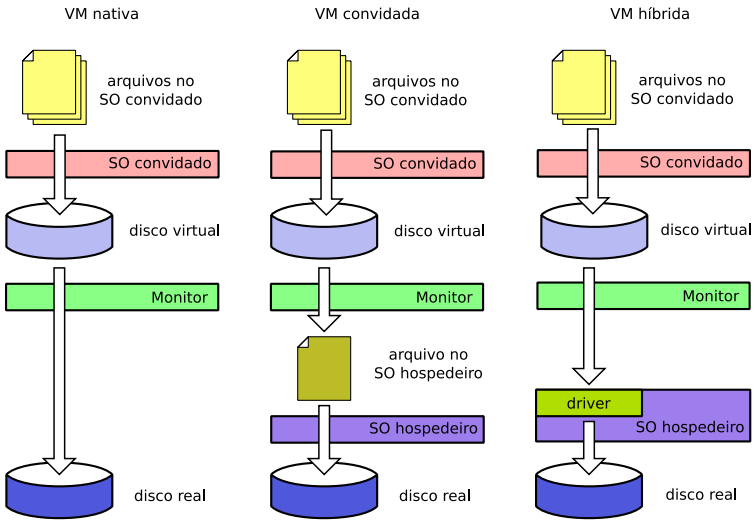

Figura 4.20. Desempenho de hipervisores nativos e convidados.

4.2. Virtualização e segurança

A evolução da tecnologia de máquinas virtuais tem permitido sua ampla adoção em sistemas de produção. Vários aspectos da construção de hipervisores, em sua maioria relacionados com o desempenho de execução dos sistemas convidados, foram resolvidos nos últimos anos [Rosenblum and Garfinkel 2005]. Atualmente, a principal utilização de máquinas virtuais no meio corporativo tem sido a consolidação de servidores, buscando a redução de custos em hardware, software e gerência do parque tecnológico [Newman et al. 2005, Ferre et al. 2006]. No entanto, vários trabalhos de pesquisa e desenvolvimento nos últimos anos comprovaram a eficácia da utilização de máquinas vir-

tuais no campo da segurança de sistemas. Esta seção discute como as propriedades dos hipervisores (apresentadas na seção 4.1.3.2) podem ser aplicadas na segurança de sistemas e discute alguns usos típicos de máquinas virtuais nesse contexto.

4.2.1. Aplicações da virtualização em segurança

Conforme [Krause and Tipton 1999], são três os princípios básicos para garantir a segurança da informação:

- *Confidencialidade*: a informação somente está visível a sujeitos (usuários e/ou processos) explicitamente autorizados;
- *Disponibilidade*: a informação deve estar prontamente disponível sempre que for necessária;
- *Integridade*: a informação somente pode ser modificada por sujeitos explicitamente autorizados e de formas claramente definidas.

Além destes, outros critérios devem ser respeitados para um sistema ser considerado seguro [Sêmola 2003]:

- *Autenticidade*: garante que a informação ou o usuário da mesma é autêntico, ou seja, garante que a entidade envolvida é quem afirma ser;
- *Não-repúdio*: não é possível negar a existência ou autoria de uma operação que criou, modificou ou destruiu uma informação;
- *Auditoria*: implica no registro das ações realizadas no sistema, identificando os sujeitos e recursos envolvidos, as operações realizadas, seus horários, locais e outros dados relevantes.

Algumas das propriedades conceituais da virtualização discutidas na seção 4.1.3.2 podem ser úteis para o atendimento desses critérios de segurança:

- *Isolamento*: ao manter os ambientes virtuais isolados entre si e do sistema real subjacente, o hipervisor provê a confidencialidade de dados entre os sistemas convidados. Adicionalmente, como os dados presentes em uma máquina virtual só podem ser acessados pelas respectivas aplicações convidadas, sua integridade é preservada. Além disso, o isolamento permite a contenção de erros de software acidentais ou intencionais no âmbito da máquina virtual [LeVasseur et al. 2004, Tan et al. 2007], o que permite melhorar a disponibilidade dos sistemas;
- *Controle de recursos*: Como o hipervisor intermedeia os acessos do sistema convidado ao hardware, é possível implementar mecanismos para verificar a consistência desses acessos e de seus resultados, aumentando a integridade do sistema convidado; da mesma forma, é possível acompanhar e registrar as atividades do sistema convidado, para fins de auditoria [Dunlap et al. 2002];

- *Inspeção*: a visão privilegiada do hipervisor sobre o estado interno do sistema convidado permite extrair informações deste para o sistema hospedeiro, permitindo implementar externamente mecanismos de verificação de integridade do ambiente convidado, como antivírus e detectores de intrusão [Laureano et al. 2007]; além disso, a capacidade de inspeção do sistema convidado, aliada ao isolamento provido pelo hipervisor, torna as máquinas virtuais excelentes "balões de ensaio" para o estudo de aplicações maliciosas como vírus e *trojans*;
- *Encapsulamento*: a possibilidade de salvar/restaurar o estado do sistema convidado torna viável a implementação de mecanismos de *rollback* úteis no caso de quebra da integridade do sistema convidado; da mesma forma, a migração de máquinas virtuais é uma solução viável para o problema da disponibilidade [Fu and Xu 2005];

Nos últimos anos, muitos projetos de pesquisa estudaram a aplicação das propriedades das máquinas virtuais na construção de sistemas computacionais seguros e confiáveis. Algumas dessas pesquisas visam proteger e aumentar a disponibilidade de sistemas e serviços, outras visam estudar o comportamento de aplicações maliciosas, mas também há estudos visando criar aplicações maliciosas que tirem proveito da virtualização, ao menos como prova de conceito. Esses trabalhos abordam diversas aplicações, como o confinamento de serviços, a detecção de intrusão, a análise de *malwares*, a construção de *honeypots* e *rootkits*, técnicas de contingenciamento e de tolerância a falhas. Algumas dessas pesquisas serão discutidas nas próximas seções.

4.2.2. Confinamento de aplicações

Em um sistema operacional, cada processo tem acesso a um conjunto de recursos para funcionar. Esse conjunto é definido através de regras de controle de acesso impostas pelo núcleo do sistema. Além disso, a lógica interna do processo também pode limitar os recursos acessíveis a seu respectivo usuário. Por exemplo, um servidor Web pode acessar os arquivos autorizados pelas permissões do sistema de arquivos, mas também possui regras internas para limitar os arquivos e diretórios acessíveis aos clientes. Todavia, serviços mal configurados, erros nas regras de controle de acesso, vulnerabilidades no serviço ou no sistema operacional podem expor informações indevidamente, comprometendo a confidencialidade e integridade do sistema. Esse problema pode ser atenuado de diversas formas:

- Monitorar a aplicação usando IDS, antivírus etc. Esta solução pode requerer um trabalho de configuração significativo; além disso, essas ferramentas também são processos, consumindo recursos e sendo passíveis de falhas ou subversão;
- Designar um equipamento separado para a aplicação sensível executar isoladamente, o que pode custar caro;
- Colocar a aplicação para executar dentro de uma máquina virtual, isolando-a do restante do sistema hospedeiro.

Em princípio, qualquer hipervisor convencional pode ser usado para confinar uma aplicação. Todavia, neste caso específico, a principal motivação para o uso de máquinas virtuais é a propriedade de isolamento (seção 4.1.3.2). Como essa propriedade não

implica necessariamente a virtualização do processador ou dos demais recursos de hardware, técnicas simplificadas e com baixo custo computacional foram concebidas para implementá-la. Essas técnicas são conhecidas como *servidores virtuais*.

Em um servidor virtual, a interface binária (ABI), composta pelas chamadas de sistema e instruções do processador, é totalmente preservada. O espaço de usuário do sistema operacional é dividido em áreas isoladas denominadas *domínios* virtuais. Cada domínio virtual recebe uma parcela dos recursos do sistema, como memória, tempo de processador e espaço em disco. Alguns recursos do sistema real podem ser virtualizados, como as interfaces de rede: cada domínio tem sua própria interface virtual e seu próprio endereço de rede. Em algumas implementações, cada domínio virtual pode impor seu próprio espaço de nomes: assim, pode-se ter um usuário `pedro` no domínio d_3 e outro usuário `pedro` no domínio d_7, sem conflitos. A distinção de espaços de nomes pode se estender a outros recursos do sistema: identificadores de processos, semáforos, árvores de diretórios, etc.

Os processos confinados em um determinado domínio podem interagir entre si, criar novos processos e usar os recursos presentes naquele domínio, respeitando as regras de controle de acesso associadas a esses recursos. Todavia, processos em um domínio não podem interagir com processos em outro domínio, não podem mudar de domínio, criar processos em outros domínios, nem usar recursos de outros domínios. Para cada domínio, os demais domínios são máquinas distintas. Normalmente é definido um domínio d_0, ou *domínio de gerência*, cujos processos têm acesso aos demais domínios. A figura 4.21 mostra a estrutura típica de um sistema de servidores virtuais. Nela, pode-se observar que um processo pode migrar de d_0 para d_1, mas que processos em d_1 não podem migrar para outros domínios. A comunicação entre processos confinados em domínios distintos (d_2 e d_3) também é proibida.

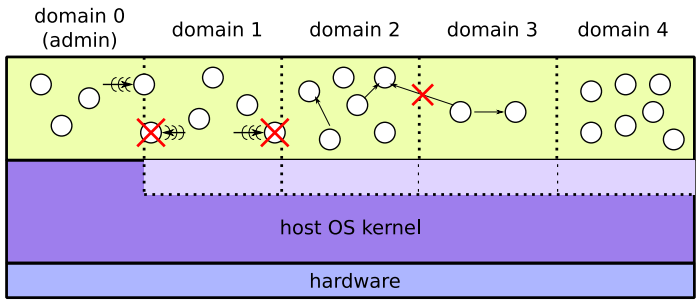

Figura 4.21. Servidores virtuais.

Há várias implementações disponíveis de mecanismos de confinamento de processos. A técnica mais antiga é realizada através da chamada de sistema `chroot`. O Sistema *FreeBSD* implementa o ambiente *Jails* [Kamp and Watson 2000] para este fim. Existem implementações similares em outros sistemas operacionais, como as *Zones* do sistema *Solaris* e os sistemas *Virtuozzo/OpenVZ* e *Vservers/FreeVPS*, para Linux.

4.2.3. Detecção de intrusão

Sistemas de detecção de intrusão (IDS – *Intrusion Detection Systems*) têm como função monitorar uma rede ou sistema computacional, buscando detectar ataques ou atividades maliciosas. Esses sistemas continuamente coletam dados do sistema e os analisam através de técnicas diversas, como reconhecimento de padrões, análise estatística e mineração de dados. Ao detectar uma atividade maliciosa, podem alertar o administrador do sistema e/ou ativar contra-medidas. De acordo com a origem da informação analisada, os IDSs podem ser classificados como:

- *IDSs de máquina* (HIDS – *Host-based IDS*): monitoram um computador para identificar atividades maliciosas locais, como subversão de serviços, vírus e *trojans*.
- *IDSs de rede* (NIDS – *Network-based IDS*): monitoram o tráfego de uma rede, para identificar ataques aos computadores a ela conectados.

Sistemas NIDS monitoram vários computadores simultaneamente, mas sua eficácia diminui na medida em que o tráfego da rede aumenta, pela necessidade de analisar pacotes mais rapidamente. Além disso, o uso de protocolos de rede cifrados torna o conteúdo dos pacotes opaco ao NIDS. Sistemas HIDS não sofrem desses problemas, pois analisam as informações disponíveis dentro de cada sistema. Todavia, um HIDS é um processo local, que pode ser desativado ou subvertido por um ataque bem-sucedido.

As máquinas virtuais podem ser úteis na proteção de sistemas HIDS. Através da propriedade de inspeção, o hipervisor pode extrair informações de um sistema convidado e encaminhá-las para análise por um detector de intrusão executando no sistema nativo, ou em outra máquina virtual. O isolamento entre máquinas virtuais assegura a integridade das informações coletadas e do próprio detector de intrusão. Por fim, a propriedade de controle de recursos permite ao hipervisor intervir no sistema convidado, para interromper atividades consideradas suspeitas pelo detector de intrusão.

No sistema *ReVirt* [Dunlap et al. 2002], uma camada construída entre o hipervisor e o sistema hospedeiro recebe as mensagens de *syslog*[2] geradas no sistema convidado e as registra no sistema hospedeiro. Essas mensagens são então usadas para detectar erros ou atividades maliciosas envolvendo os serviços do sistema convidado. Além disso, a camada *ReVirt* realiza *checkpoints* regulares do sistema convidado, que a permitem reverter o sistema a um estado anterior seguro, no caso de um ataque bem sucedido.

O trabalho [Garfinkel and Rosenblum 2003] descreve uma arquitetura para a detecção de intrusão em máquinas virtuais denominada VMI IDS (*Virtual Machine Introspection Intrusion Detection System*). Sua abordagem considera o uso de um hipervisor nativo, executando diretamente sobre o hardware. O IDS executa em uma das máquinas virtuais e observa dados obtidos das demais máquinas virtuais, buscando evidências de intrusão. Somente o estado global da máquina virtual (registradores, consumo de memória, etc) é analisado, sem levar em conta as atividades dos processos nela contidos, ou seja, o sistema proposto não tem a capacidade de avaliar processos individuais. Por isso, sua capacidade de resposta é limitada: caso haja suspeita de intrusão, a máquina virtual

[2] *Daemon* UNIX que registra as mensagens de *log* geradas pelas aplicações em execução no sistema.

comprometida é suspensa até que essa suspeita seja confirmada; nesse caso, a mesma pode ser reiniciada ou revertida a um estado anterior seguro (*rollback*).

Por outro lado, o trabalho [Laureano et al. 2004, Laureano et al. 2007] utiliza um hipervisor convidado modificado, que extrai informações do sistema convidado e as submete a um detector de intrusão executando no sistema hospedeiro. A detecção de intrusão é baseada nas seqüências de chamadas de sistema emitidas pelos processos do sistema convidado; dessa forma, cada processo convidado pode ser analisado separadamente. Caso o comportamento de um processo seja considerado anômalo, o hipervisor gera um alerta de suspeita e restringe as chamadas de sistema disponíveis ao processo.

4.2.4. Análise de programas maliciosos

[Klaus 1999, Skoudis and Zeltser 2003] consideram a existência de dois tipos de programas prejudiciais (figura 4.22):

- *Intencionais*: programas escritos para se infiltrar em um sistema, sem conhecimento de seus usuários, com a intenção de causar dano, furto ou seqüestro de informações. Esses programas são conhecidos como *malwares* (de *malicious softwares*);
- *Não-intencionais*: programas normais contendo erros de programação ou de configuração que permitam a manipulação não-autorizada das informações de um sistema; esses erros de programação ou de configuração são denominados *vulnerabilidades*.

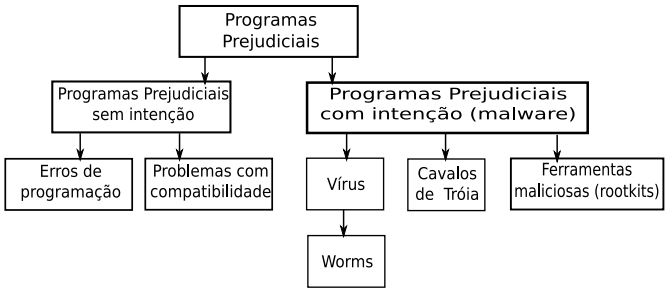

Figura 4.22. Classificação dos Programas Prejudiciais [Klaus 1999].

Para [Skoudis and Zeltser 2003], os malwares são conjuntos de instruções executadas em um computador e que fazem o sistema realizar algo que um atacante deseja. De forma geral, os malwares podem ser classificados em:

- *Vírus*: classe de software malicioso com a habilidade de se auto-replicar e infectar partes do sistema operacional ou dos programas de aplicação, visando causar a perda ou danos nos dados;
- *Worm*: designa qualquer software capaz de propagar a si próprio em uma rede. Habitualmente os *worms* invadem os sistemas computacionais através da exploração de falhas nos serviços de rede;

- *Cavalo de Tróia*: programa com função aparentemente útil, que também realiza ações escondidas visando roubar informações ou provocar danos no sistema;
- *Rootkit*: conjunto de ferramentas usadas por um invasor para ocultar sua presença em um sistema invadido. Os *rootkits* mais sofisticados envolvem modificações no próprio sistema operacional, para ocultar os recursos (como processos, arquivos e conexões de rede) usados pelo invasor.

A análise de um *malware* consiste em estudar o conteúdo (análise estática) e o comportamento (análise dinâmica) do programa, para descobrir seus objetivos, seu método de propagação, sua forma de dissimulação no sistema, encontrar evidências que possam indicar sua presença ou atividade e implementar formas de detectá-lo, removê-lo do sistema e impedir novas invasões. A análise estática se baseia no estudo do código binário do *malware*; ela é cada vez mais difícil de realizar, devido ao emprego de técnicas de dissimulação e cifragem [Beaucamps and Filiol 2007]. A análise dinâmica consiste na observação da execução do *malware* e de seus efeitos no sistema. Nesse contexto, o uso de uma máquina virtual é benéfico, pelas seguintes razões:

- Não é necessário dedicar uma máquina real "limpa" para cada análise;
- Torna-se simples salvar e restaurar estados da máquina virtual, permitindo desfazer os efeitos de uma intrusão; além disso, a comparação entre os estados antes de depois da intrusão permite compreender melhor seus efeitos no sistema;
- A verificação de informações de baixo nível (como o estado da memória, registradores, dados dentro do núcleo) torna-se mais simples, através da capacidade de inspeção do hipervisor;
- a tradução dinâmica de instruções pode ser usada para instrumentar o fluxo de instruções executado pelo *malware*.

Com base nessas constatações, diversas ferramentas e serviços de análise dinâmica automatizada de programas suspeitos têm sido propostos, como *Anubis*, *CWSandbox* e *Joebox* [Bayer et al. 2006, Willems et al. 2007]. Todavia, essa abordagem ainda não é perfeita. Vários estudos recentes têm demonstrado que as implementações atuais de hipervisores têm erros de projeto e/ou implementação que podem expor o sistema hospedeiro a ataques vindos do sistema convidado [Ormandy 2007], comprometendo a segurança do hipervisor e do próprio sistema hospedeiro. Além disso, a virtualização de plataformas correntes, como a de sistemas PC *Intel x86*, é bastante complexa, estando sujeita a problemas na virtualização de certos aspectos do hardware (conforme discutido na seção 4.1.3.3). Essas imperfeições abrem a porta para que *malwares* possam identificar o sistema invadido e comportar-se de forma distinta caso estejam em uma máquina virtual ou em um computador real [Raffetseder et al. 2007]. Com isso, a análise dinâmica do *malware* poderá não corresponder à sua execução em um sistema real. Por outro lado, outras pesquisas têm mostrado que há formas de tornar mais difícil a detecção de um sistema virtualizado por parte das aplicações convidadas [Liston and Skoudis 2006].

Em [Kwan and Durfee 2007] é proposta uma estrutura utilizando máquinas virtuais para verificação da integridade de um sistema. Neste sistema, chamado *Vault*, um agente de verificação é instalado nos hipervisores em uso. Estes agentes realizam a troca de mensagens visando garantir a segurança do sistema convidado monitorado. O principio do projeto é garantir que os sistemas monitorados não sejam subvertidos por *malwares* convencionais.

4.2.5. Honeypots e honeynets

Um *honeypot* é um sistema explicitamente preparado para ser atacado e invadido. Os *honeypots* podem ser vistos como "armadilhas", pois servem para atrair e estudar o tráfego de rede malicioso, como *worms* e ataques humanos. Como o *honeypot* não corresponde a serviços legítimos da rede e pode ser facilmente atacado, serve como alerta para a presença de atacantes na rede. Além disso, ele desvia a atenção dos atacantes dos servidores reais do sistema.

Os *honeypots* podem ser de *baixa* ou de *alta interatividade*. Nos *honeypots* de baixa interatividade, o atacante interage com emulações de serviços de rede, que não correspondem a serviços reais e portanto não serão realmente comprometidos. Um exemplo dessa abordagem é o *Honeyd*, um *daemon* UNIX que emula várias pilhas e serviços de rede [Provos 2004]. Já os *honeypots* de alta interatividade expõem sistemas e serviços reais – que podem ser comprometidos – aos atacantes. *Honeypots* de alta interatividade podem ser perigosos se mal configurados, pois podem ser invadidos e usados como plataformas para ataques ao restante da rede. Todavia, como os *honeypots* de alta interatividade são sistemas reais, as observações fornecidas por eles são mais detalhadas e correspondem melhor à realidade.

Uma *honeynet* [Honeynet Project 2001] é uma coleção de *honeypots* de alta interatividade com diferentes sistemas operacionais, configurações e serviços de rede. Por sua diversidade, essa rede tem um alto potencial de atração de ataques. Além dos *honeypots*, uma *honeynet* possui *firewalls* para evitar que tráfego malicioso se propague a partir dela para outras redes, detectores de intrusão para monitorar as atividades maliciosas e servidores de *logging* para o registro de eventos. A implantação de uma *honeynet* implica em alocar vários computadores, com sistemas operacionais e serviços distintos, o que pode significar um alto custo de implantação e operação. Nesse contexto, máquinas virtuais podem ser usadas para construir *honeynets virtuais*.

Uma *honeynet virtual* [Honeynet Project 2003] é simplesmente uma *honeynet* construída com máquinas virtuais ao invés de computadores reais. A virtualização traz vantagens, como o menor custo de implantação e gerência, mas pode também trazer inconvenientes, como limitar as possibilidades de *honeypots* aos sistemas que podem ser virtualizados sobre a plataforma computacional escolhida. Além disso, caso o sistema nativo seja comprometido, toda a *honeynet* estará comprometida. Para [Honeynet Project 2003], uma *honeynet* pode ser *totalmente virtual*, quando todos os computadores envolvidos são máquinas virtuais, ou *híbrida*, quando é composta por *honeypots* virtuais e máquinas reais para as funções de *firewall*, detecção de intrusão e *logging*.

4.2.6. *Rootkits*

Conforme apresentado na seção 4.2.4, um *rootkit* é uma ferramenta que visa ocultar a presença de um intruso no sistema. Os primeiros *rootkits* consistiam em substituir alguns comandos do sistema (como `ls`, `ps` e `netstat`) por versões com a mesma funcionalidade, mas que escondiam os arquivos, processos e conexões de rede do intruso, tornando-o "invisível". Posteriormente, foram desenvolvidos *rootkits* mais elaborados, que substituem bibliotecas do sistema com a mesma finalidade. Os *kernel rootkits* consistem em módulos de núcleo que alteram a implementação das chamadas de sistema dentro do núcleo, permitindo esconder o intruso de forma ainda mais profunda no sistema [Kruegel et al. 2004].

Recentemente, os avanços no suporte de hardware à virtualização possibilitaram o desenvolvimento de *rootkits* baseado em máquinas virtuais, ou *VM-based rootkits* (VMBR) [King and Chen 2006]. O funcionamento de um VMBR é conceitualmente simples: ao ser instalado, ele se torna um hipervisor, virtualizando todo o hardware da máquina e transformando o sistema operacional invadido em um sistema convidado (conforme apresentado na figura 4.23). Assim, toda a funcionalidade maliciosa do *rootkit* se encontra abaixo do sistema operacional (e fora do alcance dele), sendo teoricamente indetectável por detectores de *rootkits* convencionais.

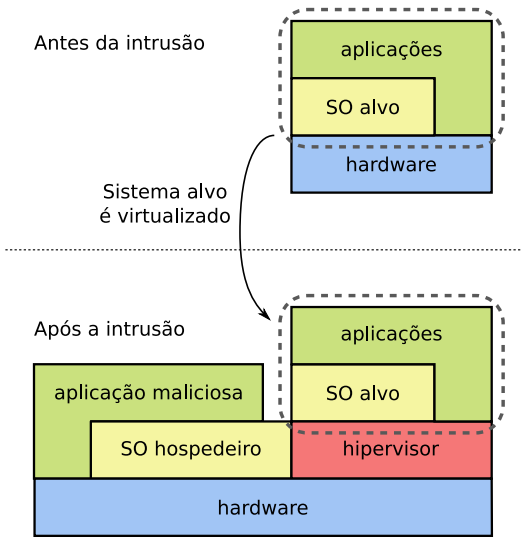

Figura 4.23. Sistema antes e depois de ser afetado por um VMBR [King and Chen 2006].

Os *rootkits* baseados em virtualização mais conhecidos são o *Subvirt* (que é uma prova de conceito) [King and Chen 2006], o *Vitriol* (que usa a tecnologia *VT-x* dos processadores Intel) [Zovi 2006] e o *BluePill* (que usa a tecnologia equivalente *SVM*, da AMD) [Rutkowska 2006]. Embora os processadores sejam diferentes, o princípio de funcionamento do suporte de hardware à virtualização é análogo em ambas as plataformas.

Da mesma forma que um hipervisor nativo, estes *rootkits* modificam a seqüência de inicialização da máquina afetada, carregando primeiro o seu código e posteriormente o sistema operacional original, em um ambiente convidado. Quando ativo, o VMBR torna-se incontornável: ele pode interceptar todos os acessos ao hardware e inspecionar todo o estado interno do sistema operacional. Embora este tipo de *rootkit* seja teoricamente indetectável, vários pesquisadores afirmam ser possível detectar anomalias em um sistema virtualizado, observado o comportamento dos processos, a alocação de recursos, diferenças de temporização nos acessos ao hardware, e falhas no suporte à virtualização, que permitiriam detectar a presença desses *rootkits* [Rutkowska 2004, Garfinkel et al. 2007].

4.2.7. Consolidação de servidores, planos de contingência e migração

Uma das aplicações mais freqüentes da virtualização é a consolidação de servidores, que consiste em transferir serviços em máquinas físicas para máquinas virtuais, para diminuir o número de equipamentos da organização. Busca-se com isso aumentar a produtividade da infra-estrutura, simplificar a gerência do ambiente, aumentar a segurança, diminuir o consumo de energia e economizar recursos humanos, físicos e financeiros. Com a virtualização, os recursos físicos podem ser dinamicamente alocados aos sistemas operacionais de acordo com suas necessidades, proporcionando balanceamento de carga dinâmico e um melhor controle de uso dos recursos dos servidores [Newman et al. 2005, Ferre et al. 2006]. Além disso, a redução do número de equipamentos permite mais investimento em tecnologias de melhoria da disponibilidade, como fontes de alimentação redundantes e controladores de disco com suporte a *hot-swapping*.

Um plano de contingência visa assegurar a disponibilidade de sistemas críticos e facilitar a continuidade das operações durante uma crise. Esse plano é aplicado após algum incidente nos sistemas de computação. As máquinas virtuais podem desempenhar um papel importante nos planos de contingência de servidores e serviços, a um custo de hardware e software relativamente baixo [Newman et al. 2005, Ferre et al. 2006]. Como máquinas virtuais podem ser salvas na forma de arquivos, *checkpoints* dos servidores críticos podem ser salvos em equipamentos distintos. Assim, quando um serviço falhar, uma cópia atualizada do mesmo pode ser rapidamente colocada em funcionamento.

Hipervisores modernos implementam facilidades de *migração* de máquinas virtuais. Na migração, uma máquina virtual e seu sistema convidado são transferidos de um hipervisor para outro, executando em equipamentos distintos. A máquina virtual tem seu estado preservado e prossegue sua execução no hipervisor de destino assim que a migração é concluída. De acordo com [Clark et al. 2005], as técnicas mais freqüentes para implementar a migração são:

- *stop-and-copy*: consiste em suspender a máquina virtual, transferir o conteúdo de sua memória para o hipervisor de destino e retomar a execução em seguida. É uma abordagem simples, mas implica em parar completamente os serviços oferecidos pelo sistema convidado enquanto durar a migração (que pode demorar algumas dezenas de segundos);
- *demand-migration*: a máquina virtual é suspensa apenas durante a cópia das estruturas de memória do núcleo para o hipervisor de destino, o que dura alguns

milissegundos. Em seguida, a execução da máquina virtual é retomada e o restante das páginas de memória da máquina virtual é transferido sob demanda, através dos mecanismos de tratamento de faltas de página. Nesta abordagem a interrupção do serviço tem duração mínima, mas a migração completa pode demorar muito tempo.

- *pre-copy*: consiste basicamente em copiar para o hipervisor de destino todas as páginas de memória da máquina virtual enquanto esta executa; a seguir, a máquina virtual é suspensa e as páginas modificadas depois da cópia inicial são novamente copiadas no destino; uma vez terminada a cópia dessas páginas, a máquina pode retomar sua execução no destino. Esta abordagem, usada no hipervisor *Xen* [Barham et al. 2003], é a que oferece o melhor compromisso entre o tempo de suspensão do serviço e a duração total da migração.

A migração de máquinas virtuais é uma solução útil para a continuidade de serviços no caso de manutenções programadas, mas ela também pode ser usada nos casos de falhas não-previstas, como demonstra o trabalho [Fu and Xu 2005].

4.2.8. Tolerância a faltas

A propriedade de isolamento das máquinas virtuais pode ser usada para restringir a propagação de erros de software nos sistemas. Alguns projetos de pesquisa exploram a possibilidade de usar máquinas virtuais para encapsular partes de sistemas operacionais ou de aplicativos, buscando aumentar assim a sua disponibilidade. Por exemplo, a maioria das falhas em um sistema operacional é causada por *drivers* construídos por terceiros [Herder et al. 2006]. Como estes *drivers* fazem parte do sistema operacional, eles executam no nível mais privilegiado do processador. Baseado na virtualização do processador é possível criar um ambiente isolado para a execução de *drivers* suspeitos ou instáveis, restringindo o alcance de eventuais erros nos mesmos.

O projeto *iKernel* (*isolation Kernel*) [Tan et al. 2007] utiliza essa abordagem. Nesse projeto, um núcleo Linux é executado como um sistema convidado. A execução de *drivers* é realizada no sistema convidado, e o resultado da execução é devolvido ao sistema hospedeiro. A comunicação entre o sistema convidado e o hospedeiro é realizada através de uma área de memória compartilhada entre ambos os sistemas, estritamente isolada do restante da memória. Eventuais erros nos *drivers* ficam restritos ao sistema convidado, que pode ser facilmente reiniciado, não afetando o hospedeiro. Testes realizados utilizando o hipervisor *QEMU* (seção 4.3.5) comprovaram a viabilidade da proposta. Todavia, alguns *drivers* de dispositivos não são facilmente isoláveis, como os *drivers* gráficos.

Outro projeto que usa a mesma abordagem é o *L4Ka* [LeVasseur et al. 2004]. A principal diferença entre este e o *iKernel* diz respeito ao nível de virtualização do sistema convidado, que neste caso não se beneficia do suporte de hardware à virtualização. Neste projeto, alguns *drivers* são executado em um sistema convidado com acesso somente aos recursos necessários para a sua execução.

4.3. Exemplos de máquinas virtuais

Esta seção apresenta alguns sistemas de máquinas virtuais de uso corrente. Serão apresentados os sistemas *VMWare*, *FreeBSD Jails*, *Xen*, *User-Mode Linux*, *QEMU*, *Valgrind* e *JVM*. Entre eles há máquinas virtuais de aplicação e de sistema, com virtualização total ou paravirtualização, além de abordagens hibridas. Eles foram escolhidos por estarem entre os mais representativos de suas respectivas classes.

4.3.1. VMware

Atualmente, o *VMware* é a máquina virtual para a plataforma x86 de uso mais difundido, provendo uma implementação completa da interface x86 ao sistema convidado. Embora essa interface seja extremamente genérica para o sistema convidado, acaba conduzindo a um hipervisor mais complexo. Como podem existir vários sistemas operacionais em execução sobre mesmo hardware, o hipervisor tem que emular certas instruções para representar corretamente um processador virtual em cada máquina virtual, fazendo uso intensivo dos mecanismos de tradução dinâmica [VMware 2000, Newman et al. 2005]. Atualmente, a *VMware* produz vários produtos com hipervisores nativos e convidados:

- Hipervisor convidado:
  - *VMware Workstation*: primeira versão comercial da máquina virtual, lançada em 1999, para ambientes *desktop*;
  - *VMware Fusion*: versão experimental para o sistema operacional *Mac OS* com processadores Intel;
  - *VMware Player*: versão gratuita do *VMware Workstation*, com as mesmas funcionalidades mas limitado a executar máquinas virtuais criadas previamente com versões comerciais;
  - *VMWare Server*: conta com vários recursos do *VMware Workstation*, mas é voltado para pequenas e médias empresas;
- Hipervisor nativo:
  - *VMware ESX Server*: para servidores de grande porte, possui um núcleo proprietário chamado *vmkernel* e Utiliza o *Red Hat Linux* para prover outros serviços, tais como a gerência de usuários.

O *VMware Workstation* utiliza as estratégias de virtualização total e tradução dinâmica (seção 4.1.5). O *VMware ESX Server* implementa ainda a paravirtualização. Por razões de desempenho, o hipervisor do *VMware* utiliza uma abordagem híbrida (seção 4.1.5.4) para implementar a interface do hipervisor com as máquinas virtuais [Sugerman et al. 2001]. O controle de exceção e o gerenciamento de memória são realizados por acesso direto ao hardware, mas o controle de entrada/saída usa o sistema hospedeiro. Para garantir que não ocorra nenhuma colisão de memória entre o sistema convidado e o real, o hipervisor *VMware* aloca uma parte da memória para uso exclusivo de cada sistema convidado.

Para controlar o sistema convidado, o *VMware Workstation* intercepta todas as interrupções do sistema convidado. Sempre que uma exceção é causada no convidado, é examinada primeiro pelo hipervisor. As interrupções de entrada/saída são remetidas para o sistema hospedeiro, para que sejam processadas corretamente. As exceções geradas pelas aplicações no sistema convidado (como as chamadas de sistema, por exemplo) são remetidas para o sistema convidado.

4.3.2. FreeBSD Jails

O sistema operacional *FreeBSD* oferece um mecanismo de confinamento de processos denominado *Jails*, criado para aumentar a segurança de serviços de rede. Esse mecanismo consiste em criar domínios de execução distintos (denominados *jails* ou celas), conforme descrito na seção 4.2.2. Cada cela contém um subconjunto de processos e recursos (arquivos, conexões de rede) que pode ser gerenciado de forma autônoma, como se fosse um sistema separado [Kamp and Watson 2000].

Cada domínio é criado a partir de um diretório previamente preparado no sistema de arquivos. Um processo que executa a chamada de sistema `jail` cria uma nova cela e é colocado dentro dela, de onde não pode mais sair, nem seus filhos. Além disso, os processos em um domínio não podem:

- Reconfigurar o núcleo (através da chamada `sysctl`, por exemplo);
- Carregar/retirar módulos do núcleo;
- Mudar configurações de rede (interfaces e rotas);
- Montar/desmontar sistemas de arquivos;
- Criar novos *devices*;
- Realizar modificações de configurações do núcleo em tempo de execução;
- Acessar recursos que não pertençam ao seu próprio domínio.

Essas restrições se aplicam mesmo a processos que estejam executando com privilégios de administrador (*root*).

Pode-se considerar que o sistema *FreeBSD Jails* virtualiza somente partes do sistema hospedeiro, como a árvore de diretórios (cada domínio tem sua própria visão do sistema de arquivos), espaços de nomes (cada domínio mantém seus próprios identificadores de usuários, processos e recursos de IPC) e interfaces de rede (cada domínio tem sua interface virtual, com endereço IP próprio). Os demais recursos (como as instruções de máquina e chamadas de sistema) são preservadas, ou melhor, podem ser usadas diretamente. Essa virtualização parcial demanda um custo computacional muito baixo, mas exige que todos os sistemas convidados executem sobre o mesmo núcleo.

4.3.3. Xen

O ambiente *Xen* é um hipervisor nativo para a plataforma *x86* que implementa a paravirtualização. Ele permite executar sistemas operacionais como Linux e Windows especialmente modificados para executar sobre o hipervisor [Barham et al. 2003]. Versões mais recentes do sistema *Xen* utilizam o suporte de virtualização disponível nos processadores atuais, o que torna possível a execução de sistemas operacionais convidados sem modificações, embora com um desempenho ligeiramente menor que no caso de sistemas paravirtualizados. Conforme seus criadores, o custo e impacto das alterações nos sistemas convidados são baixos e a diminuição do custo da virtualização compensa essas alterações (a degradação média de desempenho observada em sistemas virtualizados sobre a plataforma *Xen* não excede 5%). As principais modificações impostas pelo ambiente *Xen* a um sistema operacional convidado são:

- O mecanismo de entrega de interrupções passa a usar um serviço de eventos oferecido pelo hipervisor; o núcleo convidado deve registrar um vetor de tratadores de exceções junto ao hipervisor;
- as operações de entrada/saída de dispositivos são feitas através de uma interface simplificada, independente de dispositivo, que usa *buffers* circulares de tipo produtor/consumidor;
- o núcleo convidado pode consultar diretamente as tabelas de segmentos e páginas da memória usada por ele e por suas aplicações, mas as modificações nas tabelas devem ser solicitadas ao hipervisor;
- o núcleo convidado deve executar em um nível de privilégio inferior ao do hipervisor;
- o núcleo convidado deve implementar uma função de tratamento das chamadas de sistema de suas aplicações, para evitar que elas tenham de passar pelo hipervisor antes de chegar ao núcleo convidado.

Como o hipervisor deve acessar os dispositivos de hardware, ele deve dispor dos *drivers* adequados. Já os núcleos convidados não precisam de *drivers* específicos, pois eles acessam dispositivos virtuais através de uma interface simplificada. Para evitar o desenvolvimento de *drivers* específicos para o hipervisor, o ambiente *Xen* usa uma abordagem alternativa: a primeira máquina virtual (chamada VM_0) pode acessar o hardware diretamente e provê os *drivers* necessários ao hipervisor. As demais máquinas virtuais ($VM_i, i > 0$) acessam o hardware virtual através do hipervisor, que usa os *drivers* da máquina VM_0 conforme necessário. Essa abordagem, apresentada na figura 4.24, simplifica muito a evolução do hipervisor, por permitir utilizar os *drivers* desenvolvidos para o sistema Linux.

O hipervisor *Xen* pode ser considerado uma tecnologia madura, sendo muito utilizado em sistemas de produção. O seu código-fonte está liberado sob a licença *GNU General Public Licence* (GPL). Atualmente, o ambiente *Xen* suporta os sistemas Windows, Linux e NetBSD. Várias distribuições Linux já possuem suporte nativo ao *Xen*.

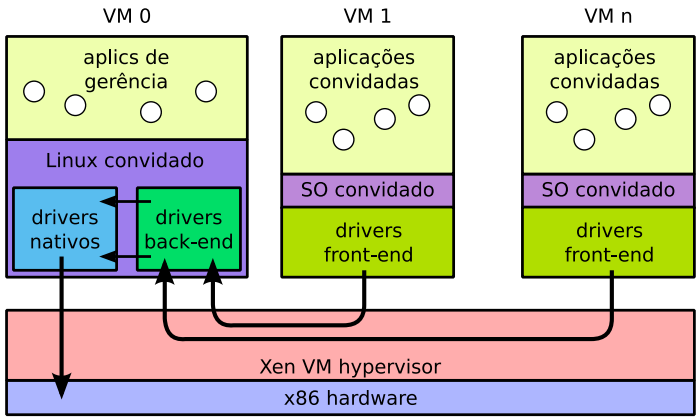

Figura 4.24. O hipervisor *Xen*.

4.3.4. User-Mode Linux

O *User-Mode Linux* foi proposto por Jeff Dike em 2000, como uma alternativa de uso de máquinas virtuais no ambiente Linux [Dike 2000]. O núcleo do Linux foi portado de forma a poder executar sobre si mesmo, como um processo do próprio Linux. O resultado é um *user space* separado e isolado na forma de uma máquina virtual, que utiliza dispositivos de hardware virtualizados a partir dos serviços providos pelo sistema hospedeiro. Essa máquina virtual é capaz de executar todos os serviços e aplicações disponíveis para o sistema hospedeiro. Além disso, o custo de processamento e de memória das máquinas virtuais *User-Mode Linux* é geralmente menor que aquele imposto por outros hipervisores mais complexos.

O *User-Mode Linux* é hipervisor convidado, ou seja, executa na forma de um processo no sistema hospedeiro. Os processos em execução na máquina virtual não têm acesso direto aos recursos do sistema hospedeiro. A maior dificuldade na implementação do *User-Mode Linux* foi encontrar formas de virtualizar as funcionalidades do hardware para as chamadas de sistema do Linux, sobretudo a distinção entre o modo privilegiado do núcleo e o modo não-privilegiado de usuário. Um código somente pode estar em modo privilegiado se é confiável o suficiente para ter pleno acesso ao hardware, como o próprio núcleo do sistema operacional. O *User-Mode Linux* deve possuir uma distinção de privilégios equivalente para permitir que o seu núcleo tenha acesso às chamadas de sistema do sistema hospedeiro quando os seus próprios processos solicitarem este acesso, ao mesmo tempo em que impede os mesmos de acessar diretamente os recursos reais subjacentes.

No hipervisor, a distinção de privilégios foi implementada com o mecanismo de interceptação de chamadas do próprio Linux, fornecido pela chamada de sistema `ptrace`[3]. Usando a chamada `ptrace`, o hipervisor recebe o controle de todas as cha-

[3]Chamada de sistema que permite observar e controlar a execução de outros processos; o comando `strace` do Linux permite ter uma noção de como a chamada de sistema `ptrace` funciona.

madas de sistema de entrada/saída geradas pelas máquinas virtuais. Todos os sinais gerados ou enviados às máquinas virtuais também são interceptados. A chamada `ptrace` também é utilizada para manipular o contexto do sistema convidado.

O *User-Mode Linux* utiliza o sistema hospedeiro para operações de entrada/saída. Como a máquina virtual é um processo no sistema hospedeiro, a troca de contexto entre duas instâncias de máquinas virtuais é rápida, assim como a troca entre dois processos do sistema hospedeiro. Entretanto, modificações no sistema convidado foram necessárias para a otimização da troca de contexto. A virtualização das chamadas de sistema é implementada pelo uso de uma *thread* de rastreamento que intercepta e redireciona todas as chamadas de sistema para o núcleo virtual. Este identifica a chamada de sistema e os seus argumentos, cancela a chamada e modifica estas informações no hospedeiro, onde o processo troca de contexto e executa a chamada na pilha do núcleo.

O *User-Mode Linux* está disponível na versão 2.6 do núcleo Linux, ou seja, ele foi assimilado à árvore oficial de desenvolvimento do núcleo, portanto melhorias na sua arquitetura deverão surgir no futuro, ampliando seu uso em diversos contextos de aplicação.

4.3.5. QEMU

O *QEMU* é um hipervisor com virtualização completa [Bellard 2005]. Não requer alterações ou otimizações no sistema hospedeiro, pois utiliza intensivamente a tradução dinâmica (seção 4.1.5) como técnica para prover a virtualização. É um dos poucos hipervisores recursivos, ou seja, é possível chamar o *QEMU* a partir do próprio *QEMU*. O hipervisor *QEMU* oferece dois modos de operação:

- *Emulação total do sistema*: emula um sistema completo, incluindo processador (normalmente um *Intel Pentium II*) e vários periféricos. Neste modo o emulador pode ser utilizado para executar diferentes sistemas operacionais;
- *Emulação no modo de usuário*: disponível apenas para o sistema Linux. Neste modo o emulador pode executar processos Linux compilados em diferentes plataformas (por exemplo, um programa compilado para um processador *x86* pode ser executado em um processador *PowerPC* e vice-versa).

Durante a emulação de um sistema completo, o *QEMU* implementa uma MMU (*Memory Management Unit*) totalmente em software, para garantir o máximo de portabilidade. Quando em modo usuário, o *QEMU* simula uma MMU simplificada através da chamada de sistema `mmap` (que permite mapear um arquivo em uma região da memória) do sistema hospedeiro.

Por meio de um módulo instalado no núcleo do sistema hospedeiro, denominado *KQEMU* ou *QEMU Accelerator*, o hipervisor *QEMU* consegue obter um desempenho similar ao de outras máquinas virtuais como *VMWare* e *User-Mode Linux*. Com este módulo, o *QEMU* passa a executar as chamadas de sistema emitidas pelos processos convidados diretamente sobre o sistema hospedeiro, ao invés de interpretar cada uma. O *KQEMU* permite associar os dispositivos de entrada/saída e o endereçamento de memória

do sistema convidado aos do sistema hospedeiro. Processos em execução sobre o núcleo convidado passam a executar diretamente no modo usuário do sistema hospedeiro. O modo núcleo do sistema convidado é utilizado apenas para virtualizar o processador e os periféricos.

O *VirtualBox* [VirtualBox 2008] é um ambiente de máquinas virtuais construído sobre o hipervisor *QEMU*. Ele é similar ao *VMware Workstation* em muitos aspectos. Atualmente, pode tirar proveito do suporte à virtualização disponível nos processadores Intel e AMD. Originalmente desenvolvido pela empresa *Innotek*, o *VirtualBox* foi adquirido pela *Sun Microsystems* e liberado para uso público sob a licença *GPLv2*.

4.3.6. Valgrind

O *Valgrind* [Nethercote and Seward 2007] é uma ferramenta de depuração de uso da memória RAM e problemas correlatos. Ele permite investigar fugas de memória (*memory leaks*), acessos a endereços inválidos, padrões de uso dos caches e outras operações envolvendo o uso da memória RAM. O *Valgrind* foi desenvolvido para plataforma *x86 Linux*, mas existem versões experimentais para outras plataformas.

Tecnicamente, o *Valgrind* é um hipervisor de aplicação que virtualiza o processador através de técnicas de tradução dinâmica. Ao iniciar a análise de um programa, o *Valgrind* traduz o código executável do mesmo para um formato interno independente de plataforma denominado IR (*Intermediate Representation*). Após a conversão, o código em IR é instrumentado, através da inserção de instruções para registrar e verificar as operações de alocação, acesso e liberação de memória. A seguir, o programa IR devidamente instrumentado é traduzido no formato binário a ser executado sobre o processador virtual. O código final pode ser até 50 vezes mais lento que o código original, mas essa perda de desempenho normalmente não é muito relevante durante a análise de um programa.

4.3.7. JVM – Java Virtual Machine

É comum a implementação do suporte de execução de uma linguagem de programação usando uma máquina virtual. Um exemplo clássico nesse sentido é o compilador *UCSD Pascal*. Nesse sistema, um programa escrito em Pascal é compilado em um código binário independente de plataforma denominado *P-Code*, que executava sobre o processador abstrato *P-Machine*. O interpretador de *P-Codes* era bastante compacto e facilmente portável, o que tornou o sistema P muito popular nos anos 1970. A estrutura da *P-Machine* é orientada a *pilha*, ou seja, a maioria das instruções realiza suas operações usando a pilha ao invés de registradores específicos.

Um exemplo atual dessa abordagem ocorre na linguagem Java. Tendo sido originalmente concebida para o desenvolvimento de pequenos aplicativos e programas de controle de aparelhos eletroeletrônicos, a linguagem Java mostrou-se ideal para ser usada na Internet. O que o torna tão atraente é o fato de programas escritos em Java poderem ser executados em praticamente qualquer plataforma. A virtualização é o fator responsável pela independência dos programas Java do hardware e dos sistemas operacionais: um programa escrito em Java, ao ser compilado, gera um código binário específico para uma máquina abstrata denominada *máquina virtual Java* (*JVM - Java Virtual Machine*). A linguagem de máquina executada pela máquina virtual Java é denominada *bytecode*

Java, e não corresponde a instruções de nenhum processador real. A máquina virtual deve então interpretar todas as operações do *bytecode*, utilizando as instruções da máquina real subjacente para executá-las.

A vantagem mais significativa da abordagem adotada por Java é a *portabilidade* do código executável: para que uma aplicação Java possa executar sobre uma determinada plataforma, basta que a máquina virtual Java esteja disponível ali (na forma de um suporte de execução denominado JRE - *Java Runtime Environment*). Assim, a portabilidade dos programas Java depende unicamente da portabilidade da própria máquina virtual Java. O suporte de execução Java pode estar associado a um navegador Web, o que permite que código Java seja associado a páginas Web, na forma de pequenas aplicações denominadas *applets*, que são trazidas junto com os demais componentes de página Web e executam localmente no navegador. A figura 4.25 mostra os principais componentes da plataforma Java.

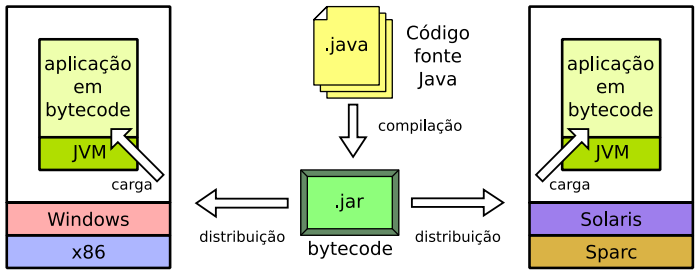

Figura 4.25. Máquina virtual Java.

É importante ressaltar que a adoção de uma máquina virtual como suporte de execução não é exclusividade de Java, nem foi inventada por seus criadores. As primeiras experiências de execução de aplicações sobre máquinas abstratas remontam aos anos 1970, com a linguagem *UCSD Pascal*. Hoje, muitas linguagens adotam estratégias similares, como Java, C#, Python, Perl, Lua e Ruby. Em C#, o código-fonte é compilado em um formato intermediário denominado CIL (*Common Intermediate Language*), que executa sobre uma máquina virtual CLR (*Common Language Runtime*). CIL e CLR fazem parte da infraestrutura .NET da Microsoft.

Em termos de desempenho, um programa compilado para uma máquina virtual executa mais lentamente que seu equivalente compilado sobre uma máquina real, devido ao custo de interpretação do *bytecode*. Todavia, essa abordagem oferece melhor desempenho que linguagens puramente interpretadas. Além disso, técnicas de otimização como a tradução dinâmica (ou compilação *Just-in-Time*), na qual blocos de instruções repetidos freqüentemente são compilados pelo monitor e armazenados em cache, permitem obter ganhos significativos de desempenho.

4.4. Perspectivas

A utilização de máquinas virtuais tornou-se uma alternativa concreta para várias soluções domésticas e corporativas. Graças a diversas pesquisas, no futuro será possível utilizar

os melhores recursos das mais variadas plataformas operacionais sem a necessidade de investir em equipamentos específicos. São vários os exemplos de sistemas que utilizam os conceitos de virtualização – alguns vistos neste trabalho -, mas, com certeza, vários outros exemplos surgirão no futuro para aproveitar os novos avanços nessa área.

Embora o uso de máquinas virtuais tenha evoluído, várias outras pesquisas na indústria e nas universidades devem ser realizadas para aprimorar as questões de segurança, mobilidade e desempenho dos hipervisores [Rosenblum and Garfinkel 2005]. Os principais campos de pesquisas nos próximos anos para melhorar o suporte à virtualização provavelmente serão:

- *Processadores*: os principais fabricantes de processadores, AMD e Intel, já disponibilizaram tecnologias para que a virtualização sobre a plataforma x86 ocorra de forma mais natural e tranqüila. Estas tecnologias vêm simplificando o desenvolvimento dos novos hipervisores.
- *Memória*: Várias técnicas têm permitido que a virtualização da memória seja mais eficiente. Pesquisas futuras devem levar aos sistemas operacionais convidados a gerenciar a memória juntamente com o hipervisor (gerência cooperativa).
- *Entrada/saída*: Os dispositivos de entrada/saída deverão ser projetados para fornecer suporte à virtualização com alto desempenho. O próprio dispositivo deverá suportar a multiplexação, de forma a permitir o acesso simultâneo por vários sistemas virtuais. A responsabilidade pelo acesso aos dispositivos deverá passar do hipervisor para o sistema convidado.

Várias pesquisas vêm sendo conduzidas sobre a aplicação de hipervisores na segurança de sistemas, mas também na melhoria da segurança dos próprios hipervisores. Alguns problemas que provavelmente merecerão a atenção dos pesquisadores nos próximos anos incluem:

- Garantir que as propriedades básicas dos hipervisores sejam providas corretamente pelas implementações; uma falha de segurança no hipervisor pode por em risco a segurança deste, do sistema hospedeiro e de todos os sistemas convidados;
- Construir métodos confiáveis para detectar a execução em sistemas convidados, para descobrir a presença de *rootkits* baseados em máquinas virtuais;
- De forma contraditória, criar técnicas para impedir a detecção de ambiente convidados, que serão úteis na construção de *honeypots* virtuais;
- Trazer para o *desktop* o conceito de *sandboxing*, que é a implementação de máquinas virtuais leves e transparentes ao usuário, para a execução isolada de aplicações sensíveis como navegadores Internet, clientes de e-mail e programas de mensagens instantâneas;

Agradecimentos

Os autores desejam agradecer as críticas e sugestões recebidas dos revisores da proposta deste minicurso; desejam também agradecer a Fundação Araucária – FAP do Estado do Paraná, pelo apoio financeiro recebido na forma de uma bolsa de doutorado.

Referências

[Barham et al. 2003] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., and Warfield, A. (2003). Xen and the art of virtualization. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, pages 164–177.

[Bayer et al. 2006] Bayer, U., Moser, A., Kruegel, C., and Kirda, E. (2006). Dynamic analysis of malicious code. *Journal in Computer Virology*, 2(1).

[Beaucamps and Filiol 2007] Beaucamps, P. and Filiol, E. (2007). On the possibility of practically obfuscating programs towards a unified perspective of code protection. *Journal in Computer Virology*, 3(1):3–21.

[Bellard 2005] Bellard, F. (2005). QEMU, a fast and portable dynamic translator. In *USENIX Annual Technical Conference*.

[Clark et al. 2005] Clark, C., Fraser, K., Hand, S., Hansen, J., Jul, E., Limpach, C., Pratt, I., and Warfield, A. (2005). Live migration of virtual machines. In *2nd Symposium on Networked Systems Design and Implementation*.

[Dike 2000] Dike, J. (2000). A user-mode port of the Linux kernel. In *Proceedings of the 4th Annual Linux Showcase & Conference*, Atlanta - USA.

[Duesterwald 2005] Duesterwald, E. (2005). Design and engineering of a dynamic binary optimizer. *Proceedings of the IEEE*, 93(2):436–448.

[Dunlap et al. 2002] Dunlap, G. W., King, S. T., Cinar, S., Basrai, M. A., and Chen, P. M. (2002). Revirt: Enabling intrusion analysis through virtual-machine logging and replay. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*.

[Ferre et al. 2006] Ferre, M. R., Pomeroy, D., Wahlstrom, M., and Watts, D. (2006). *Virtualization on the IBM System x3950 Server*. IBM RedBooks. http://www.redbooks.ibm.com.

[Fu and Xu 2005] Fu, S. and Xu, C.-Z. (2005). Service migration in distributed virtual machines for adaptive grid computing. In *34th IEEE Intl Conference on Parallel Processing*.

[Garfinkel et al. 2007] Garfinkel, T., Adams, K., Warfield, A., and Franklin, J. (2007). Compatibility is not transparency: VMM detecion myths and realities. Technical report, TRUST - The Team for Research in Ubiquitous Secure Technology.

[Garfinkel and Rosenblum 2003] Garfinkel, T. and Rosenblum, M. (2003). A virtual machine introspection based architecture for intrusion detection. In *Proceedings of the 2003 Network and Distributed System Security Symposium*.

[Goldberg 1973] Goldberg, R. (1973). Architecture of virtual machines. *AFIPS National Computer Conference*.

[Goldberg and Mager 1979] Goldberg, R. and Mager, P. (1979). Virtual machine technology: A bridge from large mainframes to networks of small computers. *IEEE Proceedings Compcon Fall 79*, pages 210–213.

[Herder et al. 2006] Herder, J. N., Bos, H., Grass, B., Homburg, P., and Tanenbaum, A. S. (2006). Reorganizing UNIX for reliability. In *Proceedings of 11th Asia-Pacific Computer Systems Architecture Conference*, pages 81–94.

[Honeynet Project 2001] Honeynet Project (2001). *Know Your Enemy: Revealing the Security Tools, Tactics, and Motives of the Blackhat Community*. Addison-Wesley.

[Honeynet Project 2003] Honeynet Project (2003). Know your enemy: defining virtual honeynets. http://project.honeynet.org.

[IBM 2007] IBM (2007). *Power Instruction Set Architecture – Version 2.04*. IBM Corporation.

[Kamp and Watson 2000] Kamp, P.-H. and Watson, R. N. M. (2000). Jails: Confining the omnipotent root. In *Proceedings of the Second International System Administration and Networking Conference (SANE)*.

[King and Chen 2006] King, S. and Chen, P. (2006). Subvirt: Implementing malware with virtual machines. In *IEEE Symposium on Security and Privacy*, pages 1–14.

[Klaus 1999] Klaus, B. (1999). From antivirus to antimalware software and beyond: Another approach to the protection of customers from dysfunctional system behaviour. In *22nd National Information Systems Security Conference*. http://csrc.nist.gov/nissc/1999/proceeding/papers/p12.pdf.

[Krause and Tipton 1999] Krause, M. and Tipton, H. F. (1999). *Handbook of Information Security Management*. Auerbach Publications.

[Kruegel et al. 2004] Kruegel, C., Robertson, W., and Vigna, G. (2004). Detecting kernel-level rootkits through binary analysis. In *20th Annual Computer Security Applications Conference*.

[Kwan and Durfee 2007] Kwan, P. C. S. and Durfee, G. (2007). Practical uses of virtual machines for protection of sensitive user data. In *Information Security Practice and Experience*, pages 145–161. Springer Berlin / Heidelberg.

[Laureano et al. 2004] Laureano, M., Maziero, C., and Jamhour, E. (2004). Intrusion detection in virtual machine environments. In *30th EUROMICRO Conference*, pages 520–525, Rennes - França.

[Laureano et al. 2007] Laureano, M., Maziero, C., and Jamhour, E. (2007). Protecting host-based intrusion detectors through virtual machines. *Computer Networks*, 51:1275–1283.

[Laureano 2006] Laureano, M. A. P. (2006). *Máquinas Virtuais e Emuladores - Conceitos, Técnicas e Aplicações*. Novatec Editora, first edition.

[LeVasseur et al. 2004] LeVasseur, J., Uhlig, V., Stoess, J., and Götz, S. (2004). Unmodified device driver reuse and improved system dependability via virtual machines. In *Proceedings of the 6th Symposium on Operating Systems Design and Implementation*, San Francisco, CA.

[Liston and Skoudis 2006] Liston, T. and Skoudis, E. (2006). On the cutting edge: Thwarting virtual machine detection. In *SANS Conference*.

[Nanda and Chiueh 2005] Nanda, S. and Chiueh, T. (2005). A survey on virtualization technologies. Technical report, University of New York at Stony Brook.

[Nethercote and Seward 2007] Nethercote, N. and Seward, J. (2007). Valgrind: A framework for heavyweight dynamic binary instrumentation. In *Proceedings of ACM SIGPLAN 2007 Conference on Programming Language Design and Implementation (PLDI 2007)*, San Diego - California - USA.

[Newman et al. 2005] Newman, M., Wiberg, C.-M., and Braswell, B. (2005). *Server Consolidation with VMware ESX Server*. IBM RedBooks. http://www.redbooks.ibm.com.

[Ormandy 2007] Ormandy, T. (2007). An empirical study into the security exposure to hosts of hostile virtualized environments. In *CanSecWest*, Vancouver BC.

[Popek and Goldberg 1974] Popek, G. and Goldberg, R. (1974). Formal requirements for virtualizable third generation architectures. *Communications of the ACM*, 17(7):412–421.

[Provos 2004] Provos, N. (2004). A virtual honeypot framework. In *13th USENIX Security Symposium*, San Diego, CA.

[Raffetseder et al. 2007] Raffetseder, T., Kruegel, C., and Kirda, E. (2007). Detecting system emulators. In *10th Information Security Conference - LNCS*.

[Robin and Irvine 2000] Robin, J. and Irvine, C. (2000). Analysis of the Intel Pentium's ability to support a secure virtual machine monitor. In *9th USENIX Security Symposium*.

[Rosenblum 2004] Rosenblum, M. (2004). The reincarnation of virtual machines. *Queue Focus - ACM Press*, pages 34–40.

[Rosenblum and Garfinkel 2005] Rosenblum, M. and Garfinkel, T. (2005). Virtual machine monitors: current tecnology and future trends. *IEEE Computer Magazine*, 38(5):39–47.

[Rutkowska 2004] Rutkowska, J. (2004). Red pill... or how to detect VMM using (almost) one CPU instruction. http://invisiblethings.org/papers/redpill.html.

[Rutkowska 2006] Rutkowska, J. (2006). Subverting Vista kernel for fun and profit. http://www.blackhat.com. Black Hat USA 2006.

[Seward and Nethercote 2005] Seward, J. and Nethercote, N. (2005). Using Valgrind to detect undefined value errors with bit-precision. In *USENIX Annual Technical Conference*.

[Skoudis and Zeltser 2003] Skoudis, E. and Zeltser, L. (2003). *Malware: Fighting Malicious Code*. Prentice Hall PTR.

[Smith and Nair 2004] Smith, J. and Nair, R. (2004). *Virtual Machines: Architectures, Implementations and Applications*. Morgan Kaufmann.

[Smith and Nair 2005] Smith, J. E. and Nair, R. (2005). The architecture of virtual machines. *IEEE Computer*, pages 32–38.

[Sugerman et al. 2001] Sugerman, J., Venkitachalam, G., and Lim, B. H. (2001). Virtualizing I/O devices on VMware workstation's hosted virtual machine monitor. In *Proceedings of the 2001 USENIX Annual Technical Conference*, pages 1–14.

[Sêmola 2003] Sêmola, M. (2003). *Gestão da Segurança da Informação - Uma visão executiva*. Campus, Rio de Janeiro.

[Tan et al. 2007] Tan, L., Chan, E. M., Farivar, R., Mallick, N., Carlyle, J. C., David, F. M., and Campbell, R. H. (2007). iKernel: Isolating buggy and malicious device drivers using hardware virtualization support. In *Third IEEE International Symposium on Dependable, Autonomic and Secure Computing*, pages 134–144.

[Uhlig et al. 2005] Uhlig, R., Neiger, G., Rodgers, D., Santoni, A., Anderson, F. M. A., Bennett, S., Kägi, A., Leung, F., and Smith, L. (2005). Intel virtualization technology. *IEEE Computer*.

[Ung and Cifuentes 2006] Ung, D. and Cifuentes, C. (2006). Dynamic re-engineering of binary code with run-time feedbacks. *Science of Computer Programming*, 60(2):189–204.

[VirtualBox 2008] VirtualBox, I. (2008). The VirtualBox architecture. http://www.virtualbox.org/wiki/VirtualBox_architecture.

[VMware 2000] VMware (2000). VMware technical white paper. Technical report, VMware, Palo Alto, CA - USA.

[Whitaker et al. 2002] Whitaker, A., Shaw, M., and Gribble, S. (2002). A scalable isolation kernel. In *Proceedings of the Tenth ACM SIGOPS European Workshop*, Saint Emilion - France.

[Willems et al. 2007] Willems, C., Holz, T., and Freiling, F. (2007). Toward automated dynamic malware analysis using CWSandbox. *IEEE Security and Privacy*, 5(2):32–39.

[Yen 2007] Yen, C.-H. (2007). Solaris operating system - hardware virtualization product architecture. Technical Report 820-3703-10, Sun Microsystems.

[Zovi 2006] Zovi, D. A. D. (2006). Hardware virtualization based rootkits. http://www.blackhat.com. Black Hat USA 2006.