**UNIVERSIDADE FEDERAL DE SANTA CATARINA**
**PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**

**Fabricio Bortoluzzi**

# Aplicação da Análise de Causa Raiz em Sistemas de Detecção de Intrusões

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dra. Carla Merkle Westphall
Orientadora

Florianópolis, junho de 2004

# APLICAÇÃO DA ANÁLISE DE CAUSA RAIZ EM SISTEMAS DE DETECÇÃO DE INTRUSÕES

Fabricio Bortoluzzi

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, Área de Concentração "Sistemas de Computação" e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

_____
Prof. Dr. Raul Sidnei Wazlawick

Banca Examinadora

_____
Prof. Dr. Carla Merkle Westphall

_____
Prof. Dr. Carlos Alberto Maziero

_____
Prof. Dr. Mário Antonio R. Dantas

_____
Prof. Dr. Vitório Bruno Mazzola

_____
Prof. Dr. Alexandre Moraes Ramos

Uma vez definido qual é o verdadeiro
problema, o que não é fácil, a solução não
demora muito a ser encontrada.
--Stephen Kanitz

Dedico este trabalho aos meus pais. Porque tenho uma mãe cuja vida - dedicada docemente à família - é um exercício contínuo de duras provas pessoais. Porque tenho no meu pai, um homem exemplar. Amante da natureza, honesto e educador. Eles souberam dizer "não" nos momentos em que essa resposta era a que eu menos queria e a que eu mais precisava. Sem bons exemplos como estes, eu estou certo que não chegaria nem na metade do caminho que me trouxe a realização deste trabalho. O mérito é de vocês!

Agradeço à Carla por ter acreditado em minha proposta e por me orientar nos estudos para a realização deste trabalho. Agradeço também a todos que participam da minha vida fora do mundo dos estudos e do trabalho: Chris, Lucas, Zé e meus afilhados Marcelo e Enzo. A todos esses e aos amigos do mestrado Felipe, Marcelo, Rafael e Alexandre, desejos sinceros de muito sucesso na vida.

# SUMÁRIO

# LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| BIOS | *Basic Input/Output System* (Sistema básico de entrada e saída) |
| CERT | *Computer Emergency and Response Team* (Equipe de emergência e resposta (a incidentes) de computadores). |
| CPU | *Central Processing Unit* (Unidade central de processamento) |
| CVS | *Concurrent Versions System* (Sistema de controle de versões) |
| FC | Fator de Causa |
| FTP | *File Transfer Protocol* (Protocolo de transferência de arquivos) |
| HTTP | *Hyper Text Transfer Protocol* (Protocolo de transferência de hipertexto) |
| IDS | *Intrusion Detection System* (Sistema de detecção de intrusões) |
| I/O | *Input, Output* (Entrada, saída) |
| IP | *Internet Protocol* (Protocolo Internet) |
| NASA | *National Aeronautics and Space Administration* (Administração nacional da aeronáutica e espaço) |
| NFS | *Network File System* (Sistema de arquivos de rede) |
| NIS | *Network Information System* (Sistema de informações de rede) |
| OpenSSL | *Open Secure Sockets Layer* (Implementação livre da camada de soquetes seguros) |
| POP3S | *Secure Post Office Protocol* (Protocolo seguro de correio) |
| RCA | *Root Cause Analysis* (Análise de causa raiz) |
| SMTPS | *Secure Simple Message Transfer Protocol* (Protocolo seguro de transferência de mensagens simples) |
| SNMP | *Simple Network Management Protocol* (Protocolo simples de gerenciamento de rede) |
| SSH | *Secure Shell* (Protocolo de shell seguro) |
| SSL | *Secure Sockets Layer* (Camada de soquetes seguros) |
| TCP | *Transmission Control Protocol* (Protocolo de controle de transmissão) |
| UDP | *User Datagram Protocol* (Protocolo de datagramas de usuário) |
| URL | *Universal Resource Locator* (Localizador universal de recursos) |
| VPN | *Virtual Private Network* (Rede privada virtual) |

# LISTA DE FIGURAS

# LISTA DE TABELAS

# RESUMO

Os Sistemas de Detecção de Intrusões são ferramentas especializadas na análise do comportamento de um computador ou rede, visando a detecção de indícios de intrusão nestes meios. Entre os benefícios de sua utilização estão a possibilidade de receber notificações na forma de alertas a respeito das intrusões, executar contramedidas em tempo real ou armazenar uma cópia dos pacotes para análise futura.

A exposição dos computadores na Internet e a crescente intensidade na freqüência e variedade de ataques vêm causando uma sobrecarga na quantidade de informações manipuladas e exibidas pelos Sistemas de Detecção de Intrusões aos administradores do sistema. Logo, a busca por novos conceitos para análise dos alertas pode ajudar na tarefa de manter computadores, redes e informações livres de ameaças.

A Análise de Causa Raiz é uma metodologia que permite a investigação detalhada e progressiva de incidentes isolados. Muito utilizada em ambientes industriais, no segmento aeroespacial e na medicina, a Análise de Causa Raiz envolve o estudo de dois ou mais acontecimentos correlacionados com o objetivo de identificar **como** e **por que** um problema aconteceu, de forma que seja possível evitar sua recorrência.

A proposta deste trabalho é aplicar a Análise de Causa Raiz nos Sistemas de Detecção de Intrusões com o objetivo de melhorar a qualidade das informações apresentadas ao administrador do sistema. Este objetivo é alcançado através da aplicação da Análise nas 10 vulnerabilidades mais críticas para os sistemas UNIX segundo o Instituto SANS e na adição de interpretação de Análise de Causa Raiz para as regras correspondentes presentes no Sistema de Detecção de Intrusões Snort.

## ABSTRACT

Intrusion Detection Systems are tools specifically designed to analyze the behavior within a computer or network, looking for intrusion signs. Among the benefits of its use are the possibility to receive notifications about intrusions alerts, to execute real time countermeasures or to store a copy of the packets for further analysis.

Computers exposed to the Internet and the increasing frequency and variety of attacks are causing an overload to the information handled and displayed by IDSs to its administrator. Furthermore, the search to new models to analyze alerts can be useful in the task of keeping computers, networks and information free of threats.

Root Cause Analysis is a methodology that allows the detailed and progressive investigation of isolated incidents. Widely used in industrial environments, aerospace and medicine, Root Cause Analysis put together two or more related incidents, aiming to identify how and why a problem happened, in a way it's possible to avoid it's recurrence.

The proposal of this work is to conceive a novel model of Root Cause Analysis that can be used within Intrusion Detection Systems aiming a better quality in the information presented to the system administrator.

# 1 Introdução

As pessoas se habituaram a utilizar computadores para realizar seus trabalhos de tal forma que fica difícil imaginar a execução da maioria das atividades modernas sem a presença deles. O aumento dessa dependência tecnológica faz emergir neste cenário uma das principais preocupações naturais do homem: a sua segurança.

Para LANDWEHR (2001), estar seguro é encontrar-se livre de ameaças. Logo, um computador está seguro se não existem vulnerabilidades em sua arquitetura, desde o *hardware* até a aplicação. Infelizmente, as arquiteturas em uso atualmente não foram elaboradas para serem seguras por padrão e por isso normalmente não é possível garantir segurança sem inviabilizar sua usabilidade.

A *International Organization for Standardization* (ISO, 2000), em sua norma 17799:2000, define que a informação é tão importante quanto os outros ativos vitais para os negócios, logo, possui valor e consequentemente precisa ser protegida adequadamente. A segurança da informação protege as informações contra uma ampla gama de ameaças, para assegurar a continuidade dos negócios, minimizar prejuízos e maximizar o retorno em investimentos e oportunidades comerciais.

O número de ataques executados contra as redes de computadores conectadas à Internet no Brasil passou dos 50.000 oficialmente registrados em 2003 pelo NIC BR Security Office (NBSO, 2004). A Tabela 1 informa os somatórios nos últimos cinco anos:

| Ano | Ataques reportados |
|---|---|
| 1999 | 3.107 |
| 2000 | 5.997 |
| 2001 | 12.301 |
| 2002 | 25.092 |
| 2003 | 54.607 |

**Tabela 1. Número de incidentes reportados ao NBSO nos últimos anos.**
**Fonte: NBSO, 2004.**

A observação destes valores indica que o número de ataques tem praticamente dobrado a cada ano. Os dados do *Computer Emergency and Response Team* (CERT, 2004) indicam que esta tendência é mundial (Tabela 2).

| Ano | Ataques reportados |
|---|---|
| 1999 | 9.859 |
| 2000 | 21.756 |
| 2001 | 52.658 |
| 2002 | 82.094 |
| 2003 | 127.529 |

**Tabela 2. Número de incidentes reportados ao CERT nos últimos anos.**
**Fonte: CERT, 2004.**

Isto se deve em princípio ao aumento do número de *hosts* que estão conectados à Internet. Entretanto, outros fatores também estão associados: Um deles é a automação das ferramentas que examinam sub-redes em busca de vulnerabilidades. Programadores mal intencionados conseguem agora elaborar, por exemplo, códigos que exploram falhas nos sistemas operacionais onde o computador infectado inicia automaticamente uma varredura em busca de outros sistemas igualmente vulneráveis, sem intervenção humana. Este procedimento automatizado infecta um número infinitamente maior de computadores do que as formas anteriores de propagação, pois neste não é necessário ações humanas como a execução de anexos de e-mail ou aceitação de instalação de código de terceiros.

Os Sistemas de Detecção de Intrusões conquistaram espaço em meio aos demais mecanismos de segurança para ajudar na proteção da informação. Com a sua utilização, é possível saber em tempo real quais os ataques em andamento e eventualmente, impedir tais ataques de obter sucesso. Entretanto, da mesma forma que os ataques se tornaram automatizados, é preciso automatizar a maior parte possível das tarefas de análise de ataques desempenhada pelos Sistemas de Detecção de Intrusões. É na busca por este objetivo que o presente trabalho está direcionado. O experimento proposto é a aplicação da metodologia de Análise de Causa Raiz, muito utilizada na investigação de acidentes de escala industrial, nos Sistemas de Detecção de Intrusões, de forma a passar para o computador a maior parte possível das tarefas associadas à análise dos alertas.

## 1.1 Justificativa

A necessidade por Sistemas de Detecção de Intrusão tem base na premissa de que os sistemas computacionais e as redes de computadores devem oferecer confidencialidade, integridade e disponibilidade para seus usuários (MOFFET, 1995).

Os Sistemas de Detecção de Intrusões são ferramentas especializadas na análise do comportamento de um computador ou rede, visando a detecção de indícios de intrusão nestes meios. Entre os benefícios de sua utilização estão a possibilidade de receber notificações na forma de alertas a respeito das intrusões, executar contramedidas em tempo real ou armazenar uma cópia dos pacotes para análise futura.

A exposição dos computadores na Internet e a crescente intensidade na freqüência e variedade de ataques vêm causando problemas no processo de exibição das informações apresentadas ao administrador destes sistemas. As duas principais deficiências encontradas foram a excessiva e quantidade de alertas gerados para um mesmo incidente e a falta de informações de como consertar os problemas de segurança relatados nestes eventos.

A busca por conceitos que venham a contribuir no aumento da qualidade dos processos de exibição de informações tratadas pelos IDSs, pode ajudar na tarefa de manter computadores, redes e informações livres de ameaças.

Este trabalho é importante porque apresenta uma nova abordagem na análise e interpretação de alertas capaz de reduzir a quantidade de informações desnecessariamente exibidas pelos IDSs ao mesmo tempo em que aumenta a sua qualidade, ao associar descrições que levam o administrador diretamente à causa raiz dos motivos pelos quais seus sistemas sofreram a intrusão, permitindo assim a aplicação de soluções que irão evitar a reincidência destes eventos.

## 1.2 Objetivos

### 1.2.1 Objetivo Geral

O objetivo geral deste trabalho é propor a aplicação da metodologia de Análise de Causa Raiz no contexto dos Sistemas de Detecção de Intrusões.

### 1.2.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Executar a Análise de Causa Raiz em um conjunto de vulnerabilidades críticas existentes nos sistemas UNIX e seus serviços de rede.

- Aplicar a Análise de Causa Raiz nos Sistemas de Detecção de Intrusões.

- Implementar as sugestões geradas durante as etapas anteriores no Snort e descrever o funcionamento da metodologia neste contexto.

## 1.3 Estrutura do trabalho

O capítulo seguinte, "Segurança da Informação", apresenta os conceitos básicos de segurança da informação. A abordagem resulta em um esquema conceitual único, que agrega a classificação dos *hackers* de acordo com suas motivações e a dinâmica entre as funções, abordagens e estratégias de segurança. O capítulo 3, "Sistemas de Detecção de Intrusão", apresenta a importância deste mecanismo, sua arquitetura, as estratégias de controle e a forma como os alertas são processados. No capítulo 4, "Análise de Causa Raiz", apresenta-se o segundo tema geral deste trabalho, a Análise de Causa Raiz, sua origem e áreas de aplicação, além de um exemplo de fácil compreensão. O capítulo 5, "Análise de Causa Raiz aplicada aos Sistemas de Detecção de Intrusões" apresenta o procedimento de utilização da Análise de Causa Raiz no contexto dos Sistemas de Detecção de Intrusão. O capítulo 6 "Análise de Causa Raiz das 10 vulnerabilidades mais críticas dos ambientes UNIX" apresenta os resultados da aplicação da Análise de Causa Raiz nas 10 vulnerabilidades mais críticas para os sistemas UNIX de acordo com o Instituto SANS (SANS, 2004). Por fim, o Capítulo 7, "Aplicação da Análise de Causa Raiz no IDS Snort" apresenta as modificações feitas no IDS Snort para contemplar a Análise de Causa Raiz e fornece um mapeamento entre os Fatores de Causa da metodologia e as regras do IDS.

# 2 Segurança da Informação

A necessidade por segurança tem sua causa principalmente devido à ação dos *hackers*. Neste trabalho, o termo *hacker* se aplica ao indivíduo que emprega esforços para acessar computadores e redes de computadores que não estão sob sua responsabilidade profissional, indiferente as suas intenções.

## 2.1 Classificação dos intrusos

ZWICKY (2000) classifica os intrusos da seguinte forma:

- Aventureiros: Invadem sistemas sem nenhuma finalidade específica além da curiosidade em ver o que encontram ao conseguir o acesso, normalmente sem modificar ou comprometer as informações acessadas.

- Competidores: intrusos, normalmente organizados em grupos, que competem entre si com o objetivo de invadir o maior número de alvos possível.

- Vândalos: invadem computadores com a finalidade de modificar informações. As motivações geralmente são normalmente o desgosto pessoal pela entidade que mantém o sistema ou a vontade de expressar uma opinião contrária aos princípios da organização alvo.

- Espiões profissionais: geralmente apresentam um conhecimento maior que os anteriores e elaboram suas próprias ferramentas para conduzir ataques objetivando a obtenção de segredos comerciais/industriais das concorrentes a empresa que lhe contrata.

É certo que, para cada tipo de *hacker* existe um nível de proteção adequado. A problemática está centrada no valor da informação. Quanto mais valiosos forem os dados, mais *hackers* estarão interessados em obtê-los e maior será a presença dos espiões profissionais entre eles. Então, é preciso haver mais e melhores mecanismos de segurança para proteção dessa informação.

## 2.2 Funções de segurança

Segundo MOFFET (1995), um sistema, para ser considerado seguro, precisa cumprir

com três premissas:

- confidencialidade: garantir que as informações sejam acessíveis apenas para aqueles que estão autorizados a acessá-las;
- integridade: salvaguardar a exatidão e a inteireza das informações e métodos de processamento;
- disponibilidade: assegurar que os usuários autorizados tenham acesso às informações e aos ativos associados quando necessário.

## 2.3 Abordagens de segurança

A garantia do cumprimento das três premissas de segurança existem sob forma de abordagens. Uma abordagem de segurança agrega conceitos que, quando adotados, podem ajudar na proteção da informação. ZWICKY (2000) definiu quatro:

- Ausência de segurança.
- Segurança através de obscuridade.
- Segurança nos *hosts*.
- Segurança na rede.

### 2.3.1 Ausência de segurança

Negligenciar qualquer uma dessas abordagens é, por resultado, uma abordagem. Este caso assume a confiança total na qualidade do *hardware* e/ou *software* fornecido pelo fabricante e não estipula planos de correção de vulnerabilidades nem o impacto da perda das informações que necessitam proteção.

Apesar de ser claramente insuficiente, essa abordagem acaba se acontecendo em muitas organizações - notadamente as de pequeno porte - que normalmente alegam não poder manter profissionais ou contratos de manutenção especializados em segurança e proteção da informação.

### 2.3.2 Segurança através da obscuridade

Outra abordagem pouco eficaz é a chamada segurança por obscuridade. Neste caso presume-se segurança através da suposição de que o computador ou o serviço não é publicamente conhecido - nem sua existência, conteúdo e/ou código-fonte.

A deficiência desta abordagem está na fragilidade de suas suposições. A ligação entre redes privadas e a Internet, por exemplo, envolve a necessidade de se expor informações, abrindo desde o início das operações, caminhos que podem ser explorados para a descoberta dos serviços ditos obscuros.

### 2.3.3 Segurança nos *hosts*

A abordagem de mecanismos de segurança inseridos para execução dentro dos computadores emerge como uma solução mais eficaz. Nesta abordagem, aplicam-se políticas, configurações, *softwares* adicionais e outros controles como forma de evitar que as vulnerabilidades da plataforma sejam exploradas e tornem-se uma ameaça.

O mecanismo mais antigo e conhecido para garantir confidencialiade é a criptografia (CAMPELLO & WEBER, 2001). Seu uso começou muito antes e foi um dos principais impulsionadores da existência dos computadores. Na abordagem de segurança em *host*, a criptografia existe sob forma de aplicativos que cifram textos e arquivos através de chaves simétricas e assimétricas, em *softwares* de esteganografia, na implementação de partições de disco rígido cifradas e várias outros.

No aspecto da garantia de integridade da informação, um dos mecanismos mais utilizados é o *software* antivírus. Através da leitura dos arquivos locais e a comparação com uma base de assinatura de vírus, este *software* acusa a presença de código malicioso e procede com a remoção do mesmo. Técnicas complementares como a monitoração residente e em tempo real melhoram a eficácia deste mecanismo.

Uma deficiência da abordagem é mostrar-se inadequada para utilização em grandes organizações. Na medida em que o número de computadores cresce, torna-se mais difícil a administração do conjunto e neste caso, inviabiliza-se uma administração única e centralizada.

### 2.3.4 Segurança na rede

Partir para soluções centradas na rede de computadores emergiu como a abordagem naturalmente mais eficaz porque permite que uma única ação de proteção tenha efeito em segmentos inteiros de estações de trabalho.

Esta abordagem contempla vários mecanismos. Alguns estendem os benefícios da anterior, como é o caso do antivírus de *gateway* e a criptografia em nível de rede, outros surgiram especificamente devido à necessidade de interligação em rede:

- Autenticação;
- *Firewalls*; e
- *Sistemas de Detecção de Intrusões*.

#### 2.3.4.1 Autenticação

A autenticação é o mecanismo concentrado em rede que verifica se um usuário é quem ele diz ser. É também a base na qual se implantam os recursos de controle de acessos e auditoria (ISRAEL & LINDEN, 1983). Os mecanismos de autenticação mais comuns são:

- *login* e senha: autenticação através da comparação entre uma senha informada e a sua correspondente armazenada de forma cifrada (SAMAR, 1996).
- cartões inteligentes: entre as várias utilidades dos emergentes *smart cards*, estes cartões podem armazenar de forma segura as informações necessárias para autenticar usuários sem que estes precisem necessariamente memorizar senhas (SHELFER & PROCACCINO, 2002).
- *tokens*: dispositivo que armazena uma chave criptográfica secreta, usada para autenticação através de negociação *challenge-response* (desafio e resposta). Este mecanismo normalmente envolve a existência do *token* em si associado à necessidade de fornecimento de uma senha, elevando consideravelmente o nível de segurança da informação (SANDHU & SAMARATI, 1996).

- biometria: identificação e autenticação do usuário com base em suas características corporais únicas, como impressão digital, voz e desenho da íris e/ou retina ocular (JAIN, HONG & PANKANTI, 2000).

2.3.4.2 Firewall

*Firewall* é uma coleção de componentes posicionados entre duas redes para proteger o segmento privado de um acesso não autorizado (SILVER, 1995). É nele que se aplica o cumprimento da maioria das restrições de privilégios impostas na política de segurança da informação (AL-TAWIL & AL-KALTHAM, 1999). A Figura 1 mostra o posicionamento de um *firewall* da forma como ele é utilizado em grande parte das topologias de rede atuais, interligando LAN, redes remotas e a Internet com as devidas restrições de segurança:

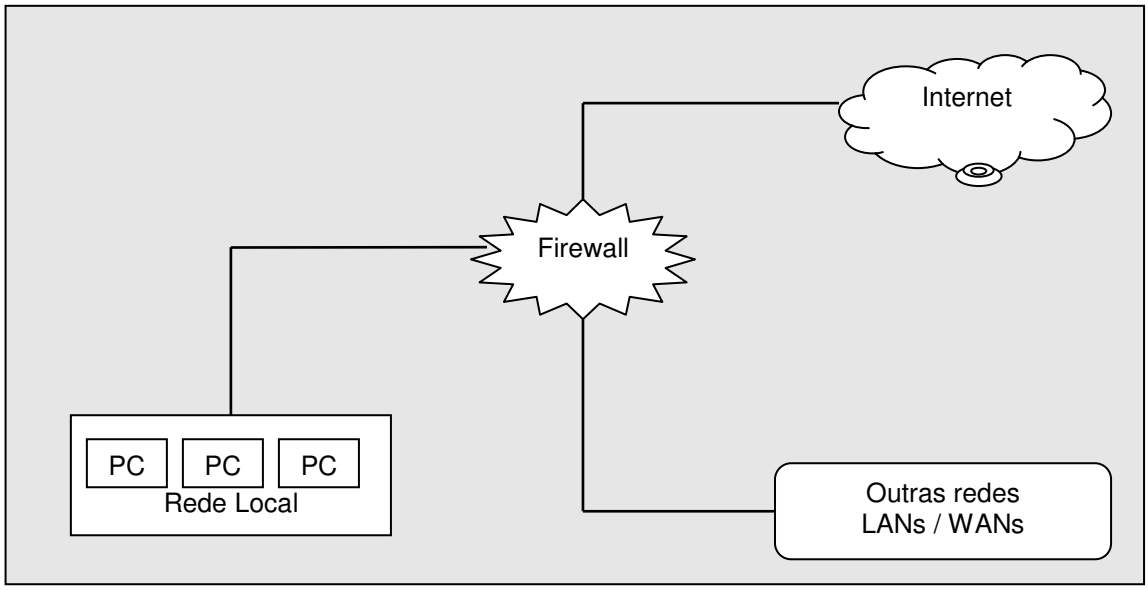



**Figura 1. Posicionamento do firewall na topologia da rede.**

O *firewall* é um ponto de choque muito eficiente para a implementação dos controles solicitados na política de segurança (CAMPELLO & WEBER, 2001). Com este mecanismo é possível:

- restringir acessos com origem na rede local e destino nas demais redes (e vice-versa);

- bloquear a ação dos *hackers* e seus ataques, impedindo que conexões em portas lógicas não especificadas sejam sempre filtradas.

- impedir a utilização de serviços que possam levar a infração de direitos autorais como o *peer-to-peer* e o FTP (*file transfer protocol*); e

- intermediar as conexões de forma a armazenar registros visando auditoria de incidentes.

Todas estas possibilidades solidificam a limitação de exposição da rede local para com as demais, principalmente a Internet. Entretanto, um *firewall* não pode proteger a rede e os computadores nas seguintes situações:

- Funcionários mal intencionados, pois o *firewall* não é ponto de choque em conexões com origem e destino na própria LAN (*Loca Area Network*).

- Novas ameaças: uma vez que a configuração do *firewall* é estática, novas ameaças irão requerer novas filtragens e até que isso aconteça, outros mecanismos devem suprir tal deficiência.

- Vírus e conteúdo da camada de aplicação: o *firewall* se restringe à análise do endereço IP e das portas lógicas, ele não pode impedir o alastramento de vírus e similares.

2.3.4.3 Sistemas de detecção de intrusões

Sistemas de Detecção de Intrusões são sistemas que analisam informações em um computador ou uma rede em busca de evidências de intrusões (VIGNA et al, 2003). Os IDSs devem ser precisos, adaptativos e extensíveis (LEE & STOLFO, 2000).

Alguns resultados comuns obtidos através do uso de IDSs:

- Ser alertado sobre a exploração de falhas de segurança em computadores.

- Ser alertado sobre varreduras de portas lógicas (*port scanning*) das redes IP.

- Reconfiguração automática do *firewall* da rede local em tempo real, de modo a bloquear conexões antes que o atacante obtenha êxito.

Detecção de intrusão é o foco deste trabalho. O capítulo 4 descreve os motivos que

definem a importância destes sistemas, bem como apresenta as diferentes classificações encontradas e estende o assunto sob nova abordagem.

## 2.4 Estratégias de segurança

Nenhuma abordagem pode, sozinha, contemplar todas as necessidades de proteção. É preciso utilizá-las em conjunto na formação de estratégias de segurança que incluam os aspectos positivos de cada estratégia. ZWICKY (2000) também apresenta essas estratégias:

- Menor privilégio: é o princípio de que um objeto (usuário, programa, processo, etc.) deve possuir apenas os privilégios necessários para a execução de suas atividades legítimas.

- Defesa em profundidade: É importante que um mecanismo, quando implantado, possua todos os recursos necessários para sua execução e que, na eventualidade de falhas no próprio mecanismo, exista outro redundante para assumir sua função.

- Ponto de choque: Existem equipamentos obrigatórios para a passagem da informação na topologia de uma rede. Roteadores e *firewalls* são pontos de choque entre a rede local e a externa. Pontos como esses oferecem possibilidades de utilização de mecanismos adicionais, como a filtragem de conexões externas e a detecção de tentativas de invasão, pois todo tráfego potencialmente malicioso com origem externa passa obrigatoriamente por este local.

- Elo mais fraco: consiste na avaliação permanente das deficiências nos mecanismos. A segurança como um todo é tão eficiente quanto seu elo ou mecanismo mais fraco. Apesar de sempre haver um ponto mais fraco, é preciso equalizar as soluções para contemplar as diferentes necessidades.

- Tolerância à falhas: para garantir a premissa de disponibilidade, é preciso elaborar medidas de tolerância à falhas. O desejável neste aspecto é que um sistema, mesmo sob ataque, deve permanecer disponível aos usuários legítimos.

Na prática isto é muito difícil de acontecer. Exemplos como o ataque de recusa (ou negação) de serviço mostram que durante a inundação de requisições, os acessos legítimos ficam sem resposta. Espera-se, entretanto que, mediante a um ataque como este, o sistema não tenha seus recursos permanentemente danificados, voltando ao estado normal de operação logo após o bloqueio do atacante.

- Participação de todos: Apesar da garantia de segurança não poder residir nas ações dos usuários, deseja-se ao menos que a conscientização quanto as ameaças exista de tal forma que as pessoas estejam mais propensas a colaborar do que a tomarem atitudes negligentes. Para isto é necessário manter canais abertos para que todas as pessoas possam expressar suas opiniões a respeito do que deve ou não ser permitido no ambiente protegido. Isto pode acontecer de forma voluntária, onde os interessados interagem com o administrador da segurança por interesses próprios, ou de forma involuntária, onde existe a obrigatoriedade de interação profissional entre os interessados na gestão da informação. Em ambos os casos, pode-se esperar comprometimento e boa vontade bem como protestos e conflitos. Cabe ao administrador da segurança saber lidar diplomaticamente com todos os casos, sem criar favorecimentos impróprios ou punições e bloqueios desnecessários.

- Diversidade de defesas: Por mais que uma única solução pareça contemplar todas as necessidades de proteção, é importante diversificar as abordagens e os fabricantes, de forma que a falha em um desses mecanismos seja amenizada pela presença de outro.

- Simplicidade: Escolher soluções de segurança simples facilita a compreensão e utilização delas. A complexidade pode esconder falhas e gerar a necessidade de mais mecanismos.

CAMPELLO & WEBER (2001) perceberam que a quinta estratégia tem duas particularidades. Primeira: tolerância à falhas é uma área mais abrangente da computação que tenta garantir a contínua disponibilidade do sistema. Neste conceito, está previsto que o *hardware* ou o *software* também podem falhar sem nenhuma

interação humana, ficando a cargo da segurança somente os casos onde existe a ação deliberada de uma pessoa. Segunda: existem duas posturas possíveis mediante a falhas: permitir o trânsito subseqüente da informação ou bloqueá-lo. Enquanto ser permissivo parece razoável porque mantém o restante das comunicações em operação, é mais prudente bloquear os acessos, mesmo que legítimos, sob risco de estar se expondo serviços não permitidos e inseguros.

## 2.5 Ameaças e vulnerabilidades

Uma vez que os mecanismos de proteção estejam corretamente inseridos nas estratégias de segurança e estes implementados nas várias estratégias, resta mapear as vulnerabilidades e ameaças que podem causar danos a informação.

SOARES et al (1995) define que a ameaça consiste na possibilidade de violação da segurança de um sistema e as classifica da seguinte forma:

- destruição da informação ou do sistema onde ela está armazenada;
- modificação da informação;
- roubo e/ou perda da informação e outros recursos;
- exposição de informação confidencial;
- interrupção no serviço de disponibilização da informação.

Notadamente, todas as ameaças têm relação direta com as funções de segurança. Já as vulnerabilidades, sob as quais as ameaças se expõem, são (BACE & MELL, 2001):

- Erro de validação de entrada. Vulnerabilidades associada à implementação dos sistemas e *softwares* servidores causada por falhas na forma como a informação entrante é tratada.
- Erro de validação de acesso. Deficiências na construção dos mecanismos (autenticação por exemplo) podem abrir caminho para que um *hacker* obtenha acesso sem a necessidade de se autenticar.

- Erro de manipulação de exceções. Exceções inesperadas e não tratadas podem tornar o sistema vulnerável.

- Erro de ambiente. Aspectos ambientais, como os recursos de segurança física, trancas, controle de temperatura, controle contra incêndio, alarmes etc., precisam estar sob controle para evitar que computadores considerados seguros sejam vulneráveis através da depredação física.

- Erro de configuração: não tem relação com a forma de construção do sistema, mas do eventual despreparo do administrador em configurá-lo.

- Condição de corrida: Esta vulnerabilidade existe quando explora-se o curto espaço de tempo em que um processo verifica se uma operação é permitida e a efetiva execução desta operação.

A concretização de alguma ameaça, ocasionada por exploração de vulnerabilidades através de ação deliberada e intencional, configura um ataque. A Figura 2 ilustra os tópicos abordados até o momento, apresentando de que forma eles se inter-relacionam:

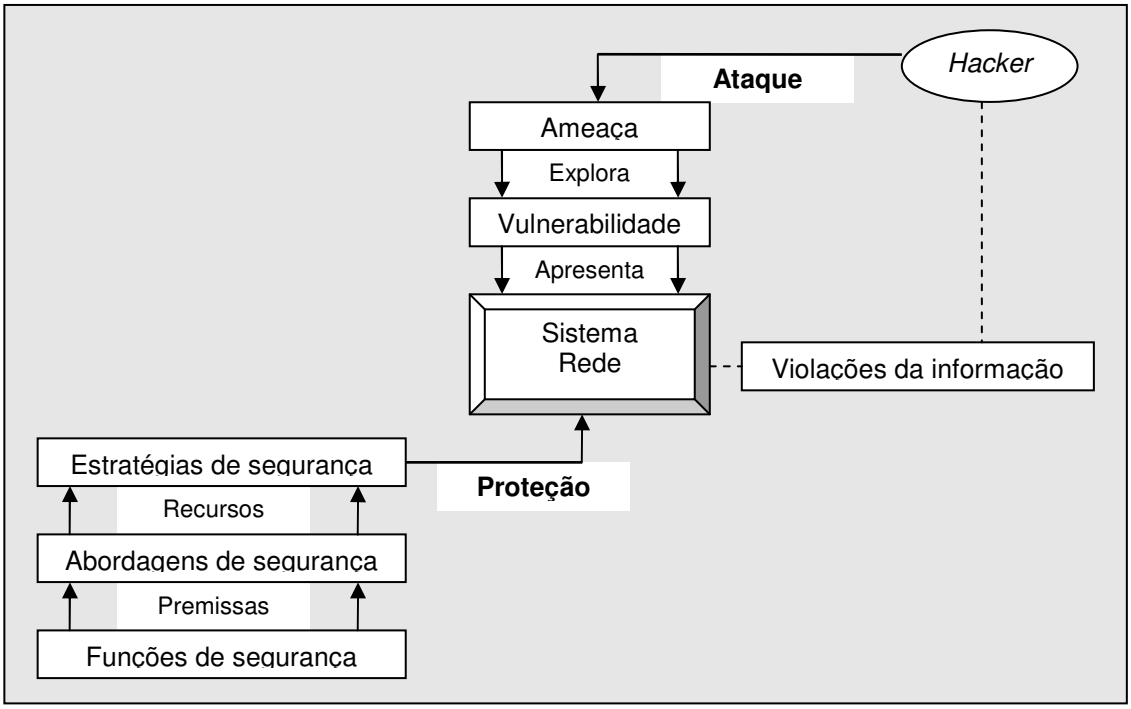



**Figura 2. Esquema conceitual de segurança e ação dos hackers.**

As funções de segurança podem ser vistas como premissas na elaboração de estratégias de proteção da informação. Essas estratégias devem ser aplicadas na efetiva proteção dos sistemas e da rede. Do ponto de vista do *hacker*, os ataques acontecem a partir de ameaças que exploram as eventuais vulnerabilidades apresentadas por estes mesmos sistemas ou equipamentos de rede, causando a violação da informação.

Este trabalho está focado nas estratégias de segurança para proteção da informação, mais particularmente na estratégia de defesa através da utilização dos Sistemas de Detecção de Intrusão.

# 3 Sistemas de Detecção de Intrusão

Sistemas de Detecção de Intrusão (do inglês, *Intrusion Detection Systems* - IDSs) são sistemas que automatizam o processo de monitoração de eventos que ocorrem em um computador ou rede, analisando-os em busca de problemas de segurança (BACE & MELL, 2001).

Este capítulo descreve os fundamentos em comum a todos IDSs, as arquiteturas existentes, os objetivos para sua implantação, as formas de gerenciamento, as abordagens de análise, as origens e o processamento dos alertas, e, por fim, as opções de resposta que os IDSs provêem, de acordo com a seguinte classificação:

- Fundamentos em comum;
- Arquitetura;
- Objetivos: contabilização ou resposta;
- Estratégia de controle e gerenciamento
- Processamento dos alertas: em lote, em tempo real;
- Origem de coleta das informações: *host*, aplicação ou rede;
- Sistema de análise: mau uso ou anomalia;
- Opções de resposta: ativa ou passiva;

## 3.1 Fundamentos em comum

Existem vários tipos de Sistemas de Detecção de Intrusão, caracterizados por diferentes formas de monitoração e abordagens de funcionamento. Entretanto, todos compartilham de três fundamentos (BACE & MELL, 2001), (CAMPELLO & WEBER, 2001):

- Origem da informação;
- Análise; e

- Resposta.

A Figura 3 exibe o ciclo de funcionamento dos IDSs. As diferentes origens de informação são usadas para determinar quando uma intrusão acontece. Essas origens podem ser obtidas nas várias camadas de construção de uma plataforma, como a rede, o computador (*host*) ou a aplicação que está em execução.

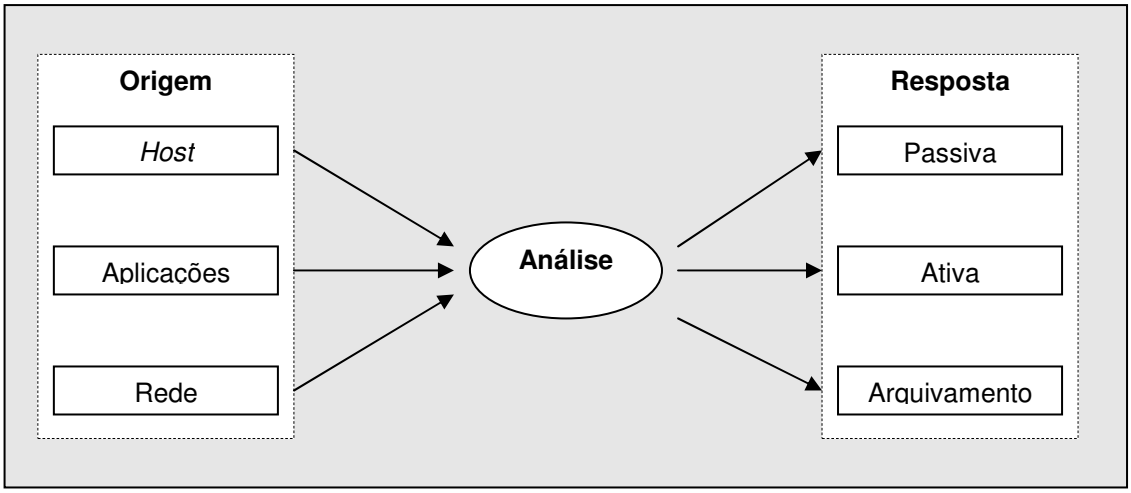



**Figura 3. Fundamentos comuns a todos IDSs.**
**Adaptado de (BACE & MELL, 2001).**

O núcleo de um IDS é seu motor de análise. Ele organiza os dados obtidos em informação que pode ser analisada e decide quando um evento lido é considerado evidência de um ataque. As abordagens mais comuns são a detecção de mau uso e a detecção de anomalia, abordadas em detalhes na seção 3.7.

Já o subsistema de resposta existe para contemplar as ações possíveis mediante a caracterização ou detecção de um ataque. As ações possíveis são classificadas em ativas quando há bloqueio ou contra-ataque, passivas quando são informativas, e de arquivamento quando são mantidas para análise estatística posterior. Elas estão mais detalhadas na seção 3.8.

## 3.2 Arquitetura de execução

A arquitetura de execução dos IDSs se refere a como os componentes funcionais estão dispostos e de que forma eles se inter-relacionam. Os componentes essenciais são o computador (*host*) que hospeda o IDS e o alvo que será monitorado (CAMPELLO & WEBER, 2001).

- Execução local: neste modo, o Sistema de Detecção de Intrusões entra em execução no mesmo computador que precisa de proteção. Foi deste modo que o IDS surgiu. Mostrou-se inadequado quando se percebeu que a proteção de execução local podia ser desabilitada e apagada caso o ataque detectado obtivesse êxito.

- Execução remota: Neste conceito, o IDS entra em execução em um computador diferente dos alvos aos quais protege. Assim, posicionado estrategicamente entre a rede pública e a rede local, pode intermediar de forma mais eficaz na detecção de intrusões.

## 3.3 Fundamentos segundo os objetivos

Apesar dos objetivos de todos os mecanismos de proteção serem praticamente os mesmos, os Sistemas de Detecção de Intrusões apresentam duas particularidades (BACE & MELL, 2001):

- Responsabilização. Os IDSs permitem a ligação de uma atividade suspeita ou alerta à origem responsável pela iniciativa. Neste enfoque, o IDS fornece recursos para levantamento de provas que possam incriminar o atacante. Isto acontece através dos próprios alertas ou então por registros (por exemplo, *sniffing*) do tráfego subseqüente à emissão do alerta.

- Resposta. Define a capacidade de um IDS em agir mediante a emissão de um alerta. Na maioria das vezes, a resposta desejável, além do armazenamento dos alertas e demais evidências, é o bloqueio preventivo da origem que inicia o ataque. Entretanto, a decisão de se implementar ou habilitar tal medida precisa ser cuidadosamente conduzida de forma a evitar que o uso legítimo também venha a ser bloqueado, causando assim a violação da premissa de disponibilidade.

## 3.4 Estratégia de controle e gerenciamento

As estratégias de controle descrevem como os elementos de um IDS são controlados e de que forma o subsistema de entrada e saída é gerenciado. As possibilidades, segundo (JACKSON, 1999), (BACE & MELL, 2004) são:

- Gerenciamento centralizado: Todos os elementos do IDS, monitoração, detecção e emissão de alarmes são instalados em um único local.

- Parcialmente distribuído: Monitoração e detecção acontecem em vários enlaces na topologia da rede. O gerenciamento de alertas é configurado no centro de operações.

- Completamente distribuído: Monitoração e detecção funcionam através da abordagem de agentes, onde as decisões de resposta são tomadas no mesmo ponto em que ocorre a análise dos alertas.

Cada caso tem particularidades que se aplicam as diferentes topologias de rede. Em redes pequenas, é comum a utilização de um só IDS, enquanto em redes de médio porte, o console também pode operar com um dos nós de detecção. Já em ambientes maiores, a tarefa do console é dedicada, envolvendo apenas a interface de apresentação e o repositório de dados.

## 3.5 Processamento dos alertas

Este aspecto diversifica os IDSs com relação a janela de tempo que existe entre a detecção, análise e resposta aos eventos detectados (DEBAR *et al*, 1998):

- Processamento em lote: Nos primeiros IDSs, a principal origem de dados eram os registros do sistema operacional, assim, a análise ficava agendada para execução de tempos em tempos, operando no modo de processamento em lotes. A deficiência deste modelo é que não há possibilidade de agir no bloqueio do ataque.

- Processamento em tempo real: A análise dos dados à medida que eles fluem pela rede é uma característica comum dos IDSs modernos, voltados para operação em rede. Neste caso, os segmentos de dados são remontados e o fluxo é analisado por inteiro. No momento em que se detecta uma anomalia ou assinatura de ataque, o IDS é capaz de tomar várias ações, inclusive de agir na reescrita ou bloqueio dos pacotes que caracterizam a invasão.

## 3.6 Origem da coleta de informações

A classificação mais comum de Sistemas de Detecção de Intrusões é o agrupamento com base no critério das diferentes origens de coleta de informações. Alguns detectam ataques através da escuta de um segmento de sub-rede, outros estão posicionados no meio de ligações entre *backbones*. Outros ainda, focam seus recursos na interpretação de dados colhidos do sistema operacional alvo da monitoração. De acordo com CAMPELLO & WEBER (2001) e BACE & MELL (2001), as três principais classificações segundo a origem da coleta de informações são:

- IDSs baseados em rede.
- IDSs baseados em *host*; e
- IDSs baseados em aplicação;

### 3.6.1 IDSs baseados em rede

A maioria dos Sistemas de Detecção de Intrusões - *open source* e proprietários - são baseados em rede. Esses IDSs detectam ataques através da captura e análise dos pacotes em uma rede. Através da escuta em um segmento, um IDS baseado em rede pode buscar evidências de ataques e proteger vários *hosts* ao mesmo tempo.

IDSs baseados em rede geralmente consistem em um conjunto de sensores posicionados em vários pontos de uma rede. Estes sensores monitoram o tráfego, executam análise local e reportam os ataques ao console central de gerenciamento. Considerando que este tipo de IDS tem *hardware* dedicado exclusivamente a sua execução, garantir a segurança deste dispositivo contra ataques é tarefa simples se comparada com os demais modelos.

### 3.6.2 IDSs baseados em *host*

Os Sistemas de Detecção de Intrusões baseados em *host* trabalham com informações originadas no sistema operacional e aplicações de um computador. Neste funcionamento, é possível coletar e analisar as atividades com grande precisão e determinar exatamente quais processos e usuários estão envolvidos em ataques. Além disto, diferentemente das demais abordagens de detecção, esta pode prever as tentativas

de ataque, pois o programa acessa e monitora diretamente os arquivos de dados, registros e os processos do sistema.

IDSs baseados em *host* normalmente utilizam origens de informações de dois tipos: rastros de auditoria e registros do sistema (*logs*). Rastros de auditoria são gerados no centro de funcionamento do sistema operacional, seu *kernel* e, portanto, são mais bem detalhados e estão mais bem protegidos contra acessos não legítimos. Porém, os registros do sistema são mais objetivos, mais simples e menores, sendo assim mais fáceis de entender e utilizar.

Alguns IDS baseados em *host* são projetados para operar de acordo com uma central de gerenciamento, localizada em computador remoto. Deste modo, o IDS recebe instruções e emite relatórios para a estação de gerenciamento, permitindo que um console reporte a análise de vários detectores.

Principais vantagens:

- IDSs baseados em *host* tem a capacidade de monitorar eventos locais e detectar ataques que não podem ser vistos por IDSs baseados em rede;

- IDSs baseados em *host* não são afetados pelo chaveamento (*switching*) e outras tecnologias que dificultam o funcionamento esperado dos IDSs baseados em rede.

- Quando os IDSs baseados em *host* se reportam a um console central, é possível analisar seus comportamentos de forma global, útil para o estabelecimento dos padrões de estado, por exemplo: normal, anormal, sob-ataque, etc.

Principais desvantagens:

- O modelo de IDS baseado em *host* é o mais difícil de ser gerenciado, porque as configurações precisam ser distribuídas e modificadas em cada um dos computadores envolvidos.

- Como o *host* que detecta intrusões é o mesmo no qual a aplicação está em execução, é possível para o atacante formular meios de ataque que desabilitem o

funcionamento do IDS.

- IDSs baseados em *host* consomem recursos de processamento nos computadores onde estão em execução, isto precisa ser considerado no planejamento de implantação destes sistemas.

### 3.6.3 IDSs baseados em aplicação

Sistemas de Detecção de Intrusões baseados em aplicação são um subtipo dos IDSs baseados em *host*. A diferença é que neste caso analisa-se os registros de eventos gerados pelo *software* na camada de aplicação e não no sistema operacional ou rede.

A possibilidade de intervir diretamente na aplicação permite a este tipo de IDS detectar comportamento suspeito inclusive nos casos de usuários legítimos que tentam danificar ou abusar de suas credenciais de autorização e demais mecanismos de acesso.

Principais vantagens:

- IDSs baseados em aplicação podem monitorar a interação entre o usuário e a aplicação, onde é possível rastrear atividades não-autorizadas, de usuários legítimos ou não.

- IDSs baseados em aplicação podem operar em ambientes onde o tráfego da rede está cifrado e analisar a informação antes dela passar pelo processo de cifragem.

Principais desvantagens:

- Os IDSs baseados em aplicação são mais vulneráveis porque os registros sob os quais operam são passíveis de adulteração por processos de usuários.

- IDSs baseados em aplicação são executados no nível mais alto de abstração das

## 3.7 Sistema de análise: mau uso ou anomalia

Há duas abordagens para analisar eventos e detectar ataques: detecção de mau uso e detecção de anomalias (CAMPELLO & WEBER, 2001), (BACE & MELL, 2001). Detecção de mau uso é a técnica usada pela maioria dos IDSs comerciais e objetiva encontrar comportamento previamente conhecido como indesejado. Já a detecção de

anomalias, na qual se monitora padrões anormais de atividade, continua sendo assunto de pesquisa. Há vantagens e desvantagens associadas a cada uma dessas abordagens e, ao menos nesta fase de evolução dos IDSs, a melhor escolha parece ser pelos sistemas de detecção de mau uso que associam algumas técnicas de detecção de anomalias.

### 3.7.1 Detecção de mau uso

Detectores de mau uso analisam a atividade do sistema, buscando por eventos ou conjuntos de eventos que correspondem com padrões previamente definidos que descrevem um ataque. Estes padrões são chamados "assinaturas", então, outra nomenclatura comum para este tipo de arquitetura é Sistema de Detecção de Intrusão baseado em assinatura.

Vantagens:

- A técnica de detecção de mau uso é muito eficiente e extremamente precisa porque gera poucos falsos positivos (situação indesejada em que o IDS gera um alerta para um evento legítimo).

- Detectores de mau uso podem rapidamente diagnosticar a ferramenta ou técnica utilizada para o ataque.

- Este tipo de detecção favorece o administrador do sistema, que pode trabalhar pro ativamente e executar contra-medidas mais eficazes.

Desvantagens:

- A detecção de mau uso está restrita a ataques previamente conhecidos pelo fabricante do IDS, logo, é necessária a existência de mecanismos eficientes de atualização das assinaturas, da mesma forma que os antivírus.

- A detecção de mau uso é estática, logo, o IDS não consegue acompanhar pequenas alterações evolutivas nas ferramentas de ataque sem aumentar proporcionalmente a possibilidade de falsos positivos.

### 3.7.2 Detecção de anomalias

Detectores de anomalia identificam comportamento não usual em um *host* ou rede. Eles assumem que os ataques são eventos que levam o sistema a um estado anormal. Estes detectores constroem perfis que representam o estado normal de comportamento dos usuários, *hosts* e conexões de rede. Estes perfis são resultado da coleta e formação de histórico a respeito de cada um dos objetos monitorados durante um certo período de execução considerada normal.

As técnicas de coleta de informações e detecção de anomalias são várias, entre elas:

- Estabelecimento de limiares: onde certos atributos do usuário e do sistema são expressos em termos de quantidade e nivelados dentro do que se estipula como aceitável.

- Medições estatísticas: através do aprendizado do que é normal ou anômalo a partir de atributos monitorados localmente (carga da CPU, memória e I/O de rede) ou através da utilização de conjuntos de dados pré-existentes, fornecidos pelo fabricante.

- Medições baseadas em regras, onde os dados observados definem o padrão aceitável de uso, com a diferença que neste caso a entrada acontece na forma de regras e não de leitura de comportamento.

- Outras medições, através de técnicas auxiliares como redes neurais, algoritmos genéticos, etc.

Detectores de anomalia geram um alto número de falsos alarmes devido à forma como o comportamento dos objetos monitorados pode variar, entretanto, são os únicos que podem vir a detectar ataques não conhecidos previamente. Além disso, é possível fazer os detectores de anomalia gerarem saída que pode ser aproveitada como entrada para detectores de mau uso.

Vantagens:

- IDSs baseados em detecção de anomalias percebem o comportamento não usual

e portanto têm a habilidade de detectar sintomas de um ataque sem conhecimento específico ou detalhes de seu funcionamento.

- Detectores de anomalias podem produzir informações que levem a definição de assinaturas para detectores de mau uso.

Desvantagens:

- Abordagens de detecção de anomalia normalmente resultam em um alto número de falsos alertas devido as dificuldades intrínsecas à previsão do comportamento dos usuários e processos monitorados.

- Este tipo de IDS requer conjuntos prévios de dados sobre o comportamento do sistema, usuários e rede, nem sempre disponíveis.

## 3.8 Opções de resposta

Depois de obter informações e analisá-las para enquadrar ou não em padrões de ataque, os IDSs geram respostas. Algumas respostas envolvem relatar resultados das buscas por anomalias, outras podem incluir respostas ativas e automatizadas. As respostas mais comuns são:

- respostas ativas; e

- respostas passivas.

### 3.8.1 Respostas ativas

Respostas ativas de IDSs são ações automatizadas tomadas quando certos tipos de intrusão são detectados. Há três categorias de respostas ativas:

- Coleta de informações adicionais. Esta é a resposta mais produtiva no sentido de que serve como evidência do acontecimento e pode levar a investigação a identificação do atacante. Neste caso, coleta-se informações que compõem o ataque, como o endereço IP e porta tcp ou udp de origem do atacante, quais os destinos explorados (se foi um único *host*, a rede toda ou parte dela) e tudo mais que fizer sentido para cada ambiente em específico. Dependendo da gravidade

dos danos, da legislação e da forma como os dados coletados estão dispostos, é possível iniciar processo judicial contra o atacante.

- Modificação do ambiente. Esta categoria representa as iniciativas que podem ser tomadas mediante um ataque visando seu bloqueio em tempo real. IDSs não pode impedir as ações de um usuário, normalmente ele modifica o sistema operacional, seja em suas configurações ou nas funções de ambiente. As ações mais comuns são:
    - Injetar pacotes TCP *reset* nas conexões envolvendo o alarme de ataque;
    - Enviar comandos de reconfiguração de rotas e listas de controle de acesso para o roteador; e
    - Enviar comandos para os *hosts* locais, de forma a desligar serviços dependendo da situação.
- Ações de contra-ataque. Estas ações envolvem medidas desde o rastreamento até a origem (*traceroute*) até varreduras e ataques de negação de serviço contra o atacante. A controvérsia sobre ética e legalidade destas ações é intensa, principalmente porque não há parâmetros claros para definir quais ações são legítimas e quais não são na Internet.

### 3.8.2 Respostas passivas

Respostas passivas estão restritas a coleta de informações para o administrador do sistema, confiando a ele a tomada de ações mediante um ataque. As respostas passivas podem ser:

- Alarmes e notificações: são geradas para informar os ataques detectados. A forma mais comum de alarme é uma mensagem na interface de administração do IDS ou então o envio de mensagem eletrônica para o administrador. O conteúdo dos alertas variam, porém geralmente incluem o IP e porta de origem, IP e porta de destino, data, hora, estado dos cabeçalhos TCP ou UDP e um código ou descrição do motivo pelo qual o alarme foi gerado.

- *Traps* SNMP (*Simple Network Management Protocol*) e plug-ins. É possível configurar alguns IDSs comerciais para gerar alertas em formato para utilização em sistemas de gerência de redes. Isto inclui os *traps* SNMP e mensagens para alertar o administrador do console de gerenciamento central. Este esquema de resposta é útil porque mantém a estrutura de gerência de rede integrada, sem necessitar de um console para cada aplicação.

- Arquivamento. Este foi o primeiro mecanismo de resposta presente nos IDSs, consistindo simplesmente na cópia do conteúdo dos alertas para um repositório de registros, na forma de arquivo-texto ou banco de dados. Sua principal utilidade é servir como recurso para elaboração de relatórios de períodos relativamente grandes, como os totais mensais e anuais, por exemplo.

## 3.9 Implementação de referência

Para aplicação dos conceitos que serão discutidos neste trabalho, é necessário haver uma implementação de IDS que sirva como laboratório para aplicação da proposta. O *software* escolhido para essa finalidade foi o Snort.

Com relação aos conceitos de classificação deste capítulo, tem-se que o Snort é um IDS de execução remota, baseado em rede, trabalha com detecção de mau uso através de sua base de assinaturas, possui gerenciamento centralizado, executa o processamento de eventos em tempo real e oferece variadas possibilidades de opções de resposta, dentre as quais estão a emissão de alertas para o console de gerenciamento (passiva) ou a intervenção do pacote ou segmento através da modificação da configuração de um *firewall* (ativa), por exemplo.

CAMPELLO & WEBER (2001) definem Snort como um Sistema de Detecção de Intrusões que opera através da comparação entre o tráfego de rede e sua base de assinaturas. Escrito em linguagem C e distribuído no modelo de *software* livre é atualmente um dos IDSs mais populares e utilizados. Devido a essas duas naturezas, livre e popular, é também o mais desenvolvido, extensível e que contém o maior número de assinaturas de ataques conhecidos, totalizando 3.015 em novembro de 2004.

Sua estrutura básica é simples, baseada na captura de pacotes de rede através da

biblioteca *libpcap* e em um analisador que trata tanto as informações de cabeçalho quanto a área de dados dos pacotes coletados. Os pacotes que coincidem com alguma das regras da base podem ser descartados, armazenados ou podem gerar um alerta para o administrador dos sistemas. Há ainda a possibilidade de utilizar regras de filtragem durante a coleta de pacotes antes que eles passem pelo analisador através de pré-processadores e processadores de saída, responsáveis respectivamente por analisar os pacotes coletados antes que a base de assinatura seja avaliada e por fazer a formatação dos resultados gerados.

O conjunto de regras do Snort é muito semelhante aos *firewalls*, embora possua diretivas complexas para análise e tratamento dos pacotes coletados. A Figura 4 ilustra uma regra usada para detectar um ataque distribuído de negação de serviço:

```
alert TCP !$H_NET any -> $H_NET 27665
(msg:"DDoS-Trin00"; flags: PA;
content:"betaalmostdone";)
```

**Figura 4. Snort: regra de exemplo.**
**Adaptado de (SNORT, 2004).**

A primeira parte de uma regra define a ação a ser tomada quando um pacote coincidir com ela. As opções são: *log*, para armazenar o pacote; *alert*, para gerar um relatório sobre a ocorrência do referido ataque; *pass*, para ignorar o pacote.

A segunda parte da regra define o padrão a ser procurado. Esse padrão é expresso utilizando-se informações de cabeçalho, como o tipo de protocolo, origem, destino, *flags*, etc, ou conteúdo do pacote, descrito através da utilização de diretivas como "*content*", "*content-list*", "*depth*" e outras.

O Snort está logicamente dividido em múltiplos componentes. Estes componentes trabalham em conjunto para detectar os ataques e gerar saída em um formato útil para interpretação (REHMAN, 2003). São eles:

- Decodificador de pacotes;
- Pré-processadores;
- Motor de detecção;
- Sistema de *logging* e alertas; e

- Módulos de saída.

A Figura 5 mostra a disposição desses componentes

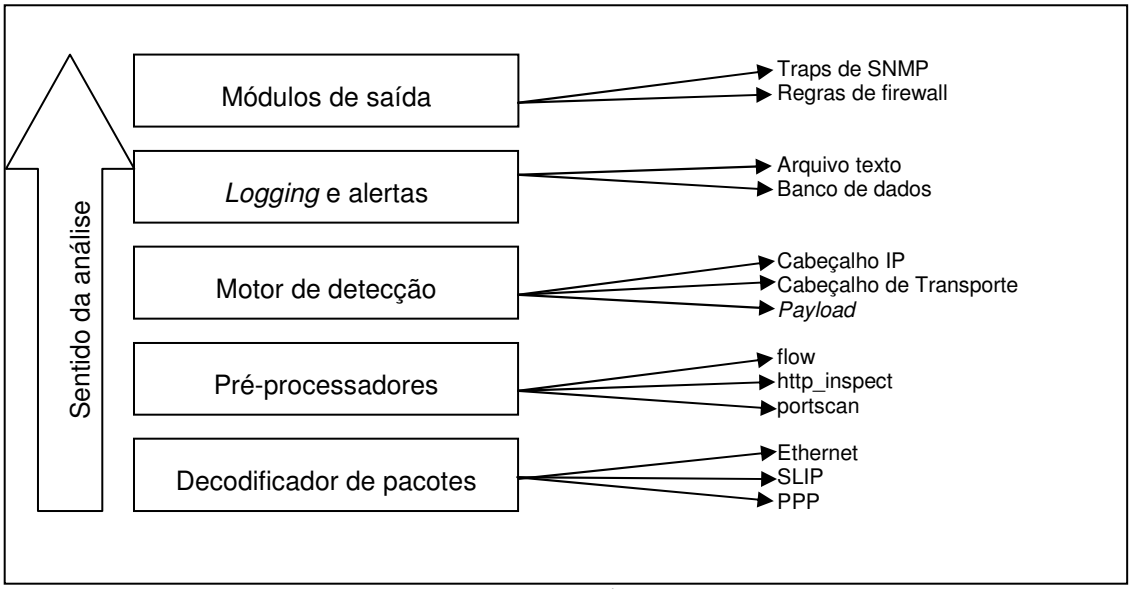



**Figura 5. Componentes lógicos do Snort.**
**Adaptado de (REHMAN, 2003).**

O decodificador de pacotes é o componente responsável por abstrair as informações sobre os diferentes formatos de quadros associados a cada tipo de interface de rede. O Snort entende quadros Ethernet, SLIP, PPP e outros. Os pré-processadores tratam a informação já no nível de pacotes de dados. Entre as atribuições desta camada estão a busca por anomalias nos cabeçalhos dos pacotes, a sua defragmentação (remontagem) e as verificações triviais de sanidade (*sanity check*, em inglês) antes do repasse das informações para comparação com a base de assinaturas. O motor de detecção é a parte mais importante do Snort, pois ele é responsável por detectar os sinais de intrusão dentro dos pacotes. Dado que o método de detecção é através de assinaturas, o *software* carrega, durante sua inicialização, o conjunto de regras e as organiza em estruturas de dados próprias para a finalidade de comparação com o fluxo de dados. O sistema de *logging* e alertas é um componente do Snort que precisa ser configurada pelo usuário para que o *software* de o tratamento desejado quando um ataque ocorrer. Entre as ações comuns estão a emissão de alertas para um arquivo ou banco de dados, a cópia do conteúdo do pacote para análise posterior ou ainda, o descarte do mesmo. Por fim, os módulos de saída são programas anexados que oferecem tratamento adicional para as informações manipuladas pelo Snort. Entre as utilidades deste componente estão a

possibilidade de comunicação com outros computadores e equipamentos de rede para solicitação de bloqueios, a geração de saída no formato XML e o envio de alertas para coletores de auditoria (servidores *syslog*).

# 4 Análise de Causa Raiz

A Análise de Causa Raiz (RCA - *Root Cause Analysis*) é uma metodologia que permite a investigação detalhada e progressiva de incidentes isolados. A RCA envolve o estudo de dois ou mais acontecimentos correlacionados demanda esforços de investigação dos fatos (ADUSKEVICZ et al, 1999). Ela deve identificar como e por que o problema aconteceu, de forma que seja possível impedir sua reincidência (NASA, 2003).

PARADIES & SKOMPSKI (2004) notaram que os desastres em geral, naturais ou resultantes das ações do homem, são sempre precedidos por avisos sob a forma de pequenos incidentes aparentemente isolados. A proposta geral da RCA é analisar e correlacionar tais incidentes, prever desastres e aplicar medidas de correção antes que eles aconteçam.

Define-se Causa Raiz como "um entre múltiplos eventos que levam a uma condição indesejada" (NASA, 2003) Logo, "é o fator causador que, se eliminado, evita o acontecimento do problema" (DEW, 2004). Em síntese, e focando diretamente nos aspectos práticos do que é a Causa Raiz, AMMERMAN (1998) apresenta estas quatro definições simples:

1. Causas Raiz são as causas fundamentais (ou de base): O objetivo do investigador deve ser identificar a causa básica. Quanto mais específica for a definição do investigador a respeito do evento, mais fácil será chegar a conclusão do por que o evento ocorreu, logo será mais fácil deduzir recomendações que impeçam sua recorrência.

2. Causas Raiz são aquelas que podem ser logicamente identificadas: A investigação de ocorrências deve ser benéfica. Não é viável manter pessoas avaliando a causa raiz de um evento por períodos longos ou intermináveis. A RCA precisa estar estruturada de forma que o uso de recursos humanos de investigação seja produtiva e resulte sempre em minimização dos custos através de investimentos certeiros na causa real do evento.

3. Causas Raiz são aquelas que o gerenciamento tem controle: O investigador deve evitar classificações genéricas como "erro de operador", "erro de equipamento"

ou "falha externa". Tais exemplos não são suficientemente claros ou tangíveis para permitir que a gerência tome ações de mudança.

4. Causas Raiz são aquelas em que é possível fornecer recomendações efetivas para que impedir sua recorrência: As recomendações de conserto devem tratar diretamente as causas raiz identificadas durante a investigação. A análise não pode terminar em recomendações vagas, como "melhorar a conscientização das pessoas com relação às políticas e procedimentos da organização", mas sim a planos concretos, menores e mais específicos, de aderência das pessoas a estes procedimentos organizacionais, por exemplo.

A NASA (*National Aeronautics and Space Administration*) enquadra a Análise de Causa Raiz como "um método estruturado de avaliação que identifica a causa raiz para uma situação indesejada" (NASA, 2003), de forma análoga, para DEW (2004) "é o processo de questionamento que provê um método estruturado para habilitar as pessoas a reavaliar as práticas em uma atividade que levam a resultados de baixa qualidade".

PARADIES & SKOMPSKI (2004) citam alguns grandes desastres que levaram a formação do conceito de Análise de Causa Raiz:

- Em 1979, na usina nuclear de Three Mile Island, em Harrisburg, Pennsylvania, nos Estados Unidos, um reator sobreaqueceu e causou a liberação de gases radioativos. Durante o ocorrido, os cientistas se dispersaram para tentar prevenir o alastramento dos gases e a defesa civil da região tentou acalmar o ânimo daquela população. Falhas humanas e erros nos equipamentos causaram o pior acidente nuclear experimentado naquele país.

- Em 1984, a falta de uma metodologia para conter desastres causou vazamento de metil isocianato em uma usina química na cidade de Bhopal, Índia. Como resultado, mais de 5.000 pessoas morreram ou sofreram deficiências irreversíveis em seus corpos.

- Em 1986, o ônibus espacial Challenger explodiu logo após seu lançamento causando a morte dos 7 tripulantes. Ainda hoje se estuda a segurança deste tipo de vôo e faz deste caso um dos maiores impulsionadores para o aperfeiçoamento

da RCA.

- Em 2000, um Concorde explodiu durante a decolagem no aeroporto Charles de Gaulle, em Paris, na França. Após um ano de investigação, encontrou-se que a causa raiz do acidente estava no desgaste de um dos pneus que estourou e causou um incêndio em uma das turbinas.

A metodologia de RCA é um processo de quatro etapas AMMERMAN (1998):

1. Coleta de dados.
2. Mapeamento de fatores de causa.
3. Identificação da causa raiz.
4. Geração e implementação de correções.

## 4.1 Coleta de dados

A aplicação da metodologia de Análise de Causa Raiz inicia-se através da busca e coleta de dados que ajudam na compreensão do evento com o maior número de detalhes possível e o mais próximo da realidade.

Não existe um método formal, entretanto a coleta de dados normalmente acontece através de procedimentos estruturados particulares de cada analista e ambiente. Geralmente estes procedimentos são entrevistas com as pessoas envolvidas, estudo dos documentos de especificação do ambiente e a observação *in loco*.

É nesta etapa que se desprendem mais esforços de tempo e recursos, isso porque sem informações suficientes sobre um evento, não é possível a um analista entendê-lo por completo, logo ela é um dos principais fatores críticos para o sucesso da metodologia RCA.

## 4.2 Mapeamento de fatores de causa

O mapeamento de fatores de causa provê uma estrutura que permite ao investigador organizar e analisar a informação coletada durante a investigação e identificar lacunas e deficiências na compreensão do evento à medida que a coleta de dados ocorre.

Esta fase inicia-se logo que os primeiros dados começam a ser coletados. Nela, gera-se um protótipo de mapa que vai sendo modificado à medida que mais dados relevantes vão sendo coletados e mais conclusões começam a surgir.

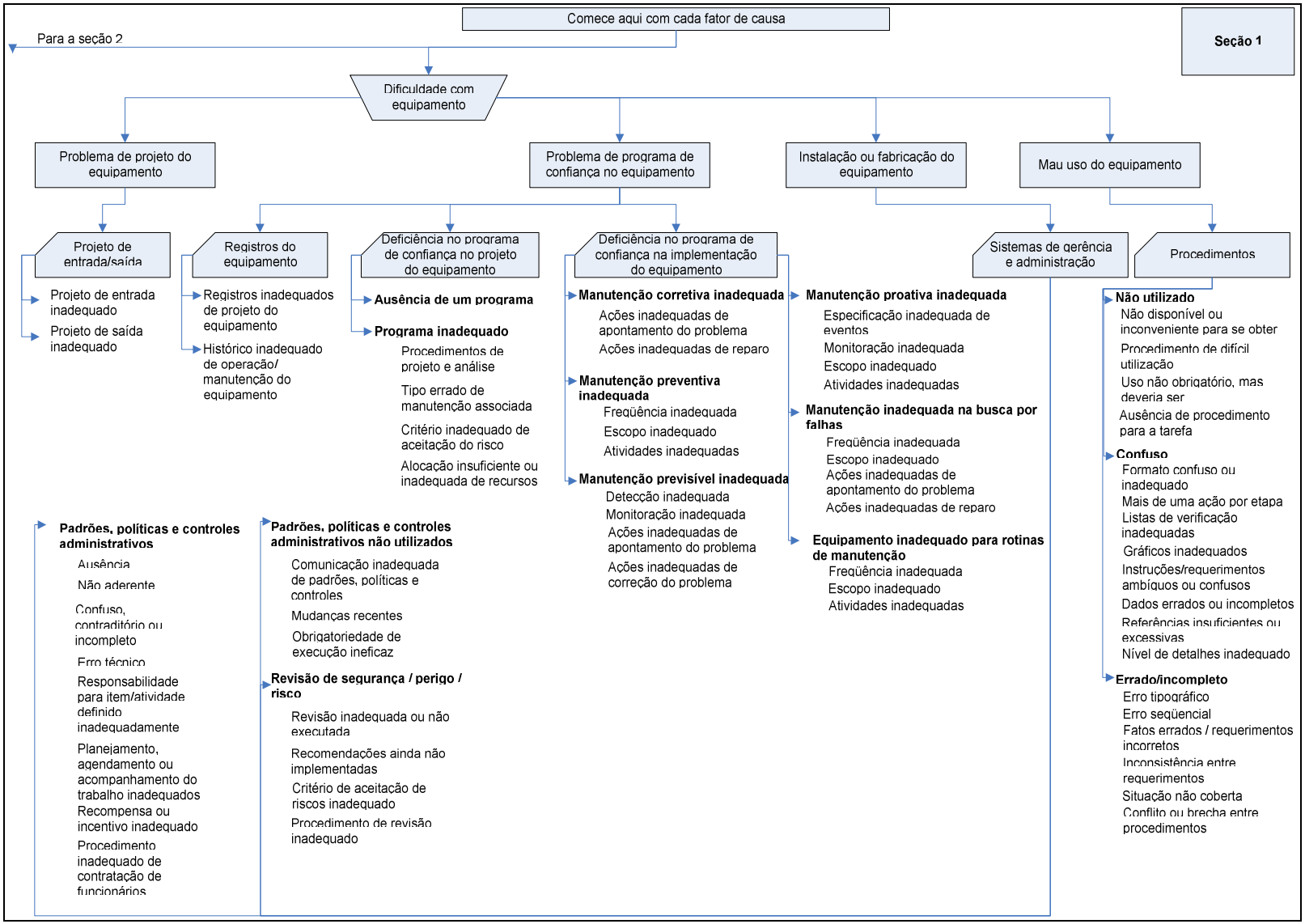
A coleta de dados (fase 1) continua até que o investigador esteja satisfeito com o Mapa de Fatores de Causa (produto da fase 2), e consequentemente, satisfeito com a investigação como um todo.

A partir deste ponto, o investigador está em uma boa situação para identificar os maiores contribuidores para o incidente, chamados de fatores de causa. Fatores de causa são aqueles colaboradores (erros humanos, falha de componentes) que, se eliminados, teriam evitado o acontecimento ou amenizado seu impacto.

Em muitas análises tradicionais, o fator de causa mais visível recebe toda a atenção. Raramente, entretanto, há apenas uma causa; os incidentes normalmente são uma combinação de vários colaboradores. Nestes casos em que se aponta somente um fator de causa, a lista final (fase 4), contendo as recomendações de remediação, não estará completa. Consequentemente, o incidente pode voltar a acontecer e a organização não aprendeu tudo o que podia com aquela experiência.

## 4.3 Identificação da causa raiz

Depois que todos os fatores de causa estiverem identificados, o investigador passa para a fase de identificação da causa raiz. Este passo envolve o uso de um diagrama de decisão chamado Mapa de Causa Raiz (Figura 6) para identificação das razões para cada fator de causa.


Comece aqui com cada fator de causa
Para a seção 2
Seção 1
Dificuldade com equipamento
Problema de projeto do equipamento
Problema de programa de confiança no equipamento
Instalação ou fabricação do equipamento
Mau uso do equipamento
Projeto de entrada/saída
Registros do equipamento
Deficiência no programa de confiança no projeto do equipamento
Deficiência no programa de confiança na implementação do equipamento
Sistemas de gerência e administração
Procedimentos
Projeto de entrada inadequado
Projeto de saída inadequado
Registros inadequados de projeto do equipamento
Histórico inadequado de operação/ manutenção do equipamento
Ausência de um programa
Programa inadequado
Procedimentos de projeto e análise
Tipo errado de manutenção associada
Critério inadequado de aceitação do risco
Alocação insuficiente ou inadequada de recursos
Manutenção corretiva inadequada
Ações inadequadas de apontamento do problema
Ações inadequadas de reparo
Manutenção preventiva inadequada
Freqüência inadequada
Escopo inadequado
Atividades inadequadas
Manutenção previsível inadequada
Detecção inadequada
Monitoração inadequada
Ações inadequadas de apontamento do problema
Ações inadequadas de correção do problema
Manutenção proativa inadequada
Especificação inadequada de eventos
Monitoração inadequada
Escopo inadequado
Atividades inadequadas
Manutenção inadequada na busca por falhas
Freqüência inadequada
Escopo inadequado
Ações inadequadas de apontamento do problema
Ações inadequadas de reparo
Equipamento inadequado para rotinas de manutenção
Freqüência inadequada
Escopo inadequado
Atividades inadequadas
Não utilizado
Não disponível ou inconveniente para se obter
Procedimento de difícil utilização
Uso não obrigatório, mas deveria ser
Ausência de procedimento para a tarefa
Confuso
Formato confuso ou inadequado
Mais de uma ação por etapa
Listas de verificação inadequadas
Gráficos inadequados
Instruções/requerimentos ambíguos ou confusos
Dados errados ou incompletos
Referências insuficientes ou excessivas
Nível de detalhes inadequado
Errado/incompleto
Erro tipográfico
Erro seqüencial
Fatos errados / requerimentos incorretos
Inconsistência entre requerimentos
Situação não coberta
Conflito ou brecha entre procedimentos
Padrões, políticas e controles administrativos
Ausência
Não aderente
Confuso, contraditório ou incompleto
Erro técnico
Responsabilidade para item/atividade definido inadequadamente
Planejamento, agendamento ou acompanhamento do trabalho inadequados
Recompensa ou incentivo inadequado
Procedimento inadequado de contratação de funcionários
Padrões, políticas e controles administrativos não utilizados
Comunicação inadequada de padrões, políticas e controles
Mudanças recentes
Obrigatoriedade de execução ineficaz
Revisão de segurança / perigo / risco
Revisão inadequada ou não executada
Recomendações ainda não implementadas
Critério de aceitação de riscos inadequado
Procedimento de revisão inadequado

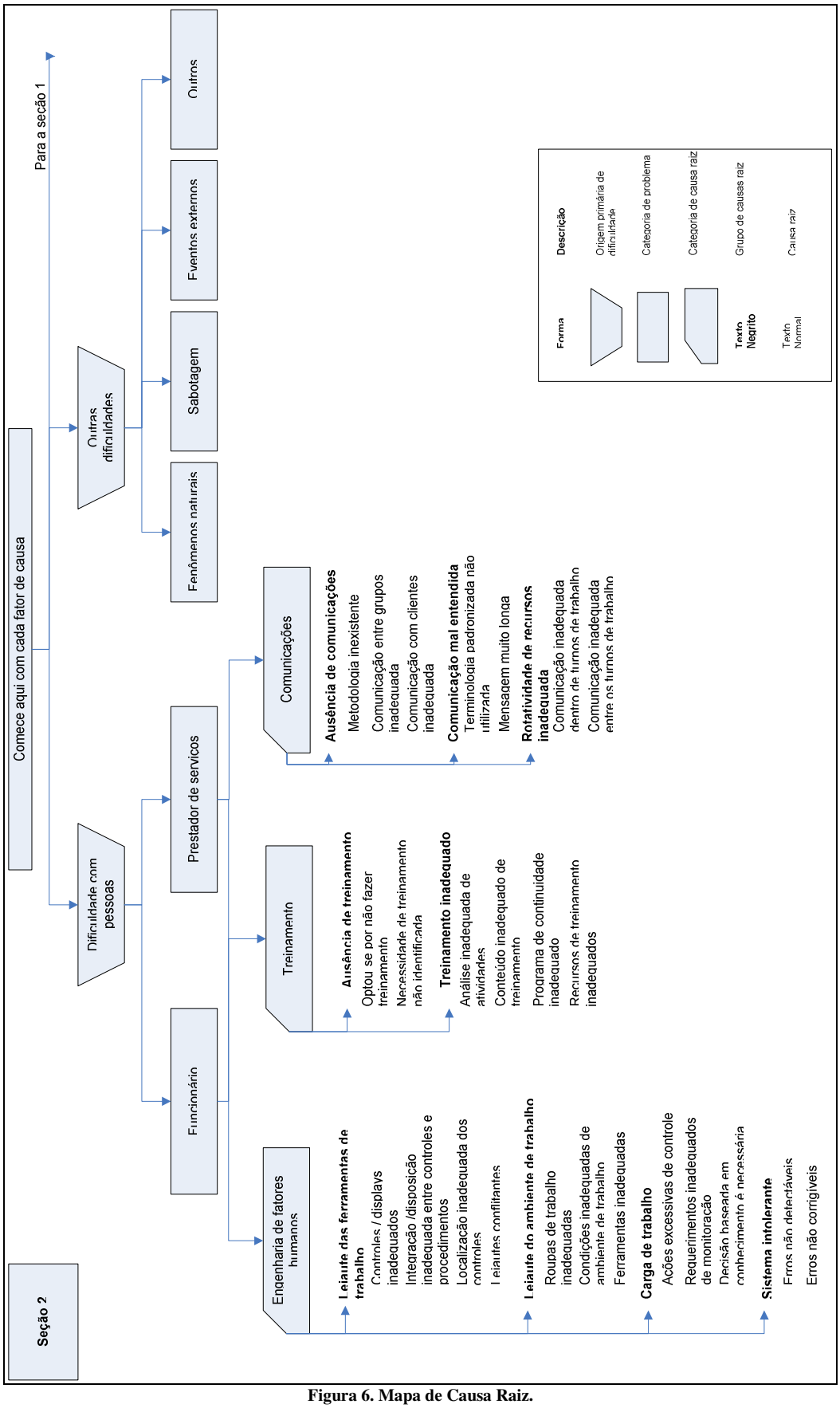



**Figura 6. Mapa de Causa Raiz.**
**Adaptado de AMMERMAN (1998).**

O mapa estrutura o processo de pensamentos do investigador, ajudando-o a responder por que um fator de causa em particular existe. A identificação das causas raiz ajuda o investigador a determinar as razões que justificam a ocorrência do evento e assim ele pode tratar os problemas que as cercam.

## 4.4 Geração e implementação de recomendações

A fase final é a geração de recomendações. Consiste em seguir a identificação das causas raiz para um fator de causa em particular e produzir recomendações tangíveis para prevenir a recorrência deste evento.

O produto gerado nesta fase é representado por uma Tabela Sumária de Causa Raiz. Esta tabela organiza a informação compilada durante as fases anteriores de análise dos dados onde cada coluna representa um dos três aspectos do processo de Análise de Causa Raiz:

- Na primeira coluna, uma descrição geral do fator de causa é apresentada com informações suficientes para o leitor ser capaz de entender o problema e propor soluções.

- A segunda coluna mostra os caminhos através do Mapa de Causa Raiz associados com o fator de causa.

- A terceira coluna apresenta as recomendações que tratam cada uma das causas raiz identificadas.

A utilização deste formato de tabela em três colunas auxilia o investigador no trabalho de garantir que as causas raiz e recomendações serão tratadas para cada fator de causa. A seção a seguir faz uso desta tabela que serve como exemplo.

## 4.5 Exemplo de aplicação da RCA

Para ilustrar a utilização da metodologia de Análise de Causa Raiz, é apresentado um exemplo livre de obstáculos técnicos adaptado do trabalho de AMMERMAN (1998). Uma dona de casa, Maria, decide preparar frangos fritos. Para isso, ela aquece uma frigideira com óleo de cozinha. Assim que inicia a atividade, Jana, sua vizinha, bate à

porta e Maria atende. Ambas começam a conversar e dez minutos se passam sem que Maria volte a dar atenção à sua fritura. Nesse intervalo, o regulador de queima de seu fogão falha e causa uma saída maior de gás pela boca, elevando demasiadamente a quantidade de calor produzida pelo fogo. O alumínio da frigideira começa a derreter até que uma abertura na forma de furo apareça no local de contato com a chama. O óleo vaza e se incendeia. Instantes depois, Maria percebe a fumaça e corre para a cozinha. Vendo o incêndio, Maria imediatamente procura pelo extintor de incêndios e tenta utilizá-lo. Ela remove o pino do extintor, direciona-o contra o fogo, mas o elemento químico não sai do cilindro. Pensando rapidamente em alternativas, Maria faz o que lhe parece mais certo, deixa o extintor de lado e joga água, o que faz com que o fogo se alastre por toda a cozinha, saindo de seu controle. Ela então chama o corpo de bombeiros que conseguem finalmente extinguir o fogo. Como resultado, a cozinha foi destruída.

O analista que estuda um caso como esse e utiliza a metodologia de RCA para apontar as causas, deve coletar o maior número de informações possíveis. Neste cenário isto inclui entrevistas com Maria e Jana, com os bombeiros, e uma visita ao local assim que possível, e em seguida formular, do melhor jeito possível, o Mapa de Fatores de Causa.

O Mapa de Fatores de Causa é apresentado na Figura 7. Ele contempla todos os eventos chave descritos dentro dos desenhos retangulares dispostos horizontalmente, enquanto os eventos de perda ou de conseqüência estão nas colunas acima e abaixo de cada evento chave. As perguntas, no formato de losango, trazem para o mapa os questionamentos do analista.

Apesar da leitura do mapa acontecer da esquerda para a direita, sua construção é feita ao contrário, da direita para a esquerda porque esse é o primeiro fato conhecido. Testes de tempo e de lógica são aplicados para a construção do mapa até o início do primeiro evento e disso resultam os vários questionamentos propostos durante sua construção.

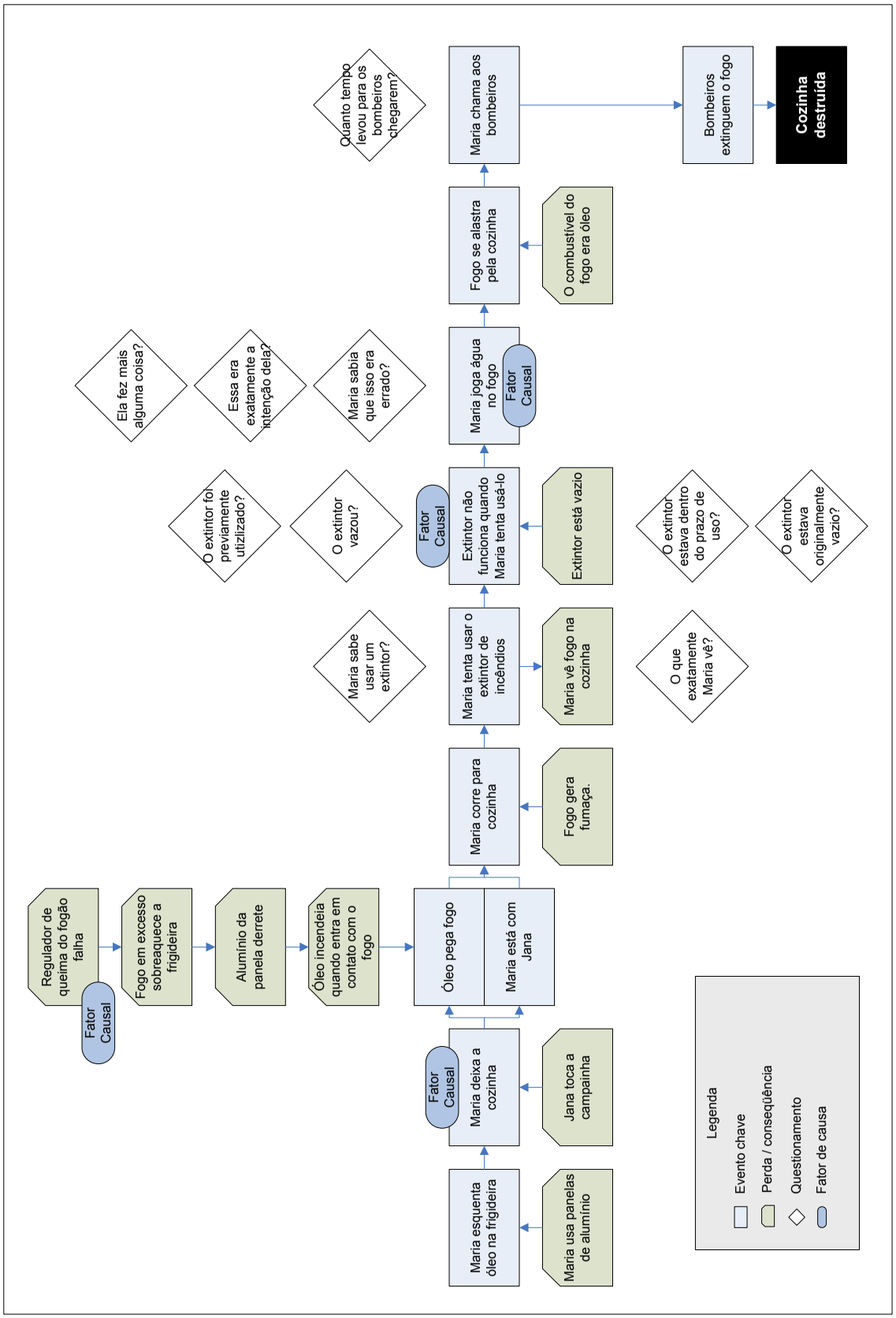



**Figura 7. Mapa de Fatores de Causa.**
**Adaptado de AMMERMAN (1998).**

Uma vez completo, é preciso que o analista identifique os fatores que influenciaram o curso dos eventos. Há quatro fatores de causa neste mapa. A remoção desses fatores ou preveniria o incêndio ou reduziria suas conseqüências.

A Tabela 3 demonstra uma possibilidade de Tabela Sumária de Causa Raiz, produto final da análise. Durante a interpretação do texto desta tabela, é necessário assumir que o ambiente de fritura do alimento seja mais próximo de onde a RCA realmente se aplica. Portanto, as recomendações propostas fazem referência a uma cozinha industrial onde Maria é uma das funcionárias responsáveis por uma das etapas do processo de produção de alimentos de uma fábrica hipotética.

| **Descriçaõ do evento:** Cozinha foi destruída por fogo. | | |
|---|---|---|
| **Fator de causa 1** | **Mapa de Causa Raiz** | **Recomendações** |
| Descrição:<br><br>Maria deixa a cozinha enquanto esquenta óleo. | Dificuldades com colaboradores.<br><br>Padrões, políticas ou controles administrativos insuficientes ou inadequados.<br><br>Ausência de sistemas administrativos e/ou de gerenciamento. | Implementar a política de que o óleo nunca pode ficar sem supervisão.<br><br>Determinar se outras políticas devem ser elaboradas para evitar incidentes parecidos na organização.<br><br>Modificar o procedimento de medição de riscos para considerar a presença de pessoas durante a execução de processos. |
| **Fator de causa 2** | **Mapa de Causa Raiz** | **Recomendações** |
| Descrição:<br><br>Regulador de queima do fogão falha. | Problema de confiabilidade dos componentes dos equipamentos.<br><br>Sem programa de avaliação da confiabilidade dos componentes. | Trocar todos reguladores de queima dos fogões.<br><br>Desenvolver um programa de manutenção preventiva que inclua a troca permanente dos componentes que sofrem desgaste nos fogões.<br><br>Considerar métodos alternativos para preparar frango de forma que a tarefa envolva menos riscos, como por exemplo, adquirir o produto já preparado pelo fabricante. |

**(continua na próxima página))**

| Fator de causa 3 | Mapa de Causa Raiz | Recomendações |
|---|---|---|
| Descrição:<br><br>Extintor não funciona quando Maria tenta usá-lo. | Problema de confiabilidade nos equipamentos de proteção e segurança.<br><br>Programa atual de manutenção dos extintores não atende as expectativas. | Recarregar os extintores de incêndio.<br><br>Inspecionar os outros extintores para certificar-se de que eles estão cheios e dentro do prazo de validade.<br><br>Adicionar os extintores de incêndio na lista de itens de auditoria.<br><br>Enviar cópia dos relatórios descrevendo o uso de equipamentos de proteção para o departamento de manutenção. |
| **Fator de causa 4** | **Mapa de Causa Raiz** | **Recomendações** |
| Descrição:<br><br>Maria jogou água sobre o fogo que tinha óleo como combustível. | Dificuldades com colaboradores.<br><br>Habilidade para executar procedimentos contra acidentes (de trabalho).<br><br>Qualidade do treinamento para emergências abaixo das expectativas. | Prover treinamento prático com extintores de incêndio.<br><br>Revisar outras necessidades de treinamento e habilidade com equipamentos de segurança.<br><br>Revisar o programa de treinamento para determinar o ambiente ideal para as aulas (laboratórios, simuladores, computadores e outros). |

**Tabela 3. Tabela Sumária de Causa Raiz.**
**Adaptado de AMMERMAN (1998).**

Uma vez finalizada a tabela, é necessário executar as recomendações propostas. A qualidade do trabalho de Análise de Causa Raiz está diretamente associada à capacidade do analista em propor recomendações eficazes, que uma vez executadas, evitam a recorrência do incidente.

O próximo capítulo aborda a transformação desta abordagem geral em uma nova abordagem, concebida com exclusividade para os ambientes computacionais e voltada para aplicação dentro dos Sistemas de Detecção de Intrusões.

# 5 Análise de Causa Raiz aplicada aos Sistemas de Detecção de Intrusões

A proposta deste trabalho é aplicar a Análise de Causa Raiz nos Sistemas de Detecção de Intrusões com o objetivo de melhorar a qualidade das informações apresentadas ao administrador do sistema. Este objetivo é alcançado através da aplicação da metodologia nas 10 vulnerabilidades mais críticas para os sistemas UNIX segundo o Instituto SANS e na adição de interpretação de Análise de Causa Raiz para as regras correspondentes presentes no Sistema de Detecção de Intrusões Snort.

## 5.1 Trabalhos correlatos

Trabalhos relacionados à melhoria dos Sistemas de Detecção de Intrusão estão emergindo e abordam várias perspectivas, que podem ser classificadas em três grupos principais:

- Esforços genéricos para melhoria dos IDSs;
- Correlação entre alarmes;
- Mineração de alarmes; e
- Inteligência artificial na detecção de intrusões.

### 5.1.1 Esforços genéricos para melhoria dos IDSs

Um dos componentes dos IDSs que tem recebido mais atenção e, por conseqüência, está se desenvolvendo mais rapidamente é o motor de eventos. Isto se dá principalmente devido a excessiva geração de falsos positivos e a dificuldade de mapeamento entre o alerta e sua causa raiz, que são resultados de múltiplos problemas, mais notadamente:

- da carência de recursos adequados de auditoria (PTACEK & NEWSHAM, 1998);
- dos pesados pré-requisitos para análise em tempo real, que inviabilizam a análise profunda e completa das fontes de coleta de dados (ILUNG, 1993), (PTACEK & NEWSHAM, 1998), como por exemplo, a monitoração de canais de rede mais

rápidos que o processamento disponível para sua análise;

- da problemática envolvida na decisão de se um comportamento é considerado legítimo ou malicioso (BELLOVIN, 1993), (PAXSON, 1999), por exemplo, duas tentativas sucessivas de *login* sem sucesso; e

- das dificuldades inerentes na escrita de assinaturas que identifiquem corretamente os ataques (LEE & STOLFO, 2000), (NING et al., 2001).

Exemplos de IDSs que são menos suscetíveis a falsos positivos são os detectores embarcáveis de ZAMBONI (2001), as ferramentas "leves" especializadas de ALMGREN et al. (2000) e o IDS baseado em rede especializado na identificação de ataques de baixo nível de SEKAR et al. (1999). Todos os três compartilham duas similaridades, primeira: utilizam assinaturas públicas que podem ser afinadas para as particularidades de cada ambiente e segunda: são de propósito específico, voltados a detecções de somente uma classe de ataques. O contraponto é que cada um precisa ser combinado a outros para uma proteção abrangente e completa.

### 5.1.2 Mineração de alarmes

A idéia da utilização de mineração de dados para suporte a investigação de alarmes é apresentada por JULISCH (2003) *apud* MANGANARIS (2000) em um trabalho de aplicação de regras de mineração e associação em conjuntos relativamente grandes de alarmes. Como resultado, os alarmes que se enquadram em um padrão de mineração previamente reconhecido são considerados normais e descartados.

Outros trabalhos que utilizam deste conceito são a mineração de dados incremental para detecção de tráfego de padrões anômalos em redes (BARBARÁ et. al. 2001) e a construção e treino de recursos classificadores de detecção de intrusão (LEE & STOLFO, 2000).

### 5.1.3 Correlação entre alarmes

Sistemas de Correlacionamento de Alarmes (ACSs) (CUPPENS, 2001) são sistemas que agrupam alarmes de forma que eles sejam classificados como pertencentes a um mesmo ataque. Assim, oferecem uma visão mais condensada do evento resultante de

vários alertas. Em particular, o maior objetivo desta técnica é facilitar a distinção entre falsos positivos e os incidentes de segurança reais.

Os ACSs variam conforme 3 critérios (JULISCH, 2003):

- Profundidade de análise: Existem ACSs de processamento *offline*, ou seja, que operam após um período de tempo (ex: processamento semanal) e os de análise em tempo real. O principal benefício do primeiro método é permitir a correlação de eventos com freqüência periódica. Já o benefício do segundo, é permitir a tomada de ações imediatas perante o ataque em andamento.

- Facididade de uso: Alguns ACSs tem dezenas de parâmetros de configuração que necessitam certa experiência do administrador do sistema antes da implantação (DEBAR & WESPI, 2001), outros requerem que o próprio usuário estipule as regras de correlação (CUPPENS & MIÈGE 2002). Para este aprendizado, o usuário executa algumas correlações e o computador as mapeia e as armazena. Infelizmente, a correlação manual é difícil e consome tempo.

- Distorções: Os ACSs são geralmente otimizados para a construção de grupos de alertas que correspondem a ataques. Outros ACSs reavaliam a severidade de cada grupo de alertas e descartam aqueles que são considerados benignos (VALDES & SKINNER 2001).

### 5.1.4 Inteligência artificial na detecção de intrusões

A aplicação de técnicas de inteligência artificial (IA) no subsistema de detecção de intrusões dos IDSs é bastante pesquisada. MUKKAMALA & SUNG (2003) apresentam a aplicação das abordagens mais relevantes, incluindo Redes Neurais Artificiais (*Artificial Neural Networks* - ANNs), Algoritmos Genéticos Lineares (*Linear Genetic Programs* - LGPs), Ranhuras de Regressão Adaptativas Multivariadas (*Multivariate Adaptive Regression Splines* - MARS) e Máquinas de Vetores de Suporte (*Support Vector Machines* - SVMs) segundo os critérios de precisão e desempenho.

BERNARDES & MOREIRA (2000) apresentam uma avaliação do uso de agentes móveis (*mobile agents*) para adição de recursos de mobilidade no processo de detecção

de intrusões. A proposta aborda um sistema modular, formada por vários pequenos agentes que monitoram os segmentos de comunicação, oferecendo os benefícios de diminuição de *overhead*, aumento de escalabilidade e flexibilidade além de tolerância à falhas.

BOTHA *et. al.* (2002) propõe a técnica de lógica difusa (*fuzzy logic*), através de um modelo dinâmico baseado em análise de tendência (*trend analysis*) para simplificar o controle do detector de intrusões dentro de um cenário organizacional.

## 5.2 Motivações para uma nova abordagem

Tomando os trabalhos correlacionados como base de comparação, pode-se dizer que este trabalho é um esforço genérico para melhoramento dos mecanismos de detecção de intrusões, especializado, entretanto, na aplicação da metodologia de análise de causa raiz. Ao contrário das técnicas de mineração e correlação, e das abordagens de IA, este trabalho não visa automatizar todo o processo de análise e detecção, mas sim tratar de forma humanamente eficaz, a parte deste processo que (ainda) não pode ser automatizada.

A fundamentação para tal objetivo apóia-se em três afirmações importantes a respeito da Análise de Causa Raiz aplicada a Sistemas de Detecção de Intrusões elaboradas por JULISCH (2003):

Primeira: "Geralmente, um conjunto pequeno de causas raiz é responsável pela vasta maioria dos alarmes gerados pelos IDSs". Os IDSs utilizam técnicas que tomam como base apenas o estado atual do *host*/rede para decidir sobre emitir ou não um alerta. Idealmente, deseja-se que a análise sobre esta decisão possa considerar outros fatores também relevantes, como por exemplo, se este mesmo alarme já foi emitido anteriormente nas mesmas circunstâncias ou então, se ele tem similaridade de características com outros alarmes recém emitidos.

Segunda: "O emprego de alarmes generalizados e agrupados simplifica a identificação de causas raiz". Agrupar alertas é necessário devido à quantidade excessiva de registros com os quais o administrador do sistema precisa trabalhar. Assim, entre as abordagens existentes para que o trabalho de agrupamento não tenha que ser feito manualmente

pelo administrador do sistema, existe Análise de Causa Raiz.

Terceira: "A remoção da causa raiz reduz significativamente a geração de futuros alarmes". Voltando aos conceitos do capítulo anterior, a Análise de Causa Raiz é eficiente quando, uma vez removida a causa, os alarmes não tornam a aparecer novamente.

Tão importante quanto às afirmações já conhecidas e relatadas por Julisch a respeito da RCA, está o fato de que a aplicação desta metodologia nos Sistemas de Detecção de Intrusões favorece o defensor de redes da forma como pode ser visto na Figura 8.

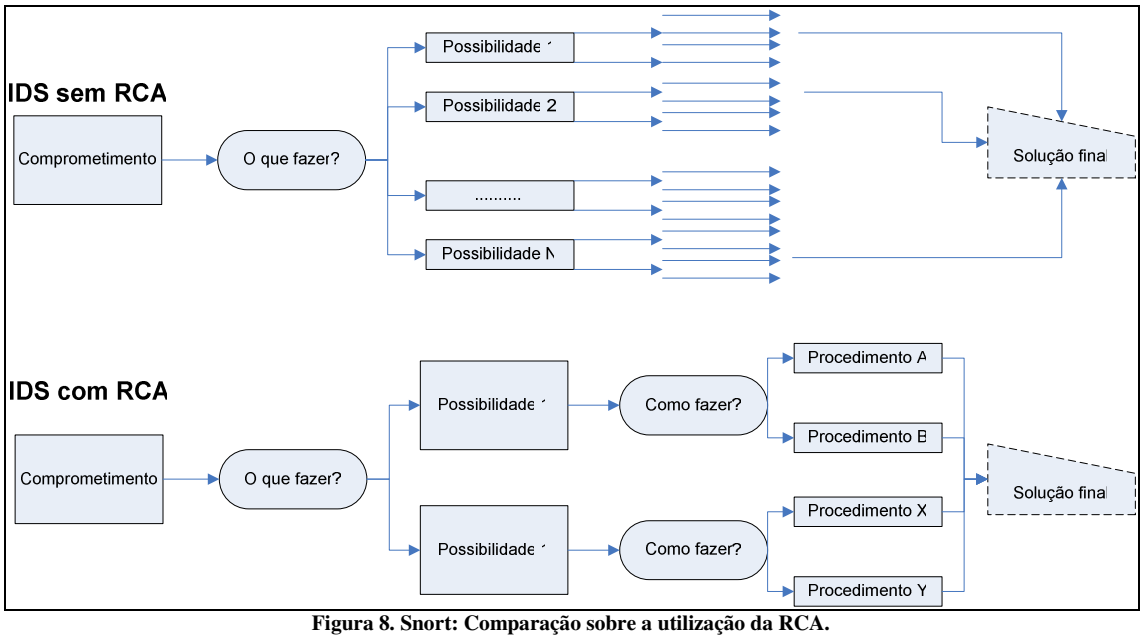


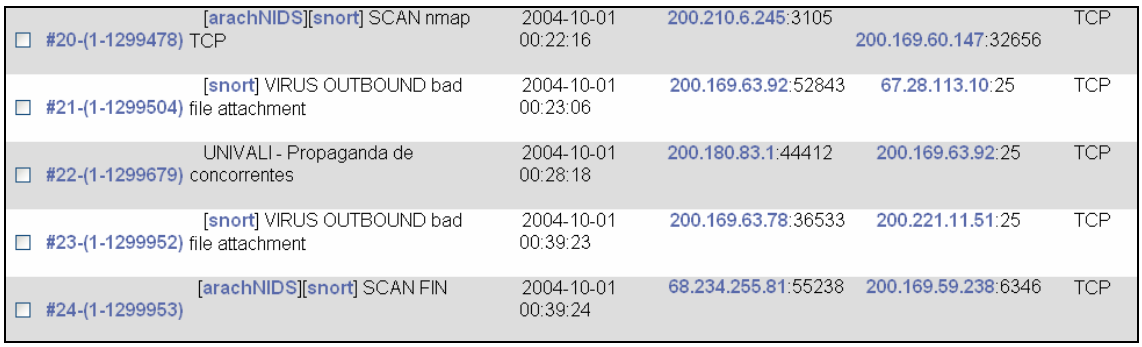

**Figura 8. Snort: Comparação sobre a utilização da RCA.**

Sem a utilização da metodologia abordada neste trabalho, o administrador do sistema, assim que é notificado do comprometimento de sua rede, parte em busca do que deve ser feito para a devida correção do problema de segurança. Geralmente há várias possibilidades de ações para cada tipo de incidente. Cabe ao administrador, com base em sua experiência, testar e descobrir quais medidas servem para que se alcance uma solução adequada.

Através do uso da Análise de Causa Raiz, o defensor da rede pode obter do próprio IDS as recomendações sobre o que deve ser feito. Basta associar cada recomendação a procedimentos documentados para que se obtenham soluções perfeitamente eficazes e, consequentemente, computadores, rede e informações sob um nível maior de proteção.

Os IDSs disponíveis para uso em ambientes de produção como o Snort (SNORT, 2004), SourceFire (SOURCEFIRE, 2004) e Dragon (ENTERASYS, 2004) estão limitados a produzir naturalmente um alerta para cada evento anômalo detectado no *host*/rede auditado. À medida que o tráfego de *host*/rede aumenta, menos viável fica a interpretação dos alertas nesta proporção de 1 alerta para 1 evento até o ponto em que os alertas começam a aparecer em um intervalo menor que o necessário para que o administrador do sistema os classifique e aplique as contramedidas adequadas.

A Figura 9 mostra um exemplo em que o IDS Snort emitiu cinco alertas para cinco atividades maliciosas distintas:



| | | | | | |
|---|---|---|---|---|---|
| ☐ #20-(1-1299478) | [arachNIDS][snort] SCAN nmap TCP | 2004-10-01 00:22:16 | 200.210.6.245:3105 | 200.169.60.147:32656 | TCP |
| ☐ #21-(1-1299504) | [snort] VIRUS OUTBOUND bad file attachment | 2004-10-01 00:23:06 | 200.169.63.92:52843 | 67.28.113.10:25 | TCP |
| ☐ #22-(1-1299679) | UNIVALI - Propaganda de concorrentes | 2004-10-01 00:28:18 | 200.180.83.1:44412 | 200.169.63.92:25 | TCP |
| ☐ #23-(1-1299952) | [snort] VIRUS OUTBOUND bad file attachment | 2004-10-01 00:39:23 | 200.169.63.78:36533 | 200.221.11.51:25 | TCP |
| ☐ #24-(1-1299953) | [arachNIDS][snort] SCAN FIN | 2004-10-01 00:39:24 | 68.234.255.81:55238 | 200.169.59.238:6346 | TCP |

**Figura 9. Snort: um alerta para cada evento.**
**Adaptado de SNORT (2004).**

Na figura é possível entender a utilidade do Snort. Em um intervalo menor do que 20 minutos, o *software* informou o acontecimento de uma varredura de portas TCP através da utilização do programa nmap (#20), a presença de vírus na rede local tentando contaminar *hosts* da Internet (#21 e #23), a entrada de mensagem de correio eletrônico não solicitada (spam) contendo propaganda de concorrentes (#23) e o início de uma varredura de *hosts* da rede local através de alguma ferramenta que implementa o método FIN Scan.

O administrador do sistema, ao ler este relatório, pode montar um diagnóstico das ameaças que cercam a sua rede. As contramedidas vão depender da habilidade de cada profissional, da disponibilidade de recursos técnicos e da política de segurança de cada organização. Entretanto, o ponto central é que as redes conectadas a Internet encontram-se tão expostas às ferramentas automatizadas de ataque que o trabalho de coleta de dados (implementada aqui pelo Snort) e a análise (dependente das qualidades do administrador) podem não ser mecanismos suficientes manter as informações sensíveis

da organização sob segurança.

A Figura 10 mostra um exemplo em que o IDS Snort gerou quatro alertas para quatro pacotes suspeitos detectados no segmento de conexão da Internet com uma rede local:

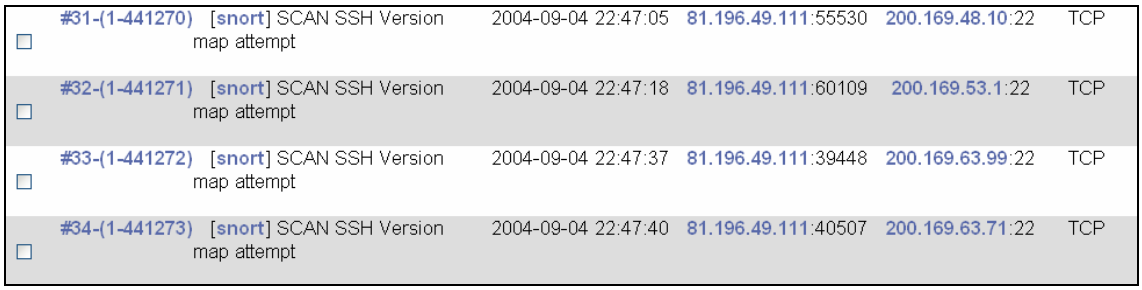



**Figura 10. Snort: muitos alertas para um mesmo evento.**
**Adaptado de SNORT (2004).**

No exemplo, o administrador do sistema recebeu em seu console de trabalho quatro alertas. Todas relatam o mesmo incidente: "SCAN SSH Version map attempt" (tentativa de mapear a versão do *software* de console seguro). Analisando os detalhes, é possível perceber que o computador de origem é um só e o que varia é o endereço IP de destino. Verifica-se também que o intervalo de tempo entre o primeiro e o último alerta é de apenas 35 segundos. Para um período amostral de 24 horas é comum que a quantidade de alertas gerados passe dos 40 mil em redes de relativo grande porte, com 2 mil *hosts* ou mais.

Neste cenário, algumas observações emergem:

- não existe correlação na exibição de um alerta recém emitido e seus predecessores.

- não existem procedimentos sobre o que pode ser feito para remediação do problema. Cabe ao administrador do sistema possuir conhecimento para lidar com cada uma das situações que lhe são apresentadas.

- devido a quantidade de alertas com os quais o administrador precisa interagir em sua tarefa de análise, ele consegue tratar apenas uma fração dos alertas emitidos em um ambiente de trabalho regular.

- os alertas que relatam incidentes mais sutis, como a ação dedicada e mal-intencionada de um hacker executando procedimentos manuais (ao invés das

varreduras e procedimentos automatizados de invasão), podem ficar perdidos em meio a excessiva quantidade de alertas apresentada ao administrador do sistema.

- metodologias que venham a contribuir na filtragem qualitativa e quantitativa de informações que chegam ao console podem ajudar o administrador na tarefa de defesa da rede e das informações.

Dentro desta proposta de trabalho, e no contexto dos Sistemas de Detecção de Intrusões, a causa raiz de um alarme será **o motivo pelo qual ele foi gerado**. Tomando como exemplo os alertas da Figura 9, têm-se os de número #20 e #24 que relatam a ocorrência de varreduras. Basicamente, a aplicação da RCA nestes alertas resultará na definição de que, para se evitar futuras tentativas de reconhecimento dos *hosts*, é necessário tomar alguma ação de bloqueio nos objetos de alvo (conexões TCP com diferentes estados, permissão de passagem de ICMP, tempo de intervalo entre solicitações de sincronia, etc...) utilizados por ferramentas como o nmap para obtenção das informações maliciosas. Então, dentre as ações cabíveis, estaria a de se restringir por exemplo, no filtro de pacotes, a possibilidade de computadores na Internet conseguirem estabelecer comunicação com computadores da rede local em portas altas, visto que a porta TCP de destino é 32656, não-registrada e sem serviço associado.

De acordo com as classificações da RCA, é possível concluir que a solução para este conjunto de alertas é idêntica para cada ocorrência, logo a exibição destas informações poderia estar condensada em um só registro ou agrupada segundo as soluções de causa raiz, aumentando a produtividade na análise executada pelo administrador do sistema.

Para os demais alertas, números #21 a #23, a porta de destino das comunicações é a de correio eletrônico (25/tcp). Aqui, a causa imediata (não raiz) está na existência de conteúdo impróprio no campo de dados payload dos pacotes IP detectados através de palavras-chave como ".exe" e ".pif", que disparam o alerta. Logo, para se tratar estes casos, cabem as seguintes causas raiz:

- Presença de vírus na rede local, consequentemente ausência ou não-cumprimento da política de utilização de antivírus no ambiente organizacional.

- Ausência de controles para auditoria das mensagens de correio eletrônico, como

ferramentas anti-spam ou anti-worms.

A Análise de Causa Raiz, enquanto método eficaz para resolução de problemas em geral, será utilizada para a construção de uma nova abordagem para mudar a forma como o administrador interage com os Sistemas de Detecção de Intrusões. Esta abordagem será mais útil para dois propósitos: melhorar a interação entre o administrador e os alertas através da sintetização de informações e apresentando soluções em maior profundidade, através da explicação do **por quê** o incidente aconteceu e não somente os atuais **o quê** e **como**.

## 5.3 Construção da abordagem

A construção do método de aplicação da Análise de Causa Raiz para Sistemas de Detecção de Intrusões consiste na transferência dos quatro passos de resolução de problemas através da RCA, apresentados no Capítulo 4, para cenários onde o assunto da investigação sejam as conseqüências de exploração de vulnerabilidades presentes em computadores e redes computacionais.

A realização desta abordagem é concebida através da obtenção de dois produtos, resultantes de cada uma destas duas etapas:

- Etapa 1: Elaborar um Mapa de Causas Raiz acompanhado de uma Tabela de Fatores de Causa para cada uma das 10 vulnerabilidades mais exploradas nos ambientes UNIX de acordo com o *ranking* TOP 20 do Instituto SANS (SANS, 2004);

- Etapa 2: Empregar a nova Metodologia de Análise de Causa Raiz, através dos Mapas de Causas Raiz e das Tabelas de Fatores de Causa gerados na etapa anterior, em um subconjunto de alertas do Snort, apresentando de que forma sua utilização ajuda o administrador do sistema a tratar os incidentes de segurança na rede de trabalho.

A Etapa 1 é o assunto do Capítulo 6 e a Etapa 2 é o assunto do Capítulo 7.

# 6 Análise de Causa Raiz das 10 vulnerabilidades mais críticas dos ambientes UNIX

A grande maioria dos *worms* e outros ataques bem sucedidos são possíveis graças a vulnerabilidades em um pequeno número de serviços mais comuns nos sistemas operacionais. Os *hackers* são oportunistas. Eles utilizam os caminhos mais fáceis e convenientes e exploram as falhas mais conhecidas com ferramentas pré-fabricadas. Eles contam com que as organizações não corrijam os problemas e geralmente varrem indiscriminadamente a Internet, procurando por sistemas vulneráveis.

A disseminação fácil e destrutiva dos *worms*, como o Blaster, Slammer e Code Red, bem como a maioria das outras formas de ataque, podem ser ligadas diretamente à exploração de vulnerabilidades não corrigidas. No ano 2000, o Instituto SANS, em conjunto com o *National Infrastructure Protection Center* (NIPC) e o FBI (*Federal Bureau of Investigation*), lançou um documento descrevendo as 20 vulnerabilidades mais críticas da Internet SANS (2004).

Esta lista SANS Top 20 é na verdade duas listas Top 10: os 10 serviços vulneráveis mais críticos no Windows e os 10 serviços vulneráveis mais críticos nos sistemas operacionais derivados do UNIX. Embora existam milhares de incidentes de segurança que afetam esses sistemas operacionais, a grande maioria dos ataques visa um ou mais destes vinte serviços vulneráveis.

Devido às qualidades mencionadas, a lista SANS Top 20 serviu como base para elaboração de 10 Análises de Causa Raiz das vulnerabilidades associadas aos sistemas UNIX, com o objetivo de validar a utilidade da metodologia de Análise de Causa Raiz. Este capítulo apresenta os resultados desta análise.

## 6.1 Composição da análise

A aplicação das Análises de Causa Raiz nas vulnerabilidades associadas aos sistemas UNIX segue os mesmos passos da sugestão de AMMERMAN (1998) para a investigação de eventos nos ambientes clássicos de aplicação da RCA. Os produtos gerados para cada análise são sempre um Mapa de Causas Raiz e uma Tabela de Fatores de causa.

### 6.1.1 Mapa de Causa Raiz

Um Mapa de Causa Raiz contêm três objetos: Fatores Causa, Causa Aparente/Imediata e Estado Final (Figura 11). A construção do mapa inicia-se a partir da observação de qual é o Estado Final do incidente. Enquanto a avaliação de um incidente tradicional costuma resultar em um Estado Final como "cozinha destruída por incêndio" ou "aeronave destruída", nos ambientes computacionais levantou-se possibilidades como "serviço indisponível" ou "computador invadido". Logo, o objeto Estado Final é sempre o desastre, ou perda provocada pela ação da exploração de alguma vulnerabilidade.

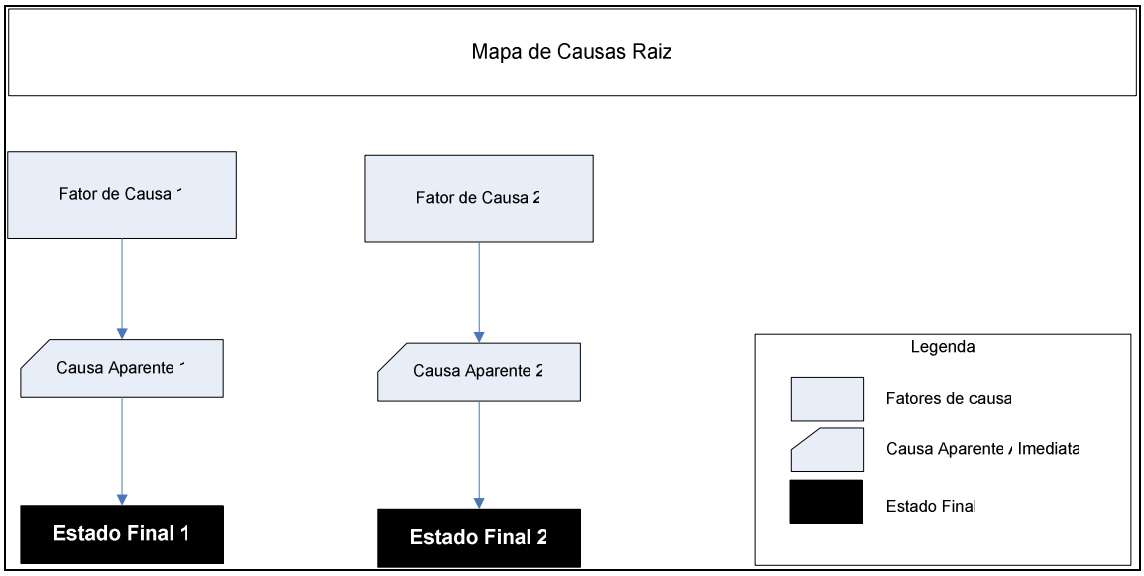



**Figura 11. Componentes de um Mapa de Causas Raiz.**

O segundo objeto, Causa Aparente, é a causa imediata que provocou o evento. Esta é a causa que quando tratada sem maior análise, não remedia a situação de forma definitiva e dá margem à indesejada recorrência do problema. Foi detectada uma relação direta entre o objeto Causa Aparente e as vulnerabilidades associadas a cada um dos 10 integrantes do *ranking* do SANS Institute. Logo, cada vulnerabilidade abordada naquele documento será um objeto de Causa Aparente neste estudo.

O terceiro objeto, Fator de Causa, é o item levantado pelo especialista da área de estudo, neste caso, segurança nos ambientes computacionais, como a verdadeira causa que leva a ocorrência do evento. Neste objeto reside, portanto, a principal contribuição de pesquisa deste trabalho, visto que ao apontar um Fator de Causa e propor soluções (presentes nas recomendações da Tabela de Fatores de Causa, que será apresentada em seguida), esteja-se tomando ações que impeçam a sua recorrência.

### 6.1.2 Tabela de Fatores de Causa

A Tabela de Fatores de Causa é o complemento do Mapa de Causa Raiz. Nela estão descritas todas as ações humanas que devem ser tomadas, sob forma de recomendações, para que seja possível evitar a recorrência do evento.

Sua estrutura é formada por três colunas. A primeira contém o Fator de Causa em questão. A segunda cita item a item o caminho percorrido desde o Fator de Causa até o Estado Final e a terceira, as recomendações que devem ser executadas (Tabela 4).

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br><br>Fator de causa 1. | Fator de causa 1<br>Causa aparente 1<br>Estado final 1I | Recomendação 1<br>Recomendação 2<br>Recomendação 3. |
| **Fator de causa 2** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Fator de causa 1. | Fator de causa 1<br>Causa aparente 1<br>Estado final 1I | Recomendação 1<br>Recomendação 2<br>Recomendação 3. |

**Tabela 4. Componentes de uma Tabela de Fatores de Causa**

De igual importância aos Fatores de Causa do Mapa de Causas Raiz, a coluna de recomendações da Tabela de Fatores de Causa contém as contribuições estudadas neste trabalho.

## 6.2 Servidor de DNS Bind

Bind (*Berkeley Internet Name Domain*) é a implementação mais utilizada na Internet para o serviço de DNS (*Domain Name System*). DNS é um serviço crítico que permite a conversão de nomes de *hosts* e domínios, como por exemplo, example.com ou www.example.com nos endereços IPs corresponentes. Devido a natureza crítica de operação deste serviço, os ataques contra o Bind normalmente levam a inacessibilidade geral da rede por ele servida.

Para o Mapa de Causas Raiz (Figura 12), foram encontrados os seguintes fatores de causa:

1. Execução desnecessária do serviço;

2. Desatualização do *software*;

3. Exibição da *string* contendo a versão do *software*;

4. Serviço executado com privilégio administrativo e/ou sem restrições de ambiente;

5. Falhas na configuração de transferência de arquivos de zonas; e

6. Serviço de consulta recursiva habilitado desnecessariamente.

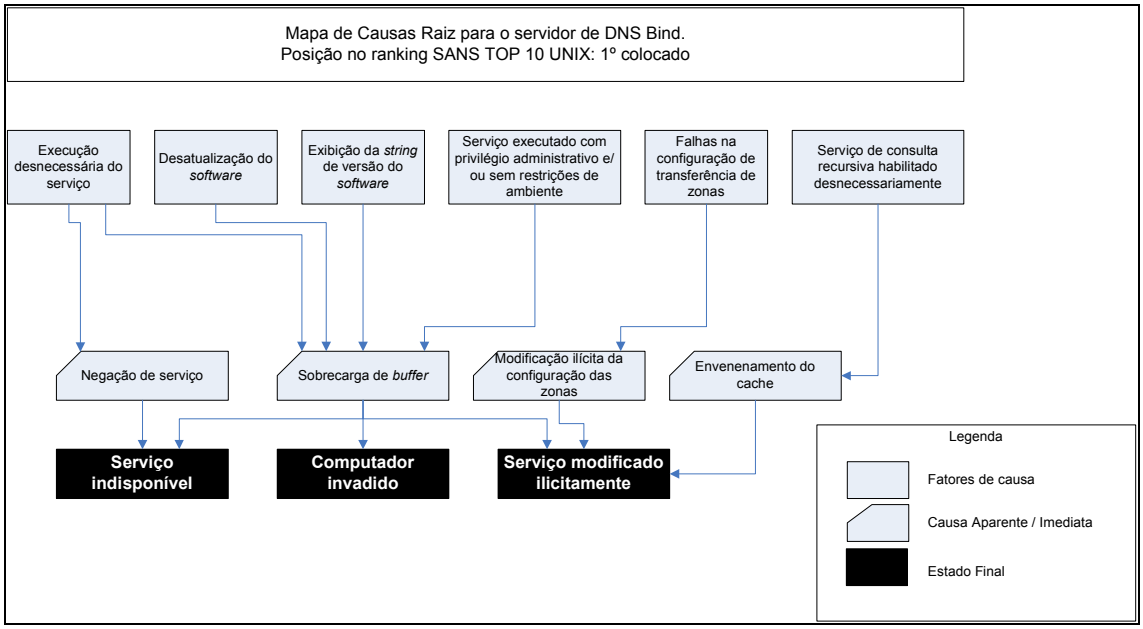



**Figura 12. Mapa de Causas Raiz para o servidor de DNS Bind.**

O Fator de Causa 1 "Execução desnecessária do serviço" ocorre quando o *software* Bind está ativado no computador sem necessidade. Este Fator de Causa é relevante porque várias distribuições do Linux, bem como muitas variantes do UNIX trazem o Bind embutido e habilitado por padrão (HARARI, E., 2000)

O Fator de Causa 2 "Desatualização do *software*" acontece quando o *software*, devidamente configurado e estabelecido, não recebe as correções de segurança à medida que as vulnerabilidades e os *upgrades* são publicados pelo fabricante.

O Fator de Causa 3 "Exibição da *string* contendo a versão do *software*" aponta que os servidores de DNS, assim como os demais, costumam fornecer a versão do *software* quando solicitado pelo cliente remoto. Esta informação é desnecessária e geralmente

utilizada por ferramentas de varredura para determinar se vale o esforço de proceder com tentativas de exploração de falhas.

O Fator de Causa 4 "Serviço executado com privilégio administrativo e/ou sem restrições de ambiente" mostra que as conseqüências resultantes da exploração de falhas de programação do Bind podem variar segundo a forma como o sistema operacional prepara o ambiente de execução deste serviço.

O Fator de Causa 5 "Falhas na configuração de transferência de arquivos de zonas" lida com os problemas associados à falta de atenção com segurança quando se define os computadores que podem estabelecer comunicação de transferência dos arquivos de zona com o servidor primário de DNS. Por padrão o Bind permite que qualquer computador solicite transferências de leitura e que nenhum está autorizado a lhe enviar pedidos de alteração ou escrita de novos dados.

O Fator de Causa 6 "Serviço de consulta recursiva habilitado desnecessariamente" relata que o serviço de DNS está dividido em duas atividades distintas. Uma é a de servir nomes de domínios e as informações de *hosts* associadas a eles e a outra é a de atuar como resolvedor de nomes para a rede local. O conhecimento desta informação pode ser importante para evitar que a exploração de falha no sistema de *caching* não venha a servidor como porta de entrada para uma ação mais grave, como a alteração das informações de zonas dos nomes de domínio servidos.

A Tabela de Fatores de Causa contendo as recomendações para remediação das falhas no servidor de DNS Bind é apresentado na Tabela 5.

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br><br>Execução desnecessária do serviço. | Negação de serviço<br>Sobrecarga de *buffer*<br>Serviço indisponível | Desabilitar a execução do *software* de DNS nos computadores que não estiverem especificamente designados para atuar como servidores.<br>Configurar o *firewall* para não permitir conexões externas a servidores de DNS que não oficiais. |
| **Fator de causa 2** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Desatualização do *software*. | Sobrecarga de *buffer*<br>Serviço indisponível<br>Computador invadido<br>Serviço modificado ilicitamente | Aplicar regularmente todos os *patches* de segurança lançados.<br>Atualizar o *software* para a última versão sempre que ocorrer publicação de falhas de segurança.<br>Documentar procedimentos de atualização do *software*, anexando-o a Política de Segurança de informações. |
| **Fator de causa 3** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Exibição da *string* contendo a versão do *software* | Sobrecarga de *buffer*<br>Computador invadido<br>Serviço indisponível<br>Serviço modificado ilicitamente | Desabilitar a exibição de informações desnecessárias ao funcionamento do serviço de DNS, como a *string* de versão do *software* e o campo HINFO (campo de descrição do ambiente - opcional ) dos arquivos de configuração de zonas. |
| **Fator de causa 4** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Serviço executado com privilégio administrativo e/ou sem restrições de ambiente | Sobrecarga de *buffer*<br>Computador invadido<br>Serviço indisponível<br>Serviço modificado ilicitamente | Isolar o serviço de DNS em um computador específico para esta finalidade, de forma que uma possível invasão através da exploração de vulnerabilidades neste serviço não venha a comprometer outros serviços compartilhados, como por exemplo um *webserver* ou banco de dados.<br>Habilitar a execução do Bind em ambiente virtual, através da utilização do recurso *jail* (comando de virtualização de ambientes) |
| **Fator de causa 5** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Falhas na configuração de transferência de arquivos de zonas | Modificação ilícita da configuração dos arquivos das zonas<br>Serviço modificado ilicitamente | Habilitar restrições nos computadores que podem solicitar transferência dos arquivos de zona. Normalmente este recurso deve ficar disponível apenas aos servidores secundários. |

| Fator de causa 6 | Mapa de Causas Raiz | Recomendações |
|---|---|---|
| Descrição:<br><br>Serviço de consulta recursiva habilitado desnecessariamente | Envenenamento do *cache*<br><br>Serviço modificado ilicitamente | Desabilitar o sistema de consultas recursivas ou dividir o as tarefas de consultas recursivas e consultas autoritárias em computadores diferentes, para evitar que a falha em um serviço não altere o funcionamento do outro. |

**Tabela 5. Tabela de Fatores de Causa para o servidor de DNS Bind.**

## 6.3 Servidores HTTP

O tráfego HTTP *Hyper Text Transfer Protocol* (Protocolo de transferência de hipertexto) é o mais intenso na Internet. Os servidores deste protocolo como o Apache e o Sun Java System Web Server (antigo iPlanet) possuem a maior fatia de usuários e, como tal, são muito visados por *hackers* para proceder com tentativas de invasão que vão desde a negação de serviços até a obtenção de privilégios de administrador do computador. Entretanto, os esforços de ataque contra este protocolo têm sua motivação justificada em muitos dos casos pela vontade do *hacker* em querer modificar as informações que são mostradas aos visitantes do *website*, ataque este conhecido como "desfiguração", ou *defacement*, em idioma inglês.

A aplicação da Análise de Causa Raiz resultou no Mapa de Causas Raiz exibido na Figura 13.

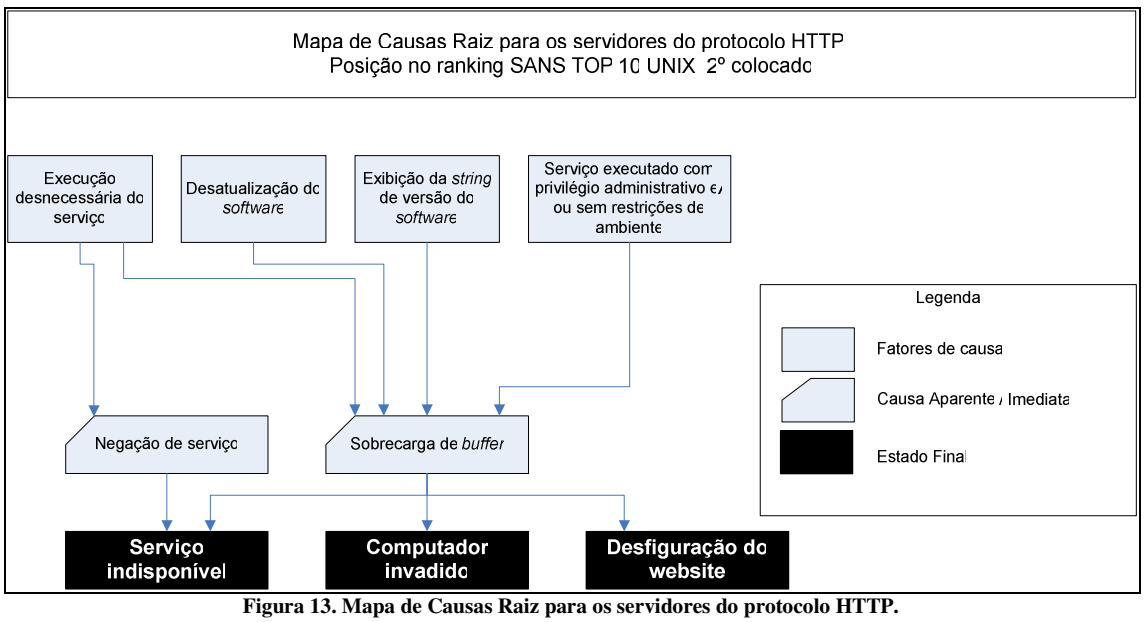



**Figura 13. Mapa de Causas Raiz para os servidores do protocolo HTTP.**

O Fator de Causa 1 "Execução desnecessária do serviço" ocorre quando o *software* (Apache, Sun Java System Web Server ou qualquer outro) está ativado no computador sem necessidade. Este Fator de Causa é relevante porque várias distribuições do Linux, bem como muitas variantes do UNIX trazem o Apache embutido e habilitado por padrão (HARARI, E., 1999).

O Fator de Causa 2 "Desatualização do *software*" acontece quando o *software*, devidamente configurado e estabelecido, não recebe as correções de segurança à medida que as vulnerabilidades e os *upgrades* são publicados pelo fabricante.

O Fator de Causa 3 "Exibição da *string* contendo a versão do *software*" define que os servidores de HTTP, assim como os demais, costumam fornecer a versão do *software* quando solicitado pelo cliente remoto. Esta informação é desnecessária e geralmente utilizada por ferramentas de varredura para determinar se vale o esforço de proceder com tentativas de exploração de falhas.

O Fator de Causa 4 "Serviço executado com privilégio administrativo e/ou sem restrições de ambiente" mostra que as conseqüências resultantes da exploração de falhas de programação do Bind podem variar segundo a forma como o sistema operacional prepara o ambiente de execução deste serviço.

O Mapa de Fatores de Causa contendo as recomendações para remediar estas falhas no servidor de DNS Bind é apresentado na Tabela 6.

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br><br>Execução desnecessária do serviço | Negação de serviço<br>Sobrecarga de *buffer*<br>Serviço indisponível | Desabilitar a execução do servidor HTTP nos computadores que não estiverem especificamente designados para atuar como *webservers*.<br>Configurar o *firewall* para não permitir conexões externas a servidores *web* não oficiais. |
| **Fator de causa 2** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Desatualização do *software* | Sobrecarga de *buffer*<br>Serviço indisponível<br>Computador invadido<br>Serviço modificado ilicitamente | Aplicar regularmente todos os *patches* de segurança lançados.<br>Atualizar o *software* para a última versão sempre que ocorrer publicação de falhas de segurança.<br>Documentar procedimentos de atualização do *software*, anexando-o a Política de Segurança de informações. |
| **Fator de causa 3** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Exibição da *string* contendo a versão do *software* | Sobrecarga de *buffer*<br>Computador invadido<br>Serviço indisponível<br>Serviço modificado ilicitamente | Desabilitar a exibição da *string* de versão do servidor HTTP que vai anexada as páginas de exibição de erro. |
| **Fator de causa 4** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Serviço executado com privilégio administrativo e/ou sem restrições de ambiente | Sobrecarga de *buffer*<br>Computador invadido<br>Serviço indisponível<br>Serviço modificado ilicitamente | Isolar o serviço de HTTP em um computador específico para esta finalidade, de forma que uma possível invasão através da exploração de vulnerabilidades neste serviço não venha a comprometer outros serviços compartilhados, como por exemplo o sistema de correio eletrônico ou um banco de dados.<br>Habilitar a execução do *webserver* em ambiente virtual, através da utilização do recurso *jail* ou similar, dependendo dos recursos do sistema operacional ou fabricante do produto. |

**Tabela 6. Tabela de Fatores de Causa para os servidores do protocolo HTTP.**

Em comparação com as vulnerabilidades do serviço de DNS implementado pelo Bind, pode-se perceber que este caso é um subconjunto do anterior. Isto ocorre por causa das características em comum existentes no modelo de desenvolvimento destes *softwares*. O Bind e o Apache são ambos *softwares* livres voltados a comunicação por soquetes TCP/IP programados em linguagem C com utilização da biblioteca *glibc* e compilador GCC (GNU Compiler Collection), da Free Software Foundation. Por essas similaridades, é natural que ambos contraiam as mesmas vulnerabilidades.

A seguir, na análise do sistema de autenticação do UNIX, que utiliza as mesmas ferramentas para programação, mas tem um propósito muito diferente, será possível constatar que seu Mapa de Causas Raiz é muito diferente dos antecessores.

## 6.4 Autenticação do UNIX

As senhas são utilizadas em virtualmente todas as interações entre usuários e sistemas de informações. A maioria das formas de autenticação, bem como a proteção de acesso a dados e arquivos, dependem fortemente nos mecanismos de autenticação fornecidos pelo fabricante do sistema.

Devido ao fato de que o acesso autenticado nem sempre fica registrado em trilhas de auditoria (*logs*), o investimento em quebrar senhas é uma oportunidade para o *hacker* que deseja acesso privilegiado de forma silenciosa. Um atacante, possuindo a senha de usuário para acesso a um sistema, tem ótimas chances de explorar uma falha ou instalar *kits* que lhe concedam o acesso administrativo desejado.

As conseqüências mais graves da fragilidade do subsistema de autenticação dos sistemas operacionais derivados do UNIX estão mapeadas no Mapa de Fatores de Causa ilustrado na Figura 14 e são três:

1. Utilização de senhas previsíveis;
2. Falta de procedimentos para troca de senhas do fabricante;
3. Falta de procedimentos para memorização ou armazenamento das senhas do usuário.

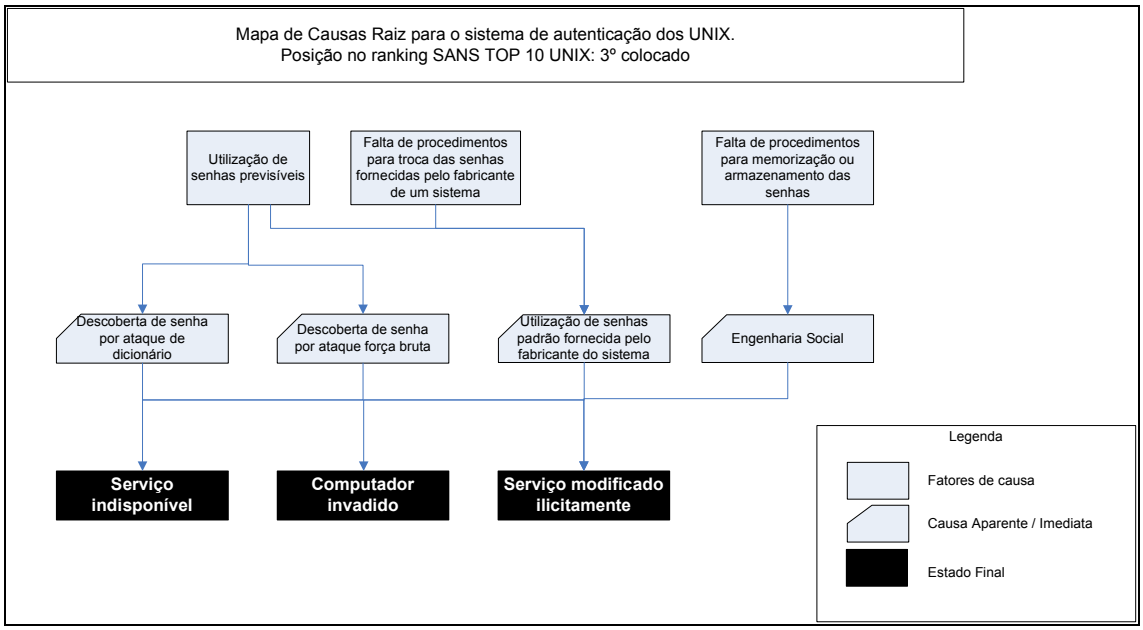



**Figura 14. Mapa de Causas Raiz para o sistema de autenticação do UNIX.**

O Fator de Causa 1 "Utilização de senhas previsíveis" define que a utilização de senhas fracas na autenticação do UNIX (assim como qualquer outro sistema de autenticação baseado em senhas) é um dos fatores de causa que levam a fácil descoberta das credenciais de acesso a este ambiente.

A Falta de procedimentos para troca das senhas fornecidas pelo fabricante de um sistema, Fator de Causa 2, trata dos problemas de falta de atenção na modificação das senhas conhecidas que são embutidas nos aplicativos de rede para que se possa fazer os ajustes iniciais de pós-instalação.

Já o Fator de Causa 3 "Falta de procedimentos para memorização ou armazenamento de senhas" traz a dificuldade de administração de pessoas que se deixam levar por ataques sociais onde a descoberta da credencial acontece a partir da simples solicitação do atacante (por telefone ou email, por exemplo) que se passa por um agente conhecido, como seu gerente ou cliente, por exemplo.

O Mapa de Fatores de Causa contendo as recomendações para o sistema de autenticação do UNIX é apresentado na Tabela 7.

| Fator de causa 1 | Mapa de Causas Raiz | Recomendações |
|---|---|---|
| Descrição:<br><br>Utilização de senhas previsíveis | Descoberta da senha por ataque de dicionário<br>Descoberta da senha por ataque de força bruta<br>Utilização de senhas padrão fornecida pelo fabricante do sistema<br>Serviço indisponível<br>Computador invadido<br>Serviço modificado ilícitamente | Habilitar os mecanismos de troca de senha compulsória, utilizando-se de dicionários similares aos dos atacantes para evitar composições fracas.<br>Habilitar os mecanismos de proteção contra solicitações repetitivas de autenticação<br>Auditar regularmente o arquivo de senhas, executando programas que tentam advinha-las da mesma forma como um *hacker* faria.<br>Aplicar diretrizes de uso aceitável de senhas como parte da Política de Segurança de Informações. |
| **Fator de causa 2** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Falta de procedimentos para troca das senhas fornecidas pelo fabricante de um sistema | Utilização de senhas padrão fornecida pelo fabricante do sistema<br>Serviço indisponível<br>Computador invadido<br>Serviço modificado ilícitamente | Habilitar os mecanismos de troca de senha compulsória, utilizando-se de dicionários similares aos dos atacantes para evitar composições fracas.<br>Auditar regularmente o arquivo de senhas, executando programas que tentam advinha-las da mesma forma como um *hacker* faria.<br>Aplicar diretrizes de uso aceitável de senhas como parte da Política de Segurança de Informações. |
| **Fator de causa 3** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Falta de procedimentos para memorização ou armazenamento das senhas | Engenharia social<br>Serviço indisponível<br>Computador invadido<br>Serviço modificado ilícitamente | Instruir o usuário a não passar sua credencial por qualquer meio e sob solicitação de qualquer pessoa.<br>Aplicar diretrizes de uso aceitável de senhas como parte da Política de Segurança de Informações. |

**Tabela 7. Tabela de Fatores de Causa para a autenticação do UNIX.**

## 6.5 Sistemas de controle de versões

Sistemas de controle de versões provêem ferramentas para gerenciar diferentes versões de documentos e códigos-fonte de *softwares*, facilitando o trabalho de equipes de

usuários que modificam os arquivos em acessos concorrentes. Tais sistemas são essenciais para o gerenciamento de qualquer projeto de desenvolvimento de *software* ou na manutenção de documentos corporativos porque proporcionam uma solução de armazenamento centralizado e permitem que diversas versões de um mesmo documento possam ser obtidas.

O CVS (*Concurrent Versions System*) é o *software* de controle de versões mais popular dos ambientes UNIX. Muitos projetos de código aberto permitem o acesso anônimo a seus repositórios que podem ser acessados remotamente através do protocolo **pserver** que fica em execução na porta 2401/tcp.

As vulnerabilidades nestes sistemas têm suas causas raiz originadas nos seguintes fatores de causa, exibidos na Figura 15.

1. Desatualização do *software*;

2. Ausência de criptografia na autenticação do serviço.

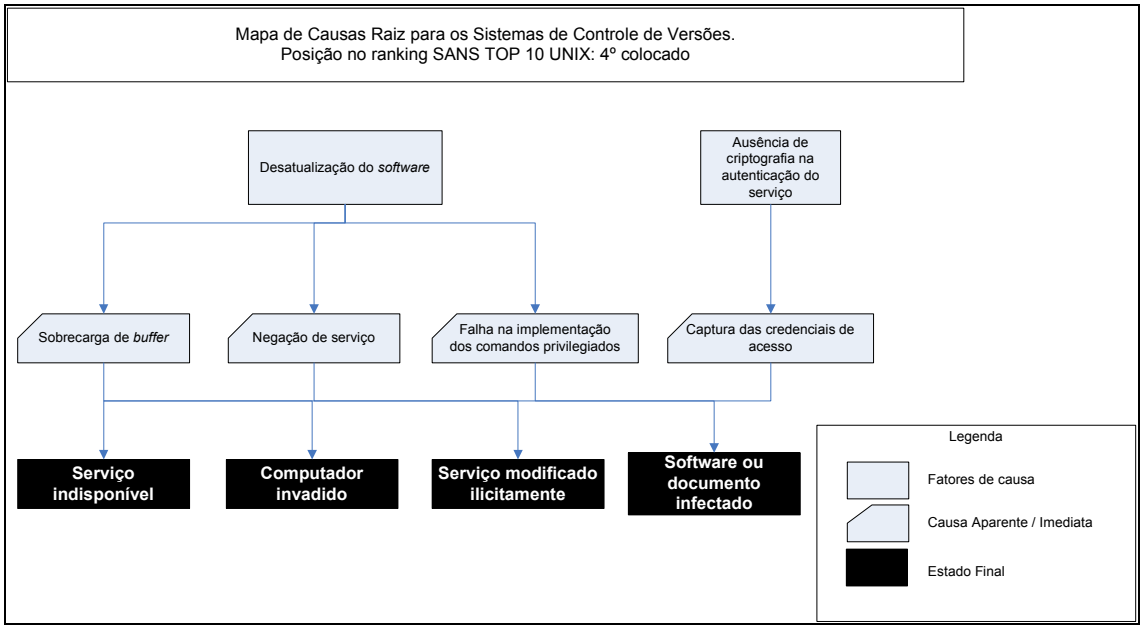



**Figura 15. Mapa de Causas Raiz para os Sistemas de Controle de Versões.**

O Fator de Causa 1 "Desatualização do *software*" aponta que a falta de atenção com as atualizações dos *softwares* que implementam o serviço de controle de versões é um dos principais fatores que geram invasões. Em geral, é muito freqüente a necessidade de atualizações destes sistemas devido à descoberta de falhas de segurança que são

aproveitadas pelos *hackers* para o comprometimento de sistemas desatualizados ou então, para execução de ataques que causam a negação (recusa) de serviços, deixando o computador e sua comunicação de rede inacessíveis.

O Fator de Causa 2 "Ausência de criptografia na autenticação do serviço" aponta que é possível com relativa facilidade capturar a senha utilizada para autenticação no serviço. No modelo de desenvolvimento de *softwares* de código aberto, o acesso anônimo é utilizado por qualquer interessado em obter a versão mais recente do sistema e o acesso privilegiado é utilizado pelo programador que desenvolve o produto. Um atacante ou programador mal intencionado pode capturar a credencial privilegiada e utilizá-la para escrever códigos de invasão como *backdoors* ou *rootkits*, afetando todas as pessoas que vierem a utilizar o programa servido pelo sistema.

O Mapa de Fatores de Causa contendo as recomendações para contenção destas falhas é apresentado na Tabela 8.

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br><br>Desatualização *do software*. | Sobrecarga de *buffer*<br>Negação de serviço<br>Falha na implementação dos comandos privilegiados<br>Serviço indisponível.<br>Computador invadido.<br>Serviço modificado ilicitamente. | Atribuir ao administrador do sistema necessidade de verificação diária da publicação de correções ou atualizações do *software* de controle de versões.<br>Aplicar regularmente todos *patches* de segurança publicados para o *software* de controle de versões. |

| Fator de causa 2 | Mapa de Causas Raiz | Recomendações |
|---|---|---|
| Descrição:<br><br>Ausência de criptografia na autenticação do serviço | Captura das credenciais de acesso<br>Serviço indisponível.<br>Computador invadido.<br>Serviço modificado ilicitamente. | Modificar o formato de conexões remotas, substituindo o protocolo **pserver** por conexões através de túneis criptografados, com auxílio do serviço de *shell* seguro (SSH).<br>Limitar a possibilidade de conexões autenticadas a usuários cadastrados em uma rede de trabalho virtual (VPN), mantendo a conexão anônima através do protocolo inseguro **pserver**. |

**Tabela 8. Tabela de Fatores de Causa para os Sistemas de Controle de Versões.**

## 6.6 Serviço de transporte de mensagens de correio eletrônico

Correio eletrônico é um dos serviços mais utilizados na Internet e o SMTP, (*Simple Messagem Transfer Protocol*), um dos protocolos mais antigos. Os agentes de transporte de correio são servidores responsáveis por receber *emails* de um remetente e repassá-lo a seus destinatários. Sendmail é o agente de transporte de correio mais utilizado, porém, devido a suas freqüentes falhas de segurança, vem recebendo concorrentes como o Qmail, Postfix, Courier-MTA e Exim (SANS, 2004).

Não é de surpreender que, sendo um dos serviços mais freqüentes da Internet, o protocolo SMTP seja alvo de tantos ataques, de acordo com a lista a seguir e como ilustrado no Mapa de Fatores de Causa da Figura 16.

1. Execução desnecessária do serviço;
2. Desatualização do *software*;
3. Ausência de mecanismos para conter *worms* e vírus;
4. Ausência de mecanismos para conter spam (mensagens de propaganda não solicitadas);
5. Ausência de mecanismos de autenticação;
6. Configuração inadequada do serviço.

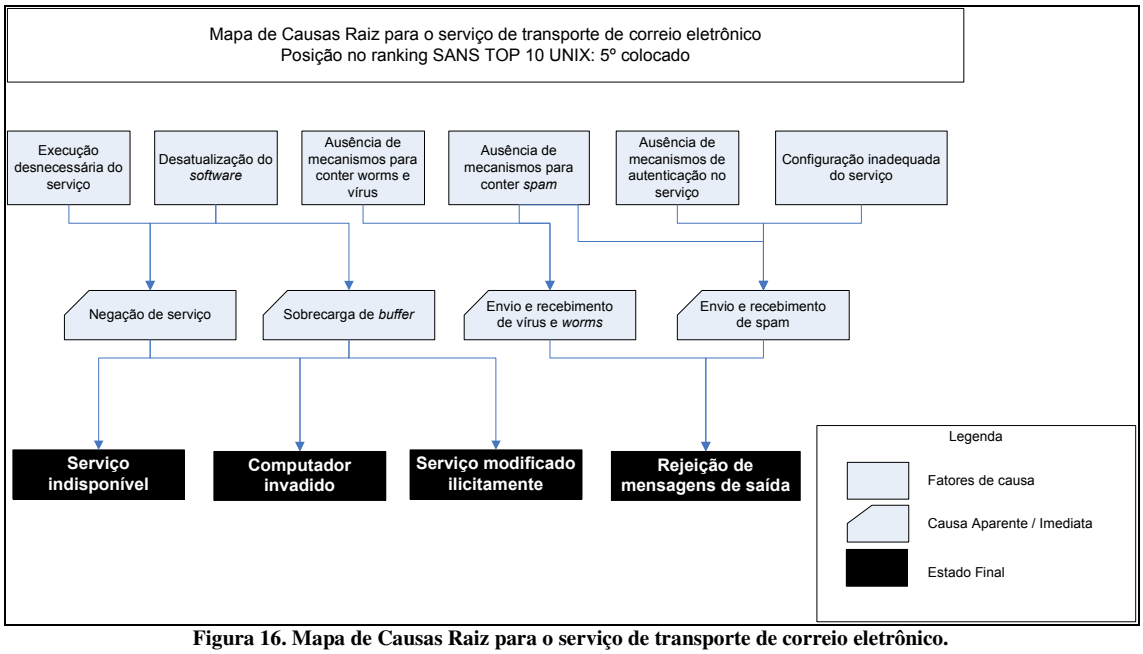



**Figura 16. Mapa de Causas Raiz para o serviço de transporte de correio eletrônico.**

O Fator de Causa 1 "Execução desnecessária do serviço" trata o problema da distribuição desnecessária de um agente de transporte de mensagens, geralmente o Sendmail, em cada instalação de um sistema operacional derivado do UNIX.

O Fator de Causa 2 "Desatualização do *software*" aponta que tanto a partir de uma instalação desnecessária como da utilização legítima de um servidor de correio eletrônico, o sistema está sujeito a vulnerabilidades caso não seja dada manutenção no *software* que desempenha o papel de transportador de mensagens de *email*.

O Fator de Causa 3 "Ausência de mecanismos para conter *worms* e vírus" trata dos problemas associados à disseminação de vírus e a utilização do serviço de correio eletrônico para esta finalidade.

O Fator de Causa 4 "Ausência de mecanismos para conter spam" trata dos problemas associados ao conteúdo das mensagens de *email* e da necessidade dos *hackers* em buscar servidores comprometidos para o envio de spam a todos os momentos.

O Fator de Causa 5 "Ausência de mecanismos de autenticação no serviço" aponta que muitas vezes não se utiliza os mecanismos já existentes de autenticação, relativamente novos nos agentes de transporte de correio, para evitar que pessoas não autorizadas façam uso ilícito do serviço de envio e da ocupação de recursos (processador, espaço em disco e banda de rede) associados a esta atividade.

O Fator de Causa 6 "Configuração inadequada do serviço" aponta que existem muitas outras configurações, além das tratadas nos Fatores de Causa anteriores, que permitem o uso mais seguro do serviço de correio, evitando que atacantes consigam a tomada do computador ou serviço de envio.

O Mapa de Fatores de Causa contendo as recomendações para contenção destas falhas é apresentado na Tabela 9.

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br><br>Execução desnecessária do serviço. | Negação de serviço.<br>Sobrecarga de *buffer*.<br>Serviço indisponível.<br>Computador invadido.<br>Serviço modificado ilicitamente. | Desabilitar a execução do agente de transporte de mensagens nos computadores que não estiverem especificamente designados para atuar como servidores.<br>Configurar o *firewall* para não permitir conexões externas a servidores de correio que não sejam os oficiais. |
| **Fator de causa 2** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Desatualização do *software*. | Negação de serviço.<br>Sobrecarga de *buffer*.<br>Serviço indisponível.<br>Computador invadido.<br>Serviço modificado ilicitamente. | Aplicar regularmente todos os *patches* de segurança lançados.<br>Atualizar o *software* para a última versão sempre que ocorrer publicação de falhas de segurança.<br>Documentar procedimentos de atualização do *software*, anexando-o a Política de Segurança de informações. |

| Fator de causa 3 | Mapa de Causas Raiz | Recomendações |
|---|---|---|
| Descrição:<br>Ausência de mecanismos para conter *worms* e vírus. | Envio e recebimento de mensagens com vírus e *worms*.<br>Rejeição de mensagens enviadas. | Considerar a utilização de *softwares* que atuam na inspeção de mensagens em busca de assinaturas de vírus ou *worms*.<br>Considerar ação conjunta com a utilização de antivírus nas estações de trabalho. |
| **Fator de causa 4** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br>Ausência de mecanismos para conter spam. | Envio e recebimento de mensagens contendo spam.<br>Rejeição de mensagens enviadas. | Considerar a utilização de *softwares* de inspeção de mensagens capazes de detectar conteúdos indesejados.<br>Considerar mecanismos avançados de autorização de envio de mensagens, como autenticação do serviço SMTP ou recursos alternativos como "POP before SMTP". |
| **Fator de causa 5** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br>Ausência de mecanismos de autenticação no serviço. | Envio e recebimento de vírus e *worms*.<br>Envio e recebimento de spam.<br>Rejeição de mensagens de saída. | Considerar a utilização de um agente de transporte de mensagens capaz de exigir autenticação antes de aceitar o envio de uma mensagem ou habilitar este mecanismo caso o agente forneça suporte.<br>Considerar mecanismos avançados de autorização de envio de mensagens, como autenticação do serviço SMTP ou recursos alternativos como "POP before SMTP". |
| **Fator de causa 6** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br>Configuração inadequada do serviço. | Envio e recebimento de vírus e *worms*.<br>Envio e recebimento de spam.<br>Rejeição de mensagens de saída. | Investir recursos humanos e tempo nas várias opções existentes em todos agentes de transporte de mensagens, visando a compreensão total de suas funcionalidades e as conseqüências em se habilitar ou desabilitar as opções.<br>Traduzir as configurações em regras documentadas na Política de Segurança, evitando que os *upgrades* ou trocas de *software* de envio de mensagens venha a trazer problemas já sanados anteriormente. |

**Tabela 9. Tabela de Fatores de Causa para o serviço de transporte de correio eletrônico.**

## 6.7 Protocolo de gerenciamento SNMP

O protocolo SNMP (Simple Network Management Protocol) é utilizado para monitoração e configuração remota de praticamente todos os equipamentos que contém o protocolo TCP/IP.

O SNMP consiste de diferentes tipos de mensagens trocadas entre estações de gerência e os dispositivos de rede que executam um *software* conhecido como agente. O método e a versão do protocolo utilizado para esta troca de mensagens, bem como o mecanismo de autenticação associado, possuem vulnerabilidades significativas apontadas na lista a seguir e tratadas no Mapa de Causas Raiz, na Figura 17.

1. Desatualização do *software*;

2. Exposição do acesso a gerência local de dispositivos;

3. Falta de política associada aos critérios de gerência de dispositivos.

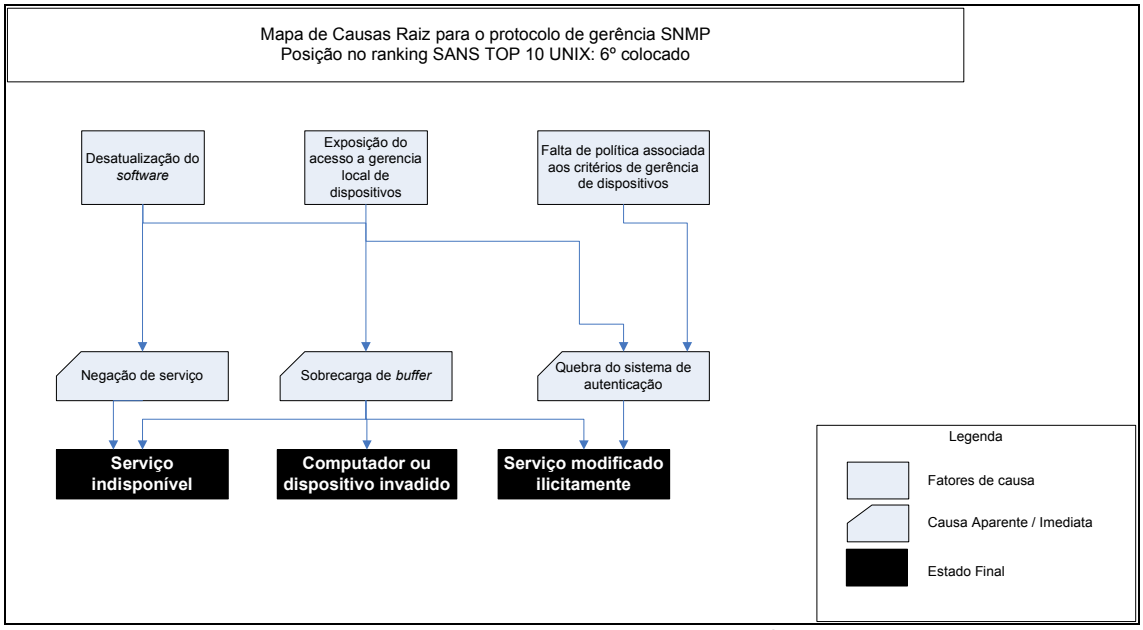



**Figura 17. Mapa de Causas Raiz para o protocolo de gerência SNMP.**

O Fator de Causa 1 "Desatualização do *software*" aponta que, nos ambientes derivados do UNIX, este também é um serviço que oferece graves vulnerabilidades quando fica exposto sem a devida manutenção de versão das implementações.

O Fator de Causa 2 "Exposição do acesso a gerência local de dispositivos" aponta que muitas vezes o serviço de gerência é local, ou seja, coletor e agente estão na mesma rede, porém o acesso fica liberado para qualquer computador da Internet, trazendo exposição desnecessária e grave para a utilização destes sistemas de gerência.

O Fator de Causa 3 "Falta de política associada aos critérios de gerência de dispositivos" trata dos casos em que não se estipula procedimentos formais de proteção

dos agentes em execução nos equipamentos.

As recomendações para tratar cada um dos Fatores de Causa encontrados é apresentado na Tabela 10.

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br>Desatualização do *software*. | Negação de serviço.<br>Sobrecarga de *buffer*.<br>Serviço indisponível.<br>Computador ou dispositivo invadido. | Considerar a utilização exclusiva da terceira versão deste protocolo (SNMP v3), tirando proveito do modelo de segurança baseado em usuário com autenticação de mensagens e a criptografia de informações.<br>Aplicar regularmente todos os *patches* de segurança lançados.<br>Atualizar o *software* para a última versão sempre que ocorrer publicação de falhas de segurança.<br>Documentar procedimentos de atualização do *software*, anexando-o à Política de Segurança de informações. |
| **Fator de causa 2** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br>Exposição do acesso a gerência local de dispositivos. | Quebra do sistema de autenticação.<br>Serviço modificado ilicitamente. | Bloquear no *firewall* a possibilidade de passagem de datagramas UDP e pacotes TCP com destino a porta de gerência de dispositivos na rede local.<br>Habilitar um Sistema de Detecção de Intrusões para relatar tentativas de coleta de informações da rede local através do protocolo SNMP. |
| **Fator de causa 3** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br>Falta de política associada aos critérios de gerência de dispositivos. | Quebra do sistema de autenticação.<br>Serviço modificado ilicitamente. | Elaborar documentação que descreva, para cada produto de rede existente no ambiente, como deve ser feita a configuração do dispositivo, considerando os recursos que cada fabricante embute nos agentes fornecidos.<br>Executar testes periódicos de tentativa de descoberta do nome da comunidade de gerência, para agentes que ainda estão implementados nas duas primeiras versões do protocolo SNMP.<br>Executar a mudança de nome da comunidade frequentemente, evitando assim que atacantes com acesso a rede local possam interferir na continuidade de prestação de serviços de rede. |

**Tabela 10. Tabela de Fatores de Causa para o protocolo de gerenciamento SNMP.**

## 6.8 Open Secure Sockets Layer (SSL)

A biblioteca de código aberto OpenSSL provê suporte criptográfico para aplicações se comunicam por meio de uma rede TCP/IP. É uma implementação muito utilizada do protocolo SSL/TLS (Secure Sockets Layer)/(Transport Layer Security) e está embutida em diversas aplicações, como servidores POP3, IMAP, SMTP, LDAP e HTTP.

Estando o OpenSSL embutido em várias aplicações, qualquer vulnerabilidade nesta biblioteca acarreta o comprometimento de diversos *softwares*, por este motivo, o OpenSSL ocupa o sétimo lugar nesta lista.

O principal Fator de Causa associado ao OpenSSL é a "Desatualização do *Software*", tratado na Figura 18. Este fator de causa, resultado da falta de manutenção na atualização e correção de falhas de segurança descobertas e publicadas para esta biblioteca é o que permite a ação dos *hackers* que aplicam tentativas de invasão por este meio.

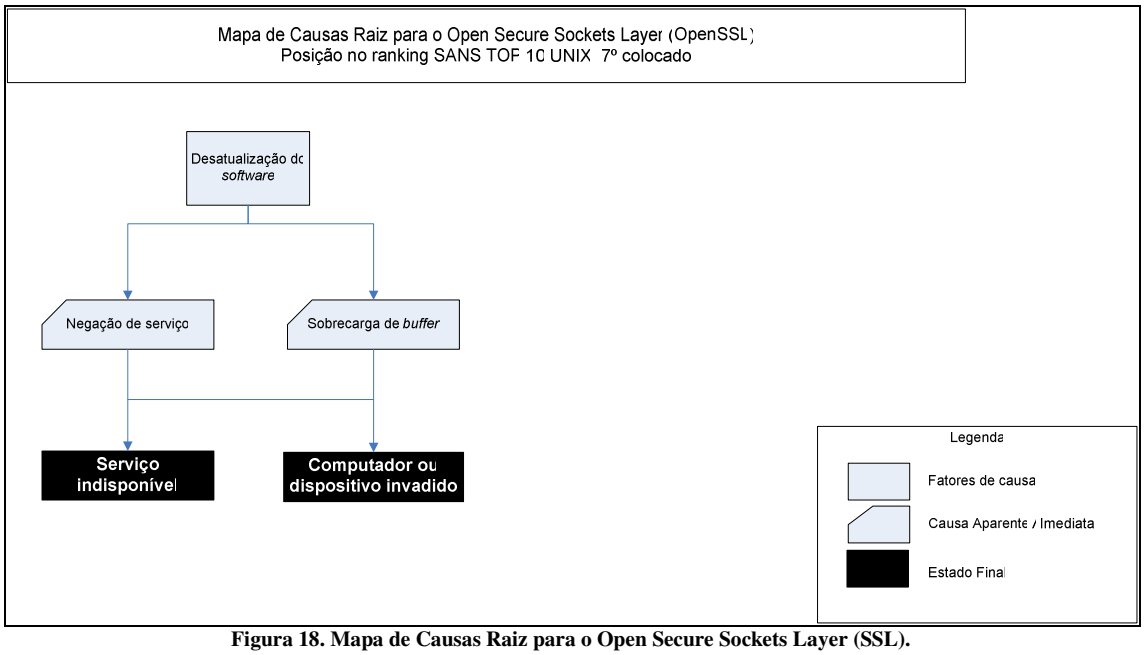



**Figura 18. Mapa de Causas Raiz para o Open Secure Sockets Layer (SSL).**

A Tabela 11 informa as recomendações para evitar o comprometimento do serviços de rede dependentes da biblioteca OpenSSL.

| Fator de causa | Mapa de Causas Raiz | Recomendações |
|---|---|---|
| Descrição:<br>Desatualização do *software*. | Negação de serviço.<br>Sobrecarga de *buffer*.<br>Serviço indisponível.<br>Computador ou dispositivo invadido. | Aplicar regularmente todos os *patches* de segurança lançados.<br>Atualizar a biblioteca OpenSSL para a última versão sempre que ocorrer publicação de falhas de segurança.<br>Documentar procedimentos de atualização do *software*, anexando-o à Política de Segurança de informações.. |

**Tabela 11. Tabela de Fatores de Causa para o Open Secure Sockets Layer (SSL).**

## 6.9 Serviços de compartilhamento de informações NIS/NFS

O *Network File System* (Sistema de arquivos de rede) e o *Network Information Service* (Serviço de informações de rede) são dois serviços importantes utilizados por sistemas operacionais derivados do UNIX para a centralização de informações de autenticação e o compartilhamento dos arquivos de trabalho na rede.

O NFS foi projetado para compartilhar (ou, exportar) sistemas de arquivos e diretórios através da rede. Por outro lado, o NIS é um conjunto de serviços que provê a localização de informações, chamadas de mapas, para outros computadores. Os mapas mais comuns são a sincronização dos arquivos *passwd* e *group*, que contém a base de usuários e senhas daquela rede.

Os dois Fatores de Causa mais ameaçadores para os serviços NIS/NFS são apresentados na lista abaixo e estão mapeados na Figura 19.

1. Desatualização do *software*;

2. Má configuração do serviço.

O Fator de Causa 1 "Desatualização do *software*" aponta que as falhas de segurança não remediadas nos serviços NIS/NFS podem deixar o sistema sujeito a ataques de negação de serviços ou a sobrecarga de *buffers*.

O Fator de Causa 2 "Má configuração do serviço" trata os casos em que, por falta de conhecimento, tempo ou atenção, o administrador do sistema deixa de aplicar as restrições de acesso para dificultar ou impedir a ação dos *hackers*.

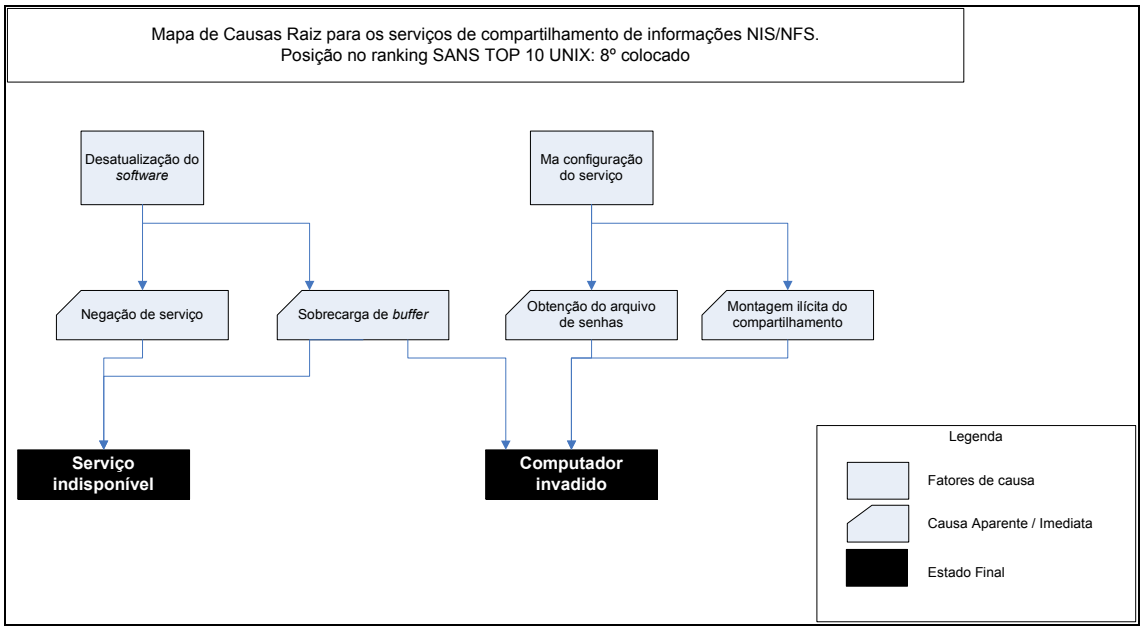



**Figura 19. Mapa de Causas Raiz para os serviços de compartilhamento de informações NIS/NFS.**

A Tabela 12 contém as recomendações que podem ser tomadas para evitar o comprometimento da rede e dos computadores por falta de medidas de segurança nos serviços NIS e NFS:

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br>Desatualização do *software*. | Negação de serviço.<br>Sobrecarga de *buffer*.<br>Serviço indisponível. | Aplicar regularmente todos os *patches* de segurança lançados.<br>Atualizar o *software* para a última versão sempre que ocorrer publicação de falhas de segurança.<br>Documentar procedimentos de atualização do *software*, anexando-o à Política de Segurança de informações.. |
| **Fator de causa 2** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br>Má configuração do serviço. | Negação de serviço.<br>Sobrecarga de *buffer*.<br>Serviço indisponível. | Investir recursos humanos e tempo nas várias opções de configuração destes serviços, visando a compreensão total de suas funcionalidades e as conseqüências em se habilitar ou desabilitar suas opções.<br>Traduzir as configurações em regras documentadas na Política de Segurança, evitando que novos computadores venham a ser configurados sem os benefícios adotados nas instalações atuais.<br>. |

**Tabela 12. Tabela de Fatores de Causa para os serviços de compartilhamento de informações NIS/NFS.**

## 6.10 Bancos de dados

Os bancos de dados são elementos presentes nos negócios eletrônicos, repositórios de informações trabalhadas por *webservers*, sistemas de ERP (*Enterprise Resources Planning*), e em várias outras finalidades. Mesmo com a necessidade implícita de integridade e confidencialidade, os Sistemas Gerenciadores de Bancos de Dados (SGBDs) não têm recebido o mesmo nível de segurança que os encontrados nos equipamentos de rede e sistemas operacionais que os hospedam.

Os Fatores de Causa que mais causam o comprometimento dos SGBDs são apresentados na lista abaixo e estão mapeados na Figura 20.

1. Falhas na configuração dos bancos de dados;
2. Ausência de criptografia no sistema de autenticação;
3. Falhas na programação do usuário;
4. Ausência de política de manutenção do SGBD.

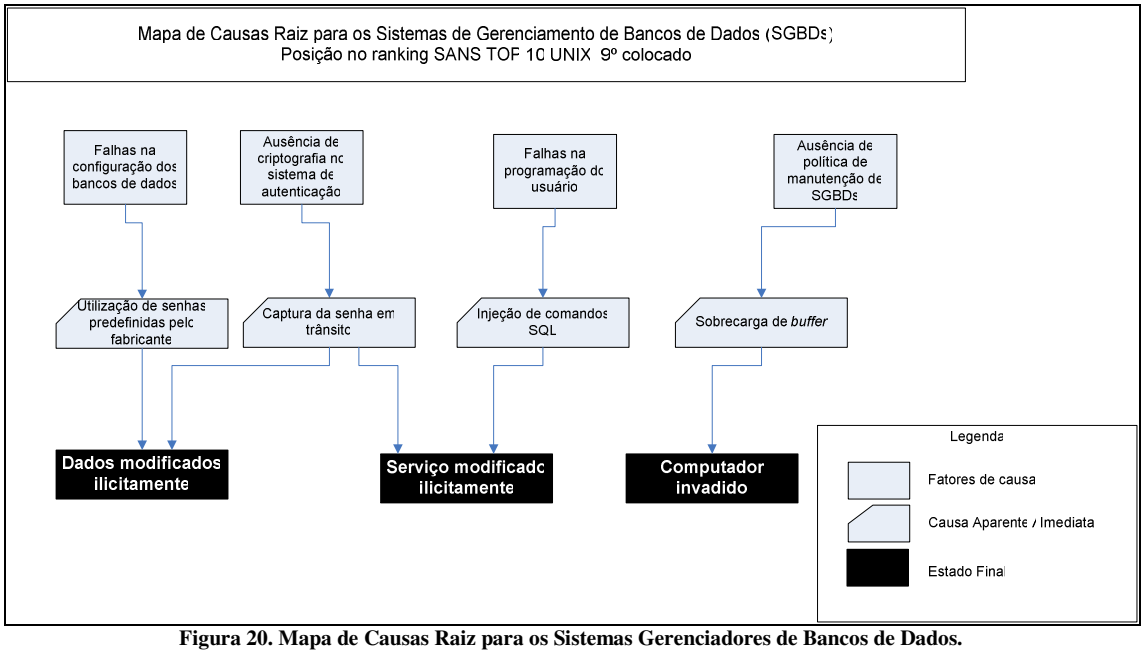



**Figura 20. Mapa de Causas Raiz para os Sistemas Gerenciadores de Bancos de Dados.**

O Fator de Causa 1 "Falhas na configuração dos bancos de dados" trata dos problemas associados às falhas humanas na etapa de configuração do funcionamento dos bancos de dados. Dependendo da implementação, pode haver contas predefinidas, habilitação da

porta de comunicações por TCP/IP ou permissão de acesso com base nos mecanismos já oferecidos pelo sistema operacional.

O Fator de Causa 2 "Ausência de criptografia no sistema de autenticação" aponta que a maioria dos SGBDs, como Oracle, Sybase, MySQL e PostgreSQL utilizam transporte de autenticação em texto em claro, onde o SSL é apenas opcional.

O Fator de Causa 3 "Falhas na programação do usuário" define que outra parcela das vulnerabilidades que levam ao comprometimento do SGBD é a falta de qualidade no código escrito pelo usuário quando este implementa algum controle de acesso ou manipula as informações de autenticação do banco. Independente do formato da interface, via console ou *web* por exemplo, é possível a um *hacker* com conhecimentos de injeção de comandos SQL, adicionar-se como usuário do sistema ou obter uma listagem dos *hashes* de senhas, por exemplo, comprometendo a integridade das informações.

O Fator de Causa 4 "Ausência de políticas de manutenção de SGBDs" aponta que muitas vezes, o motivo do comprometimento dos bancos de dados está na ausência de documentações e procedimentos que definam sob quais proteções uma nova instalação de SGBD deve estar para ser considerada segura. A falta de atenção para com este Fator de Causa pode levar, por exemplo, a descoberta de vulnerabilidades não tratadas na versão do SGBD utilizado, e por conseqüência, a exploração de uma sobrecarga de *buffer*, por exemplo.

A Tabela 13 contém as recomendações que podem ser tomadas para evitar o comprometimento dos Sistemas Gerenciadores de Bancos de Dados:

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br><br>Falha na configuração do banco de dados. | Utilização de senhas predefinidas pelo fabricante.<br>Dados modificados ilicitamente. | Tomar conhecimento das senhas predeterminadas e elaborar procedimento para evitar que estas sejam deixadas após a instalação do *software*. |

| Fator de causa 2 | Mapa de Causas Raiz | Recomendações |
|---|---|---|
| Descrição:<br><br>Ausência de criptografia no sistema de autenticação. | Captura da senha em trânsito<br>Dados modificados ilicitamente<br>Serviço modificado ilicitamente. | Impedir o uso do SGBD através de redes inseguras, como a Internet, ou providenciar mecanismos de criptografia para usuários remotos como VPN.<br>Habilitar suporte criptográfico na compilação do *software* (código aberto) ou instalar este suporte a partir de produtos de terceiro (codigo fechado/comercial).<br>Auditar a rede permanentemente visando detectar a presença de interfaces de rede em modo promíscuo. |
| **Fator de causa 3** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Falhas na programação do usuário. | Injeção de comandos SQL.<br>Serviço modificado ilicitamente. | Adotar metodologia de programação segura através de treinamento, utilização de bibliotecas preexistentes e delegando ao banco (e não à aplicação) a tarefa de autenticar entidades.<br>Instalar, na aplicação, detectores de intrusão específicos para a tarefa de detecção de comandos de injeção de SQL.<br>. |
| **Fator de causa 4** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br><br>Ausência de política de manutenção do SGBD. | Sobrecarga de *buffer*.<br>Computador invadido. | Considerar a contratação de um administrador de banco de dados com o perfil exclusivamente voltado às atividades de manutenção da base de dados e do *software* de gerência.<br>Estipular critérios, documentados na política de segurança, de como proceder para garantir que a versão instalada do banco de dados seja sempre a mais recente. |

**Tabela 13. Tabela de Fatores de Causa para os Bancos de dados.**

## 6.11 Kernel

O componente central de qualquer sistema operacional chama-se *kernel* (ou cerne/núcleo, em português). O *kernel* é responsável pelas principais interações de baixo nível entre o sistema operacional e o *hardware*, memória, escalonador, comunicações entre processos, sistema de arquivos e outros.

Devido ao acesso privilegiado do *kernel* a todos outros componentes do sistema operacional, um comprometimento em seu nível é sempre desastroso. Os riscos incluem

ataques de negação de serviços, execução de código arbitrário, acesso irrestrito ao sistema de arquivos e a tomada da conta do administrador.

Os Fatores de Causa associados às vulnerabilidades dos *kernels* dos sistemas operacionais derivados do UNIX são listados abaixo e tratados no Mapa de Causas Raiz da Figura 21.

1. Desatualização do *kernel*;

2. Má configuração do *kernel*;

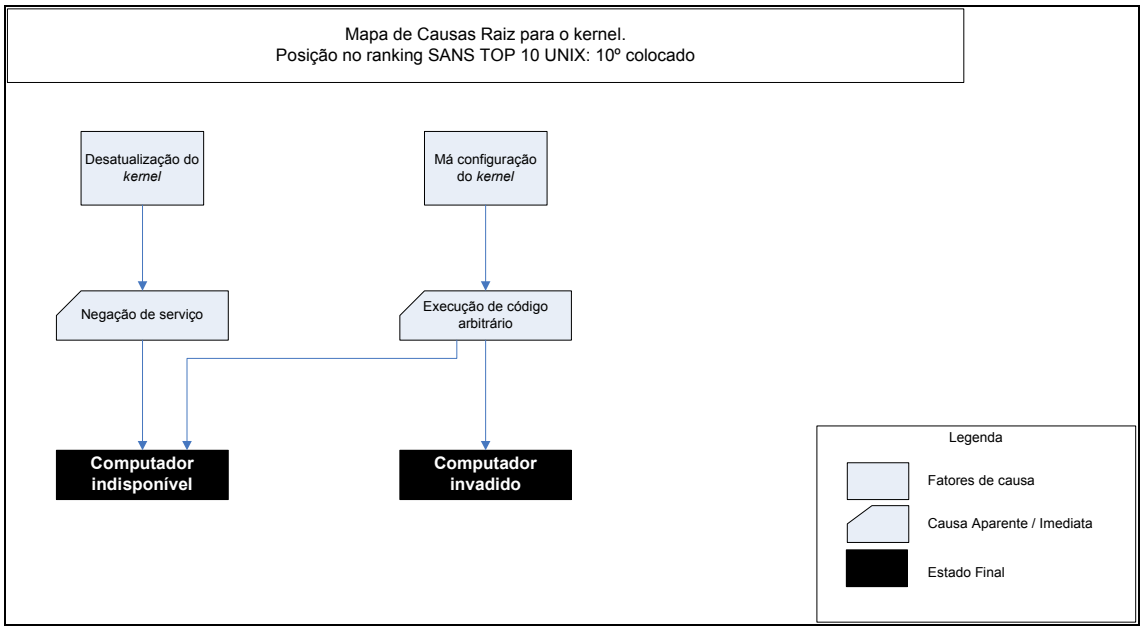



**Figura 21. Mapa de Causas Raiz para o kernel.**

O Fator de Causa 1 "Desatualização do *kernel*" aponta o caso em que a ausência de manutenção do *kernel* pode levar o sistema a um estado de indisponibilidade caso um *hacker* se utilize de uma vulnerabilidade conhecida que permita ataques de negação de serviços.

O Fator de Causa 2 "Má configuração do *kernel*" trata dos casos de *kernel* dos derivados do UNIX de código aberto, em que o usuário conta com a possibilidade de configurar cada funcionalidade para seu sistema. Apesar de haver muitos benefícios neste modelo de trabalho, muitas vezes acontece de se criar vulnerabilidades adicionais particulares para cada instalação de um mesmo sistema.

A Tabela 14 contém as recomendações que podem ser tomadas para evitar o

comprometimento do *kernel*:

| **Fator de causa 1** | **Mapa de Causas Raiz** | **Recomendações** |
|---|---|---|
| Descrição:<br>Desatualização do *kernel*. | Negação de serviço.<br>Computador indisponível. | Aplicar regularmente todos os *patches* de segurança lançados.<br>Atualizar o *kernel* para a última versão sempre que ocorrer publicação de falhas de segurança.<br>Documentar procedimentos de atualização do *kernel*, anexando-o a Política de Segurança de informações.<br>Considerar a contratação de soluções de nível corporativo para sistemas operacionais de código aberto, recebendo assim versões seguras de *kernel*, bem como pacotes pré-compilados que podem ser facilmente atualizados em caso de descoberta de vulnerabilidades.. |
| **Fator de causa 2** | **Mapa de Causas Raiz** | **Recomendações** |
| Descrição:<br>Má configuração do *kernel*.. | Execução arbitrária de código<br>Computador invadido. | Documentar procedimentos de configuração do *kernel*, anexando-o a Política de Segurança de informações.<br>Considerar a contratação de soluções de nível corporativo para sistemas operacionais de código aberto, recebendo assim versões seguras de *kernel*, bem como pacotes pré-compilados que podem ser facilmente atualizados em caso de descoberta de vulnerabilidades. |

**Tabela 14. Tabela de Fatores de Causa para o kernel.**

Com este estudo, transferiram-se as vulnerabilidades presentes no *ranking* TOP 10 do SANS Institute sob forma de Causas Aparentes, onde foi possível levantar 33 Fatores de Causa e propor um total de 75 recomendações eficazes no tratamento da recorrência de eventos, conforme a Tabela 15.

| Resumo quantitativo | |
|---|---|
| Itens vulneráveis | 10 |
| Causas aparentes apresentadas | 33 |
| Fatores de causa levantados | 33 |
| Recomendações propostas | 75 |

**Tabela 15. Resumo quantitativo do estudo.**

O próximo capítulo aborda a aplicação do conteúdo desenvolvido aqui, no Sistema de Detecção de Intrusões Snort.

# 7 Aplicação da Análise de Causa Raiz no IDS Snort

A aplicação da RCA no Snort consiste em utilizar os Mapas de Causas Raiz e as Tabelas de Fatores de Causa gerados no capítulo anterior, para definir um subconjunto de alertas correspondente as 10 vulnerabilidades mais críticas para os ambientes UNIX.

## 7.1 Procedimentos de adição de informações no IDS Snort

A Figura 22 ilustra o esquema geral de componentes que receberam informações para contemplar as Análises de Causa Raiz:

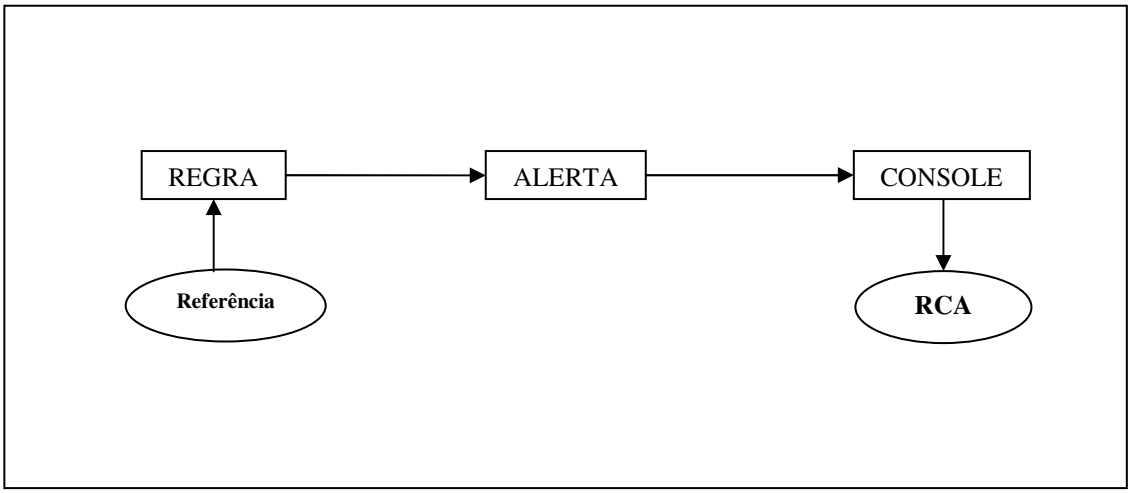



**Figura 22. Procedimento de alteração do IDS Snort.**

As regras que alimentam o Snort são extensíveis tanto em quantidade (adição de novas regras) como também na possibilidade de agregação de informações a regras já existentes. Para isso utiliza-se o campo "reference", em destaque (negrito sublinhado) na Figura 23.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS zone
transfer TCP"; flow:to_server, established; content: "|00 00 FC|";
offset:15; reference:cve,CAN-1999-0532; reference:arachnids,212;
classtype:attempted-recon; sid:255; rev:8;)
```

**Figura 23. Presença do campo "reference" em uma regra do Snort.**

A motivação original para a presença do campo "reference" é aumentar a credibilidade para a finalidade de existência de uma regra. Desta forma, quando um alerta é gerado, ele leva consigo URLs (*Universal Resource Locators*) que podem ser acessadas em tempo real, contendo explicações técnicas e detalhadas sobre os motivos pelos quais o alerta pode ter sido emitido. O alerta na Figura 24 exibe e destaca essa situação:

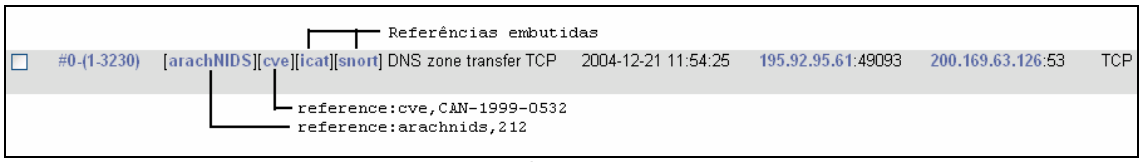



**Figura 24. Alerta exibindo as referências externas contidas na regra de origem.**

A primeira alteração consiste em aproveitar este sistema de exibição de informações e adicionar ao *software,* instruções para acesso a uma nova base de referências. Isto é feito através da edição do arquivo de configuração auxiliar denominado "reference.config". A Figura 25 exibe o arquivo e as adições (em negrito sublinhado):

```
# $Id: reference.config,v 1.4 2003/10/20 15:03:04 chrisgreen Exp $
# The following defines URLs for the references found in the rules
#
# config reference: system URL

config reference: bugtraq   http://www.securityfocus.com/bid/
config reference: cve       http://cve.mitre.org/cgi-bin/cvename.cgi?name=
config reference: arachNIDS http://www.whitehats.com/info/IDS
config reference: rootcause http://vulture.univali.br/rca/

# Note, this one needs a suffix as well.... lets add that in a bit.
config reference: McAfee    http://vil.nai.com/vil/content/v_
config reference: nessus    http://cgi.nessus.org/plugins/dump.php3?id=
config reference: url       http://
```

**Figura 25. Modificação no arquivo reference.config.**

Deste modo, assim que uma regra contiver uma referência do tipo "rootcause", o console será capaz de apontar a Análise de Causa Raiz correspondente através da passagem do código de identificação da regra como argumento no final da URL fornecida neste arquivo.

Em seguida, torna-se possível adicionar referências de Análise de Causa Raiz nas regras, como mostra a Figura 26 (adições em negrito sublinhado).

```
# (C) Copyright 2001,2002, Martin Roesch, Brian Caswell, et al.
#    All rights reserved.
# $Id: dns.rules,v 1.31 2003/11/20 20:56:56 cazz Exp $
#----------
# DNS RULES
#----------

alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS zone transfer TCP";
flow:to_server,established; content: "|00 00 FC|"; offset:15; reference:cve,CAN-
1999-0532; reference:arachnids,212; reference:rootcause,255; classtype:attempted-
recon; sid:25 5; rev:8;)
```

**Figura 26. Adição de referência de Análise de Causa Raiz em uma regra.**

Como resultado, obtém-se uma nova referência que conduz à Análise de Causa Raiz correspondente quando um alerta contemplado por esta análise é gerado (Figura 27):

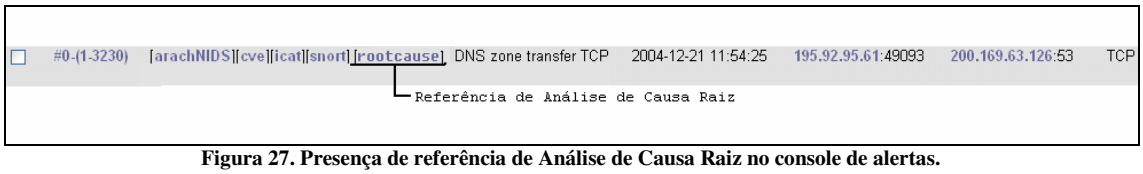



**Figura 27. Presença de referência de Análise de Causa Raiz no console de alertas.**

Assim, o administrador do sistema, além de contar com as descrições técnicas a respeito dos motivos que geraram o alarme, pode também acessar o estudo de Análise de Causa Raiz associado a este evento.

O resultado final é a apresentação da Tabela de Fatores de Causa, contendo as recomendações que, se executadas, evitam a recorrência do alerta. O exemplo da Figura 28 contém a análise de RCA para o caso de tentativa de transferência de zonas de DNS, Fator de Causa 5 do primeiro estudo de caso, abordado na seção 6.1.

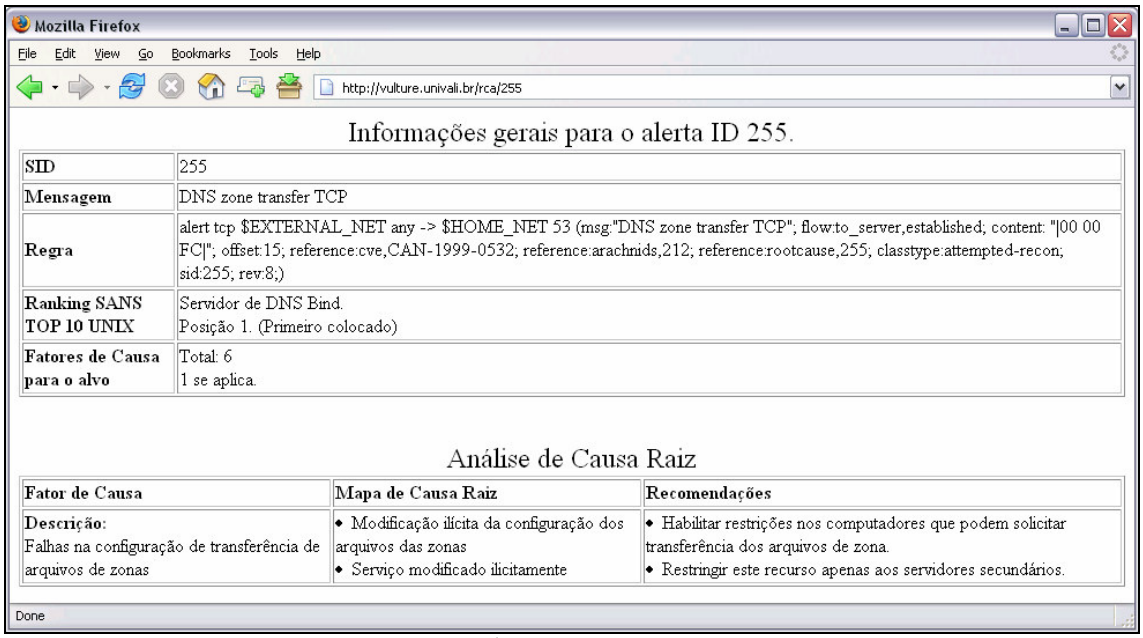



**Figura 28. Apresentação da Análise de Causa Raiz gerado a partir de um alerta.**

## 7.2 Mapeamento da relação entre Regras e Fatores de Causa

Dado que o Fator de Causa é o motivo imediato para o qual um alerta é gerado, e que uma regra de IDS tem a mesma finalidade: relatar um evento imediato e recorrente que merece atenção do analista, é cabível proceder com o mapeamento da relação entre os Fatores de Causa dos objetos estudados no Capítulo 6 e as regras fornecidas pelo IDS Snort. Este mapeamento, em que 33 Fatores de Causa tem seus correspondentes localizados em meio a 3.015 regras da atual base do Snort é também uma das principais contribuições deste trabalho.

O objetivo do procedimento é replicar os passos de alimentação do Snort apresentados na seção anterior (7.1) de forma a experimentar a Análise de Causa Raiz nas regras relacionadas às vulnerabilidades presentes no *ranking* TOP 10 do SANS Institute. Os resultados estão na Tabela 16.

| **1. Servidor de DNS Bind** | | |
|---|---|---|
| **Fator de Causa** | **Regra** | **Comentário** |
| Execução desnecessária do serviço | N/A | Saber se um serviço é necessário ou não é cabível somente ao administrador da rede. |
| Desatualização do *software* | alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS EXPLOIT named tsig overflow attempt"; flow:to_server,established; content:"\|AB CD 09 80 00 00 00 01 00 00 00 00 00 00 01 00 01\| \|02\|a"; reference:arachnids,482; reference:bugtraq,2302; reference:cve,2001-0010; **reference:rootcause,303;** classtype:attempted-admin; sid:303; rev:11;) | O IDS detecta traços de comandos binários que objetivam a sobrecarga de *buffer* dos servidores de DNS com versão de *software* desatualizada e vulnerável. |
| Exibição da *string* contendo a versão do *software* | alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS named version attempt"; flow:to_server,established; content:"\|07\|version"; offset:12; nocase; content:"\|04\|bind"; offset:12; nocase; reference:arachnids,278; reference:nessus,10028; **reference:rootcause,257;** classtype:attempted-recon; sid:257; rev:8;) | O IDS captura o trânsito da exibição da versão do serviço de DNS em execução. |
| Serviço executado com privilégio administrativo e/ou sem restrições de ambiente | N/A | Detecção de privilégios ou restrições são capacidades cabíveis somente a IDSs baseados em *host*. |
| Falhas na configuração de transferência de arquivos de zonas | alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS zone transfer TCP"; flow:to_server,established; content:"\|00 00 FC\|"; offset:15; reference:arachnids,212; reference:cve,1999-0532; reference:nessus,10595; **reference:rootcause,255;** classtype:attempted-recon; sid:255; rev:13;) | O IDS captura transferências de zona por entre as redes que audita. |
| Serviço de consulta recursiva habilitado desnecessariamente | alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS TCP inverse query"; flow:to_server,established; byte_test:1,<,16,2; byte_test:1,&,8,2; reference:bugtraq,2302; reference:cve,2001-0010; **reference:rootcause,2922;** classtype:attempted-recon; sid:2922; rev:1;) | O IDS detecta o trânsito de respostas recursivas do serviço de DNS. |
| | | |
| **2. Servidores HTTP** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Execução desnecessária do serviço | N/A | Saber se um serviço é necessário ou não é cabível somente ao administrador da rede. |
| | | |
| Desatualização do *software* | alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-PHP strings overflow"; flow:to_server,established; content:"\|BA\|I\|FE FF FF F7 D2 B9 BF FF FF FF F7 D1\|"; reference:arachnids,431; reference:bugtraq,802; **reference:rootcause,1085;** classtype:web-application-attack; sid:1085; rev:8;) | O IDS detecta traços de comandos binários que objetivam a sobrecarga de *buffer* dos *webservers* com versão de *software* desatualizada e vulnerável. |

| | | |
|---|---|---|
| Exibição da *string* contendo a versão do *software* | `alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-MISC HTTP Version String"; flow:to_server,established; content:"Apache/"; nocase; isdataat:6,relative; content:!"|0A|"; within:5; reference:bugtraq,9809; reference:nessus,11593; **reference:rootcause,2570;** classtype:non-standard-protocol; sid:2570; rev:7;)` | O IDS detecta a passagem das informações de versão do *webserver* pela porta 80/tcp. |
| Serviço executado com privilégio administrativo e/ou sem restrições de ambiente | N/A | Detecção de privilégios ou restrições são capacidades cabíveis somente à IDSs baseados em *host*. |
| | | |
| **3. Autenticação do UNIX** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Utilização de senhas previsíveis | N/A | Detecção deste tipo de Fator de Causa é uma capacidade cabível somente à IDSs baseados em *host*. |
| Falta de procedimentos para troca das senhas fornecidas pelo fabricante | `alert tcp $EXTERNAL_NET any -> $TELNET_SERVERS 23 (msg:"TELNET APC SmartSlot default admin account attempt"; flow:to_server,established; content:"TENmanUFactOryPOWER"; reference:bugtraq,9681; reference:cve,2004-0311; reference:nessus,12066; **reference:rootcause,2406;** classtype:suspicious-login; sid:2406; rev:4;)` | O IDS pode detectar a presença e tráfego de textos utilizados como senha padrão. O caso colocado aqui é apenas um exemplo do potencial de uso, porém indica, da mesma forma que os demais, a Análise de Causa Raiz correspondente. |
| Falta de procedimentos para memorização ou armazenamento de senhas | N/A | Este tipo de procedimento depende ou de ações humanas ou ferramentas auxiliares. Não tem relação direta com IDSs. |
| | | |
| **4. Sistemas de Controle de Versões** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Desatualização do *software* | `alert tcp $HOME_NET 2401 -> $EXTERNAL_NET any (msg:"MISC CVS double free exploit attempt response"; flow:from_server,established; content:"free|28 29 3A| warning|3A| chunk is already free"; reference:bugtraq,6650; **reference:rootcause,2010;** reference:cve,2003-0015; classtype:misc-attack; sid:2010; rev:4;)` | O IDS é capaz de detectar a presença de comandos que causam sobrecarga de *buffer*, cujo alvos são *softwares* desatualizados. |
| Ausência de criptografia na autenticação do serviço | `alert tcp $HOME_NET 2401 -> $EXTERNAL_NET any (msg:"MISC CVS invalid user authentication response"; flow:from_server,established; content:"E Fatal error, aborting."; content:"|3A| no such user"; **reference:rootcause,2008;** classtype:misc-attack; sid:2008; rev:4;)` | O IDS é capaz de remontar protocolos de autenticação e apontar falhas no momento do *login* em servidores CVS. |
| | | |
| **5. Serviço de Transporte de Mensagens (SMTP)** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Execução desnecessária do serviço | N/A | Detecção deste tipo de Fator de Causa é uma capacidade cabível somente à IDSs baseados em *host*. |
| Desatualização do *software* | `alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"SMTP sendmail 8.6.9c exploit"; flow:to_server,established; content:"|0A|Croot|0D` | O IDS é capaz de detectar a presença de comandos que |

| | | |
|---|---|---|
| | 0A\|Mprog"; reference:arachnids,141; reference:bugtraq,2311; reference:cve,1999-0204; **reference:rootcause,671;** classtype:attempted-user; sid:671; rev:8;) | causam sobrecarga de *buffer*, cujo alvos são *softwares* desatualizados. |
| Ausência de mecanismos para conter *worms* e vírus | alert tcp $HOME_NET any -> $EXTERNAL_NET 25 (msg:"VIRUS OUTBOUND bad file attachment"; flow:to_server,established; content:"Content-Disposition\|3A\|"; nocase; pcre:"/filename\s*=\s*.*?\.(?=[abcdehijlmnoprsvwx])(a(d[ep]\|s[dfx])\|c([ho]m\|li\|md\|pp)\|d(iz\|ll\|ot)\|e(m[fl]\|xe)\|h(lp\|sq\|ta)\|jse?\|m(d[abew]\|s[ip]) \|p(p[st]\|if\|[lm]\|ot)\|r(eg\|tf)\|s(cr\|[hy]s\|wf)\|v(b[es]?\|cf\|xd)\|w(m[dfsz]\|p[dmsz]\|s[cfh])\|xl[tw]\|bat\|ini\|lnk\|nws\|ocx)[\x27\x22\n\r\s]/iR"; **reference:rootcause,721;** classtype:suspicious-filename-detect; sid:721; rev:8;) | O IDS, a partir de uma regra simples e abrangente, é útil para detectar e quantificar a movimentação de vírus e *worms*, como nesta regra que captura extensões maliciosas para anexos de email. |
| Ausência de mecanismos para conter spam | N/A | São necessários mecanismos auxiliares não aplicáveis a IDSs para saber se um *software* está desatualizado. |
| Ausência de mecanismos de autenticação no serviço | alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"SMTP rcpt to command attempt"; flow:to_server,established; content:"rcpt to\|3A\|"; nocase; pcre:"/^rcpt\s+to\:\s*[\|\x3b]/smi"; reference:arachnids,172; reference:bugtraq,1; reference:cve,1999-0095; **reference:rootcause,663;** classtype:attempted-admin; sid:663; rev:14;) | Pode-se programar o IDS para disparar alertas para tentativas de envio de mensagens de correio sem autenticação prévia com esta regra. |
| Configuração inadequada do serviço | alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25 (msg:"SMTP TLS SSLv3 invalid Client_Hello attempt"; flow:to_server,established; flowbits:isset,sslv3.server_hello.request; content:"\|16 03\|"; depth:2; content:"\|01\|"; depth:1; offset:5; reference:cve,2004-0120; reference:nessus,12204; reference:url,www.microsoft.com/technet/security/bulletin/MS04-011.mspx; **reference:rootcause,2544;** classtype:attempted-dos; sid:2544; rev:5;) | Este é um exemplo de regra que demonstra uma tentativa de utilização de recurso do servidor de correio sem que ele estivesse configurado para tal. |
| | | |
| **6. Protocolo de gerenciamento SNMP** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Desatualização do *software* | alert udp $EXTERNAL_NET any -> $HOME_NET 161:162 (msg:"SNMP community string buffer overflow attempt"; content:"\|02 01 00 04 82 01 00\|"; offset:4; reference:bugtraq,4088; reference:bugtraq,4089; reference:bugtraq,4132; reference:cve,2002-0012; reference:cve,2002-0013; reference:url,www.cert.org/advisories/CA-2002-03.html; **reference:rootcause,1409;** classtype:misc-attack; sid:1409; rev:10;) | O IDS é capaz de detectar a presença de comandos que causam sobrecarga de *buffer*, cujo alvos são *softwares* desatualizados. |
| Exposição do acesso a gerencia local de dispositivos | alert udp $EXTERNAL_NET any -> $HOME_NET 162 (msg:"SNMP trap udp"; reference:bugtraq,4088; reference:bugtraq,4089; reference:bugtraq,4132; reference:cve,2002-0012; reference:cve,2002-0013; **reference:rootcause,1419;** classtype:attempted-recon; sid:1419; rev:9;) | O IDS alerta para tentativas de controle de recursos computacionais através de *traps* SNMP. |
| Falta de política associada aos critérios de gerência de dispositivos | N/A | Este Fator de Causa depende de ações humanas, que não podem ser implementadas em um IDS ou outro mecanismo de auxilio à segurança de redes. |
| | | |
| **7. Open Secure Sockets Layer (SSL)** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Desatualização do *software* | alert tcp $EXTERNAL_NET any -> $HOME_NET 443 (msg:"EXPLOIT SSLv2 Client_Hello Challenge Length overflow attempt"; flow:to_server,established; flowbits:isnotset,sslv2.client_hello.request; flowbits:isnotset,sslv3.client_hello.request; | Uma regra entre muitas que alerta para a execução em trânsito de tentativas de sobrecarga de |

| | | |
|---|---|---|
| | flowbits:isnotset,tls1.client_hello.request; byte_test:1,>,127,0; content:"\|01\|"; depth:1; offset:2; byte_test:2,<,768,3; flowbits:set,sslv2.client_hello.request; byte_test:2,>,32,9; **reference:rootcause,2656;** classtype:attempted-admin; sid:2656; rev:6;) | *buffers* em *softwares* dependentes de uma biblioteca OpenSSL desatualizada. |
| | | |
| **8. Serviços de compartilhamento de informações NIS/NFS** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Desatualização do *software* | alert tcp $EXTERNAL_NET any -> $HOME_NET 111 (msg:"RPC portmap proxy integer overflow attempt TCP"; flow:to_server,established; content:"\|00 01 86 A0 00\|"; depth:5; offset:16; content:"\|00 00 00 05\|"; within:4; distance:3; byte_jump:4,4,relative,align; byte_jump:4,4,relative,align; byte_test:4,>,2048,12,relative; content:"\|00 00 00 00\|"; depth:4; offset:8; reference:bugtraq,7123; reference:cve,2003-0028; **reference:rootcause,2093;** classtype:rpc-portmap-decode; sid:2093; rev:5;) | Regra para detecção de tentativa de sobrecarga de *buffer* em implementações do protocolo RPC sob o qual funcionam o NIS e o NFS. |
| Má configuração do serviço | alert udp $EXTERNAL_NET any -> $HOME_NET 111 (msg:"RPC portmap listing UDP 111"; content:"\|00 01 86 A0\|"; depth:4; offset:12; content:"\|00 00 00 04\|"; within:4; distance:4; content:"\|00 00 00 00\|"; depth:4; offset:4; reference:arachnids,428; **reference:rootcause,1280;** classtype:rpc-portmap-decode; sid:1280; rev:9;) | O IDS detecta a presença de servidores RPC com NIS/NFS expostos. |
| | | |
| **9. Bancos de dados** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Falhas na configuração dos bancos de dados | # alert tcp $EXTERNAL_NET any -> $SQL_SERVERS $ORACLE_PORTS (msg:"ORACLE dba_tables access"; flow:to_server,established; content:"dba_tables"; nocase; **reference:rootcause,1687;** classtype:protocol-command-decode; sid:1687; rev:5;) | Existem várias conseqüências em se manter um banco de dados mal configurado. Esta regra é apenas um exemplo deste mau uso dos recursos. |
| Ausência de criptografia no sistema de autenticação | alert tcp $EXTERNAL_NET any -> $SQL_SERVERS $ORACLE_PORTS (msg:"ORACLE login attempt"; flow:to_server,established; content:"scott"; content:"tiger"; nocase; **reference:rootcause,1673;** classtype:system-call-detect; sid:1673; rev:3;) | O IDS pode detectar a passagem de comandos de *login* predeterminados em texto em claro. |
| Falhas na programação do usuário | alert tcp $EXTERNAL_NET any -> $SQL_SERVERS $ORACLE_PORTS (msg:"ORACLE select like '%' attempt"; flow:to_server,established; content:" where "; nocase; content:" like '%'"; nocase; **reference:rootcause,1677;** classtype:protocol-command-decode; sid:1677; rev:5;) | Erros de programação de comandos SQLs podem ser detectados antes que estes sejam utilizados por invasores para obtenção de informações. |
| Ausência de políticas de manutenção de SGBDs | N/A | Este Fator de Causa depende de ações humanas, que não podem ser implementadas em um IDS ou outro mecanismo de auxilio à segurança de redes. |
| | | |
| **10. Kernel** | | |
| **Fator de Causa** | **Regra** | **Comentário** |
| Desatualização do *kernel* | alert ip $EXTERNAL_NET $SHELLCODE_PORTS -> $HOME_NET any (msg:"SHELLCODE Linux shellcode"; content:"\|90 90 90 E8 C0 FF FF FF\|/bin/sh"; reference:arachnids,343; **reference:rootcause,652;** classtype:shellcode-detect; sid:652; rev:9;) | O IDS abriga várias regras que alertam para a ocorrência de tentativas de exploração de falhas em versões desatualizadas do *kernel*. |
| Má configuração do *kernel* | N/A | Este Fator de Causa depende de ações humanas, que não |

| | | |
|---|---|---|
| | | podem ser implementadas em um IDS ou outro mecanismo de auxilio a segurança de redes. |

**Tabela 16. Mapa de relação entre Fatores de Causa e regras do IDS Snort.**

É possível constatar que nem todo Fator de Causa tem uma regra correspondente. Em primeira análise, pode-se sugerir que então para estes casos a ação a ser tomada seria a criação de novas regras de forma a contemplá-los. Entretanto, estudando os comentários da tabela de resultados ou ainda, analisando-se a natureza do Fator de Causa em questão, percebe-se que simplesmente não se aplica ao IDS baseado em rede tratar/identificar uma vulnerabilidade sob a qual é impossível se obter acesso, como a falta de procedimentos para trocas de senhas em um sistema de autenticação, por exemplo.

A Tabela 17 fornece os valores exatos em que foi possível ao IDS contemplar o Fator de Causa levantado.

| Resumo quantitativo | |
|---|---|
| Itens vulneráveis | 10 |
| Fatores de causa | 33 |
| Regras correspondentes no IDS | 22 |
| Índice de sucesso para o IDS | 66,6% |

**Tabela 17. Resumo quantitativo da aplicação**

Portanto, para dois terços dos casos estudados, já existe uma regra equivalente na base do Snort. Ela apenas precisou receber um novo campo de referência que informe os procedimentos necessários para a remediação do problema como proposto na RCA.

# 8 Conclusões e trabalhos futuros

A segurança de informações é uma área de pesquisa que desempenha papel vital para viabilizar a utilização de computadores e redes como meio de armazenamento, processamento e transmissão de informações. Os sistemas de detecção de intrusões, enquanto mecanismo integrante do cenário de ferramentas para proteção de informações, já conquistou seu lugar em meio a *firewalls*, VPNs, autenticadores, *honeypots* e outros no auxílio a detecção de intrusões e no cumprimento dos requisitos modernos de segurança.

A análise de causa raiz, metodologia existente indiferente aos anseios deste trabalho, é um processo maduro de correção de falhas dos ambientes modernos de produção, seja nos setores químico-industriais, petrolíferos ou aeroespaciais. Neste texto, sua utilidade foi discutida como abordagem para resolução de problemas nos ambientes computacionais e mostrou-se igualmente útil.

Também, a análise das vulnerabilidades mais críticas para os ambientes UNIX comprova a aplicabilidade da metodologia ao propor que um conjunto pequeno de boas práticas como a atualização permanente dos *softwares* servidores ou a adoção de tarefas diárias de análise de trilhas de auditoria pode, de forma eficaz, baixar drasticamente o número de incidentes na rede de trabalho.

A passagem do modelo teórico discutido nos capítulos 4 a 6 para a implementação prática, no Snort, no Capítulo 7, foi relativamente simples. Tal simplicidade se deu por alguns motivos importantes, como a facilidade com que se pode intervir em *softwares* livres em geral e no Snort e seu console ACID em específico, como também a aderência da teoria proposta na metodologia de RCA à prática de detecção desejada. Como resultado, tem-se um subconjunto de alertas do Snort modificado para conter análises de causa raiz e, como conseqüência, administradores de rede mais bem informados para agir corretamente na iminência de ataques.

Uma conclusão que emergiu durante a fase final de desenvolvimento do trabalho, mais precisamente quando do mapeamento das vulnerabilidades mais críticas para o ambiente UNIX em regras do Snort, foi que o IDS estudado não é mecanismo suficiente para cobrir todas as vulnerabilidades levantadas. Casos como o número três, "Autenticação

do UNIX", tratam vulnerabilidades locais do computador com este sistema operacional instalado. O IDS baseado em rede nunca tomará conhecimento desses problemas e, portanto, não irá tratá-los. Para este e os demais casos, outros mecanismos se fazem necessários, como programas de auditoria local ou ainda, um IDS baseado em *host*. Esta situação, demonstra que cada mecanismo tem uma finalidade específica e um escopo de atuação predeterminado e limitado.

Entre as possibilidades de trabalhos futuros, foi possível detectar a necessidade de se formular mais análises, através dos passos sugeridos nos Capítulos 4 e 5, para outros ambientes, como por exemplo, os que utilizam sistema operacional Windows e demais *softwares* de rede da Microsoft. Também, devido a excessiva quantidade de alarmes apresentados ao administrador da rede, fica evidente a necessidade do estudo de técnicas que venham a contribuir para a aglomeração de alarmes, onde o desafio é encontrar um balanço entre o excesso e a escassez (de detalhes úteis) sobre as informações apresentadas.

Outra proposta é a confecção de mapas de análise de causa raiz apenas para concepção teórica de sistemas operacionais e *softwares* de rede, com o objetivo de guiar os desenvolvedores de soluções na confecção de futuros produtos de mercado.

Entrar em níveis mais profundos de análise de causa raiz tanto no Snort, como em qualquer outra implementação de IDS certamente também é assunto para pesquisas. Existem outras possibilidades de aplicação da metodologia de RCA que vão além das apresentadas neste trabalho, como por exemplo, a implantação de mecanismos pró-ativos, como a modificação de ambiente mediante um determinado fator de causa ou através de retro alimentação visando medição estatística da funcionalidade da abordagem em relação à realidade implementada.

Espera-se que a adoção da metodologia neste texto proposta e implementada, venha a ser modificada e amadurecida em vários ambientes de produção, como parte de um esforço contínuo visando garantir o posicionamento do administrador da rede sempre um passo a frente das ameaças.

# Referências Bibliográficas


ADUSKEVICZ, P. J. *et al*. **Procedural outage reduction: addressing the human part**. Disponível em [http://citeseer.ist.psu.edu/aduskevicz99procedural.html]. Acesso em 02 de abril de 2004.

AL-TAWIL, K., AL-KALTHAM, I. A. Evaluation and testing of internet firewalls. **International Journal of Network Management**. v. 9, e. 3, p. 135-149, 1999, ISSN:1099-1190.

ALMGREN, M., DEBAR, H., DACIER,M. A lightweight tool for detecting web server attacks. **Network and Distributed System Security Symposium**, p. 157–170, 2000.

AMMERMAN, M. **The root cause analysis handbook: a simplified approach to identifying, correcting, and reporting workplace errors**. Productivity Inc. 1998. 144p. ISBN 0527763268.

ANDERSON, J. P. Computer security technology planning study. **ESD-TR-73-51 ESD/AFSC**, Bedford, MA, 1972.

BACE, R., MELL, P. Intrusion detection systems. **NIST special publication on intrusion detection systems**. Disponível em [http://csrc.nist.gov/publications/ nistpubs/800-31/sp800-31.pdf]. Acesso em 4 de setembro de 2004.

BARBARÁ, D., COUTO, J., LIN, J. Bootstrapping a data mining intrusion detection system. **Proceedings of the 2003 ACM symposium on applied computing**, Melbourne, FL, p. 178-200, 2003.

BELLOVIN, S. M. Packets found on an Internet. **Computer communications review**, New York, NY, v. 23, e. 3, p. 26–31, 1993. ISSN 0146-4833.

BERNARDES, M. C., MOREIRA, E. S. Implementation of an intrusion detection system based on mobile agents. **International symposium on software engineering for parallel and distributed systems (PDSE 2000)**. Limerick, Ireland. p. 158, 2000.

BOTHA, M., ROSSOUW, V. S., PERRY, K., LOUBSER, E., YAMOYANY, G. The utilization of artificial intelligence in a hybrid intrusion detection system. **Proceedings of the 2002 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology.** South Africa, p. 149-155, 2002. ISSN: 1-58113-596-3.

CAMPELLO, R. S., WEBER, R. F. Sistemas de detecção de intrusão. In: MACEDO, R. J. A., FARINES, J. **Livro texto dos minicursos**. 19º simpósio brasileiro de redes de computadores. Florianópolis, p. 1-40, 2001.

CERT. Computer emergency and response team: **Cert/CC statistics: number of incidents reported**. Disponível em [http://www.cert.org/stats/cert_stats.html]. Acesso em 1 de maio de 2004.

CHARI, S. N., CHENG, P. Bluebox: a policy-driven, host-based intrusion detection system. **ACM transactions on information and system security (TISSEC)**, New York, NY, v. 6, n. 2, p. 178-200, 2003.

CUPPENS, F. Managing alerts in a multi-intrusion detection environment. **17th Annual Computer Security Applications Conference** (ACSAC), p. 22–31, 2001.

CUPPENS, F. MIÈGE, A. Alert correlation in a cooperative intrusion detection framework. **IEEE Symposium on Security and Privacy**, Oakland, CA, 2002.

DEBAR, H., DACIER, M., NASSEHI, M., AND WESPI, A. Fixed vs. variable-length patterns for detecting suspicious process behavior. **Research report RZ3012**, IBM Research Division, Zurich Research Lab. 1998.

DEBAR, H., WESPI, A. Aggregation and correlation of intrusion-detection alerts. **4th Workshop on Recent Advances in Intrusion Detection** (RAID). Springer-Verlag, Berlin, p. 85–103. 2001.

DEW, J. R. **Digging deeper for root causes.** Disponível em [http://bama.ua.edu/~st497 /pdf/diggingdeeper.pdf]. Acesso em 10 de julho de 2004.

ENTERASYS. **Enterasys intrusion prevention**. Disponível em [http://www.enterasys. com/products/ids/]. Acesso em 1 de novembro de 2004.

HARARI, E. Post-installation security procedures. **Linux Journal.** New York, NY, v. 1999, n. 68, 1999.

ILUNG, K. USTAT: A real-time intrusion detection system for UNIX. **IEEE Symposium on security and privacy**, Oakland, CA, 16–28, 1993.

ISO. ISO/IEC 17799:2000 **Code of practice for information security management**. 2000. 81p.

ISRAEL, J., LINDEN, T. A. Authentication in office system internetworks. **ACM Transactions on Information Systems (TOIS)**. v. 1, e. 3, p. 193-210, 1983, ISSN:1046-8188.

JACKSON, K. A. Intrusion detection system (IDS) product review. **IBM internal confidential document**, IBM Research Division, Zurich Research Lab. 1999.

JAIN, A., HONG, L., PANKANTI, S. Biometric identification. **Communications of the ACM**. v. 43, e. 2, p. 90-98, 2000, ISSN:0001-0782.

JULISCH, K. Clustering intrusion detection alarms to support root cause analysis. **ACM Transactions on Information and System Security**, New York, NY, v. 6, n. 4, p. 443-471, 2003.

LANDWEHR, C. E. Computer Security. **International Journal of Information Security**. Springer-verlag. v. 1, p. 1-13, 2001.

LEE, W., STOLFO, S. J. A framework for constructing features and models for intrusion detection systems. **ACM transactions on information and system security (TISSEC)**, New York, NY, v. 6, n. 4, p. 227-261, 2000.

MANGANARIS, S., CHRISTENSEN, M., ZERKLE, D., HERMIZ, K. A data mining analysis of RTID alarms. **Computer Networks.** Elsevier. v. 34, n. 4, p. 571–577. 2000.

MOFFETT, J. D. Security & Distributed Systems. **Encyclopaedia of microcomputers**. Marcel Dekker Inc, New York, NY, v. 15, 1995.

MUKKAMALA, S., SUNG, A. H. A Comparative Study of Techniques for Intrusion Detection. **15th IEEE international conference on tools with artificial intelligence (ICTAI'03)**. New mexico tech. Sacramento, CA. p. 540. 2003.

NASA. National Aeronautics and Space Administration. **Root cause analysis overview**. Julho de 2003. Disponível em [http://www.hq.nasa.gov/office/codeq/rca/rootcauseppt.pdf]. Acesso em 11 de julho de 2004.

NBSO. Brazilian computer emergency response team: **Estatísticas dos incidentes reportados ao nbso**. Disponível em [http://www.nbso.nic.br/stats/incidentes/]. Acesso em 2 de abril de 2004.

NING, P., JAJODIA, S., WANG, X. Abstraction-based intrusion detection in distributed environments. **ACM Transactions on Information and System Security**. New York, NY, v. 4, n. 4, p. 407–452. 2001.

PARADIES, M., SKOMPSKI, E. **Why people don´t develop effective corrective actions**. Disponível em [http://www.taproot.com/pages/papers/correctiveAct.pdf]. Acesso em 7 de setembro de 2004.

PAXSON, V. Bro: A system for detecting network intruders in real-time. Computer Networks, Elsevier, e. 31, v. 23/24, p. 2435–2463. 1999.

PTACEK, T. H., NEWSHAM, T. N. Insertion, evasion, and denial of service: Eluding network intrusion detection. **Tech. Republic: Secure Networks.** Disponível em [http://www.snort.org/docs/idspaper/]. Acesso em 19 de setembro de 2004.

REHMAN, R. **Intrusion detection systems with snort. Advanced techniques using snort, apache, mysql, php and acid.** 1.ed. New Jersey: Prentice Hall. 275p. 2003.

SAMAR, V. Unified login with pluggable authentication modules (PAM). **Conference on computer and communications security - proceedings of the 3rd acm conference on computer and communications security**. New Delhi, India, p. 1-10, 1996, ISBN:0-89791-829-0.

SANS. **The twenty most critical internet security vulnerabilities**: The experts concensus. Disponível em [http://www.sans.org/top20/]. Acesso em 01 de outubro de 2004.

SANDHU, R., SAMARATI, P. Authentication, access control, and audit. **ACM Computing Surveys (CSUR)**. v. 28, n. 1, p. 241-243, 1996, ISSN:0360-0300.

SEKAR, R., GUANG, Y., VERMA, S., SHANBHAG, T. A high-performance network intrusion detection system. **6th ACM Conference on Computer and Communications Security**, p. 8–17, 1999.

SILVER, J. Firewalls are the net´s first, but not only, line of defense. **Government computer news**. v. 11, p. 44, 1995.

SNORT. **Snort the open source network intrusion detection system**. Disponível em [http://www.snort.org/]. Acesso em 1 de novembro de 2004.

SOARES, L. F. G., LEMOS, G., COLCHER, S. **Redes de computadores**: das LANs, MANs e WANs às redes ATM. Rio de Janeiro: Campus, 1995. 705p. ISBN: 8-57001-998-X.

SOURCEFIRE. **Sourcefire intrusion sensor**. Disponível em [http://www.sourcefire.com/products/is.html]. Acesso em 1 de novembro de 2004.

SHELFER, K. M., PROCACCINO, J. D. Smart card evolution. **Communications of the acm**. v. 45, n. 7, p. 83-88, 2002, ISSN:0001-0782.

VALDES, A., SKINNER, K. Probabilistic alert correlation. **4th Workshop on Recent Advances in Intrusion Detection** (RAID). Springer-Verlag, Berlin, p. 54–68, 2001.

VIGNA, G., VALEUR, F., KEMMERER, R. A. Designing and implementing a family of intrusion detection systems. **Proceedings of the 9th european software engineering conference held jointly with 10th ACM SIGSOFT international symposium on Foundations of software engineering**. p. 88-97, 2003, ISBN:1-58113-743-5.

ZWICKY, E. D., COOPER, S., CHAPMAN, D. B. **Building internet firewalls**. 2.ed. Sebastpol: O´Reilly, 2000. 869p. ISBN 1-56592-871-7.