ANGELITA DE CÁSSIA CORRÊA

Metodologia para análise comparativa de Sistemas de Detecção de Intrusão

Dissertação apresentada ao Instituto de Pesquisas Tecnológicas do Estado de São Paulo - IPT para obtenção do título de Mestre em Engenharia de Computação.
Área de concentração: Redes de Computadores.

Orientador: Prof. Dr. Geraldo Lino de Campos

São Paulo
Setembro 2005

Livros Grátis

http://www.livrosgratis.com.br

Milhares de livros grátis para download.

Ficha Catalográfica
Elaborada pelo Centro de Informação Tecnológica do
Instituto de Pesquisas Tecnológicas do Estado de São Paulo - IPT

C824m Corrêa, Angelita de Cássia
Metodologia para análise comparativa de sistemas de detecção de intrusão. / Angelita de Cássia Corrêa. São Paulo, 2005.
100p.

Dissertação (Mestrado em Engenharia da Computação) - Instituto de Pesquisas Tecnológicas do Estado de São Paulo. Área de concentração: Redes de Computadores.

Orientador: Prof. Dr. Geraldo Lino de Campos

1. Ataque (computador) 2. Sistema de detecção de intrusão (SDIS) 3. Sistema de prevenção de intrusão 4. SNORT 5. BRO 6. Tese I. Instituto de Pesquisas Tecnológicas do Estado de São Paulo. Centro de Aperfeiçoamento Tecnológico II.

Título

05/68 CDU 004.491(043)

Dedicatória

Dedico esse trabalho a Deus e aos meus pais que além de serem meus melhores amigos, são as pessoas que eu mais amo.

Agradecimentos

Agradeço a meus pais, que deram início à minha vida e me educaram com base em sólidos princípios morais, carinho e muito amor.

A eles e ao meu irmão, agradeço pelo incentivo e compreensão nas minhas ausências durante a realização desse trabalho.

Ao meu orientador, Prof. Dr. Geraldo Lino de Campos, pela sua dedicação, orientação, pelos conhecimentos repassados e pela confiança depositada no meu trabalho de dissertação.

À banca examinadora, Dr. Mario Yoshikazu Miyake e Dra. Graça Bressan, por aceitarem o convite para participar das bancas de qualificação e defesa final.

Ao IPT e a todos os meus professores do Mestrado, pela contribuição que forneceram ao meu crescimento pessoal, acadêmico e profissional. A todos os funcionários do IPT, e em especial, Adilson, Andréia e Ester pela colaboração e atenção recebida durante todo o Mestrado.

Ao Universo Online, especialmente a Armando Lima Amaral, agradeço a oportunidade de realizar o estudo de caso, contribuindo para meu crescimento pessoal e profissional.

À Wanda, pelo incentivo e motivação, e à Lazinha, pelo apoio e orações.

Agradeço aos meus grandes amigos Ana Hummel Fernandes, Gilberto Fernandes e Luis Laurito, que estiveram presentes em grande parte do trabalho, sempre me apoiando e torcendo para que eu pudesse concluir essa etapa; à amiga Elizabeth Rogatto que, mesmo morando nos Estados Unidos, ajudou-me muito na fase final deste trabalho, obrigada pela consideração e pela amizade e à Elizabeth de Menezes Barbosa, colega de trabalho e aluna do IPT, obrigada pela torcida e apoio.

A minha prima Andressa Begalli e à amiga Adriane Evelise Rezzaghi que, apesar da distância, vibraram e torceram por mim diariamente através da Internet.

E a todos os outros amigos, colegas, primos e tios que, direta ou indiretamente, contribuíram para a realização deste trabalho.

Um agradecimento especial a Deus que está presente em todos os momentos da minha vida.

RESUMO

Este trabalho propõe uma metodologia para avaliar SDIs, Sistemas de Detecção de Intrusão, desenvolvida para ser aplicada de modo prático no ambiente empresarial, apresentando uma seqüência de procedimentos que podem ser executados em um curto período de tempo.

Inicialmente são descritos os principais ataques a que a rede está sujeita. A seguir, é apresentada uma proposta de metodologia que consiste em cinco etapas: seleção de tipos de ataques, seleção de ferramentas de ataque, especificação de um modelo para o teste, seleção dos SDIs e análise dos SDIs.

A metodologia proposta foi aplicada a um estudo de caso para sua validação. Os SDIs escolhidos para avaliação foram Snort e Bro, e seus resultados foram analisados e comparados.

Essa metodologia mostrou-se eficaz e apta para ser utilizada em comparações de SDIs.

Palavras-chave: SDI, Snort, Bro, segurança, ataques, metodologia, estudo de caso.

ABSTRACT

Methodology for comparative analysis of Intrusion Detection Systems.

This dissertation's purpose is a methodology to evaluate IDSs, Intrusion Detection Systems, and it was developed to apply in a simple way in a corporate environment, presenting a sequence of procedures, which may be executed in a short time period.

At first, the main attacks that a network is vulnerable are described. Then, a methodology purpose is shown, which consists in five steps: kind of attacks selection, attack tools selection, specification of a test model, IDSs selection and IDSs analysis.

The methodology purpose was applied in a case study to be validated. The IDSs chosen to the evaluation were Snort and Bro and the results were analyzed and compared.

This methodology showed itself efficient and able to be used in comparisons of IDSs.

Key words: IDS, Snort, Bro, security, attacks, methodology, case study.

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

ACK	Acknowledgement
ACM	Association for Computing Machinery
CAIS	Centro de Atendimento a Incidentes de Segurança
CERT.br	Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil
CERT/CC	Computer Emergency Response Team / Coordination Center
CGI	Common Gateway Interface
CPU	Central Processing Unit
DNS	Domain Name Server
FIN	Finish
FTP	File Transfer Protocol
IDS	Intrusion Detection System
IEEE	The Institute of Electrical and Electronics Engineers
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IPS	Intrusion Prevention System
NBSO	NIC BR Security Office
NIST	National Institute of Standards and Technology
NMAP	Network Mapper
NTP	Network Time Protocol
PSH	Push
RST	Reset
RFC	Requests for Comments
RNP	Rede Nacional de Ensino e Pesquisa
SANS	SysAdmin, Audit, Network, Security
SDI	Sistema de Detecção de Intrusão
SSH	Secure Shell
SQL	Structured Query Language
SYN	Synchronize
SNS	Symantec Network Security
SPI	Sistema de Prevenção de Intrusão
TCP	Transmission Control Protocol

UDP	User Datagrama Protocol
URG	Urgent
XMAS	Xmas Tree

Sumário

1 INTRODUÇÃO

1.1 Motivação

A preocupação com segurança vem crescendo muito atualmente. Com o aumento da conectividade entre as redes de computadores, os sistemas têm-se tornado mais vulneráveis a ataques.

Segundo CHAMBERS; DOLSKE; IYER (2005), o protocolo TCP/IP - base para Internet nos dias de hoje -, falha na falta dos mais básicos mecanismos de segurança, como autenticação e encriptação.

Por outro lado, o conhecimento, as ferramentas e as técnicas disponíveis para os agressores têm crescido com rapidez. Infelizmente, as técnicas de defesa não têm crescido tão rapidamente. As tecnologias atuais vêm atingindo seus limites, sendo necessárias soluções inovadoras para lidar com o nível das ameaças atuais e futuras.

Um dos principais objetivos desses ataques é executar operações através da destruição de mecanismos de segurança dos sistemas, que deveriam impedir o acesso dos intrusos. Essas operações podem incluir o acesso às informações confidenciais, à execução de danos ao sistema ou simplesmente causar a sua indisponibilidade.

Desse modo, é extremamente importante que os mecanismos de segurança de um sistema estejam configurados de forma a prevenir acessos não autorizados aos seus recursos. No entanto, é difícil prevenir completamente a existência de falhas na segurança. Os Sistemas de Detecção de Intrusão - SDIs (*Intrusion Detection System – IDS*) - são ferramentas utilizadas para detecção dessas tentativas de intrusão. Atualmente, essas ferramentas evoluíram e pode-se identificar um ataque, externo ou interno, antes que ele afete os sistemas da empresa, podendo-se utilizar, para esse fim, o Sistema de Prevenção de Intrusão - SPI (*Intrusion Prevention System – IPS*).

O SPI funciona de forma pró-ativa no combate às ameaças, sendo designado para proteger organizações contra ameaças internas e externas. Assim como o SDI, também é designado para monitorar o tráfego de rede. Possui a capacidade de tomar ações imediatas, baseadas nas configurações de regras estabelecidas pelo administrador. Essa função tem como

vantagem agir antes de a rede ser invadida, podendo detectar um pacote malicioso e bloquear todo tráfego vindo do IP invasor ou porta.

Para gerenciar esses riscos, é necessário conhecê-los antes de tudo. Cada empresa possui diferentes características em relação à proteção de seus dados, por isso não se pode afirmar que apenas um *firewall*, ou um SDI, ou certificados digitais, criptografia, ou uma boa política de segurança possam ser suficientes para a segurança da entidade. Cada uma dessas ferramentas tem sua função essencial e todas devem ser usadas de maneira integrada.

1.2 Sistema de Detecção de Intrusão

O SDI tem como principal função detectar atividades anormais ou acessos de usuários não autorizados relativos a um computador ou a uma rede. “Esta ferramenta roda constantemente em *background* e somente gera uma notificação quando detecta alguma coisa que seja suspeita ou ilegal” (NED, 1999).

Os SDIs baseados em rede capturam e analisam pacotes de rede, realizando a busca por ataques direcionados a determinados serviços e estações existentes nesse ambiente. Através desse tipo de SDI, as informações que trafegam em um *backbone* e nos diversos segmentos de rede de uma organização podem ser monitoradas sem interferir no desempenho da rede ou das estações conectadas à mesma. Já nos SDIs baseados em *host*, as atividades são monitoradas através de dados coletados nas próprias estações, possibilitando detectar atividades não autorizadas que estejam sendo realizadas por usuários dessa estação.

As tentativas de intrusos podem causar acesso à informação, sua manipulação ou podem tornar o sistema não confiável e indisponível. Se houver ataque em um sistema, é necessário detectá-lo o mais rápido possível e tomar ações apropriadas, sendo essa uma das funções do SDI.

Para SUNDARAM (1996), um SDI geralmente não toma medidas preventivas quando um ataque é detectado, ele é reativo ao invés de agente pró-ativo.

Um SDI pode fornecer alerta indicando que o sistema está sob ataque, mesmo que o sistema não seja vulnerável a um ataque específico. Esses alertas podem ajudar o usuário a alterar sua postura de defesa nas instalações, aumentando a resistência aos ataques. Além

disso, um SDI pode servir para confirmar a configuração e o funcionamento de outros mecanismos de segurança, como os *firewalls.* (JOHN, CHRISTIE e ALLEN (2000)

Para MUKHERJEE, HEBERLEIN e LEVITT (1994), o SDI é um sistema que tenta identificar tentativas de intrusão de usuários externos e internos não autorizados que abusam de seus privilégios. Por exemplo: uma intrusão pode ocorrer quando um funcionário busca por informações confidencias; um administrador de sistemas modifica arquivos de sistema para permitir que um usuário não autorizado o acesse; um intruso acessa ou modifica arquivos de sistema; um intruso modifica tabelas de roteamento; ou um intruso instala um *trojan* em um computador para examinar dados sensíveis - como senhas de usuários - contidos num tráfego de rede.

Um SDI não pode ser usado como a única fonte de segurança de uma rede, nem em substituição a um *firewall*, mas sim, em conjunto com outros métodos para aumentar a segurança da rede.

1.3 Intrusos

Os intrusos podem ser classificados de duas formas:

- Intrusos Externos - ataques originados fora da empresa, geralmente da Internet.
- Intrusos Internos - ataques originados dentro da própria empresa. Intrusos internos são pessoas com grande conhecimento da topologia da rede e da localização dos dados importantes da empresa. Por isso deve-se ter uma atenção maior com eles do que com os intrusos externos.

1.4 Objetivo

O objetivo deste trabalho é estabelecer uma metodologia para análise de ferramentas de detecção de intrusão, em função das peculiaridades de cada organização. Serão estudados Sistemas de Detecção de Intrusão para identificar suas características, vantagens e desvantagens.

A metodologia será avaliada através de um estudo de caso realizado em um provedor brasileiro de Internet de grande porte.

1.5 Metodologia de trabalho

Inicialmente, foram pesquisados os ataques típicos e as medidas de segurança normalmente adotadas. Adicionalmente, acompanhou-se a evolução dos ataques.

Foi feita uma pesquisa bibliográfica sobre técnicas de SDI e de metodologias para sua implementação.

A seguir, propõe-se uma metodologia de utilização prática que possa adaptar-se à realidade das empresas e do mercado.

A viabilidade da metodologia proposta será avaliada através de um estudo de caso em situação real. Refinamentos serão incorporados se os resultados assim o permitirem.

A metodologia proposta fornecerá subsídios para uma decisão superior.

1.6 Organização do trabalho

O Capítulo 2 apresenta os principais conceitos que serão abordados durante o trabalho.

O Capítulo 3 apresenta os ataques típicos, vulnerabilidades, e suas perspectivas, mostrando uma visão geral das intrusões de redes de computadores e seus sistemas.

O Capítulo 4 aborda a ferramenta de defesa SDI, descrevendo seus conceitos, funções e características importantes.

O Capítulo 5 descreve a metodologia proposta para avaliação de SDIs.

O estudo de caso será apresentado no Capítulo 6, em que a metodologia criada no capítulo anterior será aplicada e os resultados obtidos na fase de teste serão apresentados.

O Capítulo 7 apresenta a conclusão.

2 ATAQUES TÍPICOS

Esse capítulo apresenta os principais tipos de ataques e vulnerabilidades encontrados atualmente.

Os ataques começam por vulnerabilidades encontradas no sistema por invasores. Essas vulnerabilidades podem ser encontradas através de ferramentas disponíveis publicamente na Internet ou desenvolvidas pelo próprio atacante.

Existe diferença entre o que é um ataque e o que é uma intrusão.

Um ataque é uma tentativa de violar a segurança de um computador ou rede. Já uma intrusão é o ataque bem sucedido, através de um acesso não autorizado a um sistema.

2.1 Ataques

2.1.1 Varredura de portas *(Port scan)*

É uma das técnicas mais populares utilizadas para descobrir e mapear serviços que estejam em estado de escuta em uma porta específica. Usando esse método, um atacante pode criar uma lista com as vulnerabilidades de uma porta aberta de um *host* ou servidor alvo para tentar a invasão.

Uma varredura de portas consiste em enviar uma mensagem para cada porta. O tipo de resposta recebida indica se a porta pode ser utilizada ou não, para que um ataque possa ser realizado. Consiste basicamente em uma coleta de dados, cujo objetivo é reunir o maior número possível de informações sobre a rede ou o servidor. O resultado de uma varredura em uma porta, geralmente se encaixa em uma destas três categorias:

- Aberta - O *host* envia uma resposta indicando que o serviço está no estado de escuta na porta;
- Fechada - O *host* envia uma resposta indicando que as conexões serão negadas para essa porta; e
- Rejeitada ou Bloqueada - Não há nenhuma resposta do *host*.

Os principais ataques são:

2.1.1.1 Varredura de portas abertas (*Open Scan*)

- **TCP *Connect***

É uma das formas mais simples de exploração do TCP. Praticamente todos os varredores de portas utilizam esse recurso. É considerado um *handshake* para cada porta definida na varredura.

Essa chamada de sistema é conhecida como *connect()*, sendo fornecida pelo sistema operacional e usada para estabelecer conexão com a porta definida do *host* alvo. Caso a porta esteja no estado de escuta, o *connect()* irá estabelecer uma conexão; caso contrário, o usuário receberá uma mensagem informando que a porta não está aberta, respondendo, normalmente, a pilha TCP/IP com um datagrama de cancelamento (TCP RST).

Uma vantagem dessa técnica é que não há necessidade de nenhum privilégio especial. Qualquer usuário de sistemas *Unix* está livre para usar essa chamada.

Esse tipo de varredura é facilmente detectado por análise de logs do *host* alvo, que mostrará o grupo de conexões que falhou pelo fato de a porta não estar em estado de conexão.

- **Varredura UDP**

Essa técnica determina quais portas UDP estão abertas em um *host.* A técnica implica em enviar 0 bytes de dados para cada porta do *host* alvo. Caso a resposta seja uma mensagem com ICMP tipo 3 (*host* não alcançável), então a porta está fechada. Caso contrário, supõe-se que está aberta.

2.1.1.2 Varredura de portas semi-abertas (*Half scan)*

- **TCP SYN**

É outra técnica bastante utilizada, também conhecida como “conexão semi-aberta”.

Esse tipo de varredura difere do TCP *connect* por não estabelecer uma conexão TCP completa. O processo inicia-se quando um datagrama TCP SYN (*Synchronize*) é enviado como se estivesse abrindo uma conexão real, aguardando uma resposta. Se o *host* alvo responder com TCP SYN/ACK, isso indica que a porta está no estado de escuta; já uma resposta com RST indica que a porta está fechada.

Caso seja recebido um TCP SYN/ACK, um TCP RST será imediatamente enviado para encerrar a conexão. O processo do servidor nunca é informado pela camada TCP porque a conexão não é completada. Essa técnica exige privilégios de *root*.

A principal vantagem desta técnica é que poucos sites irão registrá-la no arquivo de log.

2.1.1.3 Varredura de portas ocultas (*Stealth Scan*)

- **Varredura FIN**

Esta técnica consiste em enviar um pacote com a *flag* FIN (*Finish*) habilitada para uma determinada porta. Segundo a RFC 793, as portas que estiverem fechadas respondem com pacote TCP RST, enquanto as abertas devem ignorar o pacote em questão.

- **Varredura *Null***

Esta técnica consiste em enviar um pacote TCP com todas as *flags* desabilitadas para uma determinada porta do *host* alvo, sendo que as portas fechadas devem responder com TCP RST, enquanto as demais devem ignorar o pacote em questão.

- **Varredura *Xmas***

Ao contrário da técnica de varredura *Null,* a *Xmas* envia um pacote para cada porta do *host* alvo, com as *flags* URG (*Urgent*), PSH (*Push*) e FIN (*Finish*) habilitadas. É usada para identificar portas em estado de escuta e manipula essas *flags* do cabeçalho TCP. Se a porta do *host* alvo estiver fechada, receberá TCP RST como resposta; e se estiver aberta, a porta ignorará os pacotes.

2.1.1.4 Varredura de Protocolo IP

Este método determina quais protocolos IP estão sendo suportados pelo *host.* A técnica consiste em enviar pacotes IP sem nenhum cabeçalho adicional do protocolo para cada protocolo específico na máquina alvo; caso a resposta seja um ICMP tipo 3 (não alcançável), o protocolo não está em uso. Caso contrário, supõe-se que o protocolo esteja em uso.

2.1.1.5 Varredura de Fragmentação

Essa técnica utiliza várias outras técnicas de varredura de portas tais como varredura SYN, FIN, *Xmas* e *Null*. Os pacotes enviados ao *host* alvo são fragmentados, ou seja, o cabeçalho *TCP* é dividido em vários fragmentos. Com isso, os SDIs que não possuem mecanismos eficientes de remontagem de pacotes, não conseguem identificar essa forma de ataque, pois os diversos datagramas enviados individualmente não correspondem a uma ameaça. Alguns filtros de pacotes ou *firewalls* enfileiram todos os fragmentos de IP, mas muitas redes não têm recursos para suportar a perda de desempenho causada pela fila.

2.1.1.6 Outras varredura de portas

- **Varredura ACK**

Este método é geralmente usado para mapear o conjunto de regras de um *firewall*. Envia pacotes com TCP ACK para uma porta específica. Retornando um TCP RST, a porta é classificada como “não filtrada”; se não receber resposta ou se um ICMP tipo 3 (não alcançável) retornar, a porta é classificada como "filtrada". Essa varredura não mostra portas abertas.

- **Varredura *Window***

Essa técnica é semelhante à varredura ACK, pois é realizada com um pacote TCP ACK. Tem como objetivo conseguir detectar portas protegidas por *firewall*, e não portas abertas.

Quando um pacote é rejeitado e o *firewall* retorna a mensagem com ICMP tipo 3, pode-se determinar que a porta esteja filtrada. Quando retorna com TCP RST, pode-se interpretar que não existe *firewall*, ou seja, a porta não está filtrada.

Isso ocorre devido a uma anomalia no tamanho da janela TCP existente em diversos sistemas operacionais. Alguns dos sistemas que apresentam essa vulnerabilidade são: Windows, AIX, FreeBSD, SunOS, *Unix*, OS/2, etc.

- **Varredura ICMP**

Através dessa técnica, é possível determinar quais *host*s estão ativos na rede. Para isso, enviam-se pacotes ICMP do tipo 8 (ICMP *request*) nas redes especificadas e aguarda-se pelas respostas - os *hosts* que responderem estão ativos.

Alguns sites, como *microsoft.com*, bloqueiam pacotes do tipo ICMP *request*. Dessa forma, podem-se enviar também pacotes TCP ACK na porta 80. Se o RST retornar, significa que o *host* está ativo. Uma terceira técnica seria enviar um pacote SYN e esperar por um RST ou SYN/ACK.

- **Varredura *Decoy***

Essa técnica é utilizada para enganar ferramentas SDI. Utiliza endereços IPs e portas forjadas *(spoofing)* para ocultar a origem do ataque.

2.1.2 *Buffer overflow*

Ocorre quando um programa ou processo tenta armazenar mais dados que um *buffer* pode suportar. Quando ocorre o estouro de memória, os dados podem ser gravados em outros lugares da memória, e esses dados podem ser programas que permitam ao invasor consolidar o ataque.

Em um ataque *buffer overflow*, segundo SIMON (2001), o objetivo do atacante é usar a vulnerabilidade existente para corromper informação a fim de executar o ataque previamente planejado por ele; e se isso realmente acontecer, o atacante terá o controle do programa. Tipicamente, o código do ataque gera um *shell* que permite ao atacante a execução de comandos no sistema.

2.1.3 Negação de Serviço (*Denial of Service - DoS*)

Seu objetivo é esgotar os recursos de um serviço ou rede, tornando essa rede inacessível ou com respostas muito lentas. Os ataques de negação de serviços são, normalmente executados usando ferramentas que enviam diversas requisições a um determinado servidor, sobrecarregando os recursos do mesmo ou tornando o sistema inoperável.

Na grande maioria desses ataques, o endereço de origem é forjado (*spoofing*), dificultando, portanto, o processo de auditoria. Esse tipo de ataque possui uma variante denominada Negação de Serviço Distribuída - *(Distributed Denial of Service – Ddos)* -, que corresponde a ataques de DoS realizados em larga escala, partindo de várias estações e disparados simultaneamente de forma coordenada sobre um ou mais alvos.

O objetivo do ataque de negação de serviço não é ganhar o acesso não autorizado às máquinas ou dados, mas impedir que usuários legítimos de um serviço o utilizem, segundo CERT (1997).

DoS é um ataque que compromete a disponibilidade de recursos de computador. Ataques DoS comuns incluem *ping floods* e e-mails bombas, ambos pretendem consumir grande quantidade de recursos, matando processos legítimos. Outros ataques têm como alvo os erros de programação de software com a intenção de derrubar o sistema.

Os principais ataques são:

- ***Syn Flood***

Uma das formas de ataque mais conhecidas é a *Syn Flooding*, no qual um computador tenta estabelecer uma conexão com um servidor através de um sinal do TCP conhecido por SYN. Se o servidor atender ao pedido de conexão, enviará ao computador solicitante um sinal chamado ACK. O problema é que, em ataques desse tipo, o servidor não consegue responder a todas as solicitações e então passa a recusar novos pedidos.

- ***UDP Packet Storm***

Um computador faz solicitações constantes para que uma máquina remota envie pacotes de respostas ao solicitante. A máquina fica tão sobrecarregada que não consegue executar suas funções.

- ***Smurf***

É um ataque baseado em *IP spoofing e broadcast*, ocorre quando um pacote ICMP *echo request* é enviado para endereços IP de *broadcast*. Todos possuem endereços de origem forjados *(spoofed).* Todos os servidores que estiverem ativos nessa rede, responderão com pacotes ICMP *echo replay*. Se o número de respostas for muito grande, o servidor que estiver recebendo não conseguirá responder todas as solicitações e com isso irá parar de responder.

- **Ping da Morte *(Ping of Death)***

É um ataque que tenta desativar um sistema, enviando pacotes de IP fragmentados. Um pacote ICMP *echo request* com tamanho maior que 65K é enviado, fragmenta-se e é reconstituído no servidor de destino. Na reconstituição desse pacote, ocorre um *overflow* do *buffer* TCP/IP, ocasionando queda, travamento e reinicialização dos servidores atacados.

- ***Teardrop***

Ao contrário do ataque denominado *Smurf*, que utiliza a “força bruta” para gerar o ataque, o *teardrop* executa um ataque de DoS, utilizando-se de falhas em diferentes implementações da pilha TCP/IP. Este ataque explora a incapacidade de alguns sistemas operacionais de reconstituir pacotes IP fragmentados. Como resultado, os sistemas suscetíveis a esse ataque têm o funcionamento prejudicado, podendo travar o sistema operacional.

- ***Land***

Neste tipo de ataque, o atacante envia um pacote com endereços e portas de origem e destino exatamente iguais no servidor a ser atacado. Isso provocará um *loop* no TCP/IP do servidor que ficará travado.

2.1.4 *OS Fingerprinting*

Essa técnica tem como objetivo identificar o sistema operacional instalado em um determinado *host.* Para isso, utiliza-se de um conjunto de técnicas que detectam características da implementação do protocolo TCP/IP do sistema operacional que está instalado no *host* alvo.

Uma vez que essas características tenham sido identificadas, realiza-se uma comparação dessas informações com a base de dados da ferramenta de ataque, a fim de descobrir qual o sistema operacional da estação em questão.

2.1.5 *IP Spoofing*

Um ataque *IP spoofing* ocorre quando um atacante fora de sua rede finge ser um computador confiável com acesso privilegiado, usando um endereço IP que existe na sua rede ou usando um endereço IP externo, autorizado, de confiança, que podem fornecer acesso para recursos específicos na sua rede. CISCO (2002)

"O *spoofing* não é necessariamente destrutivo, mas sinaliza uma invasão próxima. O endereço pode estar fora de sua rede - para ocultar a identidade do invasor -, ou pode ser um de seus endereços internos confiáveis com acesso privilegiado." (MICROSOFT, 2004)

2.1.6 Ataques de aplicações *Web*

2.1.6.1 SQL Injection

É um tipo de *exploit* em que o atacante adiciona o código SQL a uma caixa da entrada de um formulário *web* para ganhar acesso aos recursos ou fazer mudanças em seus dados. Esses ataques podem ser facilmente evitados, quando um sistema possui uma validação de entrada de dados eficiente.

2.1.6.2 Cross-site scripting

É um tipo de *exploit* em que o atacante insere códigos maliciosos em um link que parece ser de fonte confiável. Quando o link é clicado, a programação embutida é submetida como parte de uma solicitação do cliente *web* e pode ser executada no computador do usuário, permitindo, dessa forma, ao atacante o roubo da informação.

2.1.7 Ataques de Infra-Estrutura

Ataques de infra-estrutura são ataques que afetam os componentes-chave da infra-estrutura da Internet, como servidores de nome, provedores de acesso etc. Por afetarem uma grande parte da Internet, podem seriamente impedir o funcionamento de muitos sites. Estão em crescimento por causa do grande número de organizações e usuários existentes e dependentes da Internet.

São ataques que "visam explorar deficiências na implementação de toda a infra-estrutura de rede e na própria implementação das aplicações sobre o protocolo TCP/IP." (MELCHIORI, [200-]).

O CERT (2002) cita quatro tipos de ataques de infra-estrutura conforme descritos abaixo:

- Negação de serviço distribuído - *o Distributed denial of service - DdoS* -, conforme explicado no item 2.1.3, utiliza-se de vários sistemas para atacar uma ou várias máquinas, sendo seu objetivo negar o serviço aos usuários legítimos dos sistemas da vítima.
- *Worms* - é um código malicioso capaz de se autopropagar através da Internet, enviando cópias de si mesmo de computador para computador. "Diferente do vírus, o *worm* não necessita ser explicitamente executado para se propagar. Sua propagação se dá através da exploração de vulnerabilidades existentes ou falhas na configuração de *softwares* instalados em computadores." (CERT.br, 2003). Um exemplo foi o *Code Red* que infectou mais de 250.000 sistemas em apenas 9 horas, em 19 de julho de 2001.

- Ataques em Servidores de DNS - *Domain Name Server* - incluem as seguintes ameaças:
  - Envenenamento de *cache* - *Cache poisoning* - se o DNS for feito para armazenar falsa informação, o atacante pode redirecionar o tráfego para um site real sob seu controle;
  - Dados Comprometidos - *Compromised data* - os atacantes possuem habilidade de modificar os dados fornecidos aos usuários através dos DNS vulneráveis*;*
  - Negação de Serviço - um grande ataque DoS em alguns dos servidores de nome poderia causar uma grande lentidão ou a interrupção da Internet;
  - Seqüestro de domínio - *Domain hijacking* - por investir em mecanismos inseguros utilizados por usuários para atualizar suas informações de registros de domínios, os atacantes podem se tornar membros dos processos de registro do domínio para ter controle dos domínios legítimos.
- Ataques contra ou usando roteadores - os intrusos podem usar roteadores mal configurados como plataformas para redirecionar tráfego para outros sites sob o seu controle.

2.2 Vulnerabilidades

A grande maioria dos *worm*s e outros ataques bem sucedidos acontecem devido às vulnerabilidades existentes em alguns serviços comuns nos sistemas operacionais. Os atacantes pegam os caminhos mais fáceis e exploram as falhas mais conhecidas com a utilização de ferramentas amplamente conhecidas. Eles esperam que as organizações não corrijam seus problemas e, geralmente, fazem varreduras por toda a Internet, procurando por vulnerabilidades.

A facilidade com que há a disseminação dos *worms*, como o *Blaster* e *Code Red*, pode estar ligada diretamente à exploração de vulnerabilidades não corrigidas.

A seguir, as principais vulnerabilidades encontradas em ambientes *Windows* e *Unix* segundo SANS (2004).

2.2.1 Principais vulnerabilidades

- **Servidores *Web***

As instalações padrão, de servidores *Web* mostram-se vulneráveis a vários ataques. O impacto dessas vulnerabilidades pode incluir:

• Negação de Serviço (*Denial of Service*);

• Exposição ou comprometimento de arquivos ou dados confidenciais;

• Execução de comandos arbitrários em servidores; e

• Comprometimento do servidor.

Servidores *Web* em *Unix*, como o *Apache* e o Sun *Java System Web Server* - antigo *iPlanet* -, fornecem a maioria do tráfego e por isso, precisam de uma atenção especial em relação aos aspectos de segurança. Estes aspectos incluem vulnerabilidades dentro do próprio servidor, assim como erros no *Php* e vários outros tipos de ataques.

Embora existam vários tipos de ataques diferentes, a maior causa do comprometimento de servidores *Web* em *Unix*, resulta de um sistema mal configurado no momento de sua instalação ou sem uma manutenção regular. O resultado de um

comprometimento pode ser um ataque DoS - *web defacement* -, e acesso de *root* ao sistema.

- **Serviços de Acesso Remoto**

Plataformas *Windows* suportam uma variedade de métodos e tecnologias de rede. As vulnerabilidades neste caso são algumas tecnologias de redes mal configuradas como os compartilhamentos de rede NETBIOS, sessão anônima, acesso remoto ao registro e serviços RPC (*Remote Procedure Call*).

- ***Microsoft SQL Server (MSSQL)***

Contém várias vulnerabilidades que permitem ao atacante remoto obter informação confidencial, modificar o conteúdo da base de dados e também comprometer os servidores. Dois *worms* exploraram, recentemente, várias falhas do *MSSQL*: *SQLSnake/Spida* (maio de 2002) e SQL-*Slammer/SQL-Hell/Sapphire* (janeiro de 2003).

- **Autenticação**

A maioria das formas de autenticação, bem como a proteção de dados e arquivos, baseia-se em senhas de usuários. Como o acesso autenticado normalmente não é registrado em *logs* ou não causa suspeita, uma senha comprometida é uma oportunidade de explorar um sistema.

As vulnerabilidades mais comuns relacionadas a senhas são:

• Contas de usuários com senhas fracas ou inexistentes;

• Falha do usuário em protegê-las;

• Senhas fracas ou inexistentes de contas administrativas de sistemas operacionais ou softwares adicionais; e

• Algoritmos para descoberta de senhas facilmente encontráveis.

A melhor e mais apropriada forma de defesa é o uso de uma política de senhas que incluam instruções para os bons hábitos e verificação pró-ativa da integridade das senhas.

- **Programas de Compartilhamento de Arquivo**

São utilizados para *download* de vários tipos de arquivos como textos, músicas e vídeos. Os clientes participam, baixando arquivos de outros usuários, tornando seus dados disponíveis para outros.

Várias vulnerabilidades existem quando se usam programas de compartilhamento de arquivo, podendo ser classificadas em três tipos:

1. *Vulnerabilidades técnicas* - são aquelas que podem ser exploradas remotamente;
2. *Vulnerabilidades sociais* – existem, quando um usuário malicioso ou previamente infectado cria ou altera um arquivo para que se assemelhe a algo desejado por outro usuário, podendo resultar em vírus *trojan*, *worm* ou em outros tipos de códigos maliciosos;
3. *Vulnerabilidades legais* - são aquelas que podem resultar em infração de *copyright* ou material censurável. O conteúdo disponível através desses aplicativos inclui músicas com *copyright*, filmes e programas.

- **BIND - Sistema de nome de domínio**

O pacote *Berkeley Internet Name Domain* (BIND) se tornou uma das implementações de *Domain Name Service* (DNS) mais utilizadas. O DNS é um sistema crítico que facilita a conversão de *hostnames* em um endereço IP correspondente. Devido à sua ação critica, o BIND tornou-se um alvo de ataques.

Ataques DoS que geralmente resultam em uma completa perda do serviço de nomes da Internet, por muito tempo incomodaram o BIND. Vários outros ataques, como *buffer overflow* e envenenamento de *cache* (*cache poisoning*), foram descobertos para o BIND. Embora a equipe de desenvolvimento do BIND tenha sido rápida nas respostas e no desenvolvimento de correções, um número ainda grande de servidores em produção são encontrados desatualizados, vulneráveis e mal configurados.

- ***Simple Network Management Protocol* (SNMP*)***

O *Simple Network Management Protocol* (SNMP) é utilizado para monitoramento e configuração remota de todos os tipos de dispositivos modernos que suportam TCP/IP. Enquanto o SNMP é quase onipresente na distribuição nas plataformas de rede, ele é usado, freqüentemente, como um método para configurar e gerenciar dispositivos como impressoras, roteadores, *switches*, pontos de acessos e para prover entradas para os serviços de monitoramento de rede.

A comunicação do SNMP consiste em diferentes tipos de mensagens trocadas entre estações que gerenciam SNMP e os dispositivos de rede que rodam o *software* chamado de agente. O método pelo qual essas mensagens são trocadas e o mecanismo de autenticação utilizado por elas possui vulnerabilidades significativas, capazes de serem exploradas.

- **Banco de Dados**

Bancos de dados são elementos que incluem informações críticas. Mesmo com a importância da integridade e confidencialidade de dados, os sistemas de gerenciamento de sistemas de banco de dados (DBMS) não seguem o mesmo nível de segurança de sistemas operacionais e redes.

A integridade e a confidencialidade dos dados podem ser comprometidas por vários fatores, incluindo a complexidade da implementação, uso de senhas não seguras, configurações erradas e *backdoors* desconhecidos. A maioria das organizações privadas e públicas utiliza banco de dados para armazenar informações pessoais, como informações de pagamentos, históricos médicos etc. Bancos de dados armazenam informações sensíveis como dados financeiros, incluindo informações de comércio de ações, transações financeiras, informações de clientes como números de cartões de crédito e informações confiáveis de parceiros.

Atualmente, grande parte dos bancos de dados está relacionada com aplicações para usuários finais (*frontend*). Se uma aplicação é configurada ou escrita de forma insegura, isso pode possibilitar que um invasor faça um ataque de *SQL injection* ou explore alguma das vulnerabilidades do banco de dados.

Segundo o relatório elaborado por BACE e MELL (2001), publicado pelo NIST (*National Institute of Standards and Technology*), os principais tipos de vulnerabilidade são:

- Erros na validação de entrada - classificados como vulnerabilidades de implementação, esses erros são causados por entradas de dados indevidamente tratados, ou seja, conjuntos de dados não especificados que geram resultados inesperados quando inseridos no sistema. Um dos mais importantes e conhecidos tipos de erros na validação de entrada é o chamado *buffer overflow*, conforme citado no item 2.1.2;
- Erros na validação de acesso - problemas de projeto ou de implementação, essas vulnerabilidades são causadas por erros nos próprios mecanismos de controle de acesso;
- Erros manipulando exceções - o sistema torna-se vulnerável pelo surgimento de exceções a serem tratadas e a manipulação (ou má manipulação) dessas exceções torna o sistema vulnerável;
- Erros de ambiente - o sistema pode ser seguro, mas o ambiente onde ele está pode torná-lo vulnerável;
- Erros de configuração - não estão relacionados à forma como o sistema foi construído e sim às configurações que o usuário final usou para executá-lo; e
- Condições de corrida - ocorrem quando existe um atraso entre o momento em que o sistema verifica se uma operação é permitida e a efetivação dessa operação. Nesse curto espaço de tempo, ações ilegais podem ser executadas, utilizando os privilégios do sistema em questão.

2.3 Estatísticas

Serão apresentadas estatísticas nacionais e internacionais de incidentes de segurança.

2.3.1 Internacionais

O CERT/CC - *Computer Emergency Response Team/Coordination Center* - é um centro de coordenação de informações relacionadas à segurança de redes e serviços de Internet, criado em 1998 e situado no Instituto de Engenharia de Software (*Software Engineering Institute - SEI*), um centro de pesquisa e desenvolvimento financiado e operado pela Universidade *Carnegie Mellon*. Tem como objetivo proteger sistemas contra problemas atuais e pesquisas sobre problemas futuros. Seu trabalho envolve analisar e registrar incidentes e vulnerabilidades de segurança de redes, publicando alertas de segurança, pesquisando mudanças em sistemas de rede e desenvolvendo informações e treinamento para ajudar a melhorar a segurança das nossas redes.

As tabelas abaixo exibem estatísticas de incidentes de segurança comunicados ao *CERT/CC*, no período de 1988 ao 2º quadrimestre de 2005.

Tabela 1 - Estatísticas do número de incidentes de segurança publicados pelo CERT/CC de 1988 a 2003.

Número de Incidentes

1988-1989

Ano	1988	1989
Incidentes	6	132

1990-1999

Ano	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999
Incidentes	252	406	773	1334	2.340	2.412	2.573	2.134	3.734	9.859

2000-2003

Ano	2000	2001	2002	2003
Incidentes	21.756	52.658	82.094	137.529

Total de Incidentes (1988 - 2003): **319.992**

Fonte: CERT, 2005.

A Tabela 1 compara o número de incidentes de segurança comunicados ao CERT/CC de 1998 até 2003.

A Tabela 2 apresenta o número de vulnerabilidades comunicadas ao CERT/CC de 1995 ao 2º quadrimestre de 2005.

Tabela 2 - Estatísticas de vulnerabilidades publicadas pelo CERT/CC de 1995 ao segundo quadrimestre de 2005.

Vulnerabilidades						
1995-1999						
Ano	1995	1996	1997	1998	1999	
Vulnerabilidades	171	345	311	262	417	
2000-2005						
Ano	2000	2001	2002	2003	2004	1Q-2Q 2005
Vulnerabilidades	1.090	2.437	4.129	3.784	3.780	2.874
Total de Vulnerabilidades (1995-2º Q 2005): **17.946**						

Fonte: CERT, 2005.

Pode-se identificar com esses números, um grande crescimento de vulnerabilidades de segurança. Isso mostra o motivo da preocupação que os profissionais de segurança vêm apresentando ultimamente.

2.3.2 Nacionais

O CERT.br - Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil -, anteriormente denominado NBSO/Brazilian CERT, é o Grupo de Resposta a Incidentes para a Internet Brasileira, mantido pelo Comitê Gestor da Internet no Brasil, responsável por receber, analisar e responder a incidentes de segurança em computadores, envolvendo redes conectadas à Internet brasileira.

As tabelas e gráficos apresentados abaixo exibem estatísticas de incidentes de segurança comunicados ao CERT.br em 2004 e 2005.

Tabela 3 - Incidentes mensais, classificados por tipo de ataque, comunicados ao CERT.br, de janeiro a junho de 2005.

Mês	Total	Worm (%)		af (%)		dos (%)		invasão (%)		aw (%)		varreduras (%)		fraude (%)	
jan	**4448**	1019	22	16	0	0	0	14	0	22	0	2694	60	683	15
fev	**3142**	1157	36	5	0	1	0	27	0	57	1	1433	45	462	14
mar	**4848**	1906	39	1	0	2	0	42	0	24	0	1805	37	1068	22
abr	**5253**	1432	27	17	0	0	0	20	0	25	0	1437	27	2322	44
mai	**6883**	2175	31	4	0	2	0	34	0	22	0	1489	21	3157	45
jun	**5406**	1510	27	0	0	5	0	17	0	55	1	1356	25	2463	45
Total	**29980**	9199	182	43	0	10	0	154	0	205	2	10214	215	10155	185

Fonte: CERT.br, 2005

Legendas:

- Af - Ataque ao usuário final;
- Dos - Denial of Service; e
- Aw - Ataque a servidor *Web*.

Tabela 4 - Incidentes mensais e trimestrais, classificados por tipo de ataque, comunicados ao CERT.br em 2004.

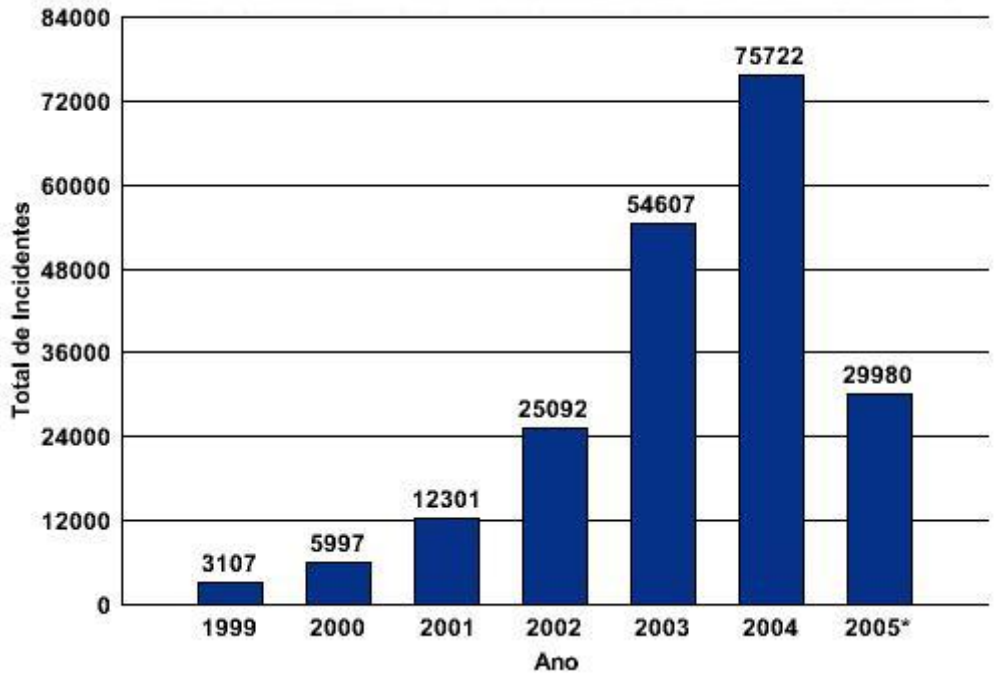
Mês	Total	worm (%)		af (%)		dos (%)		invasão (%)		aw (%)		varreduras (%)		fraude (%)	
Jan	**5886**	3013	51	39	0	6	0	9	0	55	0	2481	42	283	4
Fev	**6110**	2306	37	53	0	4	0	13	0	22	0	3542	57	170	2
mar	**6002**	2653	44	37	0	19	0	56	0	32	0	2862	47	343	5
abr	**4763**	2496	52	36	0	2	0	14	0	81	1	1946	40	188	3
mai	**5471**	2260	41	38	0	2	0	19	0	58	1	2913	53	181	3
jun	**6502**	3752	57	24	0	3	0	6	0	26	0	2498	38	193	2
jul	**6773**	4636	68	18	0	2	0	7	0	49	0	1791	26	270	3
ago	**5910**	3221	54	26	0	5	0	22	0	71	1	2194	37	371	6
set	**5167**	2997	58	56	1	4	0	13	0	52	1	1704	32	341	6
out	**11253**	8821	78	23	0	51	0	13	0	29	0	1937	17	379	3
nov	**7149**	4599	64	10	0	1	0	40	0	39	0	1888	26	572	8
dez	**4736**	1513	31	46	0	5	0	36	0	10	0	2402	50	724	15
Total	**75722**	42267	55	406	0	104	0	248	0	524	0	28158	37	4015	5

Fonte: CERT.br, 2005.

Os dados das Tabelas 3 e 4 mostram o total de incidentes de segurança comunicados no 1º semestre de 2005 e no ano de 2004 pelo Brasil.

Abaixo, são mostrados dois gráficos apresentados pelo CERT.br, mostrando incidentes de segurança em geral.

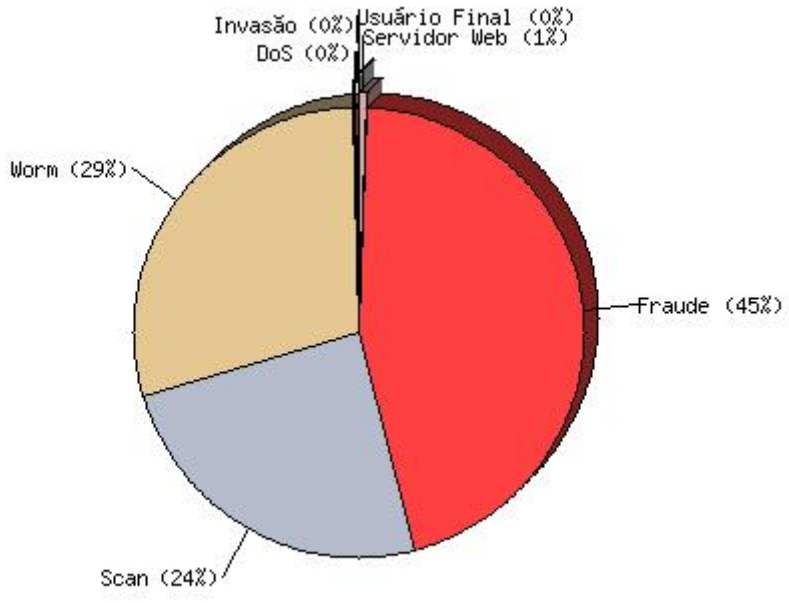
Gráfico 1 - Notificações do número de incidentes comunicados ao CERT.br, de 1999 a junho de 2005.



Fonte: CERT.br, 2005.

O Gráfico 1 compara a quantidade de notificações sobre incidentes de segurança recebidas pelo CERT.br por ano, de 1999 até junho de 2005.

Gráfico 2 - Percentual de incidentes comunicados ao CERT.br, no período de abril a junho de 2005, segundo o tipo de ataque.



Fonte: CERT.br, 2005.

O Gráfico 2 mostra, em porcentagem, estatísticas de incidentes comunicados ao CERT.br, no período de abril a junho de 2005. O gráfico é composto por dados relacionados a invasões, ataques de negação de serviço (*DoS*), *worms*, varreduras (*scan*), ataques a servidores *web*, ataques a usuários finais e fraudes.

3 SDI COMO FERRAMENTA DE DEFESA

O objetivo do SDI é monitorar o sistema contra ataques. Existe diferença entre detecção de acesso indevido por usuário autorizado e detecção de intrusão. Detecção de acesso indevido por usuário autorizado tem o objetivo de detectar problemas de segurança na rede, monitorando as atividades dos usuários autorizados. Detecção de intrusão assume que um intruso não tenha usado ou não esteja usando uma conta legítima, preocupando-se com ataques externos. Os SDIs ocupam-se com os casos de intrusão, porém também detectam atividades maliciosas de usuários internos, como varreduras de portas.

Neste capítulo, serão apresentados alguns tipos existentes de SDI e suas características.

3.1 Conceitos importantes

Alguns conceitos importantes serão usados neste capítulo, como:

- **Falso positivo -** ocorre quando a ferramenta classifica uma ação como uma possível intrusão, quando na verdade não é;
- **Falso negativo -** ocorre quando uma intrusão real acontece, mas a ferramenta permite que ela passe como se fosse uma ação normal; e
- **Subversão -** ocorre quando o intruso modifica a operação da ferramenta de SDI para forçar a ocorrência de falso negativo.

3.2 SDIs de Rede

A detecção de intrusão é o processo de monitoramento de eventos que ocorrem em um *host* ou rede e os analisa através dos sinais das intrusões, definidos como tentativas de comprometer a confidencialidade, integridade e disponibilidade de dados ou para enganar os mecanismos de segurança.

As intrusões são causadas por acessos indevidos de invasores nos sistemas. Os invasores podem ser usuários autorizados que tentam ganhar privilégios adicionais aos quais não têm direito ou que utilizam impropriamente seus privilégios. Os SDIs automatizam esse processo de monitoramento e análise.

A principal finalidade do SDI de rede é identificar ataques relacionados às falhas de segurança de recursos externos e internos de rede. Trabalha com assinaturas que são comportamentos já conhecidos de determinados pacotes de rede. Quando um pacote confere com uma assinatura é gerado um evento, sendo esse registrado em um banco de dados. Para que sua rede esteja sempre protegida, é necessário que seu SDI possua uma política atualizada de assinaturas, para que tenha proteção contra os novos tipos de ataques. Mesmo com assinaturas atualizadas, vão aparecer os falsos positivos e negativos.

A detecção de intrusão em *hosts* não é o objetivo desse trabalho.

3.3 Método de detecção do SDI

Os alertas são gerados através de dois métodos de detecção:

- **Baseado em Assinaturas**

Este método trabalha procurando regras pré-estabelecidas no tráfego da rede. Se um pacote na rede conferir com as assinaturas, será gerado um alerta ou evento no sistema.

Vantagem: foca a análise nos dados auditados e geralmente produz poucos falsos positivos.

Desvantagem: pode detectar apenas os ataques conhecidos que foram definidos pelas assinaturas.

- **Baseado em Anomalias**

Esse método possui uma base de dados do comportamento da rede, a partir da qual o sistema verifica o que é permitido. E quando encontra algo não permitido gera um alerta.

Vantagem: pode detectar ataques desconhecidos, já que não trabalha analisando uma regra específica.

Desvantagem: pode causar um número alto de falsos positivos.

3.4 Características do SDI

NED (1999) afirma que muitas ferramentas de SDI funcionam a partir da análise de atividades do sistema operacional e/ou da rede, como: atividades dos usuários, número de conexões, utilização de CPU, Entrada/Saída (E/S) de disco, uso de memória, número de tentativas de acesso, volume de dados trafegando na rede, entre outros. Através desses dados, forma-se uma base de informações que serão utilizadas pelo sistema. Com todas essas informações, o SDI pode identificar as tentativas de intrusão e até mesmo registrar a técnica utilizada.

Uma ferramenta SDI deve possuir algumas características importantes NED (1999):

- Rodar continuamente sem interação humana;
- Ser tolerante a falhas;
- Resistir a tentativas de mudança de sua base;
- Ter o mínimo de impacto no funcionamento do sistema;
- Detectar mudanças no funcionamento normal;
- Fácil configuração;
- Cobrir as mudanças do sistema durante o tempo, como no caso de uma nova aplicação que comece a fazer parte do sistema; e
- Ser difícil de ser enganado.

3.5 Funções do SDI

Os SDIs executam as seguintes funções, segundo BACE e MELL (2001):

- Monitoramento e análise dos eventos do sistema e do comportamento do usuário;
- Teste do estado da segurança das configurações do sistema;
- Determinação de um modelo para o nível de segurança do sistema e então execução das alterações para chegar a esse modelo;

- Identificação dos padrões dos eventos do sistema que correspondam aos ataques conhecidos;
- Identificação dos padrões de atividade que estatisticamente diferem da atividade normal;
- Gerência da operação do sistema de auditoria, mecanismos de *log* e dos dados que eles geram;
- Alerta a equipe apropriada de funcionários, por meios apropriados, quando os ataques são detectados;
- Medição da aplicação das políticas de segurança codificadas no mecanismo de análise;
- Fornecimento de informação-padrão sobre políticas de segurança; e
- Permissão para que peritos em “falta de segurança” executem funções importantes de monitoração da segurança.

4 METODOLOGIA PARA A ANÁLISE COMPARATIVA DE SDIs

O propósito de uma metodologia para análise de SDIs é tornar possível determinar qual ferramenta se adapta melhor aos diferentes ambientes de rede existentes.

Isso é necessário porque as características de uma rede e seu volume do tráfego variam.

Algumas metodologias têm sido estudadas como ALESSANDRI (2000), (2001); LIPPMANN et al. (2000) e PUKETZA et al. (1997); não foram encontradas metodologias mais novas. Nenhuma possui uma forma sistematizada para execução dos procedimentos citados e também são compostas por uma série de atividades exaustivas que são realizadas durante semanas e exigem que os usuários da metodologia possuam conhecimentos específicos, tais como da estrutura interna dos SDIs - o que é impossível no caso de produtos comerciais. Além disso, suas documentações não são muito claras, dificultando a sua aplicação.

4.1 Metodologias existentes para avaliação de SDIs

Este item apresenta uma síntese das principais abordagens desenvolvidas até o momento para avaliação de sistemas de detecção de intrusão. Ao final dessa síntese, é feita uma comparação entre as principais características que compõem essas metodologias e a metodologia proposta.

4.1.1 Puketza

PUKETZA et al. (1997) publicaram a primeira proposta de metodologia para avaliação de SDIs, desenvolvida na Universidade da Califórnia. A primeira atividade consiste na seleção de cenários de teste. Esses cenários são reproduzidos através de *scripts* que simulam tanto ataques quanto atividades consideradas normais. A melhor forma de escolher um cenário de teste, segundo o autor, é basear-se na política de segurança da empresa, pois é ela que define o que é uma intrusão. Uma vez selecionados os casos de teste, é possível desenvolver *scripts* que simulam diferentes comportamentos de intrusão.

PUKETZA et al.(1997) desenvolveram uma variedade de experimentos de teste que evidenciam o restante de sua metodologia. A maioria dos procedimentos se baseia nos mesmos passos básicos e permite criar um grupo de *scripts,* estabelecendo as condições desejadas para execução do teste. Os procedimentos dos testes foram divididos em três categorias: identificação de intrusão, uso de recursos e testes de saturação (*stress*).

Identificações de intrusão - são experimentos que verificam o comportamento do SDI ao detectar duas formas distintas de ataques: concorrentes e seqüenciais;

Usos de recurso - correspondem a testes que avaliam a quantidade de recursos computacionais, (carga da CPU, memória principal e espaço em disco) consumidos pelo SDI; e Testes de saturação - verificam o comportamento desses sistemas quando submetidos a situações que visam a esgotar diferentes recursos, como o aumento do número de processos concorrentes em execução na estação em que o sistema estiver instalado.

Em testes realizados, um único SDI, conhecido por *Network Security Monitor* (*NSM*), foi submetido aos experimentos previstos pela plataforma de *software* desenvolvida por PUKETZA et al.(1997). Esta plataforma tem grupos de comandos - seção básica, sincronização, comunicação, gravação e repetição -, possuindo cada grupo diferentes comandos, fornecendo ainda muitas funcionalidades necessárias para simular atividades do usuário. O *NSM* é um sistema de detecção de intrusão baseado em rede que, a exemplo de outras ferramentas, detecta ataques através da análise de assinaturas. No entanto, esse SDI possui uma característica incomum aos demais: atribui valores, entre 0 e 10, para cada conexão estabelecida na rede monitorada. Esses valores são atribuídos, considerando a freqüência e o tipo de conexão estabelecida. Para a realização dos experimentos propostos foram criados *scripts* (seqüenciais e concorrentes) para simular cada uma das seguintes ações: transmissão de um arquivo de senhas para uma outra estação da rede; descobrimento de senhas através de programas ou por tentativa e erro; e, ainda, exploração das vulnerabilidades em alguma aplicação, resultando em acesso privilegiado. A premissa existente por trás dessa estratégia é que o *NSM* atribuiria um peso maior para uma sessão composta por muitas atividades intrusivas (seqüenciais) do que para diversas sessões compostas por poucas atividades intrusivas (concorrentes).

4.1.2 Alessandri

ALESSANDRI (2000) desenvolveu pela IBM um trabalho que propõe uma abordagem de avaliação, cujo objetivo é testar as capacidades existentes nos SDIs e não as suas implementações ou base de assinaturas, sendo possível averiguar a capacidade de detecção das ferramentas, frente a um ataque para o qual o SDI ainda não tenha uma assinatura desenvolvida.

A metodologia consiste em uma técnica implementada em *prolog*, que permite descrever em forma de regras, as características existentes nos SDIs a serem avaliadas e as características exploradas em diferentes ataques. Possibilita identificar, a partir do cruzamento das regras, o comportamento dos SDIs sem que seja necessária a realização de experimentos com os sistemas avaliados.

Esta descrição de características é realizada através de duas formas. A primeira separa as propriedades dos SDIs de acordo com o nível de detalhe das características descritas, considerando um grupo de propriedades genéricas e outro de propriedades detalhadas. A segunda forma de classificação separa as propriedades dos SDIs, em função de características, tais como: posicionamento dos sensores, técnicas utilizadas para reconhecimento de padrões e o tempo entre a ocorrência de uma atividade e a geração do respectivo alarme. A descrição de uma atividade é representada por propriedades e regras que descrevem as características requeridas para que o SDI gere um alarme para essa atividade.

Com todas as características dos SDIs e as atividades identificadas e devidamente representadas, a aplicação, desenvolvida em *prolog*, está apta a executar a avaliação do SDI.

4.1.3 Lippmann

A taxonomia utilizada por Lippmann em 1998 e 1999, classifica os ataques em relação a três aspectos: nível de privilégio atual do usuário, métodos de transição e tipos de ações executadas pelo usuário.

A primeira etapa da avaliação realizada em 1998 foi coletar amostras referentes ao tráfego existente nas bases da força aérea americana para, criando um conjunto de dados sintéticos, representar diversas atividades realizadas pelos usuários como navegação na Internet, transferência de arquivos via *ftp*, compilação códigos, entre outras. A partir de

alterações no *kernel* dos sistemas operacionais das estações responsáveis pela geração do tráfego de fundo, foi possível simular atividades geradas por dezenas de usuários. Na etapa seguinte, para coletar tráfego de ataque, foram lançados diferentes tipos de ataques contra as estações Linux e Solaris da rede alvo para que o tráfego do ataque gerado fosse armazenado para posterior reprodução. Já na avaliação realizada em 1999, foram incluídos nesse tráfego ataques contra servidores *Windows NT*, ataques fragmentados e ainda ataques denominados *Stealth*, cujo objetivo é confundir o SDI em relação ao que é tráfego normal e ao que de fato é um ataque real. Segundo LIPPMANN et al. (2000), os principais objetivos dos testes realizados em 1998 e 1999 foram identificar os ataques detectados e as taxas de falsos positivos geradas pelos SDIs testados, fornecendo aos desenvolvedores de tais sistemas subsídios para corrigir falhas existentes nos mesmos e fornecer uma forma imparcial de testar o comportamento dos SDIs submetidos à avaliação.

4.1.4 Comparações das metodologias existentes

Em relação aos tipos de avaliações, foi constado que as metodologias de PUKETZA et al.(1997) e LIPPMANN et al. (2000) avaliam apenas as bases de assinaturas dos SDIs, o que além de exaustivo, considerando-se o tamanho destas bases, gera um resultado válido por um pequeno período de tempo, pois assinaturas são desenvolvidas muito rapidamente pelos fabricantes dos SDIs ou até mesmo pelos usuários destas ferramentas. Portanto, para que os resultados destas metodologias possam ser considerados confiáveis, é necessário que sejam refeitos os experimentos previstos para cada nova assinatura que surge. Por outro lado, metodologias tais como a proposta por ALESSANDRI (2000), ao invés de testarem a base de assinaturas, testam as capacidades de detecção dos SDIs e devem ter seus experimentos refeitos somente quando novos recursos forem implementados a estas ferramentas.

O tráfego de fundo é uma característica fundamental na avaliação de SDIs, pois interfere diretamente nos resultados de alguns testes, tais como verificação das taxas de falsos positivos e da escalabilidade. Propostas publicadas por LIPPMANN, em 1998 e 1999, não descrevem a composição do tráfego de fundo, fazendo com que os resultados, principalmente da avaliação de falsos positivos, sejam contestados, pois não há como afirmar se de fato existem ou não ataques inseridos nesse tráfego e também não há como identificar os motivos

que levaram os SDIs a gerar tais resultados. Portanto, os resultados destes testes podem representar valores incorretos.

Propostas como de PUKETZA et al.(1997) e LIPPMANN et al. (2000) requerem ambientes de teste complexos, com dezenas de estações - atacantes, vítimas, sistemas avaliados, geradores e coletores de tráfego -, diferentes equipamentos de interconectividade (*hub*, *switch* e roteadores) e até mesmo *firewalls*. Essas características de um ambiente de teste podem inviabilizar sua reprodução, pois além de complexos, demandam tempo e ambiente dedicado até o término dos testes. Além disso, o uso de *firewalls* faz com que diversos ataques não sejam capturados pelos SDIs, pois são bloqueados antes de chegarem à rede interna da empresa, sendo uma limitação no processo de avaliação. Nesse caso, a metodologia proposta por ALESSANDRI (2000) é uma exceção, não utilizando ambiente de teste para a realização de seus experimentos.

O resultado apresentado será a última característica a ser considerada. Propostas que prevêem o uso de ferramentas como mecanismos de auxílio no processo de avaliação, devem garantir que estes mecanismos estejam em perfeito funcionamento e que sejam facilmente utilizados para avaliação de diferentes SDIs, mediante diferentes ataques. No entanto, PUKETZA et al. (1997) criaram e testaram sua ferramenta sob características existentes em um único SDI. Embora essa proposta consista no uso de uma plataforma de *software* baseada em *scripts* que geram tanto o tráfego de fundo quanto os ataques, a quantidade de *scripts* desenvolvidos é bastante limitada. Assim, é função dos usuários desta metodologia criar *scripts* em linguagem *C* ou *TCL*, para simulação do tráfego de fundo característico da sua empresa, bem como para simular novos ataques.

LIPPMANN et al. (2000) também descreve sua abordagem com o uso de uma ferramenta como principal mecanismo na realização dos experimentos e na geração do tráfego. No entanto, conforme o mesmo autor, a ferramenta ainda não possuia uma versão estável para sua utilização, inviabilizando a reprodução de tal abordagem. O trabalho de ALESSANDRI (2000), apesar de dispensar a necessidade de reproduzir um ambiente de teste, exige conhecimentos específicos de como os SDIs tratam determinadas características dos protocolos.

Observa-se, de forma geral, que as metodologias citadas possuem ausência de proposta com aplicação voltada a empresas. Para isso, é necessário o desenvolvimento de uma

abordagem com procedimentos definidos, de uso fácil e que se reflitam na realidade dos critérios avaliados.

4.1.5 Comparação com a metodologia proposta

O principal objetivo desta metodologia é prover uma abordagem de utilização prática para avaliação de SDIs, que possa ser aplicada sem o conhecimento interno dos SDIs e possa ser executada com as limitações de tempo e recursos normalmente encontrados.

Esta metodologia exige um ambiente de teste para realização da avaliação, com um tráfego real, ou seja, dados vindos da Rede Externa (Internet.). A avaliação das bases de assinaturas dos SDIs não é o objetivo desta metodologia.

4.2 Etapas da metodologia proposta

A metodologia proposta é composta por cinco etapas: seleção de tipos de ataques, seleção de ferramentas de ataque, especificação de um modelo para o teste, seleção dos SDIs e análise dos SDIs. Todas as etapas serão descritas abaixo.

4.2.1 Seleção de tipos de ataques

A primeira fase da metodologia é identificar quais ataques devem ser identificados pelo SDI e quais podem ser detidos por outros dispositivos.

Para essa metodologia, foi adotado um modelo de ambiente de rede, no qual as funções são distribuídas entre diversos dispositivos de defesa, conforme a Figura 1.

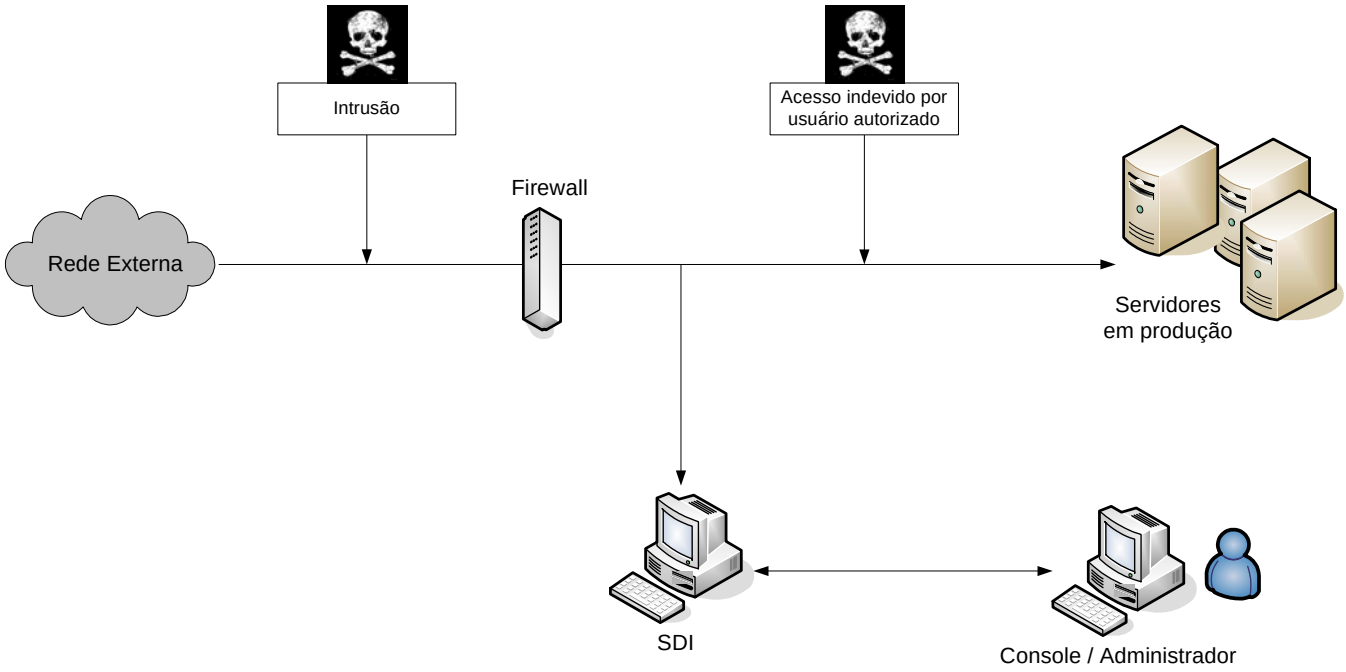


Figura 1 - Modelo de rede com dispositivos de defesa.

A Figura 1 representa um modelo de ambiente de rede composto pela Rede Externa (*Internet*), *firewall*, SDI, estação console e servidores em produção.

O *firewall*, além de detectar o acesso indevido por usuário autorizado nessa rede, pode também ter a função de antivírus, através de regras para bloquear a entrada de vírus e *worms*,

à medida que o SDI detecta a intrusão, conforme diz o Capítulo 4. Os servidores em produção também podem possuir a função de antivírus através de regras e filtros, os quais serão definidos de acordo com suas necessidades.

Esta fase da metodologia consiste em:

- Identificar os ataques possíveis;
- Verificar para quais ataques já existe proteção suficiente e excluí-los da análise.

4.2.1.1 Ataques não utilizados na aplicação da metodologia

Alguns ataques analisados no Capítulo 2 não serão discutidos, por não serem usualmente classificados como ataques de intrusão. Esses ataques são:

- **Negação de Serviço**

Os ataques de negação de serviço não serão considerados nessa metodologia por não serem classificados como ataques intrusivos. Segundo CERT (1997) e CERT (2001), esse ataque tem como objetivo impedir que usuários legítimos acessem seus sistemas e serviços, não obtendo acesso às máquinas ou dados.

- ***OS Fingerprinting***

Os ataques *OS Fingerprinting* não serão considerados nessa metodologia por não serem classificados como ataques intrusivos e usualmente serem detectados por *firewall*, segundo MANCINI (1997) e TROWBRIDGE (2003).

- ***IP Spoofing***

Os ataques de *IP Spoofing* por não serem necessariamente destrutivos e por serem detectados por *firewalls*, também não serão analisados nessa metodologia. CISCO (2002) MICROSOFT (2004).

- ***SQL Injection***

Os ataques *SQL Injection* são considerados ataques de *host* e não de rede, pois são ataques direcionados aos servidores *web* de uma organização, através da inserção de comandos SQL em aplicações *web*, a fim de manipular dados ou consultas.

- **Ataques de Infra-Estrutura**

Os ataques de infra-estrutura podem ser destrutivos, porém não são considerados intrusivos e por isso também não serão analisados nessa metodologia.

4.2.1.2 Ataques selecionados para a aplicação da metodologia

Os ataques a serem selecionados para aplicação da metodologia resultam do estudo da característica da rede a defender. Os principais ataques que devem ser considerados nessa fase são os seguintes:

- **Varredura de portas (*Port scan*)**

A varredura de portas é o primeiro tipo de ataque a ser analisado, tratando-se de um pré-requisito para as invasões. Para que um atacante consiga invadir um sistema ou rede, ele primeiramente precisa conhecê-lo e verificar quais são suas vulnerabilidades.

Alguns ataques de varreduras de portas foram selecionados para utilização dessa metodologia.

Ataques	SDI	Outros meios de defesa
Varredura de portas abertas		
TCP *Connect*		
UDP		
Varredura de portas semi-abertas		
TCP SYN		
Varredura de portas ocultas		
TCP FIN		
TCP *Xmas*		
TCP *Null*		
Outras varreduras de portas		
TCP ACK		
TCP *Window*		
Varredura ICMP		
Varredura *Decoy*		

Tabela 5 - Varreduras de Portas

- ***Buffer overflow***

Esse tipo de ataque será analisado nessa metodologia por se destacar entre uma das formas mais comuns de ataques, segundo COWAN et al. (2000), e também por ser freqüentemente detectado por SDIs.

Ataques	SDI	Outros meios de defesa
Buffer overflow		

Tabela 6 - *Buffer Overflow*

- ***Cross-site scripting***

O ataque *Cross-site scripting* pode ser detectável por assinaturas do SDI, mas não é detectado por *firewalls*.

Ataques	SDI	Outros meios de defesa
Cross-site scripting		

Tabela 7 - *Cross-site scripting*

Para cada aplicação dessa metodologia, a relação de ataques pode ser ampliada em função de novas ameaças.

4.2.2 Seleção de ferramentas de ataque

Nessa metodologia, o mais importante são os ataques reais. No entanto, como talvez não seja possível a análise de todos os ataques no período de teste, convém dispor de ferramentas de ataques para simular os casos de ataque mais conhecidos.

Os critérios adotados para seleção de ferramentas de ataque a utilizar nessa metodologia são que:

- As ferramentas devem possibilitar o teste dos ataques definidos na fase anterior;
- As ferramentas devem ser facilmente encontradas; e

- As ferramentas devem ser práticas para que o teste seja efetuado com rapidez e precisão.

4.2.2.1 Alguns exemplos de ferramentas de ataques.

- ***Nessus***

É uma ferramenta para varredura de vulnerabilidades, possuindo arquitetura cliente/servidor e recursos interessantes como trabalhar em conjunto com o *nmap* e também a possibilidade de atualização apenas baixando novos *plugins* de ataques e/ou vulnerabilidades. O *Nessus* foi desenvolvido para ser utilizado pelos próprios administradores para analisar o nível de segurança dos seus servidores.

- ***Nmap***

É uma ferramenta para varredura de portas, auditoria e exploração de segurança de redes.

- ***Whisker***

É uma ferramenta para varredura de vulnerabilidades CGI, cujo objetivo é procurar por falhas em servidores *web*, através da execução de diversos *scripts* desenvolvidos em *perl*.

4.2.3 Especificação de um modelo para o teste

O ambiente de teste a ser criado é baseado no modelo conceitual ilustrado na Figura 1. Conforme mostra a Figura 2, um ambiente de rede é criado com os SDIs sob teste, *Console* e *Atacante Fictício;* além do tráfego de fundo e servidores em produção.

Os detalhes da configuração básica de defesa e da configuração dos servidores devem refletir a realidade do ambiente de produção específico.

O tráfego de fundo é todo o fluxo de dados transmitido em paralelo aos ataques a serem reproduzidos, permitindo que o processo de teste de um ambiente de rede seja o mais real possível, sendo ele necessário para realizar a análise de escalabilidade dos SDIs. No entanto, nessa metodologia, o tráfego de fundo será representado pelo tráfego real de dados,

ou seja, dados vindos da própria Rede Externa *(Internet)*. Para o recebimento desse tráfego, é feito o espelhamento do tráfego real da rede para os SDIs. Esse fluxo de dados não vai alterar o funcionamento e o desempenho normal da rede.

A outra forma de analisar o SDI é efetuando testes com ataques conhecidos, selecionados para aplicação da metodologia no item 4.2.1.2. Dessa maneira, o tráfego de fundo não será utilizado, pois os testes serão feitos através da estação *Atacante Fictício* diretamente aos SDIs sob teste.

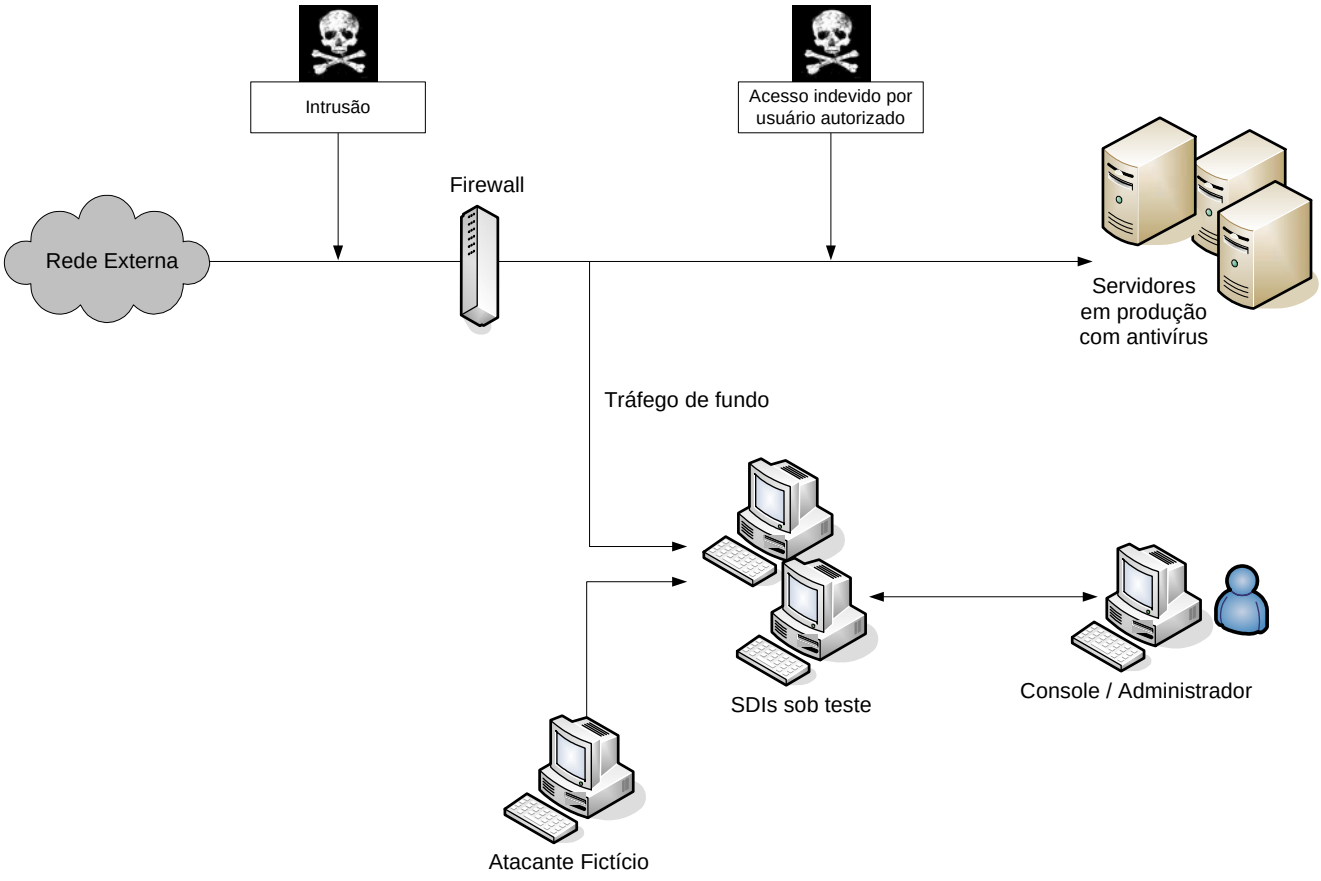


Figura 2 - Ambiente de rede do cenário de teste.

A Figura 2 representa o ambiente de teste que é composto pelo ambiente real de produção, que fornece o tráfego de fundo representado pela Rede Externa (*Internet*), pelos SDIs sob teste, que serão todos SDIs utilizados na aplicação da metodologia e que serão colocados em paralelo, para receber exatamente o mesmo fluxo de dados. Na estação *Console*, o administrador irá gerenciar todos os alertas produzidos pelos SDIs.

Na estação *Atacante Fictício*, serão instaladas as ferramentas que foram selecionadas no item 4.2.2.1 e ataques serão simulados, sem tráfego de fundo, para identificar o comportamento dos SDIs sob teste e avaliar seus resultados.

4.2.4 Seleção dos SDIs

A seleção dos SDIs a serem testados deve seguir os seguintes critérios:

- Disponibilidade - os SDIs escolhidos para análise devem estar disponíveis para avaliação;
- Adequabilidade - os SDIs devem adequar-se ao modelo conceitual ilustrado no cenário de teste, Figura 1; e
- Abrangência - o SDI deve propor-se a detectar todos os ataques determinados no item 4.2.1.2.

4.2.5 Análise dos SDIs

4.2.5.1 Identificação de parâmetros relevantes

4.2.5.1.1 Critérios quantitativos

- *Falsos Positivos*
  São alertas produzidos pelo sistema, classificados como intrusões, quando na verdade não são ações maliciosas.

- *Falsos Negativos*
  Ao contrário dos falsos positivos, eles definem intrusões reais como uma ação normal e sem perigo, enquanto deveriam ter sido detectadas pelo SDI como reais intrusões.

- *Capacidade do Sistema*
  São os recursos computacionais utilizados pelo SDI durante seu processamento.

4.2.5.1.2 Critérios qualitativos

- *Resistência do SDI à subversão*
  Esse item demonstra o quanto um SDI é resistente a uma tentativa de ataque que tente interromper sua correta operação.

- *Facilidade de atualização*
  Análise de como funciona o sistema de atualização de assinaturas ou anomalias dos SDIs.

- *Eficiência do Console em alertar o administrador de segurança.*
  Identificação, nesse item, de quais são as formas de alertar o administrador de segurança quando houver alarmes nos SDIs.

4.2.5.2 Métricas para os parâmetros

4.2.5.2.1 Quantitativos

- *Falsos Positivos*
  Há duas métricas para os falsos positivos. A primeira é a relação entre o número de falsos positivos produzidos pelos SDIs e a quantidade de pacotes analisados. É uma métrica de eficiência do método. A segunda é a relação entre o número de falsos positivos produzidos pelos SDIs e o total de eventos detectados por eles. É uma métrica da comodidade de uso do SDI.

- *Falsos Negativos*
  Há duas métricas para os falsos negativos. A primeira é a relação entre o número de falsos negativos produzidos pelos SDIs e a quantidade de pacotes analisados. A segunda é a relação entre o número de falsos negativos produzidos pelos SDIs e o total de eventos detectados por eles.

- *Capacidade do Sistema*
  Medições dos recursos computacionais são utilizados durante o funcionamento do SDI, verificando a utilização do processador, a memória principal e o espaço em disco - todos consumidos pelo SDI.

4.2.5.2.2 Qualitativos

- *Resistência do SDI à subversão*
  Analisa quando um intruso modifica a operação da ferramenta SDI para forçar a ocorrência de falso negativo.

- *Facilidade de atualização*
  Analisa como os SDIs são configurados para identificação de novos ataques e atualizações automáticas de assinaturas.

- *Eficiência do Console em alertar o administrador de segurança.*
  Testa o envio de alertas sobre ataques, através de recursos disponíveis nos consoles de gerenciamento, envio de e-mail, celular dos administradores.

4.2.6 Planejamento dos experimentos

Este item é composto por etapas que devem ser realizadas para aplicação da metodologia proposta, tais como: montagem do ambiente de teste, configurações dos sistemas operacionais e das ferramentas SDI.

É desejável que o período de teste seja suficiente para que todos os passos da metodologia sejam cumpridos com eficiência e precisão.

4.2.7 Realização dos experimentos

Consiste na realização das etapas programadas em 4.2.6, com eventuais correções determinadas pelos primeiros resultados obtidos.

A existência de ataque real e suas características devem ser informadas no período de teste.

4.2.8 Descrição dos resultados

A descrição dos resultados deve exibir as métricas descritas acima, bem como outras informações julgadas relevantes, como configurações e características do tráfego durante o teste.

O relatório final deve ser organizado para subsidiar a decisão da empresa que poderá levar em conta outros fatores.

5 ESTUDO DE CASO

Este estudo de caso tem como objetivo validar a metodologia criada. O ambiente utilizado para aplicação dos testes foi em um provedor brasileiro de Internet. Dois SDIs, Snort e Bro, foram utilizados, recebendo o espelhamento do tráfego real de e-mail e do tráfego real de alguns produtos da empresa.

5.1 Descrição dos SDIs para o Estudo de Caso.

Os SDIs selecionados para o estudo de caso serão descritos abaixo.

5.1.1 Snort

É um SDI de rede e de código-fonte aberto, tendo sido escolhido para aplicação do estudo de caso por também adequar-se à metodologia, ou seja, ser facilmente encontrado, aplicar-se ao modelo conceitual (Figura 1) e ser capaz de detectar diversos tipos de ataques.

O Snort é baseado em assinaturas. Desenvolvido por ROESCH (1998), possui um processo simples de instalação, sendo compatível com vários consoles de gerenciamento, permitindo ao administrador criar suas próprias regras.

É uma ferramenta que detecta vários tipos de ataques, como *buffer overflow*, varreduras de portas e *DoS* e realiza análise de tráfego e pacotes em tempo real, em redes IP. Pode ser utilizado em qualquer sistema *Unix* e, inclusive, em *Windows*, segundo CASWELL (2003).

Sua estrutura é simples, baseada na captura de pacotes de rede através da biblioteca *libpcap*. Os pacotes que coincidem com algumas regras podem ser descartados, armazenados ou podem gerar algum alerta aos responsáveis pelo sistema. Há ainda a possibilidade de utilizar regras de filtragem durante a coleta dos pacotes (*libpcap*), antes que eles passem pelo analisador, ou como são chamados, os pré-processadores e processadores de saída, responsáveis, respectivamente, por analisar os pacotes coletados antes que a base de assinaturas seja avaliada e por fazer a formatação dos resultados gerados.

Nesse estudo de caso, o Snort vai trabalhar juntamente com o console de detecção de intrusão *Aanval*, que terá a função de gerenciar os alertas recebidos por esse SDI. O Aanval é

composto por duas versões - a comercial e a gratuita. Será utilizada somente a segunda opção. O Aanval fornece monitoramento em tempo real e isso será importante para o acompanhamento dos alertas gerados pelo Snort; possui também outros recursos como relatórios, estatísticas diárias de endereços IPs e portas de origem e destino, suporte através de documentação *online*, fórum e grupo de discussão ou envio de mensagens através do próprio console.

Para utilização do Snort com o *Aanval*, foram necessárias a instalação e a configuração de outras ferramentas como: *Mysql, Apache, Php, Libpcap, JPgraph e Zlib*.

5.1.2 Bro

O outro SDI utilizado no estudo de caso foi o Bro, semelhante à ferramenta anterior em alguns aspectos. Trata-se de um SDI de rede, de código-fonte aberto e baseado em assinaturas, possuindo como diferencial o formato de sua base de ataques.

Foi escolhido para aplicação do estudo de caso por também adequar-se à metodologia, ou seja, pode ser facilmente encontrado, por aplicar-se ao modelo conceitual (Figura 1) e ser capaz de detectar diversos tipos de ataques.

Toda a análise é feita utilizando *scripts*, descritos em uma linguagem própria. Desenvolvido pelo LAWRENCE BERKELEY NATIONAL LABORATORY (2003), o SDI Bro pode ser utilizado em sistemas *Unix*, *FreeBSD*, *Solaris*, *SunOS* e *Linux*.

É dividido em dois componentes: um processador de eventos, responsável por reduzir um fluxo de pacotes já previamente filtrados, e um interpretador de *scripts*, responsável pelo processamento da linguagem de descrição de políticas.

Como pode ser visto, o Bro também utiliza a biblioteca *libpcap* para fazer a captura de pacotes. Filtros no formato *tcpdump* são aplicados a essa biblioteca para fazer o primeiro nível de redução de dados, agilizando o trabalho das camadas superiores.

Depois de capturados, os pacotes são encaminhados ao processador de eventos, que primeiro faz vários testes de integridade com o cabeçalho dos pacotes, descartando aqueles com problemas, e, após esses testes, processa-os na busca por eventos. Esse processamento inclui o tratamento de pacotes de conexão, como SYN, FIN ou RST, a manutenção do estado das conexões ativas bem como o tratamento de protocolos de nível mais alto.

O processamento desses pacotes gera eventos para a camada superior, informando o estabelecimento de conexões, a chegada de pacotes UDP endereçados a alguma máquina que já tenha recebido pacotes dessa natureza, considerado um *udp_request*, e outros eventos de nível mais alto.

Com esses eventos, o interpretador de *scripts*, escritos em uma linguagem especializada, aplica o código especificamente projetado para tratar de cada um desses eventos, buscando por ataques. Essa linguagem, sintaticamente semelhante à linguagem C, é usada para representar toda a base de assinaturas da ferramenta.

Também é possível a utilização de regras do Snort no SDI Bro. Essas regras serão ativadas em alguns testes exibidos ao longo desse capítulo.

5.2 Testes na rede de e-mail

Os testes a seguir foram aplicados na rede de e-mail do provedor brasileiro de Internet.

5.2.1 Seleção de tipos de ataques

Nesse item foram selecionados quais ataques são identificados pelos SDIs, e quais são detidos pelo *firewall* na rede de e-mail.

A Tabela 8 mostra que todos os ataques podem ser analisados pelos SDIs, porém, somente nas portas 25 e 110, pois se trata de uma rede de e-mail e essas são as únicas portas abertas para esses servidores. Os ataques em outras portas são detectados pelo *firewall*.

Ataques	SDI	Firewall
Varredura de portas abertas		
TCP *Connect*	x	
UDP	x	
	x	
Varredura de portas semi-abertas		
TCP SYN	x	
Varredura de portas ocultas		
TCP FIN	x	
TCP *Xmas*	x	
TCP *Null*	x	
Outras varreduras de portas		
TCP ACK	x	
TCP *Window*	x	
Varredura ICMP	x	
Varredura *Decoy*		
	x	
Buffer overflow	x	
Cross-site scripting	x	

Tabela 8 - Seleção de ataques na rede de e-mail.

A Tabela 8 é representada pelo indicador “x”, significando que todos os ataques poderão ser detectados pelos SDIs nas portas 25 e 110.

5.2.2 Ferramentas para ataques

As ferramentas utilizadas para os testes de simulação de ataques foram: Nmap e *Nessus*. Todos os ataques selecionados no item 5.2.1 foram aplicados nos dois SDIs e com isso foi feita uma análise para identificar quais foram detectados.

Para realização deste teste de detecção de ataques foi utilizada uma máquina com a função de “*Atacante Ficticio*”.

O *Nessus* também foi utlizado em testes de capacidade de sistema que serão exibidos no item 5.2.4.5.

5.2.3 Resultados da aplicação dos Ataques

O experimento foi realizado através de ataques diretos, sem tráfego de fundo. Os testes foram realizados uma única vez, nos casos em que não houve detecção o teste foi repetido para confirmação do resultado.

A Tabela 9 mostra todos os ataques aplicados nos dois SDIs, utilizando as ferramentas de ataques citadas no item 5.2.2. Esses ataques foram selecionados no item 5.2.1 para aplicação do estudo de caso.

Ataques	BRO	SNORT
Varredura de portas abertas		
TCP *Connect*	x	x
UDP	x	x
Varredura de portas semi-abertas		
TCP SYN	x	x
Varredura de portas ocultas		
TCP FIN	x	x
TCP *Xmas*	x	x
TCP *Null*	x	x
Outras varreduras de portas		
TCP ACK	x	x
TCP *Window*	x	x
Varredura ICMP	-	x
Varredura *Decoy*	x	x
Buffer overflow	x	x
Cross-site scripting	-	x

Tabela 9 - Simulação de ataques na rede de e-mail.

A Tabela 9 é representada por dois tipos de indicadores, sendo que "-" significa que o ataque não foi detectado e "x", que o ataque foi detectado.

do seu tempo de trabalho que foi utilizada para a realização do teste.O snort detectou todos os ataques, e o Bro falhou na varredura ICMP e no ataque *Cross-site scripting*.

5.2.4 Critérios quantitativos

Este teste determina o número de falsos positivos e negativos produzidos pelos SDIs. Os resultados são apresentados em função da quantidade de pacotes analisados e do total de eventos detectados pelos SDIs.

Para estes testes foi utilizado o tráfego real, sem simulação de ataques.

5.2.4.1 Falsos Positivos

Neste teste, determinou-se que os falsos positivos são os eventos detectados pelos SDIs que, através de uma análise posterior não foram classificados como ataques, tentativas de ataques ou varreduras de portas. Já os alertas são os eventos classificados como ataques, tentativas de ataques ou varreduras de portas.

O período da análise foi de 15 minutos, período julgado significativo e compatível com a capacidade de armazenamento dos pacotes para análise posterior. As medições foram feitas em horários de fluxo bastante elevado.

Para esta análise, foram ativadas as regras relacionadas a ataques de redes de e-mail. Foi feito um esforço para compatibilizar essas regras, para que a detecção fosse a mais próxima possível nos dois produtos, resultando em 17 regras para o Bro e 10 regras para o Snort.

Foram utilizadas duas máquinas de testes, uma para cada SDI e o mesmo fluxo de dados foi espelhado para cada uma. Para contagem de pacotes, foi utilizada a ferramenta *tcpdump* e para leitura, o *ethereal.*

A configuração de cada máquina foi composta por processador Pentium 4 - 2.4 GHz, memória de 2.0GB e disco rígido de 80GB de 7200 rpm.

Foi criado um *script* em *shell* que teve como função iniciar e finalizar a execução do *tcpdump* nas duas máquinas, utilizando o mesmo período, gravando arquivos com os pacotes. Isto foi feito para que os dois SDIs pudessem analisar o mesmo fluxo de dados. O horário das máquinas foi sincronizado com o servidor NTP (*Network Time Protocol*) da empresa, para que os testes fossem aplicados exatamente no mesmo horário. E o horário para iniciar e finalizar a execução do *script* foi configurado no *cron* de cada sistema operacional.

A análise dos eventos gerados pelos SDIs foi feita exatamente no período em que o fluxo de dados foi gravado pelo *tcpdump*. Isto foi necessário para que fosse possível calcular a taxa precisa de falsos positivos dos dois produtos.

5.2.4.2 Falsos Negativos

A rigor não foram detectados falsos negativos, uma vez que nenhum setor da empresa detectou a ocorrência de ataques no período de testes. Entretanto, ocorreram tentativas que deveriam ter sido detectadas. Assim, determinou-se que para este teste seriam considerados falsos negativos alertas não detectados pelos SDIs que, através de uma análise posterior, verificou-se que eram tentativas de ataques ou varreduras de portas. Esta análise foi realizada comparando os alertas gerados pelos dois SDIs, Bro e Snort, e com isso analisou-se quais alertas foram detectados em cada ferramenta.

Outros departamentos da empresa foram consultados para identificar suspeitas na rede e não foram detectadas outras tentativas de ataques ou ataques reais durante o período da análise.

O total de tentativas de ataques detectadas pelos SDIs na rede de e-mail foi de: 16.

5.2.4.3 Resultados

A tabela 10 mostra os resultados obtidos pelo teste de falsos positivos dos SDIs.

Análise de Falsos Positivos	BRO	SNORT
Eventos gerados pelos SDIs	382	67
Alertas gerados pelos SDIs	11	5
Pacotes capturados pelo *tcpdump* e analisados pelo *ethereal*	19.517.176	19.124.053
Taxa de Falsos Positivos (nº falsos positivos/total de eventos detectados pelos SDIs)	97%	92%
Taxa de Falsos Positivos (nº falsos positivos/total de pacotes capturados pelo *tcpdump*)	0,00190%	0,00032%

Tabela 10 - Análise de Falsos Positivos na rede de e-mail.

Através da Tabela 10, pode-se observar que o SDI Bro apresentou um maior número de eventos, que são as indicações de suspeitas indicadas pelos SDIs. Com isso, conclui-se que as regras utilizadas no Bro não foram muito eficientes na filtragem de pacotes à medida que foram configuradas de forma semelhante às regras do Snort.

Após análise, verificou-se uma perda de pacotes ocorrida durante a captura feita pelo *tcpdump*. Isto explica a pequena diferença no número total de pacotes.

A tabela 11 mostra os resultados obtidos pelo teste de falsos negativos dos SDIs.

Análise de Falsos Negativos	BRO	SNORT
Eventos gerados pelos SDIs	382	67
Alertas gerados pelos SDIs	11	5
Falsos Negativos	5	11
Pacotes capturados pelo *tcpdump* e analisados pelo *ethereal*	19.517.176	19.124.053
Taxa de Falsos Negativos (nº falsos negativos/total de eventos detectados pelos SDIs)	1%	16%
Taxa de Falsos Negativos (nº falsos negativos/total de pacotes capturados pelo *tcpdump*)	0,00003%	0,00006%

Tabela 11 - Análise de Falsos Negativos na rede de e-mail.

A tabela 11 mostra que o Snort apresentou uma taxa maior de falsos negativos. Com isso conclui-se que este SDI falhou na detecção de tentativas de ataques.

5.2.4.4 Capacidade do Sistema

Nesse item, foram analisados os recursos computacionais utilizados durante o funcionamento dos SDIs: utilização do processador, memória principal e espaço em disco consumidos pelos SDIs.

A Tabela 11 mostra a quantidade de recursos computacionais consumidos pelos SDIs durante a monitoração do tráfego real de e-mail.

Para a coleta dos dados, foi utilizado o comando *top* para cada processo (SDI) em cada máquina e com isso, obtiveram-se a taxa de consumo de processamento e memória de cada produto. Esses números foram coletados no minuto de fluxo mais alto dentro do período da análise.

O espaço em disco foi calculado pelo conteúdo dos seus diretórios binários e o total de regras ativas foram aquelas configuradas em cada SDI.

	Processador	**Memória**	**Espaço em disco**
BRO	8.4%	11.8%	19MB
SNORT	82.0%	2.2%	4.8MB

Tabela 12 - Capacidade do sistema durante o funcionamento dos SDIs.

A Tabela 12 mostra a taxa de consumo de processamento e memória dos SDIs. Pode-se concluir que o Bro teve uma taxa de consumo de processamento menor, apesar de utilizar mais regras. Sua taxa de consumo de memória foi mais alta. O Bro também ocupa mais espaço em disco do que o Snort.

Dado o volume de fluxo e configuração das máquinas utilizadas nos testes, conclui-se que a capacidade de sistema não é mais um fator relevante para escolha de um SDI.

5.2.4.5 Teste durante um ataque DoS simulado pela ferramenta de ataque Nessus.

Este teste analisou o consumo de processamento e memória durante a simulação de um ataque DoS. Nesse período, os SDIs estavam recebendo o tráfego real de e-mail e também o

tráfego simulado pela ferramenta de ataque *Nessus*. As condições de medição foram as mesmas, mas realizadas em outro período.

	Processador	Memória
BRO	6%	8%
SNORT	81%	2%

Tabela 13 - Simulação de ataque DoS.

Conclui-se que o ataque DoS aplicado nos SDIs não alterou, significamente, a taxa de consumo de processamento e memória.

5.2.4.6 Resultado Adicional: Bro com regras do Snort

Um teste adicional foi realizado somente com o SDI Bro, utilizando suas regras originais mais as regras do Snort.

O ambiente utilizado para realização deste teste foi outro e a configuração da máquina utilizada foi: processador Pentium 3 - 450 MHz, memória de 512 GB e disco rígido de 40GB de 7200 rpm. O tráfego analisado foi gerado pelo *tcpdump*, portanto, o teste foi realizado em modo *offl-line*.

Este teste foi aplicado para analisar a diferença de consumo de processamento e memória do SDI Bro e o estado de estabilidade (teste de "*ramp up*") comparado ao teste anterior, item 5.2.4.5.

Os períodos de análise foram de 15 minutos e o mesmo tráfego gravado foi utilizado para os dois casos.

	Processador	Memória
BRO sem regras do Snort	89.7%	72%
BRO com regras do Snort	90.1%	74%

Tabela 14 - Capacidade do Sistema utilizando Bro com regras do Snort

Conclui-se que não é devido às regras que as taxas não diferem entre o Snort e Bro. A diferença deve-se às características dos sistemas.

Cabe destacar que os índices acima não podem ser comparados diretamente com os anteriores, pois as condições dos experimentos são muito diferentes.

5.2.4.7 Gráfico comparativo do Bro com regras do Snort

O Gráfico 3 compara o consumo do Bro com regras do Snort ativadas e desativadas. O teste foi realizado no modo *off-line*, ou seja, o tráfego analisado foi gerado pelo *tcpdump.*

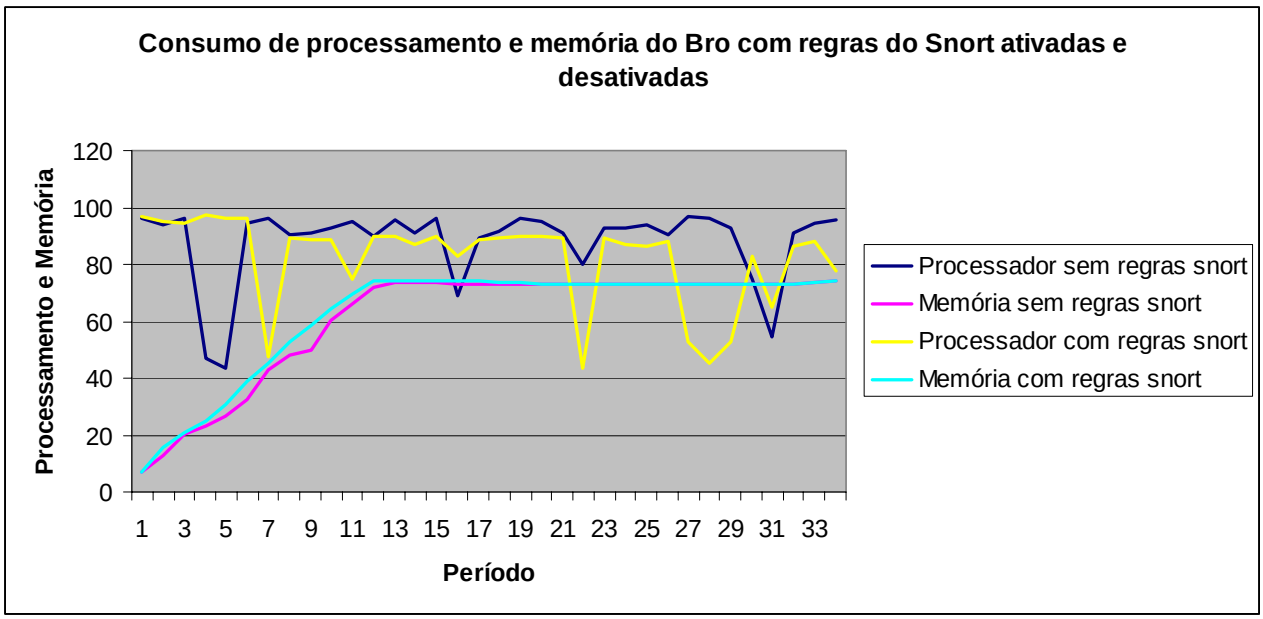


Gráfico 3 - Comparação de consumo de processamento e memória do BRO com regras do Snort.

O Gráfico 3 apresenta uma análise da taxa de consumo de processamento e memória dos Bro com regras do Snort ativadas e desativadas. Esse teste foi aplicado em modo *off-line* no período de 33 minutos. Com isso, conclui-se que a taxa de consumo não foi influenciada pelas regras do SDI e a taxa de consumo de memória atingiu estabilidade.

O tempo de 12 minutos para estabilização ocorreu na máquina de teste, de capacidade inferior. No teste real, este tempo deve ser muito menor.

5.2.5 Critérios qualitativos

5.2.5.1 Resistência do SDI à subversão

Este item demonstra o quanto um SDI é resistente a uma tentativa de ataque que tente interromper sua correta operação. Analisa quando um intruso modifica a operação da ferramenta SDI para forçar a ocorrência de falso negativo.

Os ataques foram aplicados para analisar o comportamento dos SDIs.

Para este teste, foram utilizadas ferramentas para ataques DoS como: *Jolt2, Synk4*, *Targa2* e *Nessus*.

Ferramentas de Ataque	BRO	SNORT
Synk4	x	x
Targa2	x	x
Jolt2	x	x
Nessus	x	x

Tabela 15 - Teste de resistência dos SDIs à subsversão.

A Tabela 15 é representada pelo indicador “x”, significando que todas as ferramentas foram utilizadas e o desempenho dos SDIs não foi alterado em nenhum caso.

Após a execução do teste mostrado nessa tabela, foi possível identificar que os dois SDIs resistiram aos ataques.

Não foi executado nenhum teste tipo ataque de Negação de Serviço Distribuída - *(Distributed Denial of Service – DDoS)*, pela indisponibilidade de máquinas.

5.2.5.2 Facilidade de atualização

Este item analisa como funciona o sistema de atualização de assinaturas dos SDIs.

O Bro ainda não possui uma forma de atualização automática. As atualizações são feitas somente com a atualização do produto, disponibilizada na página do fabricante. O objetivo do produto é atualizar suas assinaturas através da execução de um *script*, chamado

update-sigs, e que, através do comando *wget*, fará suas atualizações. Desta forma, orienta-se que o Bro seja reiniciado para carregar as novas assinaturas.

O processo de aviso de atualizações de assinaturas do *Aanval* funcionou corretamente, tendo enviado um e-mail de acordo com o intervalo configurado no console, avisando se possui ou não uma atualização do Snort. Para isso, o servidor precisa ter acesso à Internet para que essa opção funcione corretamente, pois ainda não é possível adquirir as atualizações de forma manual.

5.2.5.3 Eficiência do SDI/Console em alertar o administrador de segurança.

Este item identifica quais são as formas de alertar o administrador de segurança quando houver alarmes nos SDIs.

Esse item não funcionou corretamente no Bro, porque alguns *scripts* ainda estão em desenvolvimento.

No console *Aanval*, podem-se filtrar tipos de alertas recebidos pelo Snort e definir ações como: alertas por e-mail, alertas por áudio e execução de comandos, dependendo do número mínimo de alertas configurados no console.

5.3 Testes complementares na rede de produtos

Outros testes foram efetuados em outra rede que envolve alguns produtos do provedor brasileiro de Internet.

Foi realizado com tráfego real que, apesar de pequeno, é importante para o provedor.

O espelhamento desse tráfego foi feito para os dois SDIs.

As condições foram as mesmas do teste anterior, exceto pelo que segue:

5.3.1 Seleção de tipos de ataques

Nesse item, foram selecionados quais ataques são identificados pelos SDIs e quais são detidos pelo *firewall* na rede de e-mail.

A Tabela 15 mostra que todos os ataques poderão ser analisados pelos SDIs, porém, somente nas portas 80 e 443, visto tratar-se de uma rede com servidores *web* e essas serem as únicas portas abertas. Os ataques em outras portas serão detectados pelo *firewall*.

Ataques	SDI	Firewall
Varredura de portas abertas		
TCP *Connect*	x	
UDP	x	
	x	
Varredura de portas semi-abertas		
TCP SYN	x	
Varredura de portas ocultas		
TCP FIN	x	
TCP *Xmas*	x	
TCP *Null*	x	
Outras varreduras de portas		
TCP ACK	x	
TCP *Window*	x	
Varredura ICMP	x	
Varredura *Decoy*	x	
Buffer overflow	x	
Cross-site scripting	x	

Tabela 16 - Seleção de ataques na rede de produtos.

A Tabela 16, representada pelo indicador “x”, significa que todos os ataques poderão ser detectados pelos SDIs nas portas 80 e 443.

5.3.2 Ferramentas para ataques

As ferramentas utilizadas para os testes de simulação de ataques foram: *Nmap* e *Nessus*. Todos os ataques selecionados no item 5.3.1 foram aplicados nos dois SDIs e com isso foi feita uma análise para identificar quais foram detectados.

Para realização deste teste de detecção de ataques, foi utilizada uma máquina com a função de "*Atacante Ficticio*".

5.3.3 Resultados da aplicação dos Ataques

O experimento foi realizado através de ataques diretos, sem tráfego de fundo. Os testes foram realizados uma única vez, nos casos em que não houve detecção o teste foi repetido para confirmação do resultado.

A Tabela 17 mostra todos os ataques aplicados nos dois SDIs, utilizando as ferramentas de ataques citadas no item 5.3.2. Estes ataques foram selecionados no item 5.3.1 para aplicação do estudo de caso.

Ataques	BRO	SNORT
Varredura de portas abertas		
TCP *Connect*	x	x
UDP	-	x
Varredura de portas semi-abertas		
TCP SYN	x	x
Varredura de portas ocultas		
TCP FIN	x	x
TCP *Xmas*	x	x
TCP *Null*	x	x
Outras varreduras de portas		
TCP ACK	x	x
TCP *Window*	x	x
Varredura ICMP	-	-
Varredura *Decoy*	x	-
Buffer overflow	x	x
Cross-site scripting	x	x

Tabela 17 - Simulação de ataques na rede de produtos.

A Tabela 17 é representada por dois tipos de indicadores, sendo que "-" significa que o ataque não foi detectado e "x", que o ataque foi detectado.

O Snort não detectou as varreduras UPD e ICMP, e o Bro não detectou as varreduras ICMP e *Decoy*.

5.3.4 Critérios quantitativos

5.3.4.1 Falsos Positivos

Neste teste, determinou-se que os falsos positivos são os eventos detectados pelos SDIs que, através de uma análise posterior não foram classificados como ataques, tentativas de ataques ou varreduras de portas. Já os alertas são os eventos classificados como ataques, tentativas de ataques ou varreduras de portas.

O período da análise foi de 30 minutos, período julgado significativo e compatível com a capacidade de armazenamento dos pacotes para análise posterior. As medições foram feitas em horários de fluxo bastante elevado.

Para esta análise, foram ativadas as regras relacionadas a ataques de redes com servidores *web*. Foi feito um esforço para compatibilizar essas regras para que a detecção fosse a mais próxima possível nos dois produtos, resultando em 17 regras para o Bro e 14 regras para o Snort.

A aplicação deste teste foi realizada da mesma forma que na rede de e-mail.

5.3.4.2 Falsos Negativos

Os critérios de análise foram os mesmos utilizados para o caso da rede de e-mail.

O total de tentativas de ataques detectadas pelos SDIs na rede de e-mail foi de: 150.

5.3.4.3 Resultados

A tabela 18 mostra os resultados obtidos pelo teste de falsos positivos dos SDIs.

Análise de Falsos Positivos	BRO	SNORT
Eventos gerados pelos SDIs	475	44
Alertas gerados pelos SDIs	133	23
Pacotes capturados pelo *tcpdump* e analisados pelo ethereal	580.504	581.495
Taxa de Falsos Positivos (nº falsos positivos/total de eventos detectados pelos SDIs)	72%	47%
Taxa de Falsos Positivos (nº falsos positivos/total de pacotes capturados pelo *tcpdump*)	0,05891%	0,00361%

Tabela 18 - Análise de Falsos Positivos na rede de produtos.

O perfil deste resultado é um pouco diferente comparado ao teste na rede de e-mail, porque há mais ataques nessa rede. De qualquer forma, o Bro gerou mais eventos e também resultou em uma taxa maior de falsos positivos do que o Snort.

A tabela 19 mostra os resultados obtidos pelo teste de falsos negativos dos SDIs.

Análise de Falsos Negativos	BRO	SNORT
Eventos gerados pelos SDIs	475	44
Alertas gerados pelos SDIs	133	23
Falsos Negativos	17	127
Pacotes capturados pelo *tcpdump* e analisados pelo ethereal	580.504	581.495
Taxa de Falsos Negativos (nº falsos negativos/total de eventos detectados pelos SDIs)	4%	289%
Taxa de Falsos Negativos (nº falsos negativos/total de pacotes capturados pelo *tcpdump*)	0,00293%	0,02184%

Tabela 19 - Análise de Falsos Negativos na rede de produtos.

A tabela 19 mostra que na rede de produtos o Snort apresentou uma taxa maior de falsos negativos. Com isso conclui-se que este SDI falhou por não detectar algumas tentativas de ataques.

5.3.4.4 Capacidade do Sistema

Neste item, foram analisados os recursos computacionais utilizados durante o funcionamento dos SDIs: utilização do processador, memória principal e espaço em disco consumidos pelos SDIs.

A Tabela 20 mostra a quantidade de recursos computacionais consumidos pelos SDIs durante a monitoração do tráfego real de produtos. Esses números foram coletados no momento de fluxo mais alto dentro do período da análise.

Para a coleta dos dados, foi utilizado o comando *top* para cada processo (SDI) em cada máquina e com isso obtiveram-se a taxa de consumo de processamento e memória de cada produto. Esses números foram coletados no minuto de fluxo mais alto dentro do período da análise.

O espaço em disco foi calculado pelo conteúdo dos seus diretórios binários e o total de regras ativas foram aquelas configuradas em cada SDI.

	Processador	**Memória**	**Espaço em disco**
BRO	0.0%	2.3%	19MB
SNORT	0.3%	2.7%	4.8MB

Tabela 20 - Capacidade do sistema durante o funcionamento dos SDIs com o tráfego real de produtos.

Este resultado não é muito significativo devido ao pequeno tráfego.

5.3.5 Critérios qualitativos

Todos os testes aplicados em critérios qualitativos, como resistência do SDI à subversão, facilidade de atualização e eficiência do SDI/Console em alertar o administrador de segurança, foram iguais aos testes efetuados na rede de e-mail.

5.4 Versões utilizadas

As versões de todas as ferramentas utilizadas nas máquinas no período de teste dos SDIs serão descritas abaixo:

- *Aanval*: 1.55
- *Apache*: 2.0.52
- Bro: 0.9a9
- *JPGraph*: 1.8.4-11
- *Libpcap*: 0.8.3
- *Mysql*: 4.0.24-0
- *Php*: 4.3.9
- Snort: 2.3.2
- *Tcpdump*: 3.8.3
- *Zlib*: 1.1.4-8

6 CONCLUSÃO

A utilização de uma metodologia é imprescindível para garantir o bom resultado na escolha de uma ferramenta SDI. Embora ainda possuam falhas, os SDIs são considerados instrumentos importantes para a garantia da segurança de aplicações em redes e, como os *firewalls*, devem também sofrer uma grande evolução nos próximos anos.

Uma metodologia bem definida e com procedimentos práticos pode facilitar, em grande escala, o trabalho de um administrador de segurança dentro de uma organização.

No decorrer do trabalho, foi desenvolvida uma metodologia que difere das já propostas por ALESSANDRI (2000), (2001), LIPPMANN et al. (2000) e PUKETZA et al. (1997). Esta metodologia permite que uma ferramenta SDI seja avaliada em um curto período de tempo através da utilização de uma seqüência de procedimentos - seleção de tipos de ataques e de ferramentas de ataque, especificação de um modelo para o teste e seleção e análise dos SDIs. Com base nos resultados, essa metodologia pode ser aplicada em qualquer organização, de forma prática, com uso simples e requerendo pouco tempo de execução.

A metodologia foi avaliada através de um estudo de caso, em um provedor brasileiro de Internet de grande porte, que comparou os SDIs Snort e Bro e mostrou-se viável e adequada a ser utilizada em comparação de SDIs em ambiente empresarial.

Os requisitos básicos que levaram a esta proposta de metodologia foram a viabilidade, que exigia um método simples e aplicável com um volume moderado de recursos, e capaz de produzir resultados adequados a um processo de decisão. A viabilidade pôde ser verificada através da sua aplicação em um curto prazo. O processo completo, envolvendo a implantação do ambiente de teste, a aplicação do estudo de caso, a análise e descrição dos resultados, foi realizado durante cinco meses sem comprometer o funcionamento normal da empresa e com poucos recursos em equipamentos e com uma pessoa dedicando 70 % de seu tempo ao projeto.

A metodologia foi considerada adequada por fornecer uma comparação significativa, mostrando que os SDIs têm propriedades bastante diferentes. Essas diferenças fornecem subsídios para uma decisão superior, juntamente com outras informações julgadas relevantes, como preço, compatibilidade e recursos computacionais.

Referências Bibliográficas

ALESSANDRI, Dominique. **Using rule-based activity descriptions to evaluate intrusion-detection systems.** Switzerland: IBM Research Laboratory Zurich, out. 2000.

ALESSANDRI, Dominique. **Using rule-based activity descriptions to evaluate intrusion-detection** systems. RAID Symposium, set. 2001.

BACE, Rebecca; MELL, Peter. **Intrusion Detection Systems**. Scotts Valley, California: National Institute of Standard and Technology, 2001.

CASWELL, Brian. et al. **Snort 2 - Sistema de Detecção de Intruso – Open Source.** Rio de Janeiro: Editora Alta Books, 2003.

CERT.br - CENTRO DE ESTUDOS, RESPOSTA E TRATAMENTO DE INCIDENTES DE SEGURANÇA NO BRASIL. **Cartilha de Segurança para Internet.** São Paulo, mar. 2003. Disponível em: < http://www.cert.br/docs/cartilha/cartilha-01-conceitos.html>. Acesso em: 24 jun. 2003.

CERT.br - CENTRO DE ESTUDOS, RESPOSTA E TRATAMENTO DE INCIDENTES DE SEGURANÇA NO BRASIL. **Estatísticas dos Incidentes Reportados ao CERT.br.** São Paulo, 2005.
Disponível em: <http://www.cert.br/stats/incidentes/>. Acesso em: 20 ago. 2003.

CERT - COMPUTER EMERGENCY RESPONSE TEAM. **CERT/CC Statistics 1988-2005**. Pittsburgh, Pennsylvania: Carnegie Mellon University, ago. 2005.
Disponível em: <http://www.cert.org/stats/>. Acesso em: 15 ago. 2003.

CERT - COMPUTER EMERGENCY RESPONSE TEAM. **Denial of Service Attacks.** Pittsburgh, Pennsylvania: Carnegie Mellon University, 2001.
Disponível em: <http://www.cert.org/tech_tips/denial_of_service.html>. Acesso em: 03 abr. 2005.

CERT - COMPUTER EMERGENCY RESPONSE TEAM. **Overview of Attack Trends.** Pittsburgh, Pennsylvania: Carnegie Mellon University, 2002.
Disponível em: <http://www.cert.org/archive/pdf/attack_trends.pdf>. Acesso em: 27 mar. 2005.

CERT - COMPUTER EMERGENCY RESPONSE TEAM. **Security of the Internet.** Pittsburgh, Pennsylvania: Carnegie Mellon University, 1997.
Disponível em: <http://www.cert.org/encyc_article/tocencyc.html#Denial>. Acesso em: 26 mar. 2005.

CHAMBERS, Chris; DOLSKE, Justin; IYER, Jayaraman. **TCP/IP Security.** Department of Computer and Information Science. Columbus, Ohio: Ohio State University.
Disponível em:
<http://www.linuxsecurity.com/resource_files/documentation/tcpip-security.html>. Acesso em: 11 abr. 2005.

CISCO Systems. **Why You Need a Firewall.** CISCO Systems Documentation. Estados Unidos, 2002. Disponível em:
<http://www.cisco.com/univercd/cc/td/doc/product/iaabu/centri4/user/scf4ch2.htm>. Acesso em: 31 mar. 2005.

COWAN, Crispin. et. al. **Buffer Overflows:** Attacks and Defenses for the Vulnerability of the Decade. Hilton Head Island SC, jan. 2000.
Disponível em: <http://downloads.securityfocus.com/library/discex00.pdf> Acesso em: 09 abr. 2005.

JOHN, McHugh; CHRISTIE, Alan; ALLEN, Julia. **Defending Yourself:** The Role of Intrusion Detection Systems. Software, IEEE. v. 17, 5ª edição, set.-out. 2000.

LAWRENCE BERKELEY NATIONAL LABORATORY. **Bro Intrusion Detection System**. University of California. 2003.
Disponível em: < http://www.bro-ids.org/>. Acesso em: 30 mar. 2005.

LIPPMANN, Richard P. et al. **Evaluating intrusion detection systems:** The 1999 DARPA offline intrusion detection evaluation. Lexington, Massachusetts: Lincoln Laboratory, Massachusetts Institute of Techonology, 2000.

MANCINI, Alexis. **Nmap network security scanner man page.** California, 1997. Disponível em: <http://www.insecure.org/nmap/data/nmap_manpage.html>. Acesso em: 25 mar. 2005.

MELCHIORI Consultoria. **Tipos de Ataques**. São Paulo, [200-]. Disponível em: <http://www.melchiori.com.br/ataques.asp>. Acesso em: 06 mar. 2005.

MICROSOFT Corporation. **Design de firewalls de perímetro**. São Paulo: Centro de orientações de segurança, abr. 2004. Disponível em: <http://www.microsoft.com/brasil/security/guidance/prodtech/win2000/secmod156.mspx#ETAA>. Acesso em: 30 mar. 2005.

MUKHERJEE, Biswanath; HEBERLEIN, L. Todd; LEVITT, Karl N. **Network Intrusion Detection**. IEEE Network, maio-jun.1994.

NED, FRANK. **Ferramentas de IDS**. Rede Nacional de Ensino e Pesquisa. São Paulo, v. 3, n. 5, set. 1999.

PUKETZA, Nicholas. et al. **A software plataform for testing intrusion detection systems**. Davis: University of California., 1997.

ROESCH, Martin. **Snort**. 1998. Disponível em: <http://www.snort.org>. Acesso em: 15 ago. 2003.

SANS Institute. **The Twenty Most Critical Internet Security Vulnerabilities.** Estados Unidos, 2004. Disponível em: <http://www.sans.org/top20>. Acesso em: 24 mar. 2005.

SIMON, Istvan. **Comparative Analysis of Methods of Defense against Buffer Overflow Attacks.** Hayward: California State University, jan.2001.

SUNDARAM, Aurobindo. **An Introduction to Intrusion Detection**. Crossroads: The ACM Student Magazine, v. 2, 4ª edição, abr.1996.

TROWBRIDGE, Chris. **An Overview of Remote Operating System Fingerprinting.** Estados Unidos: SANS Institute, jul. 2003.
Disponível em: <http://www.sans.org/rr/whitepapers/testing/1231.php>. Acesso em: 03 abr. 2005.

Bibliografias Consultadas

AANVAL. **Intrusion Detection Console.**
Disponível em: <http://www.aanval.com/> Acesso em: 10 set. 2004.

ANDERSON, James P. **Computer Security Threat Monitoring and Surveillance,** James P. Anderson Co., Fort Washington, Pennsylvania, 1980.

BECK, Rob. Passive-Aggressive Resistance: OS Fingerprint Evasion. **Linux Journal,** Seattle Washington, 2001.
Disponível em: <http://www.linuxjournal.com/article/4750>. Acesso em:17 abr. 2005.

BIELEWICZ, Asha. **Intrusion Detection Systems: Securing a Network Environment.** Canada, 2001.
Disponível em: <http://www.cas.mcmaster.ca/~wmfarmer/SE-4C03-01/papers/Bielewicz-IDS.pdf>. Acesso em: 24 mar. 2005.

CERT - COMPUTER EMERGENCY RESPONSE TEAM. **CERT 2003 Annual Report.** Pittsburgh, Pennsylvania: Carnegie Mellon University, 2003.
Disponível em: <http://www.cert.org/annual_rpts/cert_rpt_03.html#intruder>. Acesso em: 27 mar. 2005.

CERT - COMPUTER EMERGENCY RESPONSE TEAM. **Smurf IP Denial-of-Service Attacks**. Pittsburgh, Pennsylvania: Carnegie Mellon University, 2000.
Disponível em: <http://www.cert.org/advisories/CA-1998-01.html>. Acesso em: 26 mar. 2005.

HUEGEN, Craig A. **The latest in denial of service attacks:** "smurfing" description and information to minimize effects. [S. l.], 2000.
Disponível em: <http://www.pentics.net/denial-of-service/white-papers/smurf.cgi>. Acesso em: 26 mar. 2005.

NAKAMURA, Emilio Tissato. Caminhos Para a Segurança da Informação. **DEVELOPERS' Magazine,** Rio de Janeiro**,** jul. 2003.
Disponível em: <http://www.las.ic.unicamp.br/srac/developers.html>. Acesso em: 15 fev. 2005.

NATIONAL Infrastructure Protection Center. **Swarming Attacks**: Infrastructure Attacks for Destruction and Disruption. Washington, jul. 2002.
Disponível em: <http://www.breakwatersecurity.com/resources/swarming-july2002.pdf>. Acesso em: 12 mar. 2005.

PTACEK, Thomas H.; NEWSHAM, Timothy N. **Insertion, Evasion, and deny of service:** Eluding network intrusion detection. Canada: Secure Networks, Inc., 1998.
Disponível em: <http://downloads.securityfocus.com/library/ids.ps>. Acesso em: 06 mar. 2005.

ROCHA, Cláudio. Nmap/NmapNT. **INFORMA BR**. [S.l.], 2002.
Disponível em: <http://www.informabr.com.br/port_map.htm>. Acesso em: 26 mar. 2005.

THING, Lowell. The WHATIS?COM Encyclopedia of Tecnology Terms. **Dicionário de Tecnologia**. São Paulo: Siciliano S.A./Editora Futura. 2003.

Glossário

Backbone

Uma parte de uma rede que se conecta com outras redes, cuidando do tráfego principal.

Buffer

É uma área da memória usada para o armazenamento provisório dos dados.

Exploit

É um ataque em um sistema de computador, especialmente quando aproveita de uma vulnerabilidade específica que o sistema oferece aos intrusos.

Cache

Local de armazenamento que contém dados em que o computador precisará usar em curto tempo ou usa com mais freqüência.

Cron

É um programa de agendamento de tarefas dos sistemas operacionais *Unix*. Permite programar qualquer atividade para ser executada em um determinado horário ou dia.

Ethereal

Ferramenta utilizada para análise de protocolos de rede.

Firewall

Em redes de computadores, *firewalls* são barreiras interpostas entre a rede privada e a rede externa com a finalidade de evitar os intrusos; ou seja, são dispositivos de segurança que protegem os recursos de hardware e software da empresa contra ameaças que o sistema está exposto. Estes mecanismos de segurança são baseados em *hardware* e *software* e seguem a política de segurança estabelecida pela empresa.

Handshake

Método de sinalização usado entre computadores para indicar a disponibilidade de envio e recebimento de dados.

Host

Um computador em rede dedicado a fornecer um determinado tipo do serviço.

IDS

Intrusion Detection System. Um programa, ou um conjunto de programas, cuja função é detectar atividades incorretas, maliciosas ou anômalas.

IPS

Intrusion Prevention System. Funciona para identificar ataques antes que eles atinjam sistemas, servidores, ou seja, previne invasões.

Kernel

É a parte fundamental de um sistema operacional sobre o qual o resto do sistema operacional é baseado.

Libpcap

Biblioteca para captura de pacotes.

Payload

São os dados dos pacotes trafegados pela rede.

Ping

É um comando utilizado para verificar se o protocolo TCP/IP está corretamente instalado em um computador e se está funcionando. Se o TCP/IP estiver correto, o usuário receberá uma resposta, no entanto, é utilizado para verificar se um computador, ou servidor está ligado. Utilizado também para ataques de varreduras.

Scan

Significa varrer, analisar todo um disco, um arquivo, uma rede ou uma imagem para a execução de uma determinada tarefa.

Script

Uma série de comandos, armazenados em um arquivo para a execução subseqüente ou repetitiva.

Shell

Uma camada externa que fornece uma interface ao usuário ou uma maneira de comandar o computador. Esse termo aplica-se tipicamente aos sistemas operacionais.

Spam

É o envio de mensagens não solicitadas, em grande quantidade, a destinatários desconhecidos.

Sniffer

É um programa que captura todos os pacotes que trafegam na rede para que possam ser analisados.

RFC

As *RFCs (Requests for Comments)* são documentos com anotações técnicas e organizacionais sobre a Internet (originalmente o ARPANET), iniciado em 1969. Discutem muitos aspectos de redes de computadores, incluindo protocolos, procedimentos, programas e conceitos, assim como notas de reuniões e opiniões.

Tcpdump

É uma ferramenta para monitoramento de tráfego de redes.

Trojan

Um Cavalo de Tróia (*Trojan Horse*) é um programa que executa funções normalmente maliciosas e sem o conhecimento do usuário. Por definição, distingue-se de vírus e *worm*, por não se replicar, infectar outros arquivos, ou propagar cópias de si mesmo

automaticamente. Normalmente um cavalo de tróia consiste de um único arquivo que necessita ser explicitamente executado.

Wireless

Redes sem fio.

Worm

É um programa capaz de se propagar automaticamente através de redes, enviando cópias de si mesmo de computador para computador. Diferente do vírus, o *worm* não necessita ser explicitamente executado para se propagar.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo