UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

Técnicas de Detecção de Sniffers

por

ROGÉRIO ANTÔNIO CASAGRANDE

Dissertação submetida à avaliação, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. Raul Fernando Weber
Orientador

Porto Alegre, outubro de 2003.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Casagrande, Rogério Antônio
Técnicas de Detecção de Sniffers / por Rogério Antônio Casagrande - Porto Alegre: PPGC da UFRGS, 2003.
59 f.: il.

Dissertação(mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação Computação, Porto Alegre, BR-RS, 2003. Orientador: Weber, Raul F.
1. Sistemas Computacionais; 2. Segurança de Sistemas Computacionais. I. Weber, Raul Fernando. II. Título.

Agradecimentos

À Deus;
À Universidade Federal do Rio Grande do Sul – UFRGS, pelo curso;
Ao Instituto de Informática da UFRGS, seus professores e funcionários;
À Universidade do Extremo Sul Catarinense – UNESC, pela oportunidade e apoio;
À Pró-Reitoria de Pós-Graduação da UNESC, por todo empenho e dedicação;
Aos amigos, colegas de curso, da UNESC, pelo companheirismo;
Ao Departamento de Ciência da Computação da UNESC, professores e funcionários;
Ao Orientador, Prof. Raul Fernando Weber pela sua colaboração;
À Adriana e Giulia, pelo amor e carinho e por compreenderem minha ausência;
Ao meu pai, Antônio, meus irmãos e em memória de minha mãe Carmela.

Muito obrigado.

"Não basta ensinar ao homem uma especialidade, porque se tornará assim uma máquina utilizável, mas não uma personalidade. É necessário que adquira um sentimento, um senso prático daquilo que vale a pena ser empreendido, daquilo que é belo, que é moralmente correto. A não ser assim, ele se assemelhará, com seus conhecimentos profissionais, mais a um cão ensinado do que uma criatura harmoniosamente desenvolvida. Deve aprender a compreender as motivações dos homens, suas quimeras e suas angústias, para determinar com exatidão, seu lugar preciso em relação a seus próximos e à comunidade"

Albert Einstein

Sumário

Lista de Figuras

Lista de Tabelas

Resumo

A área de Detecção de Intrusão, apesar de muito pesquisada, não responde a alguns problemas reais como níveis de ataques, dim ensão e complexidade de redes, tolerância a falhas, autenticação e privacidade, interoperabilidade e padronização. Uma pesquisa no Instituto de Informática da UFRGS, mais especificamente no Grupo de Segurança (GSEG), visa desenvolver um Sistema de Detecção de Intrusão Distribuído e com características de tolerância a falhas. Este projeto, denominado *Asgaard*, é a idealização de um sistema cujo objetivo não se restringe apenas a ser mais uma ferramenta de Detecção de Intrusão, mas uma plataforma que possibilite agregar novos módulos e técnicas, sendo um avanço em relação a outros Sistemas de Detecção atualmente em desenvolvimento.
Um tópico ainda não abordado neste projeto seria a detecção de *sniffers* na rede, vindo a ser uma forma de prevenir que um ataque prossiga em outras estações ou redes interconectadas, desde que um intruso normalmente instala um *sniffer* após um ataque bem sucedido. Este trabalho discute as técnicas de detecção de *sniffers*, seus cenários, bem como avalia o uso destas técnicas em uma rede local. As técnicas conhecidas são testadas em um ambiente com diferentes sistemas operacionais, como *linux* e *windows*, mapeando os resultados sobre a eficiência das mesmas em condições diversas.

Palavras-Chave: segurança, detecção de intrusão, sniffer, detecção de sniffer

TITLE: “SNIFFER DETECTION TECHNIQUES”

Abstract

The area of Intrusion Detection, although widely searched, does not answer to some real problems as the level of attacks, dimension and complexity of networks, fault tolerance, authentication and privacy, interoperability and standardization. A current research at the Institute of Computer Science of UFRGS, more specifically in the Security Group (GSEG), aims at developing a Distributed Intrusion Detection System with features of fault tolerance. This project, called Asgaard, is the accomplishment of a system whose objective is not only restricted to be another tool concerning Intrusion Detection, but also a platform that makes possible to add new modules and techniques, which is an advance with respect to other Intrusion Detection Systems in progress. A point which has not yet been investigated in this project would be the network sniffer detection, which is supposed to be a way to prevent that an attack proceeds to other hosts and interconnected networks, once a intruder usually installs a sniffer after a well-performed attack. This work explores the sniffers detection techniques, their sets, as well as verifies these techniques in a local area network. The known techniques are tested in an environment with different operating systems, as linux and windows, explaining the results on the efficiency of these systems in several conditions.

Keywords: security, intrusion detection, sniffer, sniffer detection

1 Introdução

Nos últimos anos, podemos acompanhar uma explosão nas tecnologias relacionadas com a *Internet*. As empresas e organizações no mundo todo estão utilizando as tecnologias da *Internet* de uma forma ou de outra. Os principais gigantes do *software* orientaram suas linhas de produto em torno do paradigma da conexão à *Internet*. Centenas de companhias se formaram oferecendo ferramentas e serviços para conectar organizações com a *Internet*. O mais importante, entretanto, é que como muitas companhias estão experimentando o uso dessas tecnologias e se conectando à *Internet*, a questão da segurança tornou-se evidente. No fundo, a preocupação principal atualmente para as empresas conectadas à grande rede é com o risco da perda dos dados ou violabilidade de seus sistemas computacionais. Com o aumento dos ataques às redes internas, tornou-se necessária a implementação de mecanismos de segurança não somente corretivos na eminência de um ataque, mas também mecanismos preventivos. Atualmente se discute e se pesquisa muito sobre Sistemas de Detecção de Intrusão – IDS (*Intrusion Detection Systems*) cujo objetivo é reagir a uma invasão ou suspeita de invasão no menor intervalo de tempo possível [BAC 2001]. Vários produtos comerciais estão disponíveis, alguns centralizados, outros distribuídos. A maior parte deles baseados em rede, ou seja, analisando o tráfego na rede à procura de indícios de tentativas de intrusão, procurando por assinaturas de ataques conhecidos. Como parte de um sistema de detecção pode-se agregar a detecção de monitores de tráfego normalmente chamados de *sniffers*[1,2], originalmente criados para auxiliar o administrador de rede a verificar o funcionamento e comportamento da rede. Após um ataque bem sucedido, onde um sistema foi comprometido, um atacante geralmente instala uma ferramenta de monitoração de tráfego de rede, visando ampliar o escopo do ataque, comprometendo assim outros sistemas ou redes interconectadas.

Atualmente os *sniffers* são ferramentas muito utilizadas em um ataque à uma rede, pois têm a habilidade de capturar qualquer informação em modo texto circulando na mesma, como nomes de usuários, senhas, além de dados dos usuários. Detectar estas atividades pode ser de grande valia na administração de uma rede, descobrindo falhas e evitando a propagação de um ataque.

Basicamente um *sniffer* é um elemento passivo na rede, ou seja, somente captura pacotes, sendo então suas atividades não tão simples de detectar [GRA 2000]. Existem algumas formas de detectar *sniffers* executando em uma rede. Um administrador pode por exemplo, observar uma determinada estação e verificar comportamentos anormais,

[1] Sniffer é marca registrada da Network Associates e refere-se ao produto "Sniffer Network Analyzer" .
[2] Popularmente conhecidos como Sniffer, são os analisadores de protocolos ou analisadores de rede.

tais como, excessiva carga de processamento, consumo de espaço em disco, processos em execução e vários registros do sistema. Certamente esta não é uma tarefa muito fácil e normalmente é feita de forma manual. Neste trabalho serão apresentadas algumas técnicas que tornam esta tarefa possível e com um certo grau de eficácia.

Na primeira parte deste trabalho, é apresentada uma visão geral dos sistemas de detecção de intrusão, classificando-os quanto à fonte de informação, quanto à análise e quanto ao tipo de resposta. São apresentadas as vantagens e desvantagens para cada tipo de IDS, e uma tabela comparativa entre sistemas de detecção de intrusão centralizados e distribuídos, conforme as características desejáveis para um IDS.

No capítulo seguinte é apresentado o Sistema de Detecção *Asgaard*, um IDS em desenvolvimento pelo grupo de segurança da UFRGS. Na próxima seção é discutido sobre os *sniffers*, seus modos de operação, os tipos mais comuns e sugestões de como fazer a prevenção. Em seguida são discutidas as técnicas de detecção de *sniffers* de forma local e remota, e os cenários onde se aplica cada técnica.

No capítulo seguinte, são apresentadas as ferramentas existentes para detecção de *sniffers* e uma classificação por técnicas adotadas.

No último capítulo, é apresentada uma avaliação destas técnicas, testes efetuados em laboratório e os resultados obtidos.

2 Sistemas de Detecção de Intrusão

Sistemas de Detecção de Intrusão ou IDSs (*Intrusion Detection Systems*) são sistemas de software e/ou hardware que automatizam o processo de monitoramento de eventos ocorridos em um sistema computacional, analisando-os à procura por indícios de problemas com a segurança [BAC 2001].
Detecção de Intrusão é o processo de monitorar os eventos que ocorrem em um sistema computacional, analisando-os à procura de sinais de intrusão. Intrusão é definida como uma tentativa de comprometer a confidencialidade, integridade, disponibilidade ou burlar os mecanismos de segurança de um sistema computacional. Intrusões são causadas por usuários não autorizados acessando os sistemas a partir da Internet, usuários autorizados que tentam ganhar privilégios adicionais não autorizados e usuários autorizados que utilizam de forma inadequada os privilégios dados a ele [ibidem].
As definições mais comuns sobre estes sistemas não contemplam o conceito de um usuário interno abusando de seus privilégios para executar ações não autorizadas, ou pelo menos tentando executá-las. Segundo Spafford e Zamboni [SPA 2000], a definição mais completa seria "detecção de intrusão e abuso interno (*intrusion and insider abuse detection*)".

2.1 Características desejáveis de um IDS

Crosbie e Spafford [CRO 95] definiram as seguintes características desejáveis para um sistema de detecção de intrusão:

- Deve executar continuamente com o mínimo de supervisão humana;
- Deve ser tolerante a falhas, sendo apto a recuperar-se de panes acidentais ou causadas por atividade maliciosa. Depois de reiniciado um IDS deve ser apto a recuperar seu estado imediatamente anterior à falha;
- Deve ser resistente à subversão. O IDS deve estar apto a monitorar ele mesmo e detectar se ele foi modificado por um atacante;
- Deve impor uma mínima sobrecarga no sistema onde está sendo executado, não interferindo na operação normal do sistema;
- Deve ser configurável para permitir implementar políticas de segurança do sistema que está sendo monitorado;

- Deve ser adaptável a mudanças no sistema e características de usuário. Por exemplo novas aplicações sendo instaladas, usuários mudando de atividade ou novos recursos disponíveis podem causar mudanças nos padrões.
- Deve ser escalável para monitorar um grande número de estações, provendo resultados de maneira rápida e eficaz;
- Deve causar leve degradação do serviço. Se algum componente do IDS pára de executar por qualquer razão, o restante do sistema deve ser afetado minimamente;
- Deve permitir reconfiguração dinâmica, permitindo que o administrador faça mudanças na configuração sem a necessidade de reiniciar todo o sistema.

2.2 Principais tipos de IDSs

Existem atualmente diversos tipos de IDSs, caracterizados por diferentes tipos de monitoramento e análise. Cada tipo possui suas vantagens e desvantagens. No entanto todos os tipos podem ser descritos em termos gerais por um modelo genérico. Eles podem ser descritos em termos de três componentes funcionais fundamentais [BAC 2001]:

- **Fontes de informação** – as diferentes fontes de informação de eventos usadas para determinar se uma tentativa de intrusão ocorreu. Estas fontes podem ser em diferentes níveis do sistema, como rede, estação e aplicação.
- **Análise** – a parte do IDS que organiza e faz a classificação dos eventos provenientes da fonte de informação, decidindo quais destes eventos indicam que uma intrusão ocorreu ou está ocorrendo. Os tipos de análise mais comuns são detecção por mau-uso ou detecção de anomalias.
- **Resposta** – o conjunto de ações que o sistema executa quando detecta uma intrusão. São tipicamente agrupadas em medidas passivas e ativas. Medidas ativas envolvem alguma intervenção por parte do sistema e medidas passivas envolvem relatórios para posterior análise e tomada de decisão por parte do administrador do sistema.

2.2.1 Classificação quanto à Fonte de Informação

A forma mais comum de classificar IDSs é agrupá-los pela fonte de informação. Alguns IDSs analisam pacotes da rede, capturados de *backbones* ou segmentos de rede, outros analisam fontes de informação geradas pelo sistema operacional ou pela aplicação à procura de sinais de intrusão.

2.2.1.1 IDS baseado em rede

A maioria dos IDSs comerciais é baseada em rede. Estes IDSs detectam ataques capturando e analisando pacotes da rede. Escutando um segmento de rede ou um *switch*, um IDS baseado em rede pode monitorar o tráfego afetando múltiplas estações que estão conectados àquele segmento de rede, portanto protegendo aquelas estações.
Um IDS baseado em rede consiste em um conjunto de sensores ou estações colocados em vários pontos de uma rede. Estas unidades monitoram o tráfego da rede, executando uma análise local e respondendo a indícios de ataques para uma central de gerenciamento.

Vantagens

- Com uma boa distribuição, poucos sensores podem monitorar toda uma rede;
- A implantação de um IDS baseado em rede causa pouco impacto na rede existente. Normalmente são dispositivos passivos que "escutam" o tráfego da rede sem interferir em sua operação normal. E ainda é relativamente fácil redefinir uma rede para poder adicionar um IDS baseado em rede;
- Um IDS baseado em rede pode ser bastante eficiente contra ataques e ainda ser invisível para muitos atacantes.

Desvantagens

- Um IDS baseado em rede pode ter muita dificuldade em processar todos os pacotes de uma grande rede, portanto pode falhar em reconhecer um ataque iniciado em períodos de alto tráfego. Alguns fabricantes estão tentando resolver este problema implementando IDSs totalmente em *hardware*, sendo muito mais rápidos e eficazes;
- Muitas das vantagens de um IDS baseado em rede não se aplicam às mais atuais redes baseadas em *switches*. Muitos *switches* não possuem uma porta universal de monitoramento e isto limita a faixa de monitoramento de um sensor para uma simples estação. E ainda, quando os *switches* possuem esta porta de monitoramento, a mesma não espelha todo o tráfego circulando no *switch;*
- Um IDS baseado em rede não consegue analisar dados cifrados. Este problema ocorre quando as organizações (e atacantes) utilizam redes privadas virtuais (VPN);
- Muitos IDSs baseados em rede não podem identificar se um ataque foi bem sucedido ou não, eles somente podem identificar que um ataque iniciou. Isto significa que após detectar um ataque, o administrador deve manualmente investigar cada estação atacada para identificar se foi invadido;
- Alguns IDSs baseados em rede têm dificuldades em lidar com ataque baseados em rede envolvendo pacotes fragmentados. Estes pacotes mal-formados podem causar instabilidade no IDS e até derrubá-lo.

2.2.1.2 IDS baseado em estação (*host-based)*

Um IDS baseado em estação funciona com informações coletadas a partir de um sistema computacional individual. Estes IDSs permitem analisar atividades com grande precisão e confiança, determinando exatamente quais processos e usuários estão envolvidos em um ataque. Ao contrário dos IDSs baseados em rede, os IDSs baseados em *estação,* na iminência de um ataque podem acessar e monitorar os arquivos e processos utilizados neste ataque. IDSs baseados em *estação* normalmente utilizam dois tipos de fontes de informação: trilhas de auditoria do sistema operacional e *logs* do sistema. Trilhas de auditoria são normalmente geradas em nível de núcleo do sistema operacional e portanto mais detalhadas e mais protegidas do que os *logs* do sistema. No entanto, os registros do sistema são bem menos complicados e menores que as trilhas de auditoria, e ainda mais fáceis de serem compreendidos.

Vantagens

- pela sua característica de monitorar eventos locais, podem detectar ataques que não seriam detectados por um IDS baseado em rede;

- podem operar em ambientes onde o tráfego da rede é cifrado, pois a fonte de informação é gerada antes de ser cifrada e/ou depois de ser decifrada na estação de destino;
- não são afetados por redes com *switches*;
- Quando um IDS baseado em *estação* opera sobre trilhas de auditoria do Sistema Operacional, ele pode ajudar a detectar "cavalos de tróia"[1] ou outros ataques que envolvam quebra de integridade de *software*.

Desvantagens

- São difíceis de gerenciar, pois as informações devem ser configuradas e gerenciadas para cada estação monitorada;
- Se o IDS residir na estação atacada, o próprio IDS pode ser atacado e desabilitado como parte do ataque;
- Não são próprios para detectar varreduras na rede, pois somente analisam os pacotes recebidos por ela própria (estação*)*;
- Podem ser desabilitados por ataque do tipo DoS;
- Estes IDSs utilizam de recursos computacionais da própria estação onde estão monitorando, afetando na performance do sistema monitorado.

2.2.1.3 IDS baseado na aplicação

IDSs baseados na aplicação são um subconjunto especial de IDSs baseados em *estação* que analisam os eventos com base no *software* de aplicação. As fontes de informação mais comum neste caso são os arquivos de *logs* de transações. A habilidade de interagir diretamente com o *software*, com domínio ou conhecimento específico da aplicação incluídos no processo de análise, permite aos IDSs baseados na aplicação detectar comportamento suspeito e ainda usuários autorizados excederem seus direitos. Isto porque os problemas são mais fáceis de aparecer na interação entre o usuário, os dados e a aplicação.

Vantagens

- Podem monitorar a interação entre o usuário e a aplicação, o que permite detectar atividades não autorizadas para cada usuário;
- Pode executar em ambientes cifrados, pois sua interface com a aplicação é feita por troca de informações não cifradas.

Desvantagens

- Podem ser mais vulneráveis a ataques que os IDSs baseados em estação, pois os registros das aplicações não são tão protegidos quanto as trilhas de auditoria do sistema operacional usadas pelos IDSs baseados em *estação*;

- Como monitoram eventos em nível de usuário, geralmente não podem detectar

[1] cavalo-de-tróia : tipo de ataque que consiste em programas que além de executar funções para as quais foi aparentemente projetado, também executa outras funções normalmente maliciosas e sem o conhecimento do usuário (NBSO).

"cavalos de tróia" ou outros ataques com outras ferramentas. Por isto é aconselhável utilizar um IDS baseado na aplicação em conjunto com um IDS baseado em *estação* e/ou baseado em rede [BAC 2001].

2.2.2 Classificação quanto à Análise

Existem duas formas primárias de analisar eventos para detectar ataques: detecção por mau-uso ou uso indevido e detecção por anomalia.

2.2.2.1- Detecção por Mau-uso

Detectores de mau-uso analisam as atividades do sistema, procurando por eventos que correspondam à um padrão pré-definido de eventos que descrevam um ataque conhecido. Devido aos padrões correspondentes a ataques conhecidos, a detecção por mau-uso é normalmente conhecida por detecção baseada em assinatura. As formas mais comuns deste tipo de detecção usadas em produtos comerciais, especificam cada padrão de eventos que correspondem à um ataque como uma assinatura separada. Entretanto, existem formas mais sofisticadas de fazer detecção por uso indevido. São as chamadas técnicas de análise baseada em estados, as quais podem se utilizar de uma simples assinatura para detectar grupos de ataques.

Vantagens

- São muito eficientes para detectar ataques sem gerar grande número de alarmes falsos[1];
- Podem diagnosticar de forma rápida e eficiente, o uso de uma ferramenta ou uma técnica de ataque;
- Podem permitir aos gerentes de sistema, sem muita experiência em segurança, rastrear problemas de segurança em seus sistemas, iniciando procedimentos de tratamento de incidentes.

Desvantagens

- Podem somente detectar os ataques conhecidos e devem ser continuamente atualizados com assinaturas de novos ataques;
- Muitos detectores deste tipo são projetados para usar poucas variações de assinaturas conhecidas para detectar as variantes de ataques comuns. Detectores que utilizam técnicas baseadas em estados podem transpor esta limitação, mas não são geralmente utilizados em produtos comerciais;

[1] alarmes podem ser classificados como falsos positivos (causados por ações consideradas inadequadas porém aceitáveis), falsos negativos (ações inaceitáveis qualificadas como ações adequadas) ou alarmes verdadeiros (ações inadequadas).

2.2.2.2 Detecção por Anomalia

Detectores de anomalias identificam um comportamento anormal em uma estação ou uma rede. Eles funcionam assumindo que os ataques são diferentes das atividades normais e podem ser detectados pelo sistema que identifica estas diferenças. Detectores de anomalias constróem modelos considerados normais de comportamento dos usuários, estações e conecções de rede. Estes padrões são construídos com dados históricos coletados em um período de operação normal. Os detectores coletam dados de eventos e usam uma variedade de medidas para determinar quando atividades monitoradas desviam de seu comportamento padrão. As medidas e técnicas utilizadas em uma detecção por anomalia incluem:

- **Detecção de Carga,** na qual certos atributos do usuário e comportamentos do sistema são expressos em termos de contagem, com algum nível estabelecido como permitido. Um atributo de comportamento pode incluir o número de arquivos acessados por um usuário em um dado período de tempo, o número de falhas de autenticação, a CPU utilizada por um processo, etc. Este nível pode ser estático ou heurístico, ou seja, projetados para mudar com valores atuais observados em um período;

- **Medidas Estatísticas,** podem ser parametrizadas, onde a distribuição de atributos modelados é assumida para um padrão particular, e não parametrizada, onde a distribuição dos atributos modelados é aprendida de um conjunto histórico de valores observados em um período;

- **Medidas Baseadas em Regras,** similares às medidas estatísticas não parametrizadas, onde os dados observados são definidos como padrões de uso aceitáveis, mas diferem nos padrões, que são especificados como regras e não como quantidades numéricas.

- **Outras Medidas**, incluindo redes neurais, algoritmos genéticos e modelos de sistemas imunes[1].

Infelizmente os detectores de anomalia e os IDSs baseados neles, produzem uma grande quantidade de falsos alarmes, pois os padrões do usuário e o comportamento do sistema podem variar muito. Não considerando este fato, pesquisadores afirmam que os IDSs baseados em anomalia podem detectar novas formas de ataques, ao contrário dos IDSs baseados em assinaturas, que só podem detectar ataques que já ocorreram. No entanto, algumas formas de detecção por anomalia produzem uma saída que pode ser utilizada como fonte de informação para detectores de mau-uso. Por exemplo, um detector de anomalias baseado na carga pode gerar um contexto representando o número "normal" de arquivos acessados por um usuário; o detector de mau-uso pode usar este contexto como parte de uma assinatura de detecção que diz: "se o número de arquivos acessados por este usuário exceder este contexto em 10%, disparar um alarme".

[1] Informações complementares sobre sistemas imunes podem ser obtidas em *http://www.cs.unm.edu/~immsec/publications/abstracts.htm* (Acesso em 20/09/03).

Vantagens

- Detectam comportamentos anormais, portanto têm a habilidade de detectar sintomas de ataques sem conhecimento de detalhes específicos;
- Podem produzir informação que pode ser usada para definir assinaturas para um detector por uso indevido.

Desvantagens

- Normalmente produzem um grande número de falsos alarmes;
- Requer um extenso conjunto de registros de eventos do sistema para caracterizar um padrão de comportamento normal.

2.2.3 Classificação quanto à Resposta

A partir do momento que os IDSs obtiveram informações sobre eventos e os analisaram à procura de sintomas de ataques, eles geram respostas. Algumas destas respostas envolvem relatórios de busca e resultados, outros envolvem respostas automáticas mais ativas. IDSs comerciais suportam uma grande faixa de opções de resposta, caracterizadas como respostas ativas ou passivas ou uma combinação das duas.

2.2.3.1 Respostas Ativas

Respostas ativas são ações automáticas que são executadas quando uma intrusão é detectada. Existem três categorias de respostas ativas:

Coletar informação adicional – A mais inócua, mas às vezes mais produtiva resposta ativa, seria coletar informações adicionais sobre uma suspeita de ataque. Cada um de nós faz o mesmo quando escuta um barulho à noite. A primeira coisa é prestar maior atenção, procurando por informações adicionais que possam permitir a tomada de decisão sobre qual ação tomar.
No caso dos IDSs, isto pode envolver um incremento do nível de sensibilidade da fonte de informação (ou seja aumentar o número de eventos de *log* pelas trilhas de auditoria do sistema operacional, ou capturando todos os pacotes ao invés de determinada porta ou sistema). Coletar informações adicionais pode ajudar na detecção do ataque e ainda podem ser utilizadas em uma possível investigação sobre a origem do ataque e dar suporte às medidas legais civis e criminais.

Mudança do Ambiente – Uma outra resposta ativa seria interromper um ataque em progresso e bloquear novos acessos pelo atacante. Tipicamente um IDS não tem a característica de bloquear um acesso específico de um usuário, mas pode bloquear o endereço IP que o atacante está utilizando. É muito difícil bloquear um determinado e reconhecido atacante, mas os IDSs podem dificultar atacantes mais espertos ou parar atacantes iniciantes executando as seguintes ações :

- Injetando pacotes de *TCP Reset* na conexão entre o atacante e o sistema invadido, portanto terminando a conexão;
- Reconfigurar roteadores e *firewalls* para bloquear pacotes provenientes da

aparente localização do atacante (IP ou sítio);
- Reconfigurar roteadores e *firewalls* para bloquear portas, protocolos ou serviços que estão sendo utilizados pelo atacante;
- Em situações extremas, reconfigurar roteadores e *firewalls* para recusar todas as conexões que utilizam certa interface de rede.

Atitude contra o Atacante

Alguns pesquisadores acreditam que a primeira ação na resposta ativa é tomar uma atitude contra o atacante. A forma mais agressiva deste tipo de resposta envolve disparar ataques contra ou tentar obter informações sobre a estação ou o *sitio* do atacante. O primeiro ponto é que isto pode ser ilegal. E ainda, muitos atacantes utilizam falsos endereços para proceder seus ataques, logo podemos causar danos a usuários ou *sitios* inocentes. Finalmente, caso o atacante perceba o contra-ataque ele pode disparar ações muito mais agressivas.

2.2.3.2 Respostas Passivas

Respostas passivas são informações sobre os usuários do sistema passadas ao gerente ou administrador do sistema para tomar uma atitude posterior baseado nestas informações.

Alarmes e Notificações

Alarmes e notificações são gerados pelos IDSs para informar aos responsáveis quando um ataque foi detectado. Muitos IDSs comerciais permitem ao usuário configurar como e quando um alarme deve ser gerado e para quem deve ser enviado. A informação fornecida em uma mensagem de alarme pode variar muito, desde uma simples notificação até uma mensagem extremamente detalhada, com o IP fonte e destino, a ferramenta utilizada para o ataque e o caminho percorrido pelo ataque. Outro conjunto de opções que são muito utilizadas, são aquelas que envolvem uma notificação remota de alertas ou alarmes. Isto permite configurar o IDS para que envie alertas para telefones celulares, *pagers* ou correio eletrônico.

Interrupções SNMP e *Plug-ins*

Alguns IDSs são projetados para gerar alarmes e alertas, enviando-os para um sistema de gerenciamento de rede. Eles utilizam interrupções SNMP[1] e mensagens para postar alertas e alarmes em uma central de gerenciamento de rede, onde podem ser vistos pelo administrador da rede. Muitos benefícios são associados com este esquema, incluindo a habilidade de adaptar toda a infra-estrutura de rede para responder a uma tentativa de ataque, a habilidade de deslocar a carga de processamento associada a uma resposta ativa para um outro sistema que não seja o invadido, e a habilidade de utilizar canais comuns de comunicação.

[1] *Simple Network Management Protocol* (RFC 1157)

Relatórios e Arquivos

Muitos, se não todos os IDSs comerciais, possuem a capacidade de gerar relatórios rotineiros e outras informações detalhadas. Alguns podem ser configurados para gerar relatórios mensais, semanais, outros podem gerar em formato próprio para inclusão em sistemas de banco de dados ou em pacotes geradores de relatórios como o *Cristal Reports*, por exemplo.

2.3 IDSs Centralizados e Distribuídos

Um sistema distribuído consiste de um conjunto de elementos ou nodos computacionais conectados em uma rede de computadores que provê a comunicação entre eles. Um dos objetivos no projeto de um software distribuído é o de minimizar a quantidade de comunicação necessária entre os nodos. Menos comunicação significa alta escalabilidade, que é a habilidade do sistema suportar mais nodos ou mais usuários [HAU 99].
Sistemas de Detecção de Intrusão são geralmente classificados pela forma como seus componentes estão distribuídos:

- Um sistema de detecção centralizado é aquele onde a análise dos dados é feita em um número fixo de locais, independente de quantas estações estão sendo monitoradas. Não se considera a localização dos componentes de coleta de dados, somente a localização dos componentes de análise. Alguns sistemas de detecção considerados centralizados são : IDES [DEN 87], IDIOT [CRO 96], NADIR [HOC 93] e NSM [HEB 90].

- Um sistema de detecção distribuído é aquele onde a análise dos dados é executada em um número de locais proporcional ao número de estações que está sendo monitorada. Também é considerado o número de componentes de análise e não o número de componentes de coleta de dados. Alguns sistemas considerados distribuídos são: DIDS [SNA 91], GrIDS [CHE 99], EMERALD [POR 97] e AAFID [BAL 98].

A Tabela 2.1 apresenta as vantagens e desvantagens de IDSs centralizados e distribuídos segundo as características desejáveis mencionadas no item 2.1 [SPA 2000].

TABELA 2.1 – Vantagens e desvantagens de IDS segundo suas características
(continua)

Características	Centralizado	Distribuído
Executar continuamente	Um pequeno número de componentes precisam estar executando.	Um grande número de componentes precisam estar executando.
Tolerância a Falhas	O estado interno do IDS é armazenado de forma centralizada, sendo fácil a recuperação após uma falha.	O estado interno do IDS é distribuído, sendo mais difícil o armazenamento de uma forma consistente e recuperável.

TABELA 2.1 – Vantagens e desvantagens de IDS segundo suas características (continuação)

Características	Centralizado	Distribuído
Resistente á subversão	Um pequeno número de componentes necessita ser monitorado. No entanto estes componentes são muito mais complexos, sendo mais difícil monitorar.	Um grande número de componentes necessita ser monitorado. No entanto, devido ao grande número de componentes, eles podem se monitorar entre si. Os componentes são usualmente menores e menos complexos.
Mínimo overhead	Impõe pequena ou nenhuma sobrecarga nos sistemas, exceto naquele onde os componentes de análise executam, onde uma grande carga é imposta. Pode ser necessário que estas estações estejam dedicadas à análise.	Impõe pequena sobrecarga pois os componentes são pequenos.
Configurável	Facilidade de configurar pelo número pequeno de componentes. Pode ser difícil ajustar para características específicas de diferentes estações a serem monitoradas.	Cada componente pode ser localizado pelo conjunto de estações por ele monitoradas, e pode ser facilmente ajustado para suas características e tarefas específicas.
Adaptável	Por ter todas as informações em poucos locais, fica mais fácil detectar mudanças no comportamento global.	Dados são distribuídos, o que torna mais difícil ajustar a mudanças globais de comportamento. Mudanças locais são mais fáceis de detectar.
Escalável	O tamanho do IDS é limitado pelo número fixo de componentes. Quando o número de estações monitorado cresce, torna-se necessário aumentar os recursos de execução e armazenamento para compensar a carga.	Um IDS distribuído pode possuir um grande número de estações, apenas adicionando componentes quando necessário. A escalabilidade pode estar limitada pela necessidade de comunicação entre os componentes, e pela existência de componentes centrais de coordenação.
Leve degradação do serviço	Se um componente de análise pára, todo o IDS pára. Cada componente é um simples ponto de falha.	Se um componente de análise pára de executar, parte da rede pode parar de ser monitorada, mas o resto dos IDSs continuam executando.
Reconfiguração dinâmica	Um pequeno número de componentes analisa todos os dados. Reconfigurar requer que o IDS seja reiniciado.	Componentes individuais podem ser reconfigurados e reinicializados sem afetar o resto do IDS.

Em [MÉ 2001] podem ser encontradas mais de 600 referências bibliográficas sobre detecção de intrusão, datadas de 1980 a 2001.

2.4 O Sistema de Detecção de Intrusão Asgaard

O Sistema de Detecção de Intrusão Asgaard é um projeto em desenvolvimento pelo Grupo de Segurança da UFRGS, cujo objetivo é o de criar um IDS completamente distribuído, com características de tolerância à falhas, modular e de fácil portabilidade. A arquitetura do sistema *Asgaard* é baseada no conceito de módulos. Distribuídos por várias estações de uma rede, esses módulos desempenham diferentes funções dentro do *Asgaard*, desde tarefas como coletar e analisar dados, consideradas atividades fim, até autenticar, controlar e efetivar a entrada e a saída de novos módulos, atividades meio. Para tanto, cada módulo *Asgaard* possui uma infra-estrutura de comunicação, que fornece primitivas de interação, confiabilidade e autenticação. Aliado a isso, todo módulo possui uma funcionalidade específica que define o seu papel no sistema, esta situada imediatamente acima da infra-estrutura de comunicação. A existência dessas camadas básicas possibilita a integração entre os diferentes módulos criados, característica fundamental no sistema *Asgaard* e um dos avanços em relação a outros IDSs. A Figura 2.1 mostra a arquitetura adotada [CAM 2001].

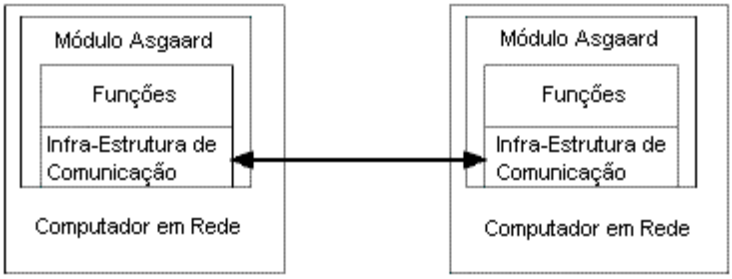

FIGURA 2.1 – Arquitetura do Sistema *Asgaard*

Um outro conceito incorporado ao Sistema *Asgaard* é o de "Domínios *Asgaard*". Um domínio *Asgaard* é a reunião de todos os módulos de uma dada rede, sem restrições de interação entre estes módulos [ibidem]. Um domínio contempla a condição mínima de segurança na troca de informações entre componentes de um IDS (módulos), como bases de assinaturas, ataques em andamento, facilitando a administração e possibilitando ações em conjunto no rastreamento ou análise de eventos. Todo módulo pertencente à um domínio passa necessariamente por um mecanismo de autenticação antes de entrar em funcionamento. Esta tarefa de autenticação é efetuada por um módulo especial chamado *Heimdal*, equivalente a uma autoridade certificadora (AC). Um outro módulo especial chamado *Odin* , autenticado manualmente, seria a principal AC de um domínio Asgaard, responsável pela autenticação dos módulos *Heimdal*. Como existe a possibilidade de interação entre diferentes domínios Asgaard, existe a necessidade de um módulo especial responsável por sua ligação, chamado *Bifrost*. Este módulo realiza a autenticação de domínios confiáveis e o tratamento das informações trocadas, evitando que dados importantes sejam interceptados [ibidem]. As funções de cada módulo encontram-se situadas em várias camadas, responsáveis por características como autenticação, detecção de falhas, etc, e cada camada baseia-se em chamadas de sistema abstratas. Todos os módulos possuem a mesma estrutura, conforme ilustra a figura 2.2.

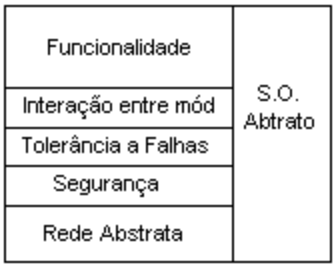

FIGURA 2.2 – Estrutura em camadas de um módulo Asgaard

A camada de rede abstrata incorpora primitivas de comunicação simples como *send, receive, listen* e *broadcast.* A exemplo da camada de Sistema Operacional abstrato (S.O.), esta camada foi projetada para isolar os detalhes de implementação e as peculiaridades de cada rede das camadas superiores.
A camada de S.O. abstrato isola as camadas do sistema, dos detalhes dependentes de sistema operacional.
A camada de segurança responde basicamente pelo sigilo e autenticidade das informações enviadas.
A camada de tolerância a falhas tenta garantir que um módulo falho seja detectado em tempo hábil e de forma distribuída, evitando qualquer ataque que vise interromper ou degradar o funcionamento do IDS.
A camada de interação entre os módulos é responsável pela interação entre os diferentes módulos funcionais, representados por diferentes sensores analisadores.

2.5 Conclusão

Uma das funções a serem agregadas neste sistema de detecção de intrusos é a capacidade de detectar *sniffers*, agregando valor e funcionalidade ao sistema. Na próxima seção serão apresentados os *sniffers*, seu funcionamento, e ainda serão discutidas, apresentadas e avaliadas as técnicas de detecção de *sniffers*.

3 Sniffers

A partir do momento que a uma rede *Ethernet* foi projetada para que os nodos compartilhassem o mesmo meio de comunicação, estes nodos estão aptos a "escutar" todo o tráfego deste meio. *Sniffers* são programas utilizados para capturar pacotes de informação trafegando em um meio compartilhado de uma rede de computadores. Programas do tipo *sniffer,* sob o ponto de vista de rede, são passivos, somente coletam dados, mas algumas vezes é possível detectar estas atividades em determinados nodos de uma rede [GRA 2000]. A seção 3.1 descreve o funcionamento dos *sniffers*.

3.1 Como funcionam os *Sniffers*

Em uma rede *Ethernet*, cada estação possui uma interface (NIC – *network interface card*), com um endereço físico (MAC – *media access control*) de 6 *bytes* atribuído pelo fabricante que a identifica na rede [HAT 2003]. Toda comunicação em uma rede Ethernet é baseada neste endereço de *hardware*. A interface de rede, no entanto pode ser configurada com diferentes filtros para receber ou rejeitar determinados tipos de pacotes, como *unicast*, *broadcast* e *multicast*, por exemplo.
Normalmente as estações estão aptas a "escutar" e responder somente a pacotes endereçados à ela, pois uma interface *Ethernet* em funcionamento normal ignora todo o tráfego que não for para o seu endereço, descartando os pacotes cujos endereços MAC não sejam correspondentes. Como estão em um mesmo meio físico compartilhado, é possível configurar esta interface para que capture todos os pacotes, independente para qual endereço o mesmo tenha sido destinado. Esta condição de funcionamento é definida como "modo promíscuo", e sob estas circunstâncias as estações podem monitorar e capturar o tráfego de rede, mesmo que o endereço de destino não seja o seu, como ilustra a Figura 3.1.

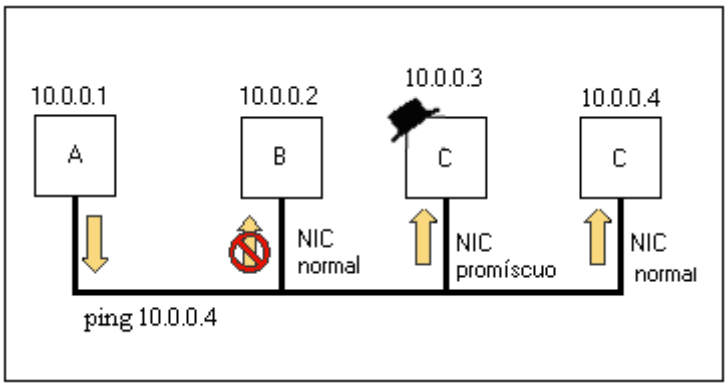

FIGURA 3.1 – Filtro de Hardware

Os *softwares* originalmente desenvolvidos para utilizar esta forma de funcionamento das interfaces são chamados analisadores de rede ou analisadores de protocolos. Porém, foram desenvolvidos programas para utilizar este modo de funcionamento com outras finalidades, como obter informações não autorizadas. As ferramentas conhecidas como *sniffers*, foram assim chamadas após ter sido lançado um produto chamado *Sniffer Network Analizer* em 1988 pela *Network General Corp.*, atualmente *Network Associates Inc*. O *Sniffer* foi um dos primeiros dispositivos com o propósito de monitorar o tráfego de uma rede, possibilitando ao administrador descobrir falhas na rede. O *Sniffer Network* foi classificado como um dos 10 produtos mais importantes da década, segundo a revista *Network Computing*[1] [JOC 2001].

Sniffers normalmente capturam os pacotes na rede, dando ao administrador, detalhes sobre os endereços de origem e destino, formação dos pacotes, além dos dados e outras informações em nível dos protocolos utilizados na comunicação. Em outras palavras, os *sniffers* foram desenvolvidos com a finalidade de ajudar os administradores de rede a avaliar e diagnosticar problemas de performance ou mau funcionamento de dispositivos de rede, de aplicações, enfim do funcionamento geral da rede ou segmento de rede. No entanto são também utilizados de forma a capturar informações não autorizadas possibilitando a invasão de outros sistemas computacionais, como nomes de contas, senhas e os próprios dados dos usuários. No dito popular, poderiam ser chamados de "faca de dois gumes" ou ainda um mal necessário. A razão da proliferação do uso de *sniffers* vem do fato da utilização de protocolos inseguros como *ftp*, *telnet*, *pop, rlogin* entre outros que enviam senhas em formato aberto[2], facilmente capturadas. A descoberta destas informações pode levar o usuário mau intencionado a iniciar ataques contra sistemas e redes conectadas, visando estações ou outros dispositivos de rede. A próxima seção descreve a utilização de *sniffers* em ambientes com *switches*, pois a adoção de redes com *switches* também foi considerada por muito tempo uma forma de proteção contra *sniffers*.

[1] Disponível em http://www.networkcomputing.com/1119/1119f1products_intro.html. (Acesso 15/03/03)
[2] As informações podem ser enviadas em formato de texto plano ou aberto, ou utilizar criptografia para dificultar a utilização das mesmas por usuários não autorizados.

3.2 Sniffing por segmentos de redes

Muitas referências tratam os *sniffers* como dispositivos de captura de pacotes em segmentos de rede, e que o simples fato de substituir *hubs* por *switches* resolveria o problema da propagação do *sniffing* para outros segmentos, como afirmou [RAY 2001]. Um segmento de rede é uma arquitetura que reside por detrás de um roteador, ponte, *hub* ou *switch*, onde cada nodo é endereçado por qualquer outro conectado ao segmento [SIP 2000]. Com a utilização de *hubs*, a propagação do sinal se dá para todas as portas do mesmo, fazendo com que todos os nodos de um segmento de rede tenham acesso aos pacotes transmitidos. Desta forma, com a instalação de um *sniffer* em qualquer nodo deste segmento, existe a possibilidade de se capturar os dados. A figura 3.2 representa o comparativo entre o funcionamento do *hub* e do *switch*, com informações sendo enviadas do nodo F para o nodo C.

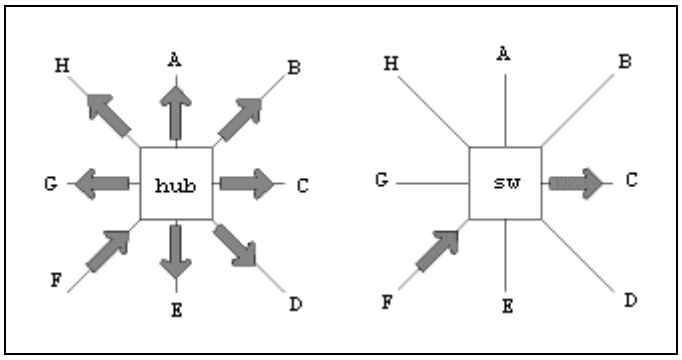

FIGURA 3.2 – Funcionamento *Hub* e *Switch*

Mas o desenvolvimento de *switches* não foi devido à questão de segurança, mas sim para segmentação de tráfego, haja visto que seu funcionamento permite o isolamento do tráfego em cada uma de suas portas. Em um ambiente com *switches*, a comunicação se dá a partir do nível 2 do modelo OSI, portanto em condições normais, os *sniffers* somente poderiam agir no segmento ao qual estivesse conectado. Isto já não é uma barreira para os *sniffers*, pois pode-se capturar pacotes entre segmentos de rede com *switches* utilizando-se algumas técnicas conhecidas de invasão do tipo *ARP*[1] *Spoofing* e *MAC Flooding* como citado em [SIP 2000], [MCC 2000] e [DHA 2002].
ARP Spoofing é uma técnica que utiliza endereços MAC e IP falsos para mascarar outra máquina na *cache* ARP. Esta *cache* contém informações para traduzir endereços IP em endereços MAC. Quando uma máquina quer se comunicar com outra ela primeiramente verifica na *cache* ARP se existe mapeado determinado IP / MAC, caso contrário utiliza o protocolo ARP mandando um pacote do tipo "*arp request*" para todos as estações (*broadcast*) requisitando o endereço MAC de determinado IP. Em condições normais, somente a estação com o endereço MAC solicitado responderá com um pacote do tipo "*arp reply*" conforme figura 3.3.

[1] ARP – *Address Resolution Protocol* (RFC 826) - protocolo que converte endereços IP em endereços físicos (MAC) em redes Ethernet.

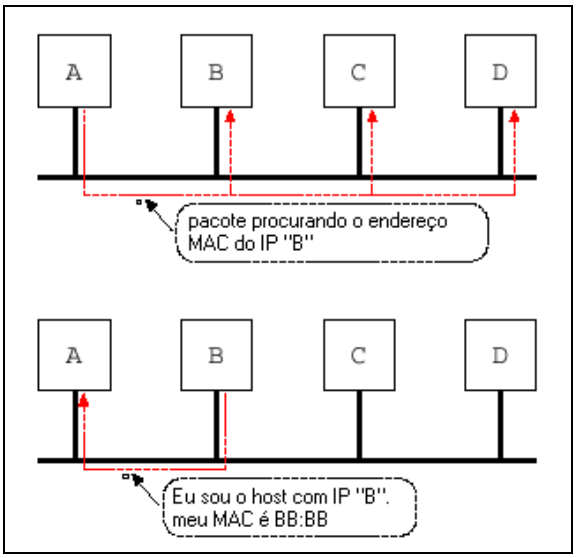

FIGURA 3.3 Funcionamento do Protocolo ARP

Uma falha neste protocolo é que não necessariamente uma estação precisa enviar um "*arp request*" para aceitar um "*arp reply*". Esta técnica se aproveita desta falha, enviando um pacote *arp reply* para uma estação que se quer atacar com uma tupla IP/MAC falsa, fazendo com que esta estação acredite que o endereço MAC recebido pertence a uma máquina confiável. Para que o ataque não seja percebido, o atacante configura sua estação para redirecionar o tráfego recebido para o destino original do pacote(*IP forwarding*).

Como os *switches* mantêm uma tabela de endereços MAC para poder estabelecer circuitos virtuais entre as estações conectadas em suas portas, a técnica de *MAC Flooding* pode ser utilizada. Esta técnica consiste em inundar um *switch* com endereços MAC falsos e tentar obter o estado "*fail open*"[1], fazendo com que o mesmo passe a funcionar como um *hub*, permitindo que se tenha acesso às informações de todos os nodos conectados. Em [SIP 2000] é citada também a técnica de *MAC Duplicating*, que consiste em configurar dois ou mais endereços MAC em uma mesma porta de *switch*. Esta técnica difere do *ARP Spoofing*, pois na primeira tenta-se confundir a estação alterando sua *cache* ARP e na segunda tenta-se confundir o *switch* fazendo com que acredite que duas portas têm o mesmo endereço MAC. Neste caso, como o tráfego está sendo enviado para ambas as portas, não é necessária a configuração de *IP forwarding*.

Em [SON 98] é descrita uma ferramenta chamada *dsniff*, que contém o utilitário "arpspoof" que como o próprio nome diz, permite o ataque de *ARP Spoofing* e outro utilitário chamado "macof" que permite o ataque de *MAC Flooding*. Como pode-se perceber, a arte de capturar pacotes em um ambiente com *switches* é mais difícil, mas não impossível. Pode-se concluir até o momento que as várias recomendações de se substituir *hubs* por *switches* para evitar a ação de *sniffers* não são eficazes. Pode-se até diminuir o raio de ação para a maioria dos invasores, mas ainda assim não impede ou bloqueia o uso do *sniffing*. A próxima seção descreve os tipos mais comuns de *sniffers* e suas plataformas.

[1] Alguns *switches* passam a se comportar como *hubs* após um bombardeio de endereços MAC ultrapassar sua capacidade de armazenamento.

3.3 Tipos de *Sniffers*

Existem atualmente centenas de programas *sniffers* criados para as mais diversas finalidades e sistemas operacionais. Uma lista bastante extensa pode ser obtida em *http://packetstormsecurity.nl/sniffers*. Os *sniffers* mais comuns, segundo [GRA 2000], estão relacionados a seguir, conforme sua plataforma:

3.3.1 *Sniffers* para Windows®

Ethereal - é uma ferramenta *UNIX-based* que também roda em *windows* e é uma das melhores soluções gratuitas de *sniffer* para *windows* atualmente. Pode ser obtido em *ftp://ethereal.zing.org/pub/ethereal/win32/* .

WinDump - é uma versão do *tcpdump* para *windows*. Pode ser obtido em *http://netgroup-serv.polito.it/windump/*

Network Associates Sniffer - Produto comercial da *Network Associates* e pode ser obtido em *http://www.nai.com*.

Network Monitor - Esta ferramenta já vem incorporada ao Windows® NT/2000/ME.

BlackICE Pro – Um *sniffer* local, não promíscuo e pode ser obtido em *http://www.networkice.com/* .

EtherPeek - Originalmente para *macintosh*, pode ser obtido versão de demonstração em *http://www.aggroup.com/* .

Sniffit - Muito utilizado para análise de dados em nível de aplicação. Pode ser obtido gratuitamente em *http:/www.symbolic.it/Prodotti/sniffit.html/*.

Snort – Sistema de detecção de intrusão gratuito portado para mais de 10 plataformas diferentes, com funções de *sniffer*. Pode ser obtido em *http://www.snort.org/*.

LanExplorer - Bastante conhecido. Pode ser obtido em *http://www.intellimax.com/* .

Analyzer - analisador de protocolo de domínio público. Pode ser obtido em *http://netgroup-serv.polito.it/analyzer/*.

3.3.2 *Sniffers* para Macintosh®

EtherPeek – Por muitos anos, o *sniffer* mais comum para *macintosh*, portado também para *windows*. Pode ser obtido em *http://www.aggroup.com/*

Snort – Sistema de Detecção de Intrusão gratuito, com funções de *sniffer*. Pode ser obtido em *http://www.snort.org/*

3.3.3 *Sniffers* para Unix®

tcpdump - Este é sem dúvida o mais antigo e mais comum programa para análise de pacotes de rede. É o padrão *Unix* e pode ser obtido em *http://www.tcpdump.org/*.

Ethereal - Este é um dos melhores sniffers para *Unix* com interface gráfica e pode ser obtido gratuitamente em *http://ethereal.zing.org/*.

sniffit - Muito utilizado para análise de dados em nível de aplicação. Pode ser obtido gratuitamente em *http://reptile.rug.ac.be/~coder/sniffit/sniffit.html/*.

hunt - Possibilidade de *sniffing* em redes com *switches*, fácil de entender. Pode ser obtido em *http://www.cri.cz/kra/index.html/*.

dsniff - Conjunto de ferramentas contendo *arpspoof*, *dnsspoff*, *macof* entre outros, testadas para *OpenBSD*, *Linux* e *Solaris*, descrito em [SON 98].

snort – Sistema de detecção de intrusão gratuito, com funções de *sniffer*. Pode ser obtido em *http://www.snort.org/*.

snoop – Para Solaris, gratuito. Pode ser obtido em *http://www.enteract.com/~lspitz/snoop.html*.

trinux - Contém o *tcpdump* e o *sniffit*, e muitos outros utilitários e cabe em um disquete inicializável. Pode ser obtido em *http://www.trinux.org/*.

karpski - Um *sniffer* para *linux* com interface gráfica. Pode ser obtido em *http://niteowl.userfriendly.net/linux/RPM/karpski.html/*.

kismet - *Sniffer* para redes *wireless* 802.11a e 802.11b, separa e identifica várias redes em uma mesma área. Executa em *linux* e pode ser obtido em *http://kismetwireless.net/*.

3.4 Prevenção contra *Sniffers*

A partir do momento que se considera a existência de um *sniffer* não autorizado, significa dizer que a rede e/ou sistemas foram comprometidos. Partindo-se deste pressuposto, ações que permitam intensificar a defesa dos sistemas computacionais podem ajudar na prevenção de ataques com *sniffers*. Isto não é uma tarefa muito fácil, considerando que o invasor pode ser interno ou externo. Downey [DOW 2000] sugere as seguintes medidas para prevenir o ataque de *sniffers*, sendo é claro, uma lista não exaustiva, pois segurança é um processo e não um produto ou uma medida [NOR 2002].

a) Atualização dos softwares com últimas versões e correções;
b) Desabilitar serviços desnecessários;
c) Verificar sítios com as últimas vulnerabilidades e correções disponíveis, tais como *CERT/CC* (*www.cert.org*), *SecurityFocus* (*www.securityfocus.com*) e *SANS Institute* (*www.sans.org*), que proporcionam guias de configuração e outras medidas de segurança;

d) Verificar freqüentemente a integridade dos arquivos utilizando uma ferramenta como *tripwire* (*www.tripwire.com*) e/ou um software antivírus atualizado;
e) Utilizar uma rede segmentada com *switches*, que muito embora não seja solução como citado anteriormente, é uma recomendação para dificultar os ataques mais simples com *sniffers*;
f) Uso de senhas fortes (mais difíceis de serem descobertas);
g) Desabilitar os LKM (*loadable kernel modules*[1]), se for o caso;
h) Utilizar criptografia.

Uma outra forma de abordar a prevenção contra *sniffers* segundo [GRA 2000], seria eliminando o modo promíscuo de operação das próprias interfaces. Algumas interfaces de rede mais antigas, como a *IBM Token Ring* original (*TROPIC chipset*), não estão aptas a funcionar em modo promíscuo. Atualmente existem interfaces de rede que bloqueiam este modo a nível de *driver*, não existindo esta funcionalidade no mesmo, portanto as aplicações falham ao tentar mudar a interface para modo promíscuo. Mas não utilizar este modo tem suas desvantagens, pois é necessário que os administradores possam analisar o tráfego de rede com o objetivo de detectar e resolver possíveis problemas.
A técnica mais efetiva contra a ação de *sniffers* segundo [BOW 2000] é a criptografia, pois a partir do momento que os pacotes não mais transitam em formato de texto compreensível, torna-se muito mais difícil a utilização das informações capturadas. Esta técnica porém, pode adicionar um tráfego considerável (*overhead*) e ter um custo elevado. Em nível de aplicação, pode-se sugerir a utilização de SSH (*secure shell*), PGP (*pretty good privacy*) ou uma camada de protocolo acima do TCP/IP como SSL (*secure socket layer*), muito utilizado atualmente para transferência "segura" de páginas (https). Em nível da camada de rede, utilizar o IPv6 (RFC 1883,1884) que provê não somente autenticação segura, mas também confidencialidade no conteúdo da sessão. Outra alternativa seria a utilização do IPsec (RFC[2] 2401 a 2412), adicionado ao IPv4 que provê autenticação, confidencialidade e integridade dos dados. IPsec baseia-se em dois protocolos, AH (*Authentication Header*) que em poucas palavras provê assinatura digital para os pacotes e o ESP (*Encapsulation Security Payload*), basicamente para cifrar os pacotes [SKO 2002].

3.5 Conclusão

Certamente não existe uma fórmula simples para a prevenção contra *sniffers*, mas sim medidas de segurança que servem não somente para impedir a instalação de *sniffers*, mas para proteção da rede como um todo. O próximo capítulo parte do pressuposto que os sistemas ou redes já foram comprometidos e que um *sniffer* pode ter sido instalado. Serão discutidas as técnicas de detecção de *sniffers*, as ferramentas disponíveis e por último a avaliação destas técnicas em diferentes plataformas.

[1] Os LKMs são arquivos que contém componentes capazes de executar dinamicamente para carregar dispositivos e outros *drivers* de hardware.
[2] RFC – *Request for Comments* – Documentos criados para serem padrões da Internet e podem ser consultadas em *www.rfc-editor.org*

4 Detecção de Sniffers

A tarefa de detectar *sniffers* tem o seu grau de complexidade, haja visto que os mesmos possuem um comportamento passivo em relação à rede. Diversos tem sido os esforços para se criar técnicas de detecção, pois uma das primeiras vantagens do atacante quando da penetração em um sistema computacional ou uma rede é ter acesso ao meio de transmissão, poder capturar informações privilegiadas, e com isto estender o ataque à outros sistemas ou redes interconectadas. As técnicas de detecção de *sniffers*, segundo [ABD 2002], podem ser classificadas em técnicas de detecção local - baseadas na estação (*host-based*) ou técnicas de detecção remota - baseadas em rede (*network-based*).
As próximas seções descrevem as técnicas conhecidas, as ferramentas disponíveis e a avaliação das mesmas em diferentes plataformas.

4.1 Técnicas de Detecção Local

Uma forma de se detectar *sniffers* em uma estação é verificar se sua interface de rede está funcionando em modo promíscuo. A maioria dos Sistemas Operacionais possui formas de se verificar a configuração das interfaces através de seus próprios utilitários. Nos sistemas operacionais Windows® 2000/XP por exemplo, o comando "*ipconfig /all*" mostra a configuração da(s) interface(s) de rede. Em sistemas UNIX®, o comando "*ifconfig*" executa esta mesma função, como mostrado no exemplo abaixo:

$ ifconfig

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Bcast:127.255.255.255  Mask:255.0.0.0
          UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
          RX packets:40 errors:0 dropped:0 overruns:0
          TX packets:40 errors:0 dropped:0 overruns:0

eth0      Link encap:Ethernet  HWaddr 00:E0:29:19:4A:68
          inet addr:172.16.0.2  Bcast:172.16.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:22 errors:0 dropped:0 overruns:0
          TX packets:23 errors:0 dropped:0 overruns:0
          Interrupt:3 Base address:0x300
```

Pode-se notar pela resposta ao comando que a interface **eth0** está funcionando em modo promíscuo, observando-se a linha 3 onde aparece o termo **PROMISC**. Esta forma de se obter as informações sobre as interfaces de rede certamente não é eficiente, pois a partir do momento que um sistema foi comprometido, os utilitários e programas executáveis não são mais confiáveis. Eles podem ter sido substituídos por outros programas com as mais diferentes funcionalidades, permanecendo transparente ao administrador dos sistemas. Segundo [ABD 2002] e [DOW 2000], existe a possibilidade de se utilizar outras ferramentas disponíveis nos sistemas operacionais para se tentar detectar *sniffers* localmente, tais como "*ls*", "*df*", "*du*", "*ps*", "*find*" e "*netstat*", mas da mesma forma elas já podem ter sido comprometidas. Uma forma de contornar este problema seria utilizando ferramentas de terceiros. A seguir são listadas as ferramentas mais conhecidas para auxiliar na detecção local de *sniffers*:

a) **cpm** (*check promiscuous mode*) – verifica se a interface está configurada para o modo promíscuo, funciona para diversas plataformas e pode ser obtida em *http://ftp.cerias.purdue.edu/pub/tools/unix/sysutils/cpm/* .

b) **ifstatus** – Outro utilitário que pode ser utilizado em conjunto com a *cron*[1], para monitorar quando a interface for colocada em modo promíscuo. Pode ser obtido em *http://ftp.cerias.purdue.edu/pub/tools/unix/sysutils/ifstatus/*

c) **lsof** (*list open files*) – Esta ferramenta possibilita a listagem de arquivos abertos, e que em conjunto com outras ferramentas do tipo "*grep*" podem auxiliar na detecção de um *sniffer*, haja visto que a maioria deles geram e mantêm abertos arquivos de *log* com os dados capturados. Funciona para a maioria das versões UNIX e pode ser obtido em *http://ftp.cerias.purdue.edu/pub/tools/unix/sysutils/lsof/*.

d) **promisc** – desenvolvida para *Linux*, detecta interface em modo promíscuo, e pode ser obtido em *http://www.attrition.org/security/newbie/security/sniffer/*.

Como estas ferramentas têm de ser executadas pelo administrador periodicamente, seria muito mais produtivo realizar este trabalho de forma automatizada, como por exemplo utilizando *scripts* ou a *cron*, no caso de sistemas UNIX. O problema é que estes procedimentos podem ser detectados pelo invasor e desabilitados antes da instalação de um *sniffer*. Em [MEL 2001] é descrita a ferramenta chamada "UPDATE", construída a partir da "*ifstatus*", que tem o objetivo de detectar interfaces em modo promíscuo, mas de uma forma mais "transparente", ou seja, sem a intervenção do usuário. A estratégia utilizada é que tradicionalmente os sistemas UNIX têm um utilitário chamado *update* que freqüentemente (=~ a cada 30 segundos) descarregam os *buffers* de disco da memória RAM para o disco (*buffer flushing*). Os sistemas operacionais mais recentes utilizam técnicas mais sofisticadas, mas a presença deste programa não é novidade alguma, portanto difícil de se imaginar que possa ter outras funções agregadas. Esta versão proposta para o *update*, implementa funções de verificação da(s) interface(s) de rede a cada 3 minutos, notificando ou desligando o sistema se a mesma encontrar-se em modo promíscuo. Como se pode perceber, existe a possibilidade de se detectar as interfaces em modo promíscuo de forma mais transparente do que simplesmente utilizar os comandos do sistema operacional.

[1] cron – serviço de agendamento de execução de programas em sistemas UNIX

Uma outra forma de detectar um *sniffer* de forma local, seria inspecionando os arquivos de *log* do sistema para verificar se a interface de rede foi colocada em modo promíscuo. Certamente estes arquivos podem ter sido modificados pelo atacante, mas sempre existe a possibilidade de que não o tenha feito por completo.
De qualquer forma em redes de grande porte esta administração torna-se inadequada e requer uma constante verificação das estações de forma manual. Para tentar resolver este problema, a seguir são descritas as técnicas para detecção remota de *sniffers* .

4.2 Técnicas de Detecção Remota

As técnicas de detecção de *sniffers* de forma remota possibilitam analisar redes ou segmentos de rede a partir de um único ponto de monitoração à procura de indícios de que um *sniffer* esteja executando. Existem várias formas de se tentar detectar que um *sniffer* esteja executando em uma rede. Um administrador de sistemas pode monitorar as estações e verificar comportamentos anormais como carga de processamento ou espaço em disco utilizado, pois quando da execução de um *sniffer*, existe o aumento considerável de consumo de recursos computacionais. Apesar de parecer razoável, estas técnicas realizadas de forma manual, na prática tornam-se inviáveis em redes de grande porte. Umas das primeiras tentativas de detectar *sniffers* de forma remota foi descrita em [GRU 98], que utilizou um falso tráfego gerado em uma rede, contendo autenticação para um serviço *telnet*, na expectativa que algum *sniffer* capturasse este tráfego e tentasse utilizá-lo. A monitoração desta utilização seria a chave para a confirmação da presença de um *sniffer*. Com o passar do tempo, outras técnicas foram sendo desenvolvidas, levando-se em consideração o funcionamento de protocolos, sistemas operacionais e *sniffers* mais comuns, como são descritas a seguir:

4.2.1 Técnicas de Detecção baseadas no endereço MAC

As técnicas de detecção pelo endereço MAC (*media access control*), somente funcionam quando a estação onde se está executando o detetor e a estação suspeita estiverem no mesmo segmento *Ethernet*. Em uma rede *Ethernet* com endereçamento IP, os pacotes são enviados e recebidos baseados em endereços físicos das interfaces ou endereços MAC. As interfaces de rede podem ser configuradas com diferentes filtros para receber diferentes tipos de pacotes. Para se entender melhor o funcionamento destas técnicas é preciso compreender o funcionamento deste filtros, os quais são listados a seguir:

a) **Unicast** – Recebe todos os pacotes com o endereço de destino correspondente ao seu endereço MAC;
b) **Broadcast** – Pacotes de *broadcast* têm o endereço de destino FF:FF:FF:FF:FF:FF. O propósito deste modo é enviar o mesmo pacote para todas as estações na rede;
c) **Multicast** – Recebe todos os pacotes que foram especificamente configurados para serem entregues a um grupo de endereços (*group bit*). Somente os pacotes dos endereços de *multicast* configurados previamente na lista de *multicast* podem ser recebidos pela interface;

d) **All Multicast** – Recebe todos os pacotes *multicast*;
e) **Promiscuous** – Recebe todos os pacotes da rede sem verificar o endereço de destino.

Os métodos a seguir foram desenvolvidos com base no funcionamento destes filtros.

4.2.1.1 Método ICMP

Este método consiste em se enviar um pacote ICMP (*internet control message protocol*)(RFC 792) do tipo "*echo request*" para o IP de uma estação suspeita, mas com o endereço MAC alterado. No funcionamento normal, a interface descarta este pacote pois o endereço MAC não corresponde ao seu. Caso a mesma esteja em modo promíscuo, ela poderá responder com um pacote do tipo "*echo reply*". A implementação desta técnica é relativamente fácil, pois requer um gerador de pacotes ICMP "*echo request*" e uma verificação se um pacote do tipo "*echo reply*" é retornado. A Figura 4.1 ilustra o funcionamento desta técnica:

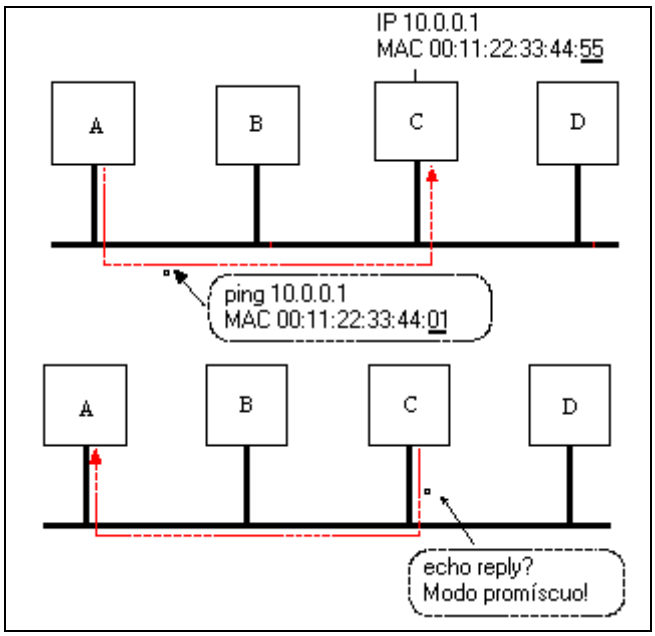

FIGURA 4.1 – Método ICMP

Este método funciona apenas em sistemas operacionais onde a pilha de protocolo TCP/IP não faz a verificação do endereço MAC. Em versões de sistemas operacionais como o *Linux* 2.0.35, esta técnica funciona perfeitamente, já o *FreeBSD* 2.2.7 faz a verificação de endereços físicos e portanto este método não é eficaz [WU 98]. Este tipo de filtro em nível de sistema operacional, pode ser chamado de "filtro de *software*", enquanto que os filtros das interfaces de rede, vistos anteriormente, de "filtros de *hardware*".

4.2.1.2 Método ARP

Este método é semelhante ao anterior, mas utilizando-se o protocolo ARP (*address resolution protocol*) (RFC 826), enviando um pacote do tipo "*arp request*" e monitorando uma resposta do tipo "*arp reply*". Para se explorar este método de forma eficaz, os pacotes devem ser criados para supostamente serem bloqueados pelo filtro de *hardware* (modo normal) e passarem pelo filtro de *software*. Desta forma, se a interface estiver em modo promíscuo, os pacotes serão respondidos. Como o filtro de software depende do sistema operacional, torna-se necessário entender o funcionamento destes filtros. Para tanto, [SAN 2001] apresenta um estudo dos filtros para *Linux* e para *Windows* utilizando pacotes ARP.

a) Linux

No módulo *Ethernet* do *Linux*, os pacotes são classificados dependendo do endereço físico MAC .

- Pacotes *BROADCAST* – FF:FF:FF:FF:FF:FF
- Pacotes *MULTICAST* – Todos os pacotes com o bit de grupo configurado, exceto pacotes de *broadcast*.
- Pacotes *TO_US* – Todos os pacotes com o mesmo endereço MAC da interface.
- Pacotes *OTHERHOST* – Todos os pacotes com endereço de destino diferente do endereço MAC da interface.

No módulo ARP do *Linux*, primeiro são rejeitados todos os pacotes *OTHERHOST*, e depois são respondidos os pacotes de *broadcast*, *multicast* e *TO_US* . A Tabela 4.2 ilustra o funcionamento dos filtros de *hardware* e de *software* , enviando pacotes "*arp request*" com tipos de endereços físicos diferentes:

TABELA 4.2 – Filtro de *software Linux (2.2/2.4)*

	group bit	**Modo Normal**			**Modo Promíscuo**		
		Filtro de Hardware	Filtro de Software	Resposta	Filtro de Hardware	Filtro de Software	Resposta
TO_US	Off	Passa	Passa	✓	Passa	Passa	✓
OTHERHOST	Off	Rejeita			Passa	Rejeita	
BROADCAST	On	Passa	Passa	✓	Passa	Passa	✓
Multicast (lista)	On	Passa	Passa	✓	Passa	Passa	✓
Multicast (fora)	On	Rejeita			Passa	Passa	✓
Group Bit	On	Rejeita			Passa	Passa	✓

Fonte: [SAN 2001]

→ Pacotes TO_US – Quando a interface está em modo normal, todos os pacotes TO_US passam pelo filtro de *hardware* e de *software*, obtendo resposta.
→ Pacotes OTHERHOST – Quando a interface está em modo normal, rejeita todos, mas quando em modo promíscuo o filtro de *software* rejeita, não obtendo resposta.
→ Pacotes BROADCAST – passam ambos os filtros e obtém resposta.
→ Pacotes MULTICAST – Em modo normal, pacotes não registrados na lista de *multicast* são rejeitados, mas em modo promíscuo, se estiverem na lista de *multicast*

ou não, passam e obtém resposta. Já os que estiverem na lista de *multicast*, sempre passam. A diferença de resposta entre *multicast* (fora da lista) com interface em modo normal e modo promíscuo, pode ser utilizado para se detectar modo promíscuo;

→ Pacotes *group bit* – Estes não são pacotes de *broadcast* nem *multicast*, mas pacotes com o bit de grupo configurado. Em modo normal o filtro de *hardware* rejeita estes tipos de pacotes, mas em modo promíscuo eles são aceitos também pelo filtro de *software* por serem considerados *multicast*, logo se obtém uma resposta.
Esta diferença de resposta também pode ser utilizada para a detecção de modo promíscuo.

b) Windows

Como nos sistemas operacionais *Windows* não se tem acesso aos códigos fonte dos programas, a análise do funcionamento do filtro de *software* é feita mediante experimentos. Para estes testes, foram utilizados vários tipos de endereços de *hardware* a seguir:

→ Endereço de *broadcast* FF:FF:FF:FF:FF:FF
Todos as estações recebem este tipo de pacote e respondem. Usado no *arp request*.

→ Endereço falso de *broadcast* FF:FF:FF:FF:FF:**FE**
Este endereço é para testar se o filtro de software verifica todos os bits

→ Endereço FF:FF:00:00:00:00
Somente os primeiros 16 bits são iguais ao endereço de *broadcast*

→ Endereço FF:00:00:00:00:00
Somente os primeiros 8 bits são iguais ao endereço de *broadcast*

→ Endereço 01:00:00:00:00:00
Este é um endereço somente com o *group bit* ativado. Usado para verificar se este endereço é considerado *multicast*.

→ Endereço 01:00:5E:00:00:00
Multicast endereço 0 normalmente não utilizado. Este é um exemplo de *multicast* que não estaria registrado na lista de *multicast* da interface. O filtro de *hardware* deve rejeitar este pacote.

→ Endereço 01:00:5E:00:00:01
Multicast endereço 1 é um endereço que todas as estações registradas recebem.

A Tabela 4.3 apresenta os resultados dos experimentos realizados utilizando pacotes ARP para verificar o funcionamento dos filtros de *software* e de *hardware* em ambientes *windows*.

TABELA 4.3 - Resposta de pacotes *arp request* em ambientes *Windows*

Endereço de HW	Windows 9x/ME		Windows NT/2k	
	Normal	Promíscuo	Normal	Promíscuo
FF:FF:FF:FF:FF:FF	✓	✓	✓	✓
FF:FF:FF:FF:FF:FE	-	✓	-	✓
FF:FF:00:00:00:00	-	✓	-	✓
FF:00:00:00:00:00	-	✓	-	-
01:00:00:00:00:00	-	-	-	-
01:00:5E:00:00:00	-	-	-	-
01:00:5E:00:00:01	✓	✓	✓	✓

Fonte: [SAN 2001]

Como esperado, todas as versões respondem a pacotes do tipo *broadcast* e *multicast* endereço 1 quando a interface está em modo normal. No entanto quando a interface está em modo promíscuo, depende da versão do sistema operacional:

- *Windows 95, 98* e *ME* respondem ao falso endereço *broadcast* de 31, 16 e 8 bits. Pode-se dizer que os filtros de *software* dos *windows* série 9x reconhecem o *broadcast* verificando somente 1 bit;

- No caso do *Windows* 2000, ele responde aos falsos *broadcast* de 31 e 16 bits. Pode-se concluir que o filtro de *software* do *Windows 2000* reconhece o *broadcast* verificando 8 bits.

Os resultados destes experimentos comprovam que pode-se utilizar o protocolo ARP para determinar de forma remota se uma interface está em modo promíscuo, tanto em sistemas operacionais *Windows* quanto *Linux*. Por outro lado depende diretamente da versão do sistema operacional de cada estação. Em ambientes heterogêneos, esta técnica pode não ser adequada, pois seriam necessários testes para cada plataforma existente.

4.2.2 Técnicas de Detecção baseadas em iscas ou armadilhas

As técnicas conhecidas como iscas ou armadilhas, também encontradas na literatura como “pote de mel”, são aquelas que fornecem um chamariz para os atacantes, normalmente utilizando um falso tráfego ou serviço e aguardando que um *sniffer* venha a capturar esta atividade e o atacante passe a utilizá-lo. A seguir são descritos os métodos para se detectar *sniffers* utilizando iscas.

4.2.2.1 Método TELNET

Um dos primeiros trabalhos a utilizar um falso tráfego foi descrito em [GRU 98], que gerava um falso TELNET (RFC 854,855) e aguardava a reutilização desta autenticação, possibilitando a detecção do *sniffer*. Muito embora esta técnica possa ser utilizada não somente em segmentos de rede, mas através de outras redes, tem a desvantagem de aguardar a ação do atacante ou seja, esperá-lo "morder a isca" e utilizar os dados capturados. Desta forma, outras estações já poderiam ter sido comprometidas. Este tipo de armadilha é muito utilizado não somente para detecção de *sniffers*, mas para detecção de intrusão como um todo, como pode ser observado no projeto *honeynet* descrito em [KUE 2001], que tem o objetivo de descobrir as técnicas, ferramentas e as ações dos atacantes. Este projeto utiliza falsos servidores desprotegidos para servirem de isca, que depois são analisados para se descobrir o que foi e pode ser comprometido e de que forma, conforme ilustra a Figura 4.2. Este trabalho resultou uma série de artigos além do livro "*Know your Enemy*"[HON 2001].

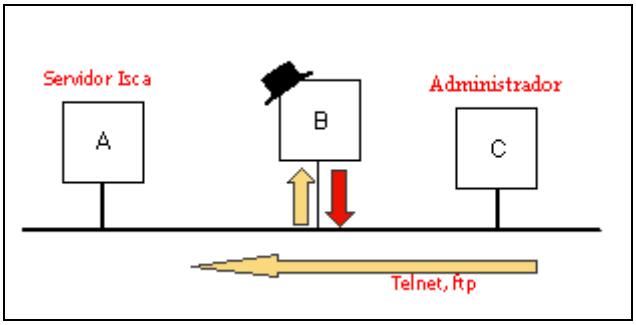

FIGURA 4.2 – Técnica da Armadilha ou Isca

4.2.2.2 Método FTP e POP3

Outras formas de iscas podem ser utilizadas por meio do uso de protocolos como FTP (RFC 959,2228) e POP3 (RFC 1939) como descrito em [ABD 2002]. Semelhante ao funcionamento do *telnet*, para utilizar estes serviços, são criadas contas de usuário em servidores de arquivos e de correio, com nomes especialmente atraentes para que sirvam como iscas, tipo *root, admin,* ou administrador. Logicamente, por motivo de segurança, devem ser contas sem grandes privilégios. Estes servidores não necessariamente precisam ser servidores falsos, pois o custo pode se tornar elevado, mas por outro lado, em caso de servidores válidos o risco de segurança também é elevado. A análise de custo/benefício neste caso deve levar em consideração a importância das informações que se quer proteger. A partir do momento em que estes serviços estejam ativos, basta gerar falsas solicitações de FTP e POP3 com estas contas de usuários, e aguardar a reutilização destes serviços.
Os métodos apresentados, *telnet*, *ftp* ou *pop3*, seguem o mesmo princípio, ou seja, que o atacante seja atraído pelos serviços disponíveis e efetivamente decida explorar os mesmos. Neste caso não se tem certeza se o atacante vai ou não utilizar as informações, mas estas soluções podem ser utilizadas em conjunto com as demais técnicas aumentando o escopo de ação da detecção. A seguir é descrito um outro método também baseado em isca, que visa detectar os *sniffers* quando já estão em execução.

4.2.2.3 Método DNS

O sistema de nomes de domínio DNS (*Domain Name System*) é um protocolo padrão (RFC 1034,1035), que além de manter as regras para sintaxe dos nomes e a delegação de autoridade no âmbito da Internet, contém um mecanismo de mapeamento de endereços IP para nomes de domínio, que serve para facilitar a utilização e a memorização de dispositivos de uma rede. Segundo [COM 98], o DNS é um mecanismo que implementa uma hierarquia de nomes de máquinas para as redes TCP/IP. Este serviço pode ser comparado ao funcionamento das listas telefônicas convencionais, onde se tem o nome das pessoas associado a determinado número de telefone. Por exemplo, ao invés de assimilar que um servidor de páginas tem o endereço IP 200.10.10.10, fica muito mais fácil lembrar de seu nome como *www.casagrande.org*. Um servidor DNS, além de conter um banco de dados de informações de nomes e endereços referentes a determinada zona ou domínio, aceita requisições do tipo *DNS lookup*, solicitando o endereço IP de um determinado nome, e do tipo *DNS lookup* reverso solicitando o nome referente a determinado endereço IP (*resolver*).
Esta técnica explora o fato de que a maioria dos *sniffers* executa uma solicitação do tipo *DNS lookup* reverso de forma automática, visando a descoberta dos nomes das estações, possibilitando ao atacante selecionar as estações mais importantes ou que contenham os serviços mais importantes. Obviamente que estas requisições por si só não revelam a presença de *sniffers*, pois sua presença faz parte do funcionamento normal de uma rede. Uma forma de utilizar este serviço para a descoberta de *sniffers*, descrita em [WU 98], consiste em gerar um falso tráfego de determinado serviço, *telnet* ou *ftp* por exemplo, para um endereço IP falso e monitorar uma requisição de *DNS lookup* reverso para o mesmo. Desde que o tráfego gerado normalmente será ignorado pelas estações no segmento de rede, se um *DNS lookup* reverso para aquele endereço IP falso for detectado, identificará a presença de um *sniffer* neste segmento.
A Figura 4.3 ilustra o funcionamento desta técnica:

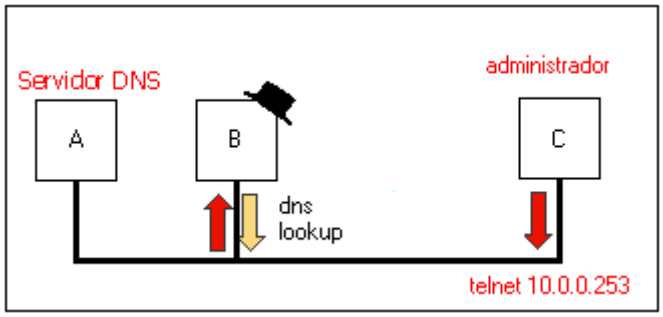

FIGURA 4.3 – Funcionamento da Técnica DNS

Segundo [WU 98], esta técnica mostrou-se eficiente em todos os testes realizados em um segmento de rede, independente do sistema operacional utilizado. Foram utilizados os *sniffers tcpdump*, *esniff*, *linsniff* e *sniffit*, e todos realizaram o *DNS lookup* reverso, permitindo a identificação de sua presença. Esta técnica é utilizada no próprio servidor DNS, contando o número de *DNS lookup* reverso feito para o endereço falso. Já em [SST 2001] a técnica foi utilizada colocando a interface de rede da própria estação em modo promíscuo e enviando os pacotes pela rede, e a seguir monitorando por um *DNS lookup* reverso para aquele endereço falso. Obviamente que esta técnica pode ser evadida se os *sniffers* forem previamente configurados para não realizarem o *DNS lookup* reverso, o que não ocorre na maioria dos casos até então conhecidos.

4.2.3 Técnica de Detecção baseadas em Carga ou Latência

Esta técnica segue o princípio de que a partir do funcionamento de um *sniffer*, existe uma degradação da performance da estação comprometida, haja visto que a interface de rede estando em modo promíscuo precisa processar todos os pacotes para enviar ao sistema operacional. Este excessivo número de interrupções e chamadas ao sistema operacional reflete na performance da estação e isto pode ser medido, calculando-se o tempo de resposta médio em funcionamento normal e promíscuo. Uma das formas utilizadas para se tentar detectar *sniffers* é gerando um alto tráfego no segmento e comparando o desempenho das estações. Uma possibilidade de fazer esta análise de desempenho foi discutida em [WU 98], que consiste em enviar diversas solicitações de serviço do tipo *ftp*, por exemplo, para uma estação e estabelecer uma média para o funcionamento normal. A seguir enviar diversas solicitações com o *sniffer* executando, verificar a média e comparar com o uso normal. Estes testes podem ser realizados para um tráfego normal ou para um alto tráfego gerado de forma proposital, comprometendo muito mais o desempenho da estação com o *sniffer* e com isto obtendo resultados mais conclusivos. A Tabela 4.3 ilustra testes realizados com *ftp* em três estações diferentes, com sistemas operacionais diferentes, alternando com *sniffer* em funcionamento ou não (*sniff on/off*) e gerando um falso tráfego ou não (*fake on/off*).

TABELA 4.3 – Testes de Carga com FTP

	Tipo de Teste	**Média RTT (msec)**	**% Aumento c/ tráfego**
Linux Pentium 200	Sniff off, fake off	45,34	
	Sniff off, fake on	46,32	2,16%
	Sniff on, fake off	49,99	
	Sniff on, fake on	59,65	19,32%
Linux Pentium 300	Sniff off, fake off	32,39	
	Sniff off, fake on	33,18	2,43%
	Sniff on, fake off	34,46	
	Sniff on, fake on	42,42	23,09%
FreeBSD Pentium 300	Sniff off, fake off	12,42	
	Sniff off, fake on	15,38	23,81%
	Sniff on, fake off	12,49	
	Sniff on, fake on	20,34	62,79%

Fonte: [WU 98]

Por estes testes pode-se verificar que com o *sniffer* executando e com alto tráfego, houve o aumento considerável em média do tempo de resposta RTT (*roundtrip time*). Esta técnica, apesar de não determinística, pode ser utilizada para a detecção de *sniffers*, desde que estabelecidos parâmetros e métricas a serem utilizadas. Em [REI 2002], estes testes foram feitos utilizando pacotes ICMP (*ping*), mas este tipo de pacote não chega ao nível da aplicação, sendo pouco indicado para medir latência segundo [WU 98].
Para esta técnica, deve-se levar em consideração também o tráfego existente no momento da realização dos testes, pois podem levar a obter falsos positivos, e considerar também a possibilidade de existirem estações invisíveis (não autorizadas).

4.2.4 Técnica de Detecção baseada em Rota da Origem

Esta técnica utiliza uma das opções contidas no datagrama IP, onde é possível o transmissor estabelecer uma rota ou caminho por onde a informação deve seguir. Um dos objetivos iniciais desta opção seria medir o desempenho da rede forçando os datagramas a seguirem determinado caminho, independente de roteadores. O protocolo IP aceita duas formas de roteamento de origem. A primeira, chamada livre ou solta (LSSR – *Loose-Source and Record Route*), oferece um meio para que a origem de um datagrama IP forneça a informação de roteamento para ser usada pelos roteadores ao encaminharem o datagrama até o seu destino, além de registrarem a rota seguida. A segunda, chamada de restrita (SSRR – *Strict Source and Record Route*) que é semelhante ao roteamento de origem livre, exceto que não podem existir rotas intermediárias entre as pré-estabelecidas, ou seja o roteador intermediário precisa enviar o datagrama para o próximo endereço IP na rota de origem via uma rede conectada diretamente e não por um roteador intermediário, o que causaria erro [COM 98].
Por meio destas opções, existe a possibilidade de se detectar *sniffers*. O método descrito em [GRA 2000] consiste em se enviar pacotes do tipo ICMP para uma estação suspeita, configurando a informação LSSR de forma a forçar a rota desejada.
Por exemplo, seja o envio de pacotes ICMP para a estação 10.0.0.3 configurados para serem roteados por 10.0.0.2 primeiramente. Estando as duas estações no mesmo segmento e se a 10.0.0.2 não estiver configurada para rotear, caso se obtenha resposta, então tem-se fortes indícios de que a estação 10.0.0.3 pode estar executando um *sniffer*. Certamente a estação 10.0.0.2 receberá o pacote, e não estando configurada para rotear, descarta o mesmo. Pode-se observar no campo TTL (*time-to-live*), se o pacote recebido veio diretamente da estação 10.0.0.3 (caso *sniffer*) ou roteado pela estação 10.0.0.2 (caso esteja configurada para tal). A Figura 4.4 ilustra esta técnica:

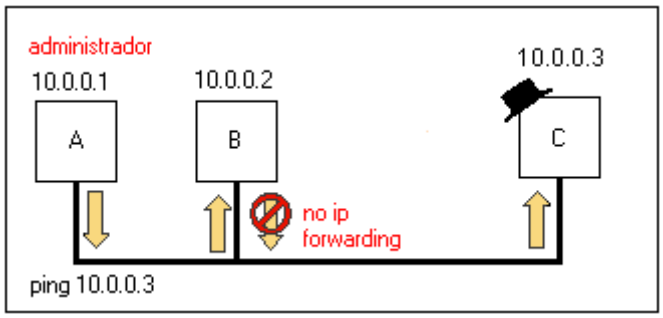

FIGURA 4.4 – Técnica Rota de Origem

4.2.5 Outras Possibilidades

A seguir são descritas algumas técnicas alternativas para a detecção de *sniffers*, pouco difundidas, mas que também podem vir a ser exploradas.

4.2.5.1 Técnica de Detecção baseada em SNMP

O gerenciamento de rede não é definido como parte do protocolo de transporte ou de rede, mas sim em nível de aplicação. Isto é, quando um gerente de rede necessita interagir com um dispositivo de *hardware* específico, o programa de gerenciamento utiliza o modelo cliente-servidor. Um programa de aplicação na estação do gerente age como um cliente e um programa no dispositivo age como um servidor [STE 94]. O cliente utiliza os protocolos de transporte convencionais TCP ou UDP para estabelecer a comunicação com o servidor. Para evitar a confusão entre programas de aplicação que utilizam os termos cliente e servidor, em gerenciamento de redes os mesmos são chamados de gerente e agente respectivamente. O protocolo SNMP (*Simple Network Management Protocol*) (RFC 1157) define como um gerente se comunica com um agente, obtendo e definindo informações para a administração da rede [COM 2001]. Segundo [COM 98], o SNMP em vez de definir um grande conjunto de comandos, lança todas as operações em um paradigma de busca e armazenamento (*fetch-store*)[1]. Teoricamente o SNMP tem basicamente dois comandos que permitem ao gerente buscar o valor de um dado ou armazenar o valor de um dado. As demais operações são definidas como resultados destas duas operações básicas. Este protocolo pode ser utilizado como técnica de detecção de *sniffer*, haja visto seu funcionamento permitir a busca de informações nos dispositivos de uma rede. Pode ser utilizado por exemplo para medir a carga ou desempenho das estações em funcionamento normal e promíscuo em alto tráfego, desde que um agente esteja executando nestas estações. Certamente não é um método muito efetivo pois um atacante quando compromete uma estação pode detectar agentes executando e desabilitá-los. Até o momento não existem experimentos conclusivos sobre a eficiência desta técnica, no entanto teoricamente utilizando informações das MIBs (*Management Information Base*) pode-se chegar a identificar a existência de *sniffer* executando nas estações. Certamente é uma área para futuras pesquisas, no que diz respeito à detecção de intrusão em redes de computadores.

4.2.5.2 Técnica de Detecção baseada no WINPCAP

Uma das bibliotecas para captura de pacotes mais comuns em sistemas *Windows* é a biblioteca WINPCAP[1], que é semelhante à biblioteca *libpcap* dos sistemas *Unix*. Normalmente os *sniffers* necessitam da instalação de uma biblioteca de captura de pacotes, então uma possibilidade seria utilizar um programa para identificar estações com esta biblioteca instalada. Esta técnica consiste de um programa em linguagem "*perl*" que pode ser obtido em *http://patriot.net/~carvdawg/perl.html* (Acesso em 25/05/03), e que tem a função de detectar a biblioteca *winpcap* instalada em sistemas *Windows*.
Certamente não é um método eficaz, pois a biblioteca pode estar sendo utilizada para outras finalidades como para gerência de redes, por exemplo, o que resultaria em falsos positivos; ou ainda os *sniffers* utilizarem outras bibliotecas diferentes desta, mas é também uma alternativa que pode e deve ser considerada.

---

[1] O paradigma de busca e armazenamento tem sua origem em um protocolo de gerenciamento conhecido como HEMS (RFC 1021 a 1024).
[1] Pode ser obtida em *http://netgroup-serv.polito.it/winpcap/*.

4.3 Ferramentas para Detecção

A primeira iniciativa que se tem conhecimento sobre a detecção de *sniffers* de forma remota, foi descrita em [GRU 98], que utilizou basicamente a técnica de isca ou armadilha, gerando falsas conecções *ftp* e *telnet* via *Internet* entre dois laboratórios da IBM situados em países diferentes. Esta comunicação utilizava contas de usuário e senhas falsas, de grande interesse como "*root*" e "*admin*", servindo de isca para os atacantes, pois bastava monitorar a reutilização destas contas/senhas para se ter a indicação da presença de um *sniffer*. A seguir são descritas as ferramentas disponíveis para a detecção de *sniffers* de forma remota.

a) **Neped** - (*network promiscuous ethernet detector*) descrita em [SAV 98], utiliza a técnica de detecção baseada no endereço MAC, forjando pacotes ARP no segmento de rede e verificando a resposta das estações em modo promíscuo. Funciona para sistemas operacionais *linux* versões 2.0.x e 2.1.x .

b) **Sentinel** – Esta ferramenta utiliza os métodos ARP, ICMP e DNS para a verificação de um segmento de rede. Utiliza as bibliotecas *libpcap* e *libnet*, para possibilitar portabilidade. Executa corretamente em *openBSD* 3.0-beta, *freeBSD* 4.0, *netBSD* 1.5.2 e *linux* 2.4.x , conforme [BIN 2001].

c) **Promiscan** – Esta ferramenta, descrita em [SAN 2001], utiliza exclusivamente pacotes ARP, executa em sistemas *windows* NT/2K/XP e pode detectar estações *windows* 9x,ME,2k e *linux* 2.4.x em modo promíscuo.

d) **Antisniff** – Uma das primeiras ferramentas de detecção comercial, utiliza os métodos ICMP, ARP, DNS e Latência, com versões para diferentes sistemas operacionais, é descrita em [SST 2002].

e) **SnifferWall** – Esta ferramenta descrita em [ABD 2002] implementa os métodos ICMP, ARP e Armadilha com os protocolos FTP, DNS e POP3 , e pode detectar interfaces em modo promíscuo de sistemas *windows* 9x,ME,NT,2K e *linux* versões 2.0 a 2.4 .

f) **Sniffdet** - Uma das ferramentas mais recentes para detecção de *sniffers*, sob licença GPL[1], descrita em [REI 2002], implementa os métodos ICMP, ARP, DNS e Latência, e executa em sistemas *linux*.

A seguir é apresentada a Tabela 4.4 comparativa entre as ferramentas atualmente disponíveis para detecção de *sniffers* remotamente, em relação às técnicas utilizadas.

[1] GPL – *General Public License* – Licença de software que garante a livre distribuição e alteração.

TABELA 4.4 – Ferramentas de detecção por técnica

	Ano	Plataforma	ICMP	ARP	DNS	LATÊNCIA	ISCA
IBM Detector	1998	Linux					ftp,telnet
Neped	1998	Linux		✓			
Sentinel	2000/01	Linux, BSD	✓	✓	✓		
Promiscan	2001	Windows NT+		✓			
SnifferWall	2002	Windows	✓	✓	✓		ftp,telnet,pop3
AntiSniff	1999/02	Win/Linux/BSD	✓	✓	✓	✓	
SniffDet	2002	Linux	✓	✓	✓	✓	

Esta tabela apresenta os esforços realizados para a construção de ferramentas para detecção de *sniffers*, classificadas pelo ano em que foram criadas, as plataformas que podem ser executadas e as técnicas implementadas.
Na próxima seção, será apresentada uma avaliação das técnicas conhecidas, utilizando as ferramentas disponíveis contra ambientes *windows* e *linux*.

4.4 Avaliação e Testes

Uma avaliação das técnicas de detecção de *sniffers* requer o estabelecimento de um cenário para testes, devido às inúmeras opções a serem consideradas, como tipo de sistema operacional e versão utilizada, diversidade de *sniffers* e as técnicas e ferramentas disponíveis.
Foram escolhidos arbitrariamente os sistemas operacionais *windows* e *linux* como plataforma de testes, unicamente devido à popularidade dos mesmos. Para o ambiente *windows* foram escolhidos os *sniffers* "ethereal" e "windump", e para o ambiente *linux* o "ethereal" e "tcpdump".
Os testes são realizados a partir da instalação dos *sniffers* em uma estação *windows* e disparadas as técnicas de uma estação *linux* com as ferramentas "sentinel", "antisniff" e "sniffdet", testando cada opção implementada. A seguir os *sniffers* são instalados na estação *linux* e disparados os mesmos testes a partir da estação linux e ainda os testes a partir de outras estações *windows* com as ferramentas "antisniff" e "promiscan", verificando também a eficiência das mesmas.
Algumas variações de sistema operacional também foram testadas, como *Windows* XP e *FreeBSD*, e seus resultados serão também apresentados.
A seguir são descritos os testes realizados, classificados por técnica e por sistema operacional utilizado.

4.4.1 Testes de Detecção em Sistemas Operacionais Windows®

Os primeiros testes realizados foram utilizando o sistema operacional Windows 98, versão 4.10.98, instalado em um microcomputador Intel Pentium® 200 Mhz com 64 MB de memória RAM. Nesta estação com endereço IP 10.0.0.54 foram instalados os *sniffers* "ethereal 0.9.2" obtido a partir de *http://ethereal.zing.org* e "windump 3.8" obtido a partir de *http://netgroup-serv.polito.it/windump* . Para o funcionamento dos *sniffers* foi instalada a biblioteca "winpcap" obtida em *http://netgroup-serv.polito.it/winpcap*.

Na outra estação utilizada para testes, um microcomputador Intel Pentium 75 Mhz com 32 MB de memória RAM, foi instalado o sistema operacional Linux Conectiva 6.0 kernel 2.2.17-14cl. Foram instaladas as ferramentas "antisniff", "sentinel" e "sniffdet" para efetuar os testes com as técnicas implementadas. Para o correto funcionamento destas ferramentas, é requerida a instalação das bibliotecas "libpcap" obtida a partir de *http://www.tcpdump.org* e "libnet" obtida a partir de *http://www.packetfactory.net/projects/libnet*. A seguir são apresentados os resultados destes experimentos, classificados por ferramenta e por técnica utilizada:

a) **Ferramenta Antisniff versão 1.1.2**: Esta ferramenta implementa várias técnicas e possui as seguintes opções, em seu arquivo de ajuda:

```
AntiSniff Researchers version 1-1-2
Usage: as -1|2|3|5|6 [-t target -n pkts -e ether -f opts]
  by mudge@l0pht.com
  -1  Enable DNS test (requires -t)
  -2  Enable ICMP time delta test (requires -t, optional -n)
  -3  Enable Ether Ping test (requires -t, optional -e)
  -4  Enable NT EtherPing test (requires -t)
  -5  Enable UDP Echo test (requires -t, optional -n)
  -6  Enable multicast arp test (requires -t)
  -t  target IP address
  -e  target ether addr for raw frame
  -n  number of packets to send (optional)
  -f TCPSYN,SIXTYSIX,THREEWAY for -2 and -5 this defines the
floodtraffic that will be generated (defaults running all types)
```

A seguir são apresentados os resultados obtidos classificados por técnica utilizada:

```
---------------------------------
Ferramenta: AntiSniff
linux --> asniff
windows98 -->windump e ethereal
método : DNS
---------------------------------
comando : #antisniff -1 -t 10.0.0.54

[*]--Results of test--[*]
     status       : SUCCESS
     checktype    : DNSCHECK
     icmpTestType : ---
     promisc cnt  : 1
     errStr       :
     avg1         : 0
     avg2         : 0
     sent1        : 0
     recv1        : 0
     sent2        : 0
     recv2        : 0
     overflow     : NO
    machines list:
      10.0.0.54
```

Esta técnica mostrou-se eficiente utilizando o *sniffer windump*, pois realizou o *DNS lookup reverso* para o endereço IP fictício utilizado, e também para o *sniffer ethereal*, desde que configurado para realizar o DNS *lookup*.

```
--------------------------------
Ferramenta: AntiSniff
linux --> asniff
windows98 -->windump e ethereal
método : ICMP Latência
--------------------------------
comando : #antisniff -2 –f THREEWAY -t 10.0.0.54

[*]--Results of test--[*]
     status       : SUCCESS
     checktype    : ICMPTIMECHECK
     icmpTestType : THREEWAY
     promisc cnt  : 0
     errStr       :
     avg1         : 898
     avg2         : 1201
     sent1        : 10
     recv1        : 10
     sent2        : 10
     recv2        : 10
     overflow     : NO
    machines list:
```

Este teste, utilizando inundação de pacotes THREEWAY não obteve resposta. Para os outros dois tipos de pacotes, TCPSYN e SIXTYSIX[1] foram obtidos os resultados esperados, como mostrado a seguir:

```
comando : #antisniff -2 –f TCPSYN -t 10.0.0.54

 [*]--Results of test--[*]
     status       : SUCCESS
     checktype    : ICMPTIMECHECK
     icmpTestType : TCPSYN
     promisc cnt  : 1
     errStr       :
     avg1         : 875
     avg2         : 7292
     sent1        : 10
     recv1        : 10
     sent2        : 10
     recv2        : 10
     overflow     : NO
    machines list: 10.0.0.54

comando : #antisniff -2 –f SIXTYSIX -t 10.0.0.54

[*]--Results of test--[*]
     status       : SUCCESS
     checktype    : ICMPTIMECHECK
     icmpTestType : SIXTYSIX
     promisc cnt  : 1
     errStr       :
     avg1         : 929
     avg2         : 9242
     sent1        : 10
     recv1        : 10
```

[1] Pacotes SIXTYSIX são pacotes com endereço MAC 66:66:66:66:66:66 cujo objetivo é não ser aceito por nenhuma estação, somente com sniffer executando. Pacotes TCPSYN consistem em inundar com solicitações de conexão (SYN) e pacotes THREEWAY consistem do mesmo princípio do TCPSYN, mas com todos os passos de uma conexão (3way) [SST 02].

```
 sent2        : 10
 recv2        : 10
 overflow     : NO
machines list: 10.0.0.54
```

Para os testes realizados com pacotes ICMP (*etherping*) com falso endereço MAC, apesar de utilizados os endereços FF:00:00:00:00:00, FF:FF:00:00:00:00 e FF:FF:FF:FF:FF:00 não houve a detecção. Também para os testes realizados com a técnica ARP não foram obtidos resultados positivos.

b) **Ferramenta Sentinel 0.9:** Esta ferramenta implementa as técnicas ICMP, ARP e DNS (latência não implementada, apesar de constar na documentação) e possui as seguintes opções em seu arquivo de ajuda:

```
The Sentinel Project (Version 0.9)
Copyright (c) 1999-2000 Subterrain Security Group
Written by bind@subterrain.net

Usage:
  sentinel [method] [options] [-t <target>]

Methods:
  [ -l ICMP Ping Latency test ]
  [ -e Etherping test ]
  [ -a ARP test ]
  [ -d DNS test ]

Options:
  [ -n Number of packets to send ]
  [ -i Set default interface ]
```

A seguir são apresentados os resultados obtidos classificados por técnica utilizada:

```
---------------------------------
Ferramenta: Sentinel
linux --> sentinel
windows98 -->windump e ethereal
método : ICMP
---------------------------------
comando : #sentinel -e -t 10.0.0.54

Running on: 'lab15' running Linux 2.2.17-14cl on a(n) i586

Device: eth0
Target: 10.0.0.54

Sending out 10 fake ICMP echo packets..........
```

```
--------------------------------
Ferramenta: Sentinel
linux --> sentinel
windows98 -->windump e ethereal
método : ARP
--------------------------------
comando : #sentinel -a -t 10.0.0.54

Running on: 'lab15' running Linux 2.2.17-14cl on a(n) i586

Device: eth0
Target: 10.0.0.54

Sending out 10 ARP packets.
Positive to ARP test: 10.0.0.54
--------------------------------
Ferramenta: Sentinel
linux --> sentinel
windows98 -->windump e ethereal
método : DNS
--------------------------------
comando : #sentinel -d -t 10.0.0.54

Running on: 'lab_15' running Linux 2.2.17-14cl on a(n) i586

Device: eth0
Target: 10.0.0.54

Creating 10 fake TCP connections..
Positive to DNS test: 10.0.0.54
```

Percebeu-se nestes testes que com a técnica ICMP não se obteve resultados positivos, mas com as técnicas DNS e ARP foi detectada a presença do *sniffer* .

c) **Ferramenta Sniffdet 0.8:** Esta ferramenta implementa as técnicas DNS, ICMP, ARP e Latência e possui as seguintes opções em seu arquivo de ajuda:

```
sniffdet 0.8
A Remote Sniffer Detection Tool
Usage: sniffdet [options] TARGET
  Where:
  TARGET is a canonical hostname or a dotted decimal IPv4 address

  -i  --iface=DEVICE     Use network DEVICE interface for tests
  -c  --configfile=FILE  Use FILE as configuration file
  -l  --log=FILE         Use FILE for tests log
  -f  --targetsfile=FILE Use FILE for tests target
      --pluginsdir=DIR   Search for plugins in DIR
  -p  --plugin=FILE      Use FILE plugin
  -u  --uid=UID          Run program with UID (after dropping root)
  -g  --gid=GID          Run program with GID (after dropping root)

  -t  --test=[testname]  Perform specific test
      Where [testname] is a list composed by:
        dns         DNS test
        arp         ARP response test
        icmp        ICMP ping response test
        latency     ICMP ping latency test
```

A seguir são apresentados os resultados obtidos classificados por técnica utilizada:

```
--------------------------------
Ferramenta: Sniffdet
linux --> sniffdet
windows98 -->windump e ethereal
método : DNS
--------------------------------
comando : #sniffdet -t dns 10.0.0.54



------------------------------------------------------------
Sniffdet Report
Generated on: Mon Jul 14 18:39:05 2003
------------------------------------------------------------
Tests Results for target 10.0.0.54
------------------------------------------------------------
Test: DNS Test Watch for DNS queries for hostnames who don't exist
Validation: OK
Started on: Mon Jul 14 18:38:38 2003
Finished on: Mon Jul 14 18:39:05 2003
Bytes Sent: 0
Bytes Received: 83
Packets Sent: 0
Packets Received: 1
------------------------------------------------------------
RESULT: POSITIVE
------------------------------------------------------------

--------------------------------
Ferramenta: Sniffdet
linux --> sniffdet
windows98 -->windump e ethereal
método : Latência
--------------------------------
comando : #sniffdet -t latency 10.0.0.54

------------------------------------------------------------
Sniffdet Report
Generated on: Mon Jul 14 20:33:46 2003
------------------------------------------------------------
Tests Results for target 10.0.0.54
------------------------------------------------------------
Test: Latency test Ping response with custom packet flood
Validation: OK
Started on: Mon Jul 14 20:33:33 2003
Finished on: Mon Jul 14 20:33:46 2003
Bytes Sent: 6293062
Bytes Received: 0
Packets Sent: 101501
Packets Received: 91
------------------------------------------------------------
RESULT:
Normal time: 0.7
Flood round-trip min/avg/max: 8.9/11.2/42.1 ms
------------------------------------------------------------
------------------------------------------------------------
Number of valid tests: #1
Number of tests with positive result: #0
------------------------------------------------------------
```

Os testes com as técnicas ICMP, ARP e latência não obtiveram resultados positivos, entretanto a técnica DNS funcionou perfeitamente.
Em seguida foram instaladas as ferramentas "AntiSniff" para Windows 9x e "Promiscan" para Windows NT/2K/XP e disparados os testes contra a estação Windows 98.
Acrescentou-se neste momento uma outra estação com Windows XP Professional para executar a ferramenta "promiscan". A seguir é apresentada a Tabela 4.5 comparativa resultante das técnicas utilizadas contra Windows 98, classificadas por ferramenta utilizada. Para os próximos testes realizados com diferentes sistemas operacionais, serão apresentadas somente as tabelas comparativas resultantes.

TABELA 4.5 – Resultados obtidos com *sniffer* em Windows 98

	ICMP	ARP	DNS	LATÊNCIA	Sniffer
AntiSniff			✓	✓	Windump
			✓	✓	Ethereal
Sentinel		✓	✓		Windump
		✓	✓		Ethereal
SniffDet			✓		Windump
			✓		Ethereal
Promiscan XP		✓			Windump
		✓			Ethereal
AntiSniff W98		✓	✓		Windump
		✓	✓		Ethereal

Percebeu-se que a detecção de interface em modo promíscuo foi independente do *sniffer* utilizado, desde que o mesmo utilize este modo de operação.
Um outro teste foi realizado contra uma estação Windows XP Professional com o sniffer Ethereal, que apresentou poucos resultados significativos, ou seja houve a detecção apenas com a técnica de latência da ferramenta AntiSniff para Linux, e a técnica ARP das ferramentas Promiscan e AntiSniff para Windows. Cabe salientar que a ferramenta AntiSniff para linux era uma versão de pesquisa e a versão para Windows era uma outra versão comercial mais recente. Deve-se a isto o fato de algumas técnicas da mesma ferramenta diferirem nos resultados. Outro aspecto que deve ser considerado é a interface de rede e o driver utilizado nesta estação. Como tratava-se de uma máquina recém adquirida, os drivers e a interface de rede (Intel Pro 100 VENetwork) eram muito recentes e provavelmente por esta razão não responderam às ferramentas utilizadas.
Para os próximos testes, o *sniffer* utilizado será somente o *tcpdump*, haja visto não ser este o fator decisivo da detecção.

4.4.2 Testes de Detecção em Sistemas Operacionais Linux e BSD

Para os testes de detecção com *sniffers* executando em Sistemas Operacionais Linux, foram utilizadas as mesmas instalações anteriores. Uma alteração é que onde rodava o Windows 98 foi instalada uma nova versão do Linux, o Conectiva 7.0 kernel 2.2.19, e o

sniffer “tcpdump”. As ferramentas “antisniff”, “sentinel” e “sniffdet” foram disparadas a partir do Linux Conectiva 6.0 kernel 2.2.17. A ferramenta “promiscan”, a partir do Windows XP e a ferramenta “antisniff” para Windows em uma outra estação Windows 98. A seguir é apresentada a Tabela 4.6 resultante dos testes realizados.

TABELA 4.6 – Resultados obtidos com *sniffer* em Linux

	ICMP	ARP	DNS	LATÊNCIA
AntiSniff	✓		✓	
Sentinel	✓	✓	✓	
SniffDet			✓	
Promiscan XP		✓		
AntiSniff W98		✓	✓	

Uma curiosidade nestes testes, diz respeito á técnica ICMP, onde a ferramenta “antisniff” em sua opção 3 (icmp) não mostrou resultados positivos e sua opção 4 (icmp NT) apresentou resposta positiva. Isto se deveu ao fato de que na injeção de pacotes com endereços MAC falsos, para a opção 3 por *default* é utilizado um MAC do tipo 66:66:66:66:66:66 o qual não foi respondido. Na opção 4 foram utilizados pacotes com endereço MAC FF:FF:00:00:00:00, obtendo-se respostas positivas. Com diversos testes realizados concluiu-se que a partir dos primeiros 16 bits ativados (FF:00:00:00:00:00) a estação linux considerou como *broadcast* e respondeu, revelando a presença do *sniffer*.
Um outro conjunto de testes foi realizado, desta vez contra o Sistema Operacional FreeBSD. Foi realizada a instalação completa do FreeBSD versão 4.7, por ser a disponível no momento, e utilizado o *sniffer* “tcpdump”. A seguir é apresentada a Tabela 4.7 resultante dos testes contra o Sistema Operacional FreeBSD, utilizando as ferramentas já descritas, tanto em Linux como em Windows.

TABELA 4.7 – Resultados obtidos com *sniffer* em FreeBSD

	ICMP	ARP	DNS	LATÊNCIA
AntiSniff				✓
Sentinel		✓	✓	
SniffDet			✓	
Promiscan XP		✓		
AntiSniff W98		✓	✓	

Como foi percebido através dos testes realizados, existe uma variação muito grande entre técnicas utilizadas, versões de sistemas operacionais e ferramentas utilizadas. As técnicas apresentadas realmente comprovaram que a detecção de *sniffer* é possível e que muito embora dependa dos sistemas operacionais utilizados, pode-se obter uma composição das mesmas, com um estudo aprofundado das diferentes técnicas implementadas, resultando em uma ferramenta muito mais poderosa e abrangente. Mais testes precisam ser realizados, e outras técnicas implementadas, mas com certeza, os Sistemas de Detecção de Intrusão podem e devem incluir em suas implementações esta funcionalidade, agregando valor e confiabilidade à detecção de intrusos, mais precisamente na detecção de ferramentas de monitoração e captura de tráfego de rede.

5 Conclusão

Durante muito tempo considerou-se difícil ou quase impossível detectar *sniffers* em uma rede, pois eles têm a característica de serem passivos, ou seja, somente capturam pacotes. As técnicas apresentadas neste trabalho mostram que a detecção de *sniffers* é possível e que se forem utilizadas em conjunto, aumentam consideravelmente a eficácia da detecção. O resultado deste conjunto de técnicas pode servir como nova funcionalidade para os Sistemas de Detecção de Intrusão (IDS), muito utilizados atualmente como parte de um sistema de defesa contra ataques nas redes. Obviamente que algumas destas técnicas funcionam para determinados sistemas operacionais, e outras não. O certo é que são um grande passo no sentido de contribuir com a prevenção de conseqüências indesejáveis na eminência de um ataque em uma rede de computadores, evitando sua proliferação e o comprometimento de outros sistemas ou redes.

Os sistemas de detecção de intrusão precisam estar em permanente desenvolvimento, pois novas técnicas de ataque são criadas diariamente e, portanto, necessitam conhecer as assinaturas de novos ataques para manterem-se atualizados. Neste intervalo de desconhecimento de novos ataques, as técnicas de detecção de *sniffers* podem ser empregadas auxiliando no processo de defesa, pois podem revelar a presença de um intruso tanto externo quanto interno no momento que ele passe a capturar informações.

Estas técnicas têm a função de agregar valor aos IDSs, possibilitando um maior raio de ação na defesa dos sistemas computacionais. Como o *sniffer* é uma das ferramentas mais utilizadas em um ataque, acrescentar técnicas de detecção de *sniffers* aos sistemas de detecção de intrusão vem a ser mais uma opção indispensável a se considerar na implementação destes sistemas.

Dentre as técnicas testadas, as que apresentaram melhores resultados foram aquelas onde utilizaram-se as características de funcionamento do sistema de nomes de domínio (DNS), e do protocolo ARP. O método utilizando DNS apresentou uma eficácia de 100%, desde que os *sniffers* estivessem aptos a efetuar o DNS *lookup* reverso de forma automática. O método utilizando o protocolo ARP foi o que mais apresentou resultados significativos, ou seja, nas variações de sistemas operacionais e versões, foi aquele que mais revelou a presença do *sniffer*. No caso da técnica de latência, os resultados foram pouco significativos, pois o cenário e o momento dos testes levaram à obtenção de falsos positivos. Pelo fato do protocolo ARP não possuir nenhum tipo de autenticação, pode-se considerar que técnicas baseadas nele são as mais indicadas. No entanto, deve-se considerar que na implementação das técnicas, o conjunto delas aumenta o escopo da detecção, portanto todas devem ser consideradas.

Os sistemas operacionais estão evoluindo, novos protocolos adotados, novos ataques estão sendo desenvolvidos, novas vulnerabilidades encontradas, ferramentas novas estão disponíveis. Técnicas de defesa, métodos e ferramentas precisam ser desenvolvidas para acompanhar esta evolução. Apesar de parecer uma corrida contra o tempo, também tem sua contribuição, estimulando o desenvolvimento da ciência da computação, encontrando novas dificuldades e como conseqüência, novos desafios a serem superados. Além da contribuição desta pesquisa para com a defesa dos sistemas computacionais, proporcionou também um engrandecimento pessoal, aprimorando os conhecimentos acerca da área de segurança, mais especificamente na questão da detecção de intrusão. A disseminação deste conhecimento torna-se fundamental, pois a cada dia somos mais dependentes da tecnologia que criamos.

6 Trabalhos Futuros

Com a disseminação destas técnicas, certamente os atacantes também levarão em consideração a existência de ferramentas de detecção com estas funcionalidades, portanto desenvolvendo novos métodos de evasão. Novos testes necessitam ser realizados, pois existe uma grande diversidade de versões de sistemas operacionais, versões de *drivers* de interfaces dependentes do fabricante, centenas de *sniffers* configuráveis, novos protocolos de comunicação, além da possibilidade de se descobrir novas técnicas explorando outros protocolos que possam ser empregadas na detecção de *sniffers*.

As técnicas conhecidas precisam ser aprimoradas e novas técnicas serem criadas, de preferência explorando características independentes do sistema operacional, devido à diversidade de sistemas e versões. A técnica envolvendo o protocolo SNMP também ainda não tem experimentos conclusivos, portanto uma área para possíveis trabalhos de pesquisa.

Há que se considerar para novas pesquisas também as redes sem fio, muito difundidas atualmente e sem nenhum estudo conhecido até o momento sobre *sniffers* e a possibilidade de detecção.

A área de segurança é uma das áreas mais dinâmicas da computação, necessitando atualização e desenvolvimento constante, pois cada vez mais os serviços essenciais são controlados por computadores. Portanto tem-se aí um excelente campo para novos trabalhos e pesquisas, pesquisando para o bem da ciência e evolução da humanidade, nunca para apropriação indevida de informações em benefício próprio.

Referências

[ABD 2002] ABDELALLAHELHADJ, H. et al. **An Experimental Sniffer Detector: SnifferWall**. Sécurité des Communications sur Internet-SECI02, France, Sept. 2002. Disponível em: <http://www.lsv.enscachan.fr/~goubault/SECI-02/Final/actes-seci02/pdf/Abdelallahelhadj.pdf>.Acesso em:14 maio 2003.

[BAC 2001] BACE, R.; MELL, P. **Intrusion Detection Systems**. Scotts Valley: NIST, 2001. (NIST Special Publication SP 800-31).

[BAL 98] BALASUBRAMANIYAN, J.S. et al. **An Arquitecture for intrusion detection using autonomous agents**. West Lafayette: Coast Group, Universidade de Purdue, 1998. (Technical Report 98-05).

.

[BIN 2001] BIND. **The Sentinel Project**. Disponível em: <http://www.packetfactory.net/Projects/sentinel/sentinel-1.0.tar.gz>. Acesso em: 20 maio 2003.

[BOW 2000] BOWERS, B. **Dsniff and Switched Network Sniffing**. SANS Institute, 2000. Disponível em: <http://www.giac.org/practical/Brad_Bowers.doc>. Acesso em: 14 maio 2003.

[CAM 2001] CAMPELLO, R.S.; WEBER, R.F.; SERAFIM, V.S.; RIBEIRO, V.G. O Sistema de Detecção de Intrusão Asgaard. In: WORKSHOP EM SEGURANÇA DE SISTEMAS COMPUTACIONAIS, 2001, Florianópolis. **Anais do WSeg**. Florianópolis: UFSC, 2001.

[CHE 99] CHEUNG, R. et al. **The design of GrIDS:** a graph-based intrusion detection system. Davis: Universidade da Califórnia, 1999. (Technical Report CSE-99-2)

[COM 98] COMER, D.E. **Interligação em Rede com TCP/IP.** Rio de Janeiro: Campus,1998. v.1.

[COM 2001] COMER, D.E. **Computer Networks and Internets.** 3rd ed. Upper Saddle River: Prentice Hall, 2001.

[CRO 95] CROSBIE, M. et al. **Active defense of a computer system using autonomous agents**. West Lafayette: Coast Group, Universidade de Purdue, 1995. (Technical Report 95-008).

[CRO 96] CROSBIE, M. et. al. **IDIOT – Users Guide**. West Lafayette: Coast Group, Universidade de Purdue, 1996.

[DEN 87] DENNING, D.E. et al. **A prototype IDES – a real-time intrusion detection expert system.** [S.l.]: Computer Science Laboratory, SRI Internacional, 1987. Technical Report.

[DHA 2002] DHAR, S. **Switchsniff**. Seattle: Linux Journal, Mar. 2002. Disponível em <http://www.linuxjournal.com/article.php?sid=5869>. Acesso em: 14 maio 2003.

[DOW 2000] DOWNEY, J. **Sniffer Detection Tools and Countermeasures**. SANS Institute, 2000. Disponível em: <http://www.giac.org/practical/GSEC/James_Downey_GSEC.pdf>. Acesso em: 14 maio 2003.

[GRA 2000] GRAHAM, R. **Sniffing (network wiretap, sniffer) FAQ**. Sept. 2000. Disponível em: <http://www.robertgraham.com/pubs/sniffing-faq.html>. Acesso em: 14 maio 2003.

[GRU 98] GRUNDSCHOBER, S. **Sniffer Detector Report**. Zurich: IBM Research Division, Zurich Research Laboratory, Global Security Analysis Lab., 1998. Disponível em: <http://packetstormsecurity.nl/unix/ids/>. Acesso em: 14 maio 2003.

[HAU 99] HAUSWIRTH, M.; JAZAYERI, M. A component and communication model for push systems. In: JOINT EUROPEAN SOFTWARE ENGINEERING CONFERENCE, ESEC/FSE, 7., 1999, Toulouse, France. **Proceedings…** Toulouse: ACM Press, 1999.

[HAT 2003] HATCH, B.; LEE, J. **Hacking Linux Exposed**. 2nd ed. Osborne: McGraw-Hill, 2003.

[HEB 90] HEBERLEIN, L. et al. A network security monitor. In: IEEE SYMPOSIUM ON RESEARCH IN SECURITY AND PRIVACY, 1990, Oakland, USA. **Proceedings…** Oakland: IEEE, 1990.

[HOC 93] HOCHBERG, J. et al. NADIR: an automated system for detecting network intrusion and misuse. **Computers & Security**, [S.l.], p. 235-248, May 1993.

[HON 2001] HONEYNET PROJECT. **Know Your Enemy:** Revealing the Security Tools, Tactics, and Motives of the Blackhat Community. 1st ed. Boston: Addison-Wesley, 2001.

[JOC 2001] JOCH, A. Network Sniffers. **Computerworld Magazine**, Framingham, v. 35, p. 53, 23 July 2001.

[KUE 2001] KUEHL, K. **The Honeynet Project**. México, 2001. Disponível em: <http://project.honeynet.org>. Acesso em: 15 maio 2003.

[MCC 2000] McCLURE, S.; SCAMBRAY, S. **Switched networks lose their security advantage due to packet-capturing tool**. Infoworld. USA, Sept. 2000. Disponível em <http://www.infoworld.com/articles/op/xml/00/05/29/000529opswatch.xml>. Acesso em: 14 maio 2003.

[MÉ 2001] MÉ, L.; CÉDRIC, M. **Intrusion Detection – A Bibliography.** Sup'elec, Rennes, France, Sept. 2001. Disponível em: <http://www.supelec-rennes.fr/ren/perso/cmichel/bibid_raid2001.ps>. Acesso em: 14 maio 2003.

[MEL 2001] MELLANDER, J. UPDATE – A stealthy Sniffer Detection**. Information Security Bulletin,** USA, Apr. 2001. Disponível em: <http://www.lbl.gov/ICSD/Security/news/mellander_update-art1.pdf>. Acesso em: 14 maio 2003.

[NOR 2002] NORTHCUTT, S. et al. **Desvendando Segurança em Redes.** Rio de Janeiro: Campus, 2002.

[POR 97] PORRAS, P.A.; NEUMANN, T.L. EMERALD: Event monitoring enabling responses to anomalous live disturbances. In: NATIONAL INFORMATION SYSTEMS SECURITY CONFERENCE, 20., 1997, Baltimore, USA. **Proceedings…** Baltimore: NIST, 1997.

[RAY 2001] RAY, J. et al. **Maximum Linux Security**. 2nd ed. Indianápolis: Sams Publishing, 2001.

[REI 2002] REIS, A.S.; SOARES, M. **Um Sistema de Testes para Detecção Remota de Sniffers**. UFPR, 2002. Disponível em: <http://sniffdet.sourceforge.net>. Acesso em: 25 maio 2003.

[SAN 2001] SANAI, D. **Detection of Promiscuous Nodes Using ARP Packets**. Securityfriday, 2001. Disponível em: <http://www.securityfriday.com>. Acesso em: 14 maio 2003.

[SAV 98] SAVAGE. **Neped - Network promiscuous ethernet detector**. Disponível em: <http://www.apostols.org/projectz/neped>. Acesso em: 20 out. 2002.

[SKO 2002] SKOUDIS, E. **Counter Hack:** A Step-by-Step Guide to Computer Attack and Effective Defenses. Indianápolis: Prentice-Hall PTR, 2002.

[SIP 2000] SIPES, S. **Why your switched network isn't secure**. SANS Institute, 2000. Disponível em <http://www.sans.org/resources/idfaq/switched_network.php>. Acesso em: 14 maio 2003.

[SNA 91] SNAPP, S.R. et al. DIDS - Distributed intrusion detection system. In: NATIONAL COMPUTER SECURITY CONFERENCE, 14., 1991, Washington, USA. **Proceedings…** Washington: NIST, 1991.

[SON 98] SONG, D. **Dsniff.** Disponível em: <http://www.monkey.org/~dugsong/dsniff>. Acesso em: 14 maio 2003.

[SPA 2000] SPAFFORD, E.H.; ZAMBONI, D. **Intrusion Detection using autonomous agents**. West Lafayette: Coast Group, Universidade de Purdue, 2000.

[SST 2002] SECURITY SOFTWARE TECHNOLOGY. **AntiSniff – Promiscuous Mode Interface Detector**. Security Software Technology. USA, Sept. 2002. Disponível em: <http://www.securitysoftwaretech.com/antisniff>. Acesso em: 25 março 2003.

[STE 94] STEVENS, W.R. **TCP/IP Illustrated -The Protocols**. Boston: Addison-Wesley, 1994.

[VER 99] VERWOERD, T. **Active Network Security**. University of Canterbury, New Zealand, Nov. 1999. Disponível em: <http://www.cosc.canterbury.ac.nz/research/reports/HonsReps/1999/hons_9909.pdf>. Acesso em: 10 dezembro 2002.

[WU 98] WU, D.; WONG, F. **Remote Sniffer Detection.** Universidade de Berkeley. USA, Dec. 1998. Disponível em: <http://www.cs.berkeley.edu/~daw/teaching/cs261-f98/projects/final-reports/fredwong-davidwu.ps>. Acesso em: 14 maio 2003.