Centro Universitário de Brasília – UniCEUB

Faculdade Tecnologia e Ciências Sociais Aplicadas – FATECS

Curso de Engenharia da Computação

Luís Cláudio Burdino

Honeypots como Estratégia de Segurança para Redes

Trabalho de conclusão de curso apresentado à banca examinadora do Centro Universitário de Brasília, como exigência parcial para conclusão do curso de Engenharia de Computação, sob orientação do Professor M.Sc. Antonio José Gonçalves Pinto.

Brasília – Distrito Federal

Junho/2009

RESUMO

Um dos problemas que os Administradores de Redes precisam tratar é a dificuldade em capturar e registrar informações relevantes sobre os ataques direcionados à sua rede. Este trabalho de pesquisa se propõe a implantar, testar e analisar a solução de Honeypot de baixa interatividade, que capture as informações sobre os ataques direcionados à rede interna. Adicionalmente, responder como estes dados podem auxiliar o administrador em incrementar o nível de segurança de seu ambiente.

Palavras chaves: honeypot, honeyd, arpd, segurança de redes.

ABSTRACT

One of the problems that Network Administrators need to address is the difficulty to capture and record relevant information about the attacks and attackers. This project has the proposal to deploy, test and analyze a low-interaction honeypot solution that captures information about the attacks targeted to the internal network. And responding how such data can help the Network Administrators to enhance the security level of their environment.

Keywords: honeypot, honeyd, arpd, network security.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

ASCII	*American Standard Code for Information Interchange*
APIDS	*Aplication Protocol-Based IDS*
AT&T	*American Telephone and Telegraph.*
CSIRTs	*Computer Security Incident Response*
DMZ	*DeMilitarized Zone*
DNS	*Domain Name Server*
DDoS	*Distributed Denial of Service*
DoS	*Denial of Service*
DTK	*Deception Toolkit*
IDS	*Intrusion Detection System*
ICMP	*Internet Control Message Protocol*
IP	*Internet Protocol*
IRC	*Internet Relay Chat*
FTP	*File Transfer Protocol*
HIDS	*Host-Based IDS*
HTTP	*Hyper Text Transfer Protocol*
IANA	*Internet Assigned Numbers Authority*:
LBL	*Lawrence Berkley Laboratory*
MAC	*Media Access Control.*
NAT	*Network Adress Translation*
NDS	*Novell Directory System*
NIDS	*Network-Based* IDS
NTFS	*Network File System*
PAT	*Port Adress Translation*
PIDS	*Protocol-Based* IDS.
POP3	Post Office Protocol 3.
RFC	*Request For Comment.*
S.O.	Operational System, Sistema Operacional.
SMTP	Simple Mail Transfer Protocol.
SQL	*Structured Query Language, ou Linguagem de Consulta Estruturada.*
TCP	*Transmission Control Protocol.*
TTL	*Transistor-Transistor Logic.*
UDP	*User Datagram Protocol.*
VPN	*Virtual Private Network*
www	World Wide Web significa, rede de alcance mundial; é um sistema de documentos em hipermídia que são interligados e executados na Internet.

1 INTRODUÇÃO

1.1 Motivação

O crescimento da demanda por recursos de rede aumentou muito nos últimos anos, resultando na necessidade de rápido desenvolvimento de sistemas e tecnologias que respondessem a essas expectativas.

Como a utilização dos serviços "*on-line*" cresce muito rápido, como conseqüência natural é o incremento, também, das vulnerabilidades nos sistemas operacionais: vírus, *worms* e o acesso ilegal, a informações confidenciais, por pessoas não autorizadas.

Há pouco tempo, a violação das informações nas empresas ocorria através de indivíduos com grande saber técnico, porém, ultimamente, esses conhecimentos podem ser dispensados, pois existem na Internet várias ferramentas e publicações que facilitam e até ensinam como realizar os ataques a pessoas ou a sistemas de empresas, sem a necessidade de grande domínio das técnicas a serem utilizadas.

A gravidade dos ataques a informações tem suas variações, que podem ser desde a simples curiosidade, ou até mesmo a paralisação de sistemas inteiros, causando vários problemas graves para as empresas que não dispõem de serviços de segurança que impeçam que estes ataques aconteçam e que protejam todo o sistema.

A rede dentro do contexto pessoal e organizacional, passou a ser vista como grande suporte ao negócio que elas mantêm, sendo essencial sua disponibilidade e segurança.

Estando a segurança dessas redes expostas a diversas falhas e vulnerabilidades que podem comprometer a sua disponibilidade, diversos métodos e ferramentas foram desenvolvidos com objetivo de corrigir as falhas e evitar ataques, no entanto, o problema é que muitos se resumem a soluções passivas de prevenção e resposta aos ataques. Surgem então os sistemas que tem como objetivo antecipar-se aos ataques através da monitoração, captura e leitura de dados, conhecidos como *honeypots*.

Neste contexto, atua o *honeypot*, opção de estudo de ataques e de segurança da rede e tema central deste projeto, que tem como objetivo a implantação de solução em sistema operacional livre, visando à análise e entendimento de comportamentos, ferramentas utilizadas no ataque e até mesmo o tipo de atacantes, possibilitando assim a criação de novas ferramentas e rotinas de proteção pró-ativas.

1.2 Objetivo Geral

Implementar uma solução de *honeypot,* onde através da coleta das informações sobre ataques existentes, promover maior facilidade aos administradores de rede no desenvolvimento pró-ativo de novas ferramentas, bem como permitir uma melhor compreensão das ameaças e conseqüentemente auxiliar a defesa das organizações contra esses ataques.

1.3 Objetivos Específicos

Criar uma solução de Honeypot utilizando o Sistema Operacional livre, OpenBSD, os *daemons, Honeyd, Arpd,* e uma interface gráfica para que o administrador possa analisar os resultados.

1.4 Justificativa e Relevância do Tema

Com o avanço da Internet, milhões de pessoas e instituições estão interligadas via rede mundial de computadores. A segurança passou a ser uma necessidade fundamental tornando-se foco de discussão das comunidades envolvidas com a tecnologia de redes [SCHNEIER, 2001]. Nesse contexto, os administradores de rede procuraram formas de poderem tornar suas redes mais seguras. Diversos mecanismos e soluções de segurança foram sendo cada vez mais utilizados. Entre esses, os mais difundidos são os *firewalls* e os Sistemas de Detecção de Intrusão (IDS).

Através de uma postura meramente passiva, começou-se a buscar atrair os atacantes para sistemas especialmente construídos de modo a propiciar a inspeção

e o estudo das técnicas e estratégias de ataque, adotando assim uma postura mais ativa no combate aos incidentes na Internet. Tais sistemas foram denominados *honeypots* e surgiram para atender a necessidade de compreender o perfil dos ataques bem como de detectar as últimas tendências relativas às vulnerabilidades mais exploradas [FRANCO, 2003].

1.5 Escopo do Trabalho

Desenvolver uma solução de *honeypot* baseada em software livre utilizando o *honeyd,* com sistema operacional livre OpenBSD, permitindo que todos os recursos de hardware fiquem dedicados para a função especificada Será feito recompilação do sistema operacional com *deamon*[1] *honeyd* para que ele rode no kernel do sistema operacional, baseada em *honeypots* para capturar os ataques feitos à rede.

1.6 Resultados Esperados

O resultado esperado com este trabalho de pesquisa, consiste no desenvolvimento de uma solução de *honeypot* que facilite o trabalho dos administradores no que se refere à coleta de dados dos ataques originados em sua rede, para que esses dados possam ser analisados de forma gráfica e eficaz. Além disso, ao longo do tempo, o administrador de dados terá condições de criar uma base de conhecimento, à partir dos registros dos ataques mapeados, com intuito de desenvolver melhorias e contramedidas de segurança aos ataques.

1.7 Estrutura do Trabalho

A estrutura do trabalho está disposta em capítulos, os quais serão apresentados de acordo com a definição a seguir:

Capítulo I - define a motivação, o objetivo e estrutura do trabalho que será desenvolvido.

[1] Programa que é executado num computador e está pronto a receber instruções/pedidos de outros programas para a execução de determinada acção. [Disponível em:<http://www.prof2000.pt/users/pedroff/dicio.htm#D>. [Acesso em: 05.05.2009].

Capítulo II - apresenta as principais soluções existentes no mercado e as conseqüências do problema identificado e principais benefícios da solução proposta.

Capítulo III - traz os conceitos pertinentes a pesquisa realizada, relativos à área de segurança de redes e *honeypot*, visando o bom entendimento do trabalho como um todo.

Capítulo IV - descreve o modelo proposto de *honeypot*, sua instalação e configuração.

Capítulo V - apresenta detalhadamente a aplicação da solução proposta e seus resultados.

Capítulo VI - conclusão do trabalho.

2 APRESENTAÇÃO DO PROBLEMA

2.2 Problema Identificado

Atualmente a maior parte das soluções de segurança baseia-se em fatos e padrões de ataque conhecidos. No entanto, para tornar conhecidos os diferentes ataques que surgem todos os dias, é necessário um mecanismo para capturar os ataques e estudá-los. [VERÍSSIMO, 2001]

Muitos ataques atualmente são barrados pelos sistemas de detecção de Intrusão - (*IDS - Intrusion Detection System*) que bloqueiam inicialmente os ataques conhecidos e aqueles que fogem da normalidade da rede, mas não se sabe como este ataque é executado ou mesmo de onde é proveniente.

As soluções baseadas em *software* livre que atuam como detectores de intrusão (IDS) ainda não dispõem de mecanismos capazes de detectar determinados ataques mais sofisticados, não sendo possível aos administradores de rede identificar informações importantes sobre o mesmo, dificultando a realização de medidas preventivas necessárias para controlar ou evitar a ocorrência desses eventos [VAZ, 2004]. Além disso, a falta de uma interface de gerenciamento desses eventos contribui para a complexidade dessas análises e elaboração de soluções efetivas para solucionar esse problema. Mesmo os sistemas atuais que fornecem essas informações, são baseados em assinaturas de ataques conhecidos, não dando subsídios ao administrador de rede para lidar com as diversidades e complexidade de situações vividas no seu dia a dia.

Desta forma, a carência tecnológica identificada nos sistemas de detecção de intrusão baseados em software livre, evidencia e reforça a necessidade de estudo e desenvolvimento de soluções para esse tipo de problema, como proposto neste trabalho, no que diz respeito a dificuldade dos administradores em obter informações detalhadas sobre os atacantes da rede.

2.3 Soluções Existentes

Algumas das soluções de segurança existentes no mercado são:

2.3.1 Mantrap

Mantrap é um sistema comercial desenvolvido para atuar na plataforma Solaris, sendo utilizado tanto para segurança externa quanto interna.

Ele cria quatro ambientes, chamados de jaulas, podendo assim suportar até quatro sistemas operacionais. Com apenas uma máquina são criados quatro sistemas visíveis na rede, cada um com sua própria interface, podendo emular qualquer aplicativo compatível com o sistema operacional da jaula. O *Mantrap* somente limita aqueles aplicativos que interagem como o *kernel*[2].

Os ambientes são isolados, sendo personalizados distintamente. O módulo de geração de conteúdo gera, por exemplo, mensagens de e-mail utilizando nomes conhecidos da instituição, diferentes para cada ambiente criado [*Matrap*, 2009].

Todas as atividades relevantes ocorridas dentro de cada jaula são armazenadas localmente ou no servidor remoto de *logs*, além de possuir uma ferramenta gráfica de análise de eventos.

2.3.2 Specter

O *Specter* é projetado para ambiente Windows, para organizações comerciais de pequeno e grande porte. Pode monitorar até quatorze portas de TCP, sendo sete de armadilhas e sete de serviços. As armadilhas bloqueiam e registram as tentativas de ataques, conforme pode ser observado na figura 2.1; as portas de serviços interagem com o invasor, emulando o aplicativo de acordo com o serviço utilizado.

Pode emular até quatorze sistemas operacionais diferentes (*Windows* 98, *Windows* NT, *Windows* 2000, *Windows* XP, *Linux*, *Solaris*, Tru64 (Digital *Unix*), NeXTStep, *Irix*, *Unisys Unix*, AIX, Maços, MacOS X, FreeBSD), mas não consegue emular na pilha IP, assim um atacante fazendo uso de uma ferramenta de obtenção de impressões digitais ativa pode detectar o *honeypot* [*Specter*, 2009].

[2] O *kernel* consiste, em abstrair a interface do *hardware*, permitindo que processos utilizem este recurso concorrentemente, de forma segura e padronizada. [Disponível em:<http:// pt.wikipedia.org/wiki/Kernel>, Acesso em: 05.04.2009].

Possui grande variedade de configuração, características de notificação, banco de dados dos incidentes, além da facilidade no uso. A figura 2.1 tela do *Specter*.

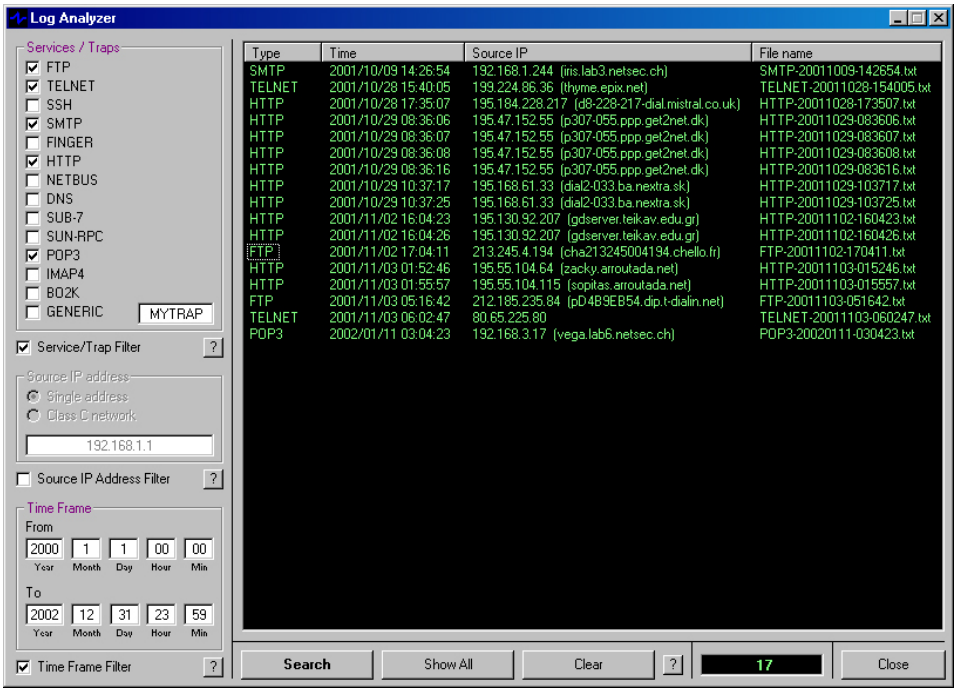

Figura 2.1 - Tela de gerenciamento de incidentes do *Specter*.

2.3.3 NetBait

O *NetBait* é uma solução de segurança de rede que redireciona ataques contra espaços de IP não utilizados para “fazendas de *honeypots*”, anulando o intruso pelo uso da ilusão. A figura 2.2 ilustra o ambiente criado pelo NetBait.

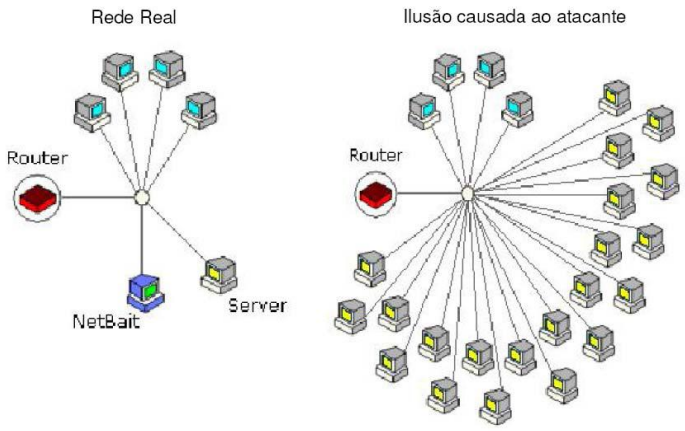

Figura 2.2 - Ambiente ilusório criado pelo NetBait.

O *NetBait* cria esses ambientes projetando um desvio da rede real, assim os nós da rede ficam cercados por múltiplos nós falsos; cada nó falso pode ser configurado com qualquer combinação de sistemas operacionais, serviços e aplicativos. Possui gerenciamento remoto centralizado e configuração de comportamento dinâmico [*NetBait*, 2009].

2.3.4 Smoke Detector

O *Smoke Detector* emula até dezenove sistemas operacionais por máquina entre Linux, Solaris8, HP-UX, AIX4, FreeBSD4, AS400, WindowsNT4 e Windows2000, confundindo e estendendo o tempo de resposta ao invasor. A figura 2.3 ilustra o funcionamento do *Smoke Detector*.

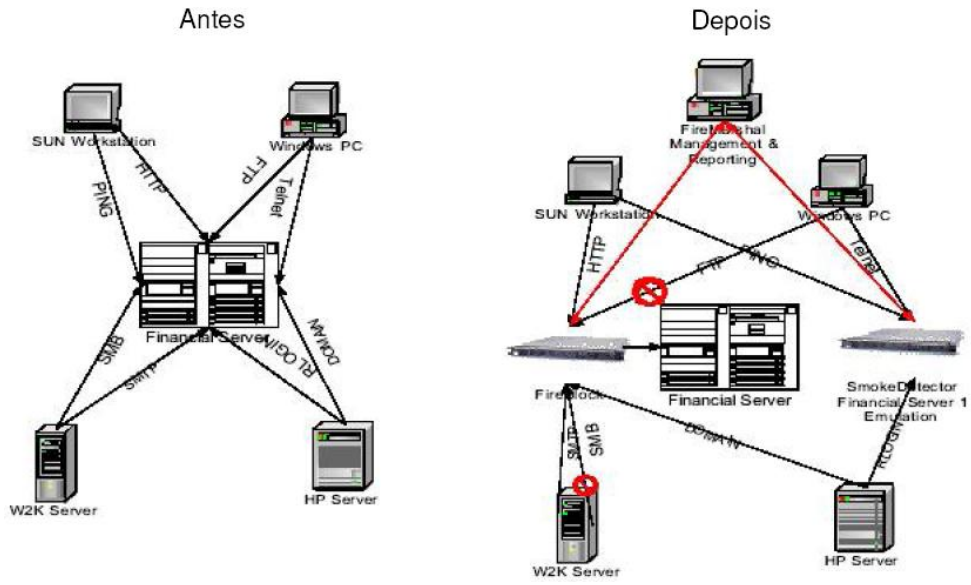

Figura 2.3 - Funcionamento *Smoke Detector*.

Ele é usado para disfarçar servidores importantes, enviando alerta de tentativas de invasão ao administrador da rede, para permitir o bloqueio dos recursos reais em tempo hábil [*Smoke Detector*, 2009].

2.3.5 KFSensor

O KFSensor é um *honeypot* utilizado em plataforma Windows, projetado principalmente para proteção. Ele é preparado para bloquear ataques de recusa de serviço e estouro de *buffer*; registra os *logs* do atacante, podendo ser filtrado de várias maneiras para facilitar a análise em determinada porta ou protocolo [KFSensor, 2009]. A figura 2.4 ilustra o monitoramento do KFSensor.

Figura 2.4 - Monitoramento do KFSensor.

2.3.6 Honeypots Free / OpenSource

- *BackOfficer Friendly:*

É um programa para Windows que emula alguns serviços básicos de rede, como TELNET, FTP, SMTP, POP3. Quando o *BackOfficer Friendly* recebe conexão para um dos serviços citados, gera algumas respostas falsificadas, distraindo o invasor e proporcionado tempo para o bloqueio do ataque. Ele somente consegue monitorar sete portas por vez [*BackOfficer Friendly*, 2009]. É indicado para iniciantes que desejam verificar o funcionamento de um *honeypot*. A figura 2.5 demonstra a tela do BackOfficer Friendly.

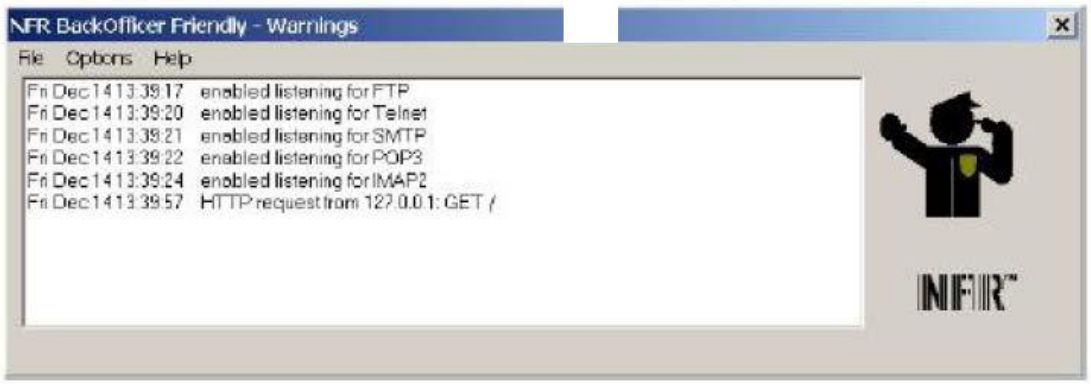

Figura 2.5 - Tela do BackOfficer Friendly.

- *Honeyd:*

O *Honeyd* é um dos principais aplicativos utilizados na construção de *honeypots*, sendo projetado para sistemas Unix. Ele emula centenas de sistemas operacionais, podendo simular aplicações no espaço de endereços IP não utilizados, utilizando vários simultaneamente.

Pode assumir a identidade de um IP que não esteja sendo utilizado e interagir com um atacante, respondendo suas requisições e registrando seu ataque; pode monitorar todas as portas baseadas em UDP e TCP. Grande facilidade de configuração. *Honeyd* é *OpenSource*[3], ou seja, permite alteração no código fonte, podendo surgir novos *scripts* que emulam novos serviços [*Honeyd*, 2009].A figura 2.6 demonstra o monitoramento de endereços Ip não utilizados

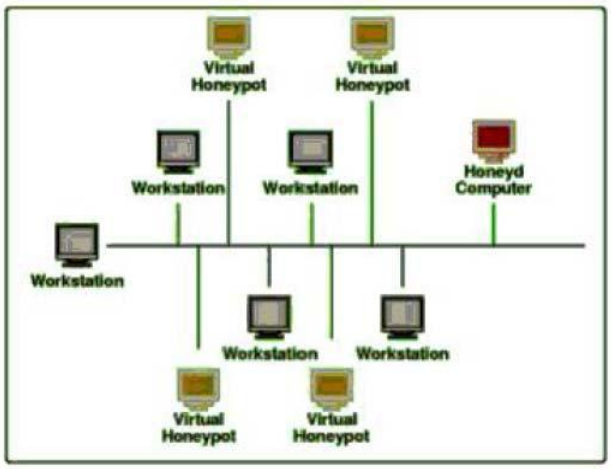

Figura 2.6 - Honeyd monitorando endereços IP não utilizados.

[3] O termo código aberto, ou *open source* em inglês, foi criado pela *OSI (Open Source Initiative)* e se refere ao mesmo software também chamado de software livre. [Disponível em:<http://pt.wikipedia.org/wiki/Open_source>, Acesso em: 05.04.2009].

- *LaBrea Tarpit:*

O *LaBrea Tarpit* é um *honeypot OpenSource* desenvolvido para sistemas Windows ou Unix, projetado para diminuir a velocidade ou parar um ataque. *LaBrea* é um programa que cria um tarpit (poço de piche), assumindo o comando de endereços de IP novos em uma rede e criando máquinas virtuais que respondem a tentativas de conexão. As respostas forjadas enviadas pelo *honeypot* são utilizadas para manter uma conexão aberta com o atacante, diminuindo a velocidade ou até parando o ataque automatizado.[BizSytem, 2009]

- *Deception ToolKit:*

O *Descption Toolkit* é utilizado para simular que um sistema possui um grande número de vulnerabilidades, atendendo as entradas do atacante e fornecendo respostas personalizadas falsas. São utilizadas rotinas para registrar todos os passos do atacante. É um produto para plataformas Linux; exige um compilador de C e um interpretador de PERL[4].

Foi um dos primeiros *honeypots* desenvolvidos de forma completa, sendo bastante interessante para estudo [*Descption Toolkit, 2009].*

- *Tiny Honeypot:*

O *Tiny Honeypot* é um software *OpenSource*, desenvolvido para plataforma Linux. Simula serviços de TELNET e FTP, provê uma série de respostas falsas aos pedidos do atacante. A meta não é enganar atacantes qualificados, e sim deixar o processo mais difícil com uso de serviços falsos. [Tiny Honeypot, 2009]

[4] Linguagem de programação estável e multiplataforma, usada em aplicações de missão crítica em todos os setores, sendo destacado o seu uso no desenvolvimento de aplicações web de todos os tipos. [Disponível em:<http://pt.wikipedia.org/wiki/Perl>, Acesso em: 05.04.2009].

2.4 Efeitos da Persistência do Problema

À medida que as tecnologias de rede vêm sofrendo um avanço enorme promovendo o compartilhamento de dados, sobretudo na Internet, é necessário encontrar algum meio de proteger informações de usuários não autorizados a obtê-las. A cada dia, é mais fácil obter ferramentas automatizadas ou o código-fonte de explorações de falhas de sistemas prontos que promovem acesso aos servidores de redes, comprometendo a segurança das organizações e indivíduos que nela atuam. O mais preocupante é que hoje em dia não é mais necessário ter conhecimentos avançados em um determinado sistema operacional ou protocolo para explorar suas vulnerabilidades.

Atualmente, existem diversas ferramentas que juntas contribuem para a melhoria da segurança de uma rede. Por exemplo, a criptografia estabelece um nível de proteção para dados, estejam eles em trânsito ou armazenados em disco. Outro exemplo seria o uso de sistemas de segurança como *firewalls*, sistemas de IDS, entre outros, que funcionam de modo defensivo. Estes sistemas são adaptados para reagir sempre que seja detectada uma falha de segurança na organização. Deste modo, pretende-se proteger (defendendo) a organização e os seus recursos [PITANGA & MARCELO, 2003].

Em complemento a isso, de um modo pró-ativo, os *honeypots* objetivam a recolha de informação sobre ataques existentes. O recurso a esta informação facilita o desenvolvimento de novas ferramentas, assim como permite uma melhor compreensão das ameaças, o que conseqüentemente auxilia a defesa das organizações contra esses mesmos ataques.

Assim, é cada vez mais necessário que as organizações e usuários de rede estejam munidos de conhecimento e de ferramentas apropriadas, que lhes ofereçam meios de proteção para suas informações. Neste sentido, provas-se a importância do uso de soluções de rede pró-ativas, como o *honeypot* e o estudo do comportamento dos invasores de uma rede para o desenvolvimento de novas ferramentas e sistemas de defesa. Como benefício, prover o aumentando do nível de segurança das redes e capacitar seus administradores no desenvolvimento de mecanismos de proteção mais eficientes [SPITZNER, 2003].

2.5 Fatores Associados ao Problema

Quando se fala de sistemas de detecção de intrusão, existem duas situações indesejáveis que freqüentemente nos deparamos. Uma é o que chamamos de falsos positivos (ou falsos alertas) e a outra são os falsos negativos, ou seja, os ataques reais, porém não detectados. A primeira situação representa inconsistência na detecção de eventos. Ocorre geralmente quando um ataque é efetuado na rede e o IDS o detecta, gerando alertas deste e de outros supostos ataques que possuem características semelhantes, mas que na realidade não aconteceram. Falsos positivos podem ser explorados de forma maliciosa.

A segunda situação implica num comprometimento mais imediato caso o ataque seja bem sucedido, tendo em vista que se o administrador não sabe que um ataque está acontecendo na rede, obviamente uma atitude imediata não será tomada por ele nem por algum mecanismo de defesa [FAGUNDES, L. LEONARDO, 2002].

Quanto ao sistema de detecção de intrusão do *honeypot,* apesar de ser um poderoso dispositivo de segurança, sua utilização possui algumas desvantagens, das quais se destacam [J. PEIXOTO, 2002]:

- Possuem uma visão reduzida, pois apenas conseguem identificar ataques quando existe uma interação direta. Desta forma, é impossível a detecção de um ataque efetuado a outro sistema quando esse ataque não interage também com o *honeypot*;
- Os *honeypots* podem ser utilizados para atacar outros sistemas, ou seja, um atacante que obtenha controle do *honeypot* pode utilizá-lo para atacar outros sistemas;
- Quando um *honeypot* é detectado a sua importância diminui drasticamente. Esta situação pode tornar-se perigosa quando os responsáveis do *honeypot* desconhecem este fator, pois, os atacantes podem injetar informações erradas no *honeypot*, de maneira a induzir os responsáveis em erro;

- O aproveitamento dos recursos dos *honeypots* para realização de procedimentos ilícitos, como distribuição de material ilegal (filmes, música), pode também ser encarado como uma desvantagem.

2.5.1 Controle de Dados

O controle existe principalmente para mitigar o risco que se corre de danificar a rede de terceiros. Os invasores precisam de um bom grau de liberdade para agirem, até porque quanto mais liberdade tiver, mais se aprenderá sobre eles. No entanto, quanto mais liberdade tiver, maior o risco que se corre de conseguirem vencer o sistema de controle. O grande desafio da construção de uma *honeypot* é saber dosar esse grau [NORTHCUTT, MC LACHLAND, 2001].

Outro desafio é ter que esconder o controle do atacante e garantir que ele seja efetivo. Uma prática essencial para evitar que se tenha apenas um ponto de falha é realizar o controle em camadas. Quanto mais elementos de controle houver tanto melhor para prevenir ataques novos e não conhecidos. Contador de conexões externas, *gateway* de prevenção de intrusão e restrição de banda são alguns dos mecanismos possíveis. Porém, é necessário ter em mente que qualquer que seja a proteção, ela serve apenas para minimizar o risco, não para eliminá-lo.

2.5.2 Captura de Dados

É o processo de monitorar e registrar todas as atividades realizadas pelos invasores sem que eles percebam. É a coleta da matéria-prima de qualquer análise posterior com o objetivo de aprender as técnicas, ferramentas e motivos do ataque. O grande desafio é capturar o máximo de informações possíveis sem que o atacante saiba. Novamente a arquitetura em camadas é a melhor solução. Quanto mais mecanismos houverem de captura (*logs* de *firewall*, *sniffers*[5], *logs* de IDS, *logs* do sistema, etc), mais aprendizado eles trarão.

[5] Ferramenta, constituída de um *software* ou *hardware*, é capaz de interceptar e registrar o tráfego de dados em uma rede de computadores. [Disponível em:< http://pt.wikipedia.org/wiki/Sniffer>, Acesso em: 10.04.2009].

Para evitar que os atacantes detectem a armadilha, uma solução é manter o mais estável possível o *honeypot*, ou seja, armazenar as informações da captura em outro local, longe do invasor. Se este tiver acesso aos dados, não apenas detecta, mas modifica ou exclui.

2.5.3 Riscos

O risco em um *honeypot* é o preço que se paga por autorizar aos atacantes o acesso privilegiado a sua rede. Cada organização detentora de uma *honeypot* deve medir por si própria o risco que pretende correr. Não existe um consenso quanto às questões legais de um *honeypot* e cada organização é responsável pela sua. Os principais riscos percorrem quatro áreas: dano, detecção, desabilitação e violação [A. WESLEY, 2002].

Quando o atacante detecta que está num *honeypot*, ele passa a ignorá-lo ou procura apagar os registros de sua invasão, eliminando toda a capacidade de captura e, portanto, diminuindo o valor da solução. O atacante pode fazer pior, registrar informações falsas e confundir a análise posterior dos *logs*.

Além de detectarem a identidade de um *honeypot*, atacantes podem querer desabilitar seus mecanismos de controle e captura sem que o administrador saiba. Por exemplo, um atacante ao conseguir acesso a um *honeypot,* desabilita seu mecanismo de captura, mas simula algumas informações para que o administrador pense que o mecanismo ainda está funcionando perfeitamente. A maneira de reduzir esse risco é novamente através de múltiplas camadas de controle e captura de dados.

O risco de violação refere-se ao sentido legal. Em todos os casos, a maneira mais eficiente de diminuir os riscos, além de se construir uma arquitetura em camadas, é a constante monitoração humana. É aconselhável que se tenha profissionais treinados capazes de fazer uma análise do *honeypot* em tempo real. Para um ataque ser identificado, é necessário monitorar e analisar todos os dados capturados. Ter um profissional de segurança nessa tarefa é essencial para evitar que ataques novos e desconhecidos provoquem os riscos citados.

As técnicas automáticas de análise neste caso não são eficientes devido à complexidade das informações e a possibilidade de novos ataques.

A mensagem principal a respeito dos riscos é que não importa quão eficiente seja o controle e captura de dados em um *honeypot,* o risco nunca é eliminado, ele apenas pode ser mitigado.

2.6 Benefícios da Solução do Problema

A solução proposta, baseada em *honeypot* contempla não só a captura, bem como o desenvolvimento de uma camada de interação que proporcione ao administrador da rede maior agilidade e facilidade na análise das informações coletadas, conseqüentemente, para elaborar contra medidas de proteção da sua rede.

Além disso, alguns outros benefícios podem ser obtidos com a utilização da solução proposta:

- A solução foi desenhada para capturar qualquer interação ilícita ou não autorizada, mesmo novos tipos de interações nunca antes identificados;
- Devido ao registro de suas interações diretas, necessitam apenas de um conjunto mínimo de recursos;
- Permitir a coleta de informação mais detalhada e específica que a maioria dos restantes dispositivos de segurança;
- Prover uma simples configuração e gestão das informações coletadas, sem a utilização de algoritmos complexos;
- Além de eficaz na captura das informações sobre os ataques, a solução de detecção de intrusão, apresenta uma interface amigável que facilita a análise e manipulação das informações captadas.

3 REFERENCIAL TEÓRICO

3.1 Segurança em Redes - Aspectos Gerais

A proteção de uma rede é sempre construída de camadas de segurança. Um atacante pode conseguir passar por uma barreira de proteção, então deve haver outra para barrá-lo, garantindo assim a integridade das informações. As camadas são constituídas por dispositivos físicos, softwares ou políticas de segurança. Quanto mais diversificada for a proteção, mais segurança existirá. Os danos de ter uma segurança invadida, informações roubadas e credibilidade no mercado abalada são financeiramente altas e isso justifica o investimento em boas soluções de segurança de rede [TANENBAUM, 1997].

Ao planejar uma solução de segurança, é necessário haver uma constante revisão das ações, para evitar problemas futuros. Sendo assim, a manutenção também é muito importante, uma vez que nenhum software é concebido sem *bugs* ou falhas.

A implantação de práticas de segurança minimiza as chances de ocorrerem problemas de segurança e facilitam a administração das redes e recursos de forma segura, com o objetivo de prover aos usuários serviços de alta qualidade e estabelecer um comportamento ético e profissional. As diretrizes de segurança devem definir as normas para uma boa utilização da rede e as atividades consideradas como violação da utilização de recursos e serviços, bem como punições no caso da má utilização destes.

3.2 Atacantes, Alvos e Motivação

3.2.1 Atacante

O conceito *hacker* surgiu no início da década de 1960. Atualmente os *hackers* são conhecidos como pessoas com grandes conhecimentos em técnicas de programação, análise de sistemas e especialistas em segurança de redes. Conhecem detalhadamente o computador e os sistemas operacionais, protocolos de rede e todas as vulnerabilidades conhecidas, e ainda são capazes de descobrir

novas vulnerabilidades, que muitas das vezes são informadas aos desenvolvedores para que possam ser corrigidas [OLIVEIRA, 2002].

Um segundo tipo de *hacker* são os *crackers*, os verdadeiros vilões da Internet. Esses indivíduos utilizam seus conhecimentos de maneira maliciosa, aproveitando as vulnerabilidades existentes para obter informações sigilosas, invadir sistemas, destruindo informações, alterando programas e desfigurando sites. Um *cracker* é mais facilmente identificado, pois na maioria de seus ataques deixam evidências indicando a invasão. [OLIVEIRA, 2002].

A habilidade desses usuários mal intencionados muda com rapidez, à medida que os ataques se tornam mais modernos, tornado-se uma grande ameaça para segurança das informações trafegadas na rede. Um grande problema são as ferramentas, cada mais automatizadas, algumas possuindo até interface gráfica que facilita sua utilização por indivíduos que não possuem conhecimento, fazendo com que eles consigam hackear centenas de sistemas em uma só noite. Além disso, a utilização de criptografia nos ataques, que dificulta o monitoramento do tráfego e identificação de ataques.

- Tipos de Atacantes:

Script Kiddies: Possuem pouco conhecimento de programação, utilizam programas pré-compilados e rodam *scripts*[6] automatizados que fazem todo o serviço, sendo necessário somente definir qual *exploit* vai ser utilizado. Desejam comprometer o maior número de sistemas possíveis com o menor esforço possível, não importando se o computador pertence a uma empresa ou a um usuário comum. A maior parte dos ataques, *scans* e sondagens são feitas por *Script Kiddies.* [SPITZNER, 2002]

Advanced Blackhats: São *hackers* experientes que utilizam seus conhecimentos para invadir e comprometer sistemas importantes e bem protegidos. Normalmente utilizam ferramentas desconhecidas e de criação própria, tornando difícil a identificação. Sabem como apagar seus rastros limpando arquivos de *logs*, históricos ou modificando o *kernel* da máquina, escondendo processos do

[6] *Script (também conhecido como linguagem de scripting, ou linguagem de extensão)* são linguagens de programação executadas do interior de programas e/ou de outras linguagens de programação, não se restringindo a esses ambientes. [Disponível em:<http://pt.wikipedia.org/wiki/Scripts>, Acesso em: 20.04.2009].

administrador, fazendo com que os usuários atacados talvez nunca descubram que sua máquina foi invadida. Seus ataques são geralmente patrocinados financeiramente ou por motivações nacionalistas. Apesar de serem minoria eles são muito mais perigosos por seus conhecimentos e técnicas. [SPITZNER, 2002]

Lammers: são atacantes curiosos que simplesmente realizam seus ataques repetindo fórmulas de invasão. Tem certo conhecimento em sistemas operacionais e são facilmente descobertos, pois costumam deixar sua assinatura eletrônica por onde passam e cometem uma série de falhas no momento da invasão confirmando que estiveram por lá [SYMANTEC, 2009]

Carders: *hackers* criminosos que utilizam suas experiências para roubar cartões de crédito e utilizar em benefício próprio. [SYMANTEC, 2009]

Além dos ataques cometidos por pessoas, também existem ataques oriundos de programas planejados com um determinado objetivo. Eles têm sua classificação subdividida nas seguintes características [SPITZNER, 2002]:

- Vírus: São programas de computador, e como tal, só podem agir em softwares, e na maioria das vezes não poderá causar danos ao hardware (micro, drives de disco rígido e flexível, mouse, vídeo, teclado, etc). Eles têm comportamento semelhante ao do vírus biológico: multiplicam-se, precisam de um hospedeiro, esperam o momento certo para o ataque e tentam esconder-se para não serem exterminados.

- *Worms*: Podem ser interpretados como um tipo de vírus mais inteligente que os demais. A principal diferença entre eles está na forma de propagação: os *worms* podem se propagar rapidamente para outros computadores seja pela Internet, seja por meio de uma rede local. Geralmente, a contaminação ocorre de maneira discreta e o usuário só nota o problema quando o computador apresenta alguma anormalidade. O que faz destes vírus inteligentes é a gama de possibilidades de propagação. O *worm* pode capturar endereços de e-mail em arquivos do usuário, usar serviços de SMTP (sistema de envio de e-mails) próprios ou qualquer outro meio que permita a contaminação de computadores (normalmente milhares) em pouco tempo.

- Cavalo-de-tróia (Trojan): É um tipo de praga digital que, basicamente, permitem acesso remoto ao computador após a infecção. Os cavalos-de-tróia podem ter outras funcionalidades, como captura de dados do usuário e execução de instruções presentes em *scripts*. Entre tais instruções, pode haver ordens para apagar arquivos, destruir aplicativos, entre outros. Quando um cavalo-de-tróia permite acesso ao computador, o que ocorre é que a praga passa a utilizar portas TCP e de alguma maneira informa a seu criador a "disponibilidade" daquele computador. Ainda, a praga pode se conectar a servidores e executar instruções que estejam disponíveis no momento do acesso.

- *Spyware*: É um software que recolhe informações pessoais sem primeiramente informar ao usuário o que está fazendo, e sem que ele possa decidir se aceita, ou recusa a sua presença no sistema. As informações que o *spyware* recolhe podem ser desde informação relativa a todos os sites que visitou, até informações mais sensíveis, tais como nomes de usuário e senha.

3.2.2 Alvos

Para se efetuar um ataque é necessário se eleger uma empresa que apresente em sua estrutura vulnerabilidades.

O usuário tem a errada impressão de que nunca será alvo desses ataques, pois acreditam que seus sistemas não têm valor ou que seja difícil de ser encontrado e atacado por possuir um IP dinâmico. Porém, os atacantes podem utilizá-lo para atacar outros computadores ou guardarem informações roubadas.

3.2.3 Classificação dos Ataques

Os ataques podem ser classificados segundo o objetivo principal, conforme definido abaixo:

- *Ataques de Exploração:*

O ataque de exploração é um tipo de ataque que consiste em obter o maior número de informações, sem causar nenhum tipo de dano ao sistema.

Alguns exemplos de ataques são: a espionagem digital, o furto de senhas e fraudes bancárias. As entidades mais vulneráveis são as instituições financeiras e entidades governamentais, devido ao tipo de informação que trafegam.

Um dos programas mais utilizados pelos atacantes são os *scanners* que tem por função varrer a rede em busca de máquinas com problemas, ou com portas de serviços abertas. Uma das ferramentas mais utilizadas pelos scanners é o nmap, que detecta os dispositivos na rede, e em seguida verifica se eles apresentam alguma vulnerabilidade (portas, serviços, S.O.). Com o nmap é possível fazer a varredura de redes inteiras, com opções variadas [MARCELO e PITANGA, 2003].

- *Ataques de Paralisação:*

Este ataque tem como objetivo principal indisponibilizar temporariamente recursos de um sistema. É também conhecido como negação de serviços. Este ataque pode ser iniciado com ferramentas básicas (ping, tracert, etc.), causando uma sobrecarga de conexões aos sistemas alvos, que acarretará na negação do serviço [STEIN e STEWART, 2004].

- *Ataques de Comprometimento:*

Esses tipos de ataques têm a finalidade de comprometer o funcionamento dos dispositivos de uma rede através da desativação de serviços críticos em servidores, comprometimento ou destruição de informações do alvo, desperdício de recursos ou até mesmo comprometer fisicamente os recursos de um sistema.

Os ataques de comprometimento podem ser exemplificados por pichações de sites, destruição intencional de dados, suspensão dos recursos do sistema como: processamento, memória [HIJAZI, 2004].

3.2.4 Motivação:

Normalmente a motivação de ataques é dada por fatores como dinheiro, ego, diversão, ideologia, status e/ou inclusão em um grupo social. Esses fatores podem ser obtidos de diversas maneiras como obtenção de números de cartões de crédito, tentativas de quebrar a segurança de uma empresa e fazer espionagem industrial, possuir controle de centena de milhares de computadores e derrubar ou fazer *defacement* (desfiguração) de sites.

3.3 Ferramentas e Tipos de Ataques

Seguem abaixo diversas ferramentas que servem de arsenal para os *crackers*. Essas ferramentas são utilizadas diversas vezes em conjunto. A figura 3.1 mostra os ataques mais reportados ao CERT.br nos meses de janeiro a março de 2009, conforme mostrado na figura 3.1[CERT.BR, 2009]:

Figura 3.1 - Estatísticas do CERT.br mostrando os ataques mais reportados.

Legenda:

- **Worm**: notificações de atividades maliciosas relacionadas com o processo automatizado de propagação de códigos maliciosos na rede.
- **DOS** (DoS -- *Denial of Service*): notificações de ataques de negação de serviço, onde o atacante utiliza um computador ou um conjunto de computadores para tirar de operação um serviço, computador ou rede.
- **Invasão**: um ataque bem sucedido que resulte no acesso não autorizado a um computador ou rede.
- **Web**: um caso particular de ataque visando especificamente o comprometimento de servidores Web ou desfigurações de páginas na Internet.
- **Scan**: notificações de varreduras em redes de computadores, com o intuito de identificar quais computadores estão ativos e quais serviços estão sendo disponibilizados por eles. É amplamente utilizado por atacantes para identificar potenciais alvos, pois permite associar possíveis vulnerabilidades aos serviços habilitados em um computador.
- **Fraude**: segundo Houaiss, é "qualquer ato ardiloso, enganoso, de má-fé, com intuito de lesar ou ludibriar outrem, ou de não cumprir determinado dever; logro". Esta categoria engloba as notificações de tentativas de fraudes, ou seja, de incidentes em que ocorre uma tentativa de obter vantagem.
- **Outros**: notificações de incidentes que não se enquadram nas categorias anteriores.

3.3.1 Engenharia Social

Um método utilizado para se obter informações sigilosas e importantes, por meio da confiança das pessoas. Esse método não necessita necessariamente de computadores ou de uma rede e depende muito da capacidade de persuasão da pessoa. Um exemplo seria um *cracker* tentando se passar por um funcionário que presta suporte técnico para uma empresa e tenta obter informações da empresa, alegando que estas são necessárias para o seu serviço.

3.3.2 BOT

Têm esse nome por agirem de forma automatizada; são utilizados para manter controle de canais no IRC e fazer ataques DoS. Cada computador hackeado conta como um BOT, sendo que já foram descobertos milhares de BOTs sobre domínio de um único *hacker* (*botnets*). Também são conhecidos como computadores zumbis.

O Dr. Alan Solomon [SOLOMON, 2002] explicou a utilização das *botnets* da seguinte maneira: "Imagine que você tenha 100.000 computadores espalhados pelo mundo inteiro e cada um deles rodando e sendo monitorado em uma casa, escritório ou escola qualquer. Imagine que com apenas um comando, você possa mandar esses computadores fazerem o que você quiser. Você pode mandar eles acessarem o mesmo site simultaneamente, procurar uma falha de segurança nos computadores próximos ou apenas que eles mandem grandes quantidades de SPAM".

3.3.3 Backdoor

É um método de fazer um "*bypass*" no método normal de autenticação, provendo um acesso remoto escondido do usuário legítimo a um computador. Normalmente são introduzidos por um tipo de *malware*.

3.3.4 Analisador de rede

Popularmente conhecido como *sniffer*, é uma ferramenta que faz a captura e análise de pacotes em redes *ethernet* ou *wireless*, de acordo com a *request* for *coment* (RFC) ou alguma outra especificação.

O *sniffer* é utilizado para monitoramento de redes como no caso do *Intrusion Detection System* (IDS). Porém podem ser utilizados para fins maliciosos, como obter arquivos confidenciais, senhas e outras informações [TANENBAUM, 1997].

3.3.5 Rootkits

É um conjunto de ferramentas que faz modificações no nível do *kernel*, e com isso liberando o acesso ao sistema para o atacante, enquanto se esconde do sistema operacional (escondendo arquivos, conexões, endereços de memória e registros de entradas usados por programas que detectam acessos privilegiados aos recursos do sistema) [ROOTKITS, 2005].

Rootkits escondem a sua presença no sistema, escondendo suas chaves no registro (para que você não possa vê-las) e escondendo os seus processos no Gerenciador de Tarefas, além de retornar sempre erros de “arquivo inexistente” ao tentar acessar os arquivos do trojan.

Diversos trojans utilizam essas tecnologias com o objetivo de dificultar sua remoção e o fazem com sucesso: os *rootkits* mais avançados são bem difíceis de serem removidos.

3.3.6 Phishing/Scam

É uma fraude eletrônica que utiliza engenharia social para atingir seus objetivos. Essa atividade criminosa aumentou exponencialmente nos últimos anos. Baseia-se no envio de e-mails fraudulentos com o objetivo de obter códigos de acesso e dados bancários de usuários [D’AVILA, 2004].

3.3.7 Brute Force

Método de quebrar a criptografia, tentando um grande número de possibilidades, tais como tentar utilizar todas as chaves possíveis para decifrar uma mensagem criptografada.

3.3.8 Exploit

Pode ser uma seqüência de comandos, uma porção de dados ou um trecho de código que se aproveita de vulnerabilidades de um software para se obter um acesso ilícito [MCDONALD,1999].

É muito utilizado por *worms* e scripts automatizados. Um exemplo bem conhecido desse tipo de ataque é o SQL *Injection* que explora vulnerabilidades de um banco de dados mal configurado, conseguindo obter dados confidenciais contidos nas tabelas do banco, tais como *logins* e senhas de usuários.

3.3.9 Arp Spoofing

Método no qual o atacante se passa pela vítima, informando que o IP da vitima está no endereço físico (MAC) do atacante, recebendo todos os dados e fazendo um processo análogo ao servidor. O atacante então funciona como um roteador e captura todos os pacotes.

3.3.10 Denial of Service (DoS)

Têm como objetivo interromper ou causar lentidão numa atividade legítima, como pessoas acessando páginas WEB, escutando uma rádio online, transferindo dinheiro de suas contas bancárias ou até mesmo a comunicação de um barco atracando num porto [MIRKONIC, 2004]. Esse efeito de negação de serviço é obtido enviando-se mensagens para a interface que está em operação e fazê-la cair, reiniciar ou executar trabalhos inúteis. Esse tipo de ataque consiste num envio massivo de pacotes ao alvo, fazendo com que sua cache estoure e com isso a vítima começaria a descartar pacotes de usuários legítimos.

3.3.11 Métodos de DoS

SYN Flood é uma inundação de pacotes TCP/SYN, onde normalmente o IP é *spoofado* (não corresponde ao IP da máquina que origina o ataque).

Essas solicitações fazem com que o alvo responda com pacotes TCP/ACK-SYN, deixando conexões TCP parcialmente abertas e aguardando por pacotes TCP/ACK que não serão respondidos pelo IP forjado. Essas conexões TCP semi - abertas consomem recursos do alvo e limitam o número de conexões possíveis [MIRKONIC, 2004].

- *Smurf Attack*: É um tipo de ICMP *Flood*, utilizado em redes mal configuradas que permitem que pacotes sejam enviados para uma rede inteira utilizando o endereço de *broadcast*. Nesse caso o IP forjado é o IP da vítima.

- *Fraggle Attack*: É um tipo de UDP *Flood* que funciona de modo parecido ao *Smurff Attack*, onde é enviada uma grande quantidade de pacotes UDP *echo traffic* para um endereço de *broadcast* contendo o IP da vítima como requisitante.

3.3.12 Tipos de DoS

DDoS: É utilizada quando o atacante possui várias máquinas (bots) sobre seu comando e faz com que elas ataquem simultaneamente o alvo. Funciona do mesmo método que o DoS, porém com muito mais recursos o que acarreta em muito mais tráfego dirigido ao alvo e com uma maior dificuldade do atacante ser identificado.

DRDoS: Ocorre quando são enviados vários pacotes forjados com o IP da vítima para uma grande quantidade de computadores e estes respondem para o IP informado. Podem ser, por exemplo, pacotes ICMP *Echo Request*, que serão respondidos com pacotes ICMP *Echo Reply* para o IP da vítima. Esse ataque é conhecido como *Ping Flood*.

3.3.13 Malware

Qualquer software feito para danificar ou infiltrar um computador sem o consentimento de seu proprietário. Seu nome provém da junção das palavras *malicious* e software.

O *malware* também pode ser denominado de *badware*. Muitas vezes os *malwares* instalam *backdoors* para possibilitar o acesso posterior dos *crackers*.

Podem ser classificados da seguinte maneira: Vírus, *Worm,* Trojan ou *Spyware.*

3.4 Firewall

É um conceito que pode ser implementado em *hardware* ou *software* feito para segregar duas redes de diferentes níveis de confiabilidade, controlando o trafego que entra e sai de acordo com regras especificadas pela política de segurança, normalmente uma rede interna (intranet) e a Internet.

A funcionalidade de um *firewall* se assemelha a uma porta corta fogo, que tem como objetivo impedir que o fogo se alastre para outros compartimentos, no nosso caso a nossa rede interna. Atualmente, além do controle de conexões os *firewalls* possuem recursos como controle de conteúdo, *Network Adress Translation* (NAT) / *Port Adress Translation* (PAT), estabelecimento de *Virtual Private Network* (VPN) entre outros foram incorporados.

William Cheswick e *Steven Bellovin* [WILLIAM, 2005] definem *firewall* como: "Uma coleção de componentes ou um sistema colocado entre duas redes possuindo as seguintes propriedades: todo o tráfego de dentro para fora e vice-versa tem que passar por ele, somente trafego autorizado, como foi definido pela política local de segurança, é permitido passar por ele e o sistema é altamente resistente à penetração". Os firewalls não estão restritos a somente os elementos de rede e devem estar presentes também em servidores e estações de trabalho.

3.4.1 Tipos de Firewall

Várias tecnologias de *firewall* estão disponíveis atualmente e o grande diferencial entre eles é o nível de interação com o modelo OSI e de acordo com essa interação pode-se obter maior velocidade ou flexibilidade.

- *Packet Filter*: Todos os *firewalls* fazem um tipo de filtragem de pacote IP, normalmente por um roteador com filtragem de pacotes. Essa filtragem segue alguns critérios como IP de origem e destino e porta TCP/UDP de destino e origem, podendo bloquear acessos a um host ou rede específica, assim como portas específicas. Fazem uma combinação de filtragem de IP com filtragem de portas TCP ou UDP. Tem grandes vantagens como, por exemplo, permitir acesso à porta 22 (SSH) somente pelos IPs especificados.

- O filtro de pacotes não faz uma verificação se o pacote faz parte de um fluxo de dados, portanto, não há uma informação sobre o estado de conexão. Podem ser feitos também bloqueios sobre o protocolo ICMP. Caso um pacote se enquadre num dos filtros ele pode ser descartado silenciosamente ou rejeitado, ou seja, descartado e uma mensagem de erro é enviada ao IP de origem.

- *Proxy*: Conhecidos também como *Aplication Gateway*, ele supre as fraquezas do *packet filter*, possibilitando o encaminhamento e a filtragem de conexões como, por exemplo, Telnet e FTP. Utilizada em conjunto com o *Packet Filter*, eles disponibilizam um nível muito maior de segurança e flexibilidade, porém essa combinação coloca limites na transparência, conectividade e possui implementação e gerenciamento trabalhoso. As principais vantagens são: esconder informações, providenciar robustez na autenticação, menor custo, mais eficiência e regras menos complexas de filtragem. As desvantagens são: mais lento que outras tecnologias, exige um *Proxy* especializado para cada protocolo e exige a configuração dos clientes.

- *Packet Inspection*: Como seu nome diz, é feita uma inspeção nos pacotes em vez de somente filtrá-los. Esse método é aplicável a todos os protocolos, desde a camada de rede (Cabeçalhos IP) até a camada de aplicação. Essa tecnologia também permite monitorar o estado das conexões, liberando os pacotes que chegam de acordo com seus pacotes de solicitação, porém, os pacotes disfarçados de respostas a solicitação inexistente são descartados. Esse método também é chamado de *Stateful Inspection*. Suas vantagens são: velocidade, análise de vários aspectos do pacote e bloqueio do pacote em aspectos mais básicos. As desvantagens são: a permissão de conexão direta entre rede interna e externa e não examinar todos os pacotes.

- *Firewalls* híbridos: Atualmente muitos dos *firewalls* comerciais utilizam-se de uma combinação de tecnologias.

3.4.2 Arquiteturas de Firewall

As arquiteturas de um *firewall* podem ser do tipo:

Dual-Homed Bastion Host: utiliza uma única máquina com no mínimo duas interfaces de rede, interligando a rede interna com a Internet. Essa máquina possui a missão crítica de ser uma barreira entre Internet privada e pública e é denominada *Bastion Host* e funciona como um *proxy* como ilustrado na figura 3.2.

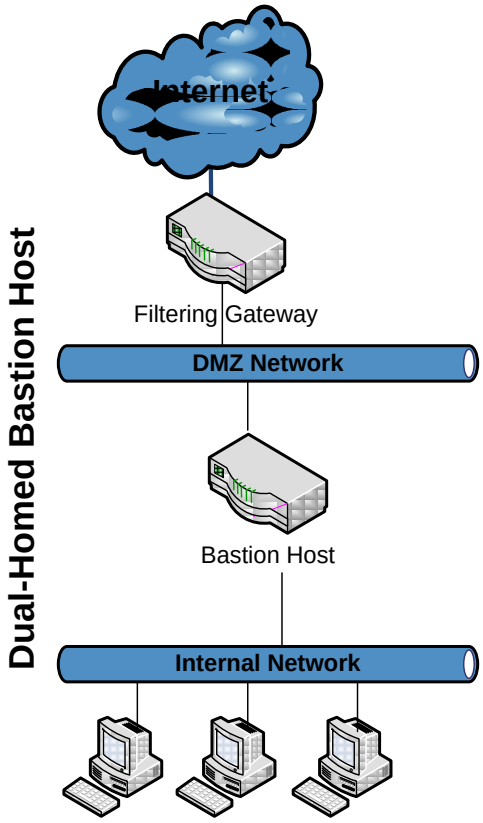

Figura 3.2 - Arquitetura *Dual-Homed Bastion Host.*

Screened Host Gateway: O *Bastion Host* se localiza na rede interna, rodando dedicadamente como um *proxy*. Todo tráfego que não for direcionado a ele é descartado pelo filtro de pacotes. Isso impede a conexão direta entre a rede interna e a Internet. Sua vantagem é o acesso aos serviços sem concorrência, pois todo processo de filtragem é feito no gateway. Sua desvantagem é o invasor já se encontrar dentro da rede no caso de comprometimento. Sua arquitetura está ilustrada na figura 3.3.

Screened Subnet: Adota uma subrede chamada zona desmilitarizada (DMZ), ou seja, um local com o nível de segurança intermediário, normalmente utilizado para colocar servidores WEB e e-mail, que necessitam prover serviços a usuários externos. O *Bastion Host* fica localizado dentro dessa DMZ e serve como único ponto de acesso para a rede interna. É considerada uma das arquiteturas mais seguras e esta ilustrada na figura 3.4.

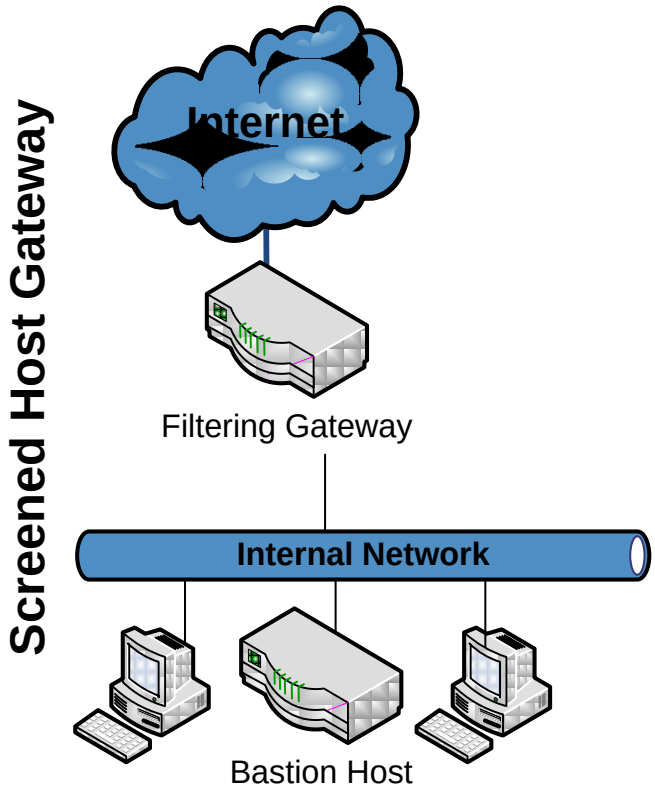

Figura 3.3 - Arquitetura *Screened Host Gateway.*

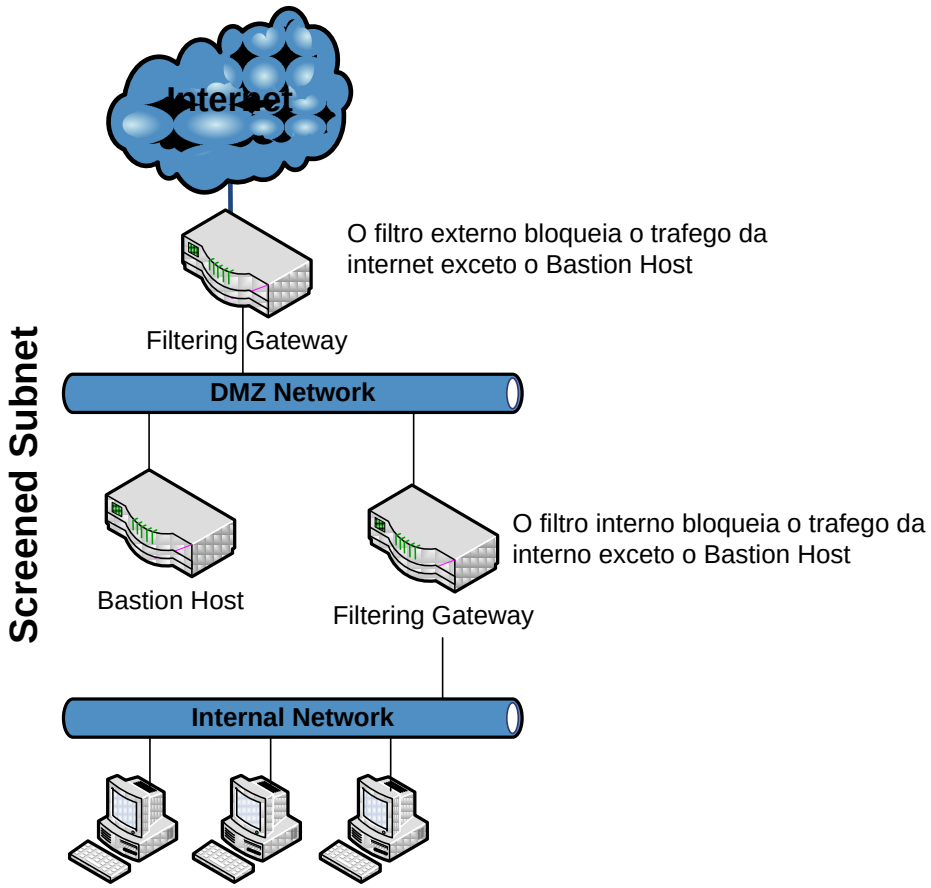

Figura 3.4 - Arquitetura *Screened Subnet.*

3.5 Sistema de Detecção de Intrusão (IDS)

É uma ferramenta de segurança que coleta e analisa dados, buscando detecção e contenção de ataques à rede e é utilizado para detectar tráfego malicioso que não é detectado por um *firewall*. Ele é composto de sensores que geram os eventos de segurança, um console que controla os sensores e monitora os eventos/alertas e um *engine* que grava os eventos num banco de dados e utiliza as regras configuradas para gerar alertas.

Michael Wilkinson [WILKINSON, 2009] define um IDS como "Qualquer sistema designado a detectar tentativas de comprometimento da integridade, confidencialidade ou disponibilidade da rede protegida e sistemas de computadores associados".

Um IDS somente deve reportar uma tentativa que realmente comprometa o sistema atacado ou um ataque que ainda não foi visto, por exemplo, um *exploit* conhecido que explora uma vulnerabilidade do Windows XP sendo utilizada em um Linux não deve gerar um alerta. Uma má configuração do IDS pode acarretar nos seguintes problemas:

- Falso-Positivo: Quando o tráfego legítimo é considerado um ataque. É considerado um sério problema, pois o IDS pode bloquear tráfego legítimo, parando serviços fundamentais para uma empresa.
- Falso-Negativo: Quando um ataque não é percebido pelo IDS, passando como se fosse tráfego legítimo.

3.5.1 Métodos de Detecção e Tipos de Reação

- Métodos de Detecção:

Anomalia: Funciona com base na definição do tipo de tráfego normal da rede. Tudo que foge desse padrão é considerado uma anomalia gerando alertas. O grande problema desse método é definir o que é normal no tráfego de sua rede, sendo necessária para a utilização deste uma rede estável e bem definida. Tudo que é novo gera um alerta.

Assinatura: Funciona com base em assinaturas de ataque que estão armazenadas em sua base de dados e são comparadas com os dados coletados em sua rede. Quando os dados batem é gerado um alerta.

- Tipos de Reação:

Passiva: Apenas gera alerta ao administrador de rede e ele deve tomar uma providência caso seja necessário. O problema é o tempo de reação, pois, após o administrador receber a notificação, é necessário um tempo para que o administrador tome uma providência.

Reativa: Executa alguma ação, como deslogar um usuário, acrescentar uma regra no *firewall* ou uma ACL no roteador. O problema são os falso-positivos, causando uma negação de serviço em sua rede.

3.6 Tipos de IDS

Segundo *Gerald Fink* [G. A. FINK, 2002] são classificados em:

HIDS (Host-Based IDS): Mais fácil de ser implementado, configurado e é dedicado a um único *host*. São baseados em software e é utilizado para monitoramento de logs (*syslog*), acesso às portas (portas que não devem ser acessadas, ao acontecer uma tentativa um alerta é gerado) e cálculo/verificação de *checksums* (certos arquivos têm os seus *checksums* verificados periodicamente, para verificar se houve modificações).

NIDS (Network-Based IDS): Dispositivo (*hardware*) que monitora todos os dados de um segmento estratégico da rede, normalmente em zonas desmilitarizadas (DMZ) ou nas bordas da rede como ilustrado na figura 3.5. O acesso é feito por *hubs*, *switchs* configurados com espelhamento de porta ou com um *tap*. Trabalha em modo promíscuo (não possui um endereço IP) e *stealth* (somente recebe os dados, não há transmissão de dados). Análise em tempo real, por meio de padrões, freqüência, correlação de eventos e detecção de anomalia.

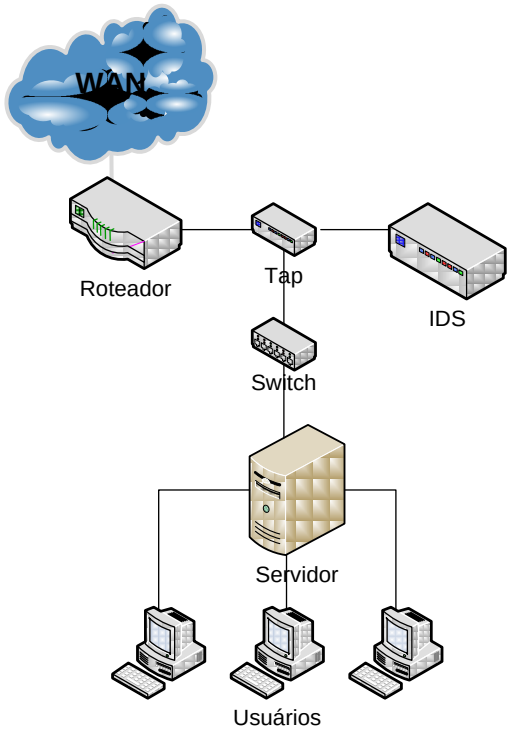

Figura 3.5 - Figura Mostrando um NIDS.

PIDS (Protocol-Based IDS): Dispositivo ou software que monitora e analisa o comportamento e estado dinâmico do protocolo de comunicação entre o dispositivo conectado e o sistema protegido, normalmente utilizado no *front-end* de servidores WEB, monitorando as conexões http/https com os clientes.

APIDS (Aplication Protocol-Based IDS): É um IDS que monitora e analisa o comportamento e estado dinâmico do protocolo de aplicação entre dois dispositivos, normalmente utilizado na comunicação entre um servidor WEB e um sistema de gerenciamento de banco de dados, monitorando o protocolo SQL.

Hybrid IDS: Combina dois ou mais tipos de IDS, tentando buscar um complemento deixando a rede mais segura.

3.7 Honeypots

3.7.1 Conceitos

Honeypot é um recurso computacional de rede cujo único propósito é ser atacado e reunir o máximo de informações possíveis sobre os atacantes [SPITZNER, 2002]. A forma como um *honeypot* é desenvolvido resulta em um sistema onde o atacante pensa estar interagindo com objeto que compõe a rede e não um ambiente preparado para colher e monitorar informações de acessos não autorizados.

Diversos serviços podem ser simulados ou atendidos por um *honeypot*, por exemplo: FTP, servidor de e-mail, SSH, portas específicas para alguns programas etc. Segundo proposto por *Marty Roesch*, desenvolvedor do *Snort* [SPITZNER, 2002] *honeypots* podem ser divididos em duas categorias: *honeypots* de produção e *honeypots* de pesquisa. No primeiro, a utilização tem como objetivo a proteção, enquanto o segundo é utilizado para pesquisa, entendimento e aprendizado dos ataques.

Normalmente o conceito de *honeypot* é associado ao tipo de proteção onde as pessoas o acrescentam como um dispositivo de segurança nas organizações ajudando, por exemplo, a detectar ataques.

Em contra partida, os *honeypots* de pesquisa não agregam valor ativamente na segurança de uma rede, pois sua prioridade é reunir as informações sobre a comunidade de invasores. Isto é feito através da análise e coletas de dados relacionados à identificação do perfil do atacante, como estão organizados e as ferramentas que utilizam.

Essa classificação tem como objetivo apenas definir e orientar o tipo de *honeypot* que venha a ser desenvolvido e não é absoluta, uma vez que o mesmo *honeypot* pode ser tanto de pesquisa quanto de produção.

3.7.2 Histórico

O conceito de monitorar e pesquisar invasores tem como primeira referência *Cliffor Stoll* [STOLL, 2002], quando ele tornou pública a invasão ocorrida nos sistemas *Lawrence Berkley Laboratory* (LBL). Ele decidiu acompanhar os passos do invasor ao invés de simplesmente fechar as portas. O resultado, quase um ano depois de monitoração, revelou a origem do ataque assim como os motivos do atacante. Em seqüência ocorreram os seguintes fatos:

- 1992 - Foi publicado um artigo por *Bill Cheswick* [CHESWICK,1992] onde ele descreve a maneira como ele acompanhou a invasão em um dos sistemas da AT&T;
- 1997/1998 - Desenvolvido por *Fred Cohen o Deception Toolkit* (DTK). A comunidade de segurança teve acesso a uma das primeiras soluções de *honeypots*;
- 1998 - Primeiro *honeypot* comercial é vendido para o público, a Cyber-*Cop Sting* onde são introduzidos conceitos de sistemas virtuais em um único *honeypot*. É neste mesmo ano que surge a *release* do *BackOfficer*, um *honeypot* baseado em *Windows*, livre e simples que ajudou muito a popularizar o conceito de *honeypot*.
- 1999 - Nasce o Projeto *Honeynet*;

- 2000/2001 - Diversas empresas passaram a utilizar *honeypots* para avaliação e detecção de ataques, assim como captura e estudo de atividades relacionadas à *worms*.

Atualmente o uso de *honeypots* tem diversas aplicações e como houve grande crescimento de usuários e desenvolvedores, as ferramentas disponíveis atendem as necessidades propostas que são identificação e monitoração de atacantes.

Há desenvolvimento intenso na área de *honeypots* voltados para área de aplicações WEB. Neste caso, por exemplo, o sistema é desenvolvido de tal forma que permite que uma aplicação WEB qualquer seja transformado em um *honeypot* para captura e análise de dados com informações de ataques. Especialmente esta área é de grande importância pela quantidade de aplicações que tem sido desenvolvida especificamente para WEB e o uso é cada vez maior como sites de compras, sistemas de pesquisa e intranet de grandes empresas. Nesses casos o *honeypot* simula todos estes serviços para iludir o atacante.

3.7.3 Abrangência, Vantagens e Desvantagens

O maior proveito que podemos tirar de um *honeypot* está justamente no seu conceito, pois como os serviços deles não são críticos nem reais, qualquer acesso a ele é por definição suspeito ou malicioso. Esta afirmação confronta diretamente, por exemplo, com outros dispositivos de segurança como IDS e o *Firewal,l* explicados anteriormente, onde o volume de dados que trafega é muito maior e nem sempre com valor significativo. O uso de *honeypots* permite a formatação dos dados com informação precisa tornando a análise de informação muito mais rápida e simples.

Outra grande vantagem associada ao uso de *honeypots* é a simplicidade e baixa necessidade de recursos, pois não há necessidade de desenvolvimento de algoritmos complexos, manutenção extensiva de tabelas de dados com informações, etc. Certamente, o uso e aplicação de um *honeypot* está diretamente atrelado a necessidade de complementar a segurança de sua rede e por isso a complexidade e demanda de recursos varia muito.

Uma das desvantagens apresentadas pelo uso de *honeypots* é que este só é capaz de ver a atividade que é direcionada especificamente para ele. Dessa forma, se ocorre um ataque em determinada parte da rede, na maioria das vezes o *honeypot* não terá esta informação. Além disso, é possível que um atacante perceba que está atacando um *honeypot* pelas características de respostas e do comportamento padrão que é esperado de um *honeypot*, por exemplo, quando emula a um serviço qualquer. De posse dessa informação, o invasor pode simplesmente evitar o *honeypot* e focar em outros pontos da rede. Por último o risco associado ao uso de *honeypots* está no fato que uma vez comprometido este pode ser utilizado como ponte para ataques em outras partes da rede da organização a qual o invasor está infiltrado.

Os riscos e aplicações são associados diretamente ao tipo de *honeypot* implementado, pois como veremos mais tarde, são classificados em níveis de interatividade

3.7.4 Classificação através de níveis de interatividade

Através do nível de interação proporcionado pelo *honeypot*, estes podem ser classificados em baixa, média e de alta interatividade. A quantidade de informação coletada assim como o valor desta informação, o risco de comprometimento e o nível de implementação é diretamente proporcional ao nível de interação oferecido pelo *honeypot*.

- Baixa interatividade: Normalmente *honeypots* de produção que são utilizados para proteger e fazer detecção de ataques. São de instalação mais simplificada e emulam os serviços, ou seja, os invasores poderão ter resultados de *scan* assim como realizar conexão. Devida a baixa interatividade, a quantidade de informação obtida é limitada. Como a instalação e manutenção são mais simples, podemos, por exemplo, simular diversos serviços rodando em um ambiente Unix como FTP. A detecção através de *honeypots* de baixa interatividade se da através do conhecimento de que muito ataques correspondem a comportamentos pré-definidos facilitando, portanto a configuração do serviço a ser emulado.

- Média interatividade: Como enuncia o nome, é o ponto intermediário, oferecendo mais interação que *honeypots* de baixa interatividade, mas menos que de alta interatividade. Indo um ponto além dos serviços emulados pelos de pouca ou quase nula interação, o comportamento e resposta deste serviço pode ser configurado para não só responder a testes de conexões, mas também acrescentar uma iteração fazendo com que o atacante faça mais solicitações e envie mais comandos. São também comumente associados aos ambientes virtuais onde os *honeypots* são implementados. Neste nível já podemos ter *honeypots* de produção e pesquisa ao mesmo tempo. O nível de iteração é incrementado, assim como o risco e a complexidade de manutenção; em contrapartida temos a possibilidade de reunir informação mais precisa e com maior volume.

- Alta interatividade: O maior nível de interação deste *honeypot* proporciona uma quantidade imensa e valiosa de informação sobre os invasores. O objetivo principal é disponibilizar ao invasor um ambiente real onde ele pode interagir de acordo com seu método e livre vontade. Dentre as vantagens de um *honeypot* de alta interatividade podemos citar: grande oportunidade de aprendizado; identificação de novas vulnerabilidades; identificação de novas ferramentas. Em compensação, estes sistemas exigem esforço muito maior de manutenção (Exemplo: atualização de regras de *firewall*), instalação e configuração muito mais alto. Devido a estes fatores, normalmente estes *honeypots* são alocados em ambientes protegidos, por exemplo, por *firewall*, pois o *honeypot* uma vez comprometido pode obter acesso ilimitado aos recursos da máquina tendo a possibilidade de comprometer outros sistemas.

3.8 Honeynet

Honeynet são *honeypots* de alta interatividade, desenvolvidos para pesquisa e coleta de dados de invasores. É também definido como uma ferramenta de pesquisa composta por uma rede cujo objetivo principal é ser comprometida, mas com controle de captura e saída de dados, prevenindo outros ataques com base nos sistemas que foram comprometidos [PROJECT, 2001].

Com um volume maior de dados para análise, o estudo dos motivos, procedimentos e ferramentas dos invasores são mais precisos, resultando em informações úteis que podem agregar valor na segurança da rede. A topologia como de uma *honeynet* está ilustrada na figura 3.6.

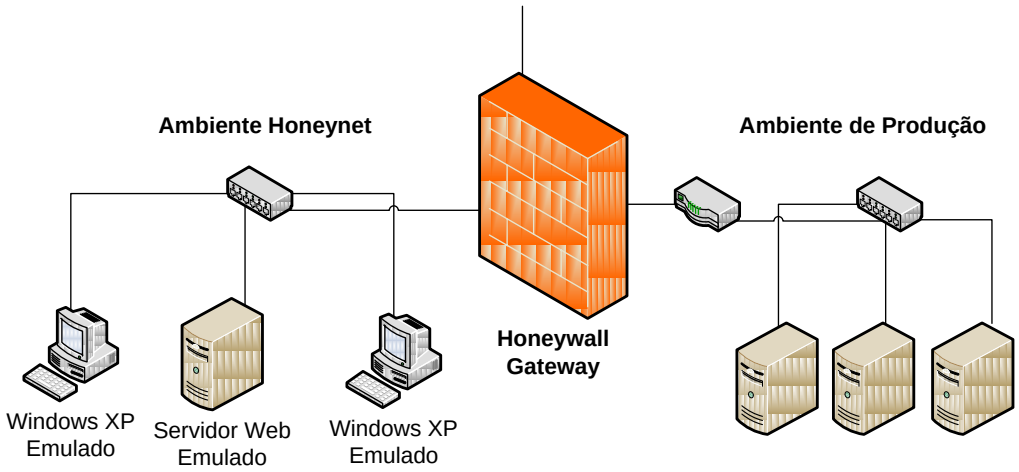

Figura 3.6 - Exemplo de topologia *honeynet*.

3.8.1 Arquitetura

Para a implementação de *honeynet* devemos ter o conjunto dos seguintes elementos: Captura de Dados, Controle de Dados, Análise de Dados e opcionalmente para ambientes onde temos *honeypots* distribuídos, o agrupamento de Dados.

São estes elementos que definem os limites e a liberdade que será dada ao atacante, a forma como os dados são capturados, analisados e depois agrupados e consolidados.

Desde o início do desenvolvimento das *honeynets*, naturalmente surgiram evoluções e demandas que resultaram em novas arquiteturas para implementação. Estas evoluções são chamadas de Gerações e atendem por acrônimos, como exemplo a Geração 2 atende pelo acrônimo Gen II.

A principal alteração que caracteriza a Gen II é a introdução de um dispositivo que lida ao mesmo tempo com os mecanismos de controle e a captura de dados da *honeynet* através do chamado IDS *Gateway* ou com nome mais usual, o *honeywall*. Essa alteração que é mais evidente no controle de dados torna as *honeynets* de Gen II e Gen III mais difíceis de serem detectadas pelos atacantes [HONEYNET, 2009].

- Controle de Dados

Sempre há risco do comprometimento de uma *honeynet* e para isto devemos tomar atitudes que possam minimizar a possibilidade de novos ataques a outros sistemas a partir daquele que foi comprometido. O Controle de Dados deve garantir que os outros sistemas que compõem a rede não sejam afetados pelos *honeypots* através do controle de tráfego do que entra e sai da *honeynet*. A característica importante de uma *honeynet* é justamente proporcionar interação ao atacante e por isso é difícil definir exatamente os limites onde você proporciona esta interação e o risco aceitável que ela representa. É exatamente por isso que o Controle de Dados deve ser considerado de grande importância no planejamento da *honeynet*. São utilizados três modos para o controle:

i. Limite de conexões: limitação convencional do número de conexões só que mais inteligente, pois compara com uma tabela de dados com comportamentos ofensivos.
ii. Descarte de pacotes: pacotes são comparados com uma tabela de dados contendo assinaturas de comportamento padrão. Se este é reconhecido, o pacote é então descartado.

iii. Substituição de pacotes: funciona de forma similar ao descarte de pacotes, mas em vez de descartar o pacote inteiro, as partes ofensivas dos pacotes são modificas e substituídas para torná-lo menos perigoso.

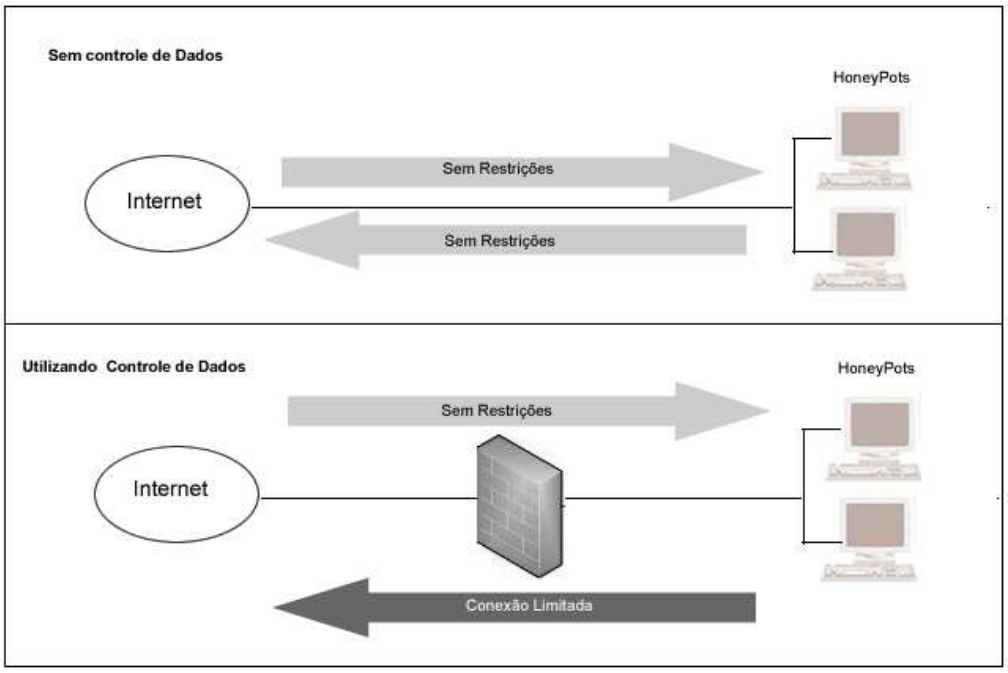

Figura 3.7 - Ilustração controle de dados – *Honeynet.*

- Captura de Dados

Resume-se a atividade de monitoração e registro de *logs*. É essa captura de dados que irá permitir posteriormente a análise do comportamento das ferramentas e objetivos do atacante. Dessa forma, é imprescindível que este não perceba que está sendo monitorado, pois ao menor sinal, certamente o atacante se empenhará em modificar os registros gravados ou até mesmo deletá-los.

Para captura de dados efetivamente, devemos implementar diferentes formas de registros. Alguns *honeypots*, por exemplo, são implementados em máquinas que tiveram os *drivers* de placa de redes recompilados, permitindo algumas alterações que fiquem imperceptíveis ao atacante. Outra forma comum de alteração é a modificação do *shell*, permitindo que os comandos sejam gravados em *logs* cujo diretório não seja o padrão.

Os registros de logs devem ser gravados em local fora da *honeynet*, de forma segura para não haver o risco de serem deletados ou modificados pelo invasor.

Na geração atual de *honeynets*, a captura e controle de dados estão concentrados em apenas um dispositivo, uma espécie de forma híbrida entre IDS e *Firewall* atuando na camada 2 como *bridge*. Dessa forma, como não estão na camada 3 [PROJECT, 2001] não possuem IP, tornando muito mais difícil sua detecção, pois não ocorre decremento TTL, não existe endereço MAC e tampouco roteamento de pacotes. Nas gerações 2 e 3 as camadas de captura são [HONEYNET, 2009]:

Firewall: basicamente limitando o número de conexões de saída de cada *honeypot*

IDS: nesta camada todo pacote é conferido para comparação com a tabela de assinaturas de comportamentos padrões, capturando neste processo todo o tráfego que entra e sai. Geralmente nesta camada são capturados os dados do ataque, ferramental utilizado, etc.

Captura de dados do *honeypot*: nesta camada são capturados os *logs* do sistema e cópia dos dados entrados pelo atacante, capturando tudo que ele digita. A intenção dos *logs* do sistema é acompanhar na análise, quais os processos do atacante. Com esta camada, mesmo que ele utilize conexão SSH e não seja capturado pela camada do IDS, o *honeypot* terá um cliente de captura instalado realizando os registros que são depois enviados para o servidor.

- Análise de Dados

É aqui que reside todo o objetivo e o valor de um *Honeypot*. De posse dos dados capturados, cabe ao administrador fazer a análise e extrair informação útil. A análise deve ser apurada, pois normalmente a quantidade de informação gravada tem volume muito grande, exigindo uma percepção boa para distinguir realmente a utilidade dela. É aconselhável e praticamente necessário o uso de ferramentas que permitam a visualização rápida destes dados através de tabelas e gráficos como na figura 3.8. É aconselhável que estas ferramentas façam a pré-formatação dos dados,

mas que permitam eventualmente a análise manual dos dados, possibilitando ao administrador fazer os ajustes necessários.

A segunda alternativa é utilizar um menu de diálogo com limitada capacidade gráfica, mas que também permite alteração remota.

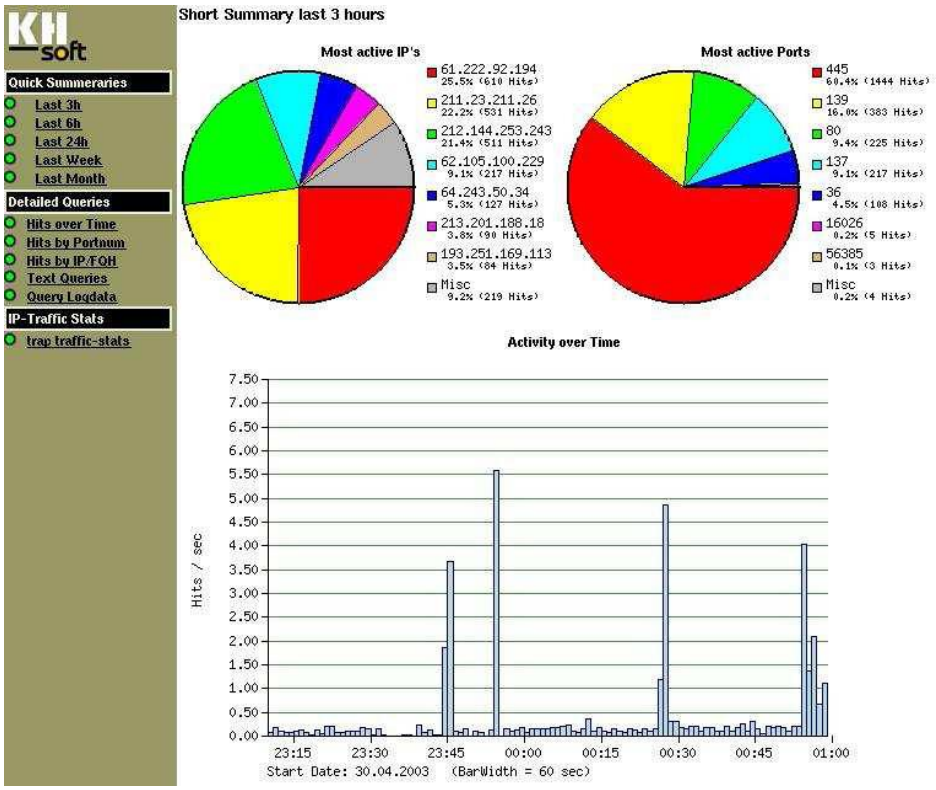

Figura 3.8 - Análise de dados com a ferramenta *honeyview*.

- Agrupamento Dados

Este item é adicionado quando a *honeynet* envolve diversos *honeypots* distribuídos fisicamente como é o caso, por exemplo, do projeto Honeynet.br, explicado mais tarde, que envolve um consórcio de *honeypots* distribuídos. A informação reunida agrega alto valor para a pesquisa.

3.9 Honeynets Virtuais

A montagem de uma *honeynet* requer no mínimo dois computadores, sendo que no primeiro será instalado o *honeywall* e no segundo ficará o *honeypot*. Uma alternativa é utilizar uma solução onde todos os recursos necessários ficam alocados em um só computador, configurando-se um ambiente chamado de *Honeynet* virtual.

Essa solução baseia-se na virtualização de *software* que permite executar ao mesmo tempo múltiplos sistemas operacionais.

As vantagens vão desde redução do custo, às facilidades de instalação e manutenção. Em contrapartida, como está tudo centrado em apenas uma máquina, o risco de ter todo o sistema comprometido é alto, pois uma vez que ele assume o controle do *software* de virtualização, ele obtém o controle de todos os demais sistemas que estão virtualizados. A capacidade de virtualização da maioria dos sistemas disponíveis é outro fator limitante, pois a maioria deles baseia-se apenas no chip x86 da Intel, excetuando-se alguns programas.

Com o uso de virtualização podemos ter, por exemplo, uma *Honeynet* inteira instalada e configurada em um *notebook* permitindo grande portabilidade. Isso possibilita um teste rápido em diversos pontos de uma rede de grande porte, apenas plugando e desplugando cabos de rede seguido dos ajustes de configuração necessários. A solução mais comum para virtualização é o *VMWare Workstation* que suporta diversos sistemas operacionais e é fácil de usar.

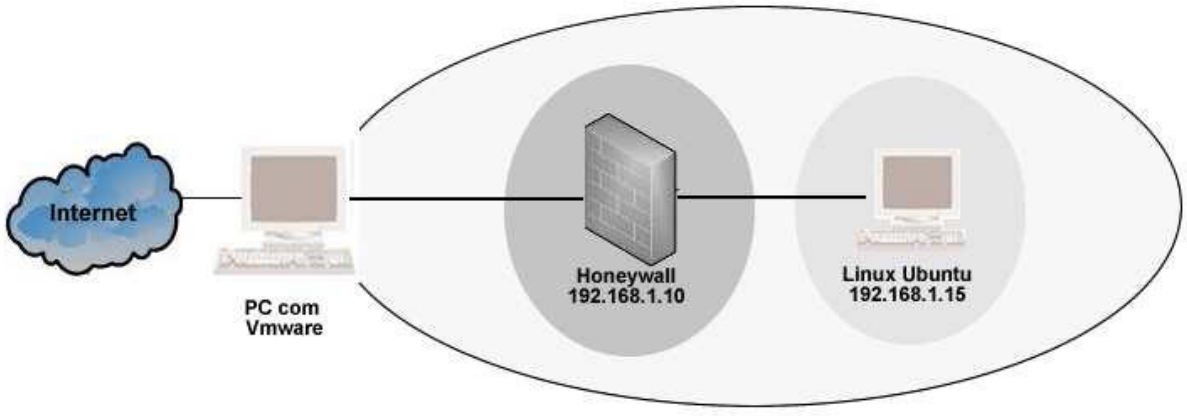

Figura 3.9 - Exemplo *honeynet* virtual.

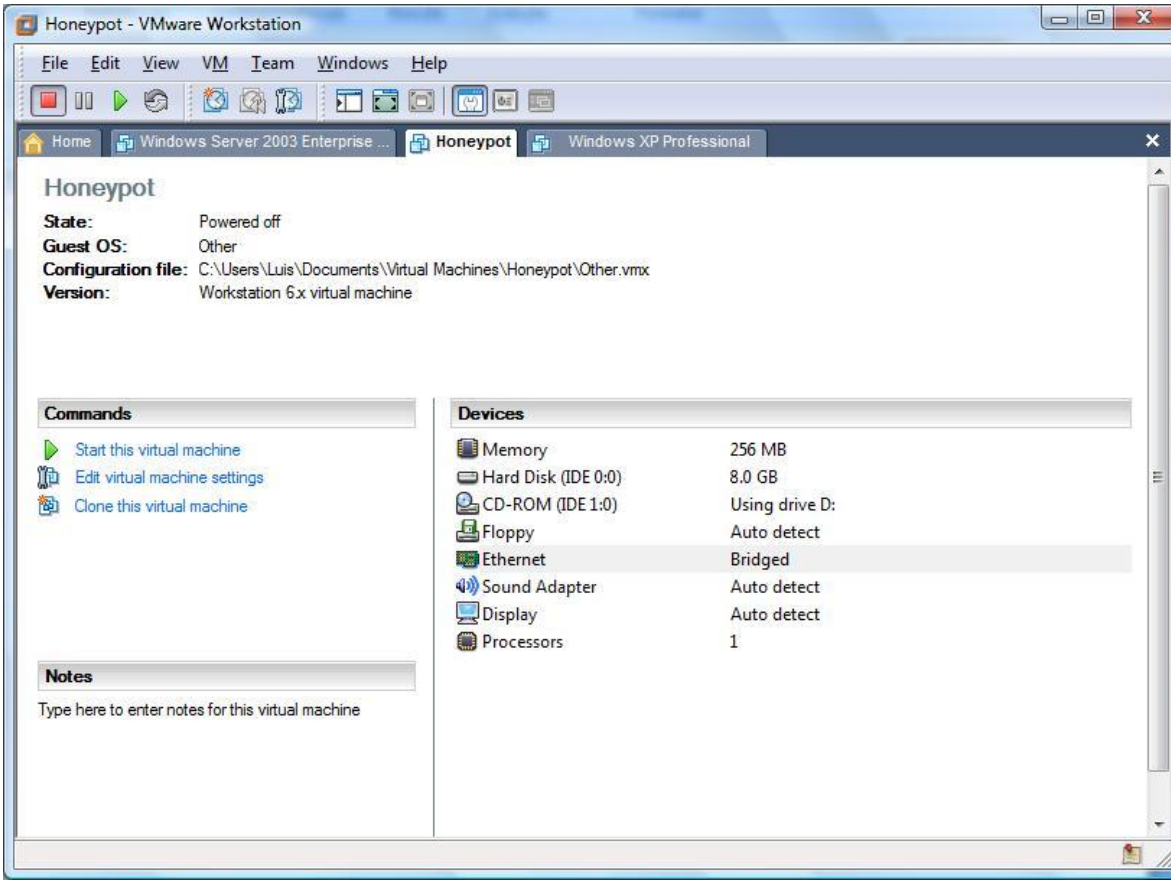

Figura 3.10 - Tela do VMware workstation.

3.10 Projeto Honeynet

É uma organização de pesquisa sem fins lucrativos, composta essencialmente por voluntários com interesse em promover o aumento da segurança na Internet, sem custos para o público em geral [PROJECT, 2008]. Dentro do Projeto seus membros publicam artigos, apresentações e ferramentas para avaliação e aprendizado da comunidade. Para alcançar este objetivo, seguem as seguintes ações em ordem:

- Conscientização: Enfatizam os riscos e vulnerabilidades que existem na Internet. Diversas organizações e indivíduos não têm consciência destes riscos e nem suas conseqüências. Como resultado são informados e orientados a diminuir seus potenciais riscos.

- Informação: Para aqueles que já estão cientes das vulnerabilidades, são ensinados técnicas e informações cruciais para melhorar a segurança de suas redes.
- Ferramentas: São fornecidas ferramentas e apoio para os que desejam continuar a pesquisa por conta própria.

O *Honeynet Research Alliance Group* é composto por grupos independentes que desenvolvem e implementam tecnologias associadas ao uso de *honeypots* e dividem a informação entre si. A reunião dos diversos projetos *honeynet* espalhados pelo mundo permite uma análise mais apurada e precisa no entendimento dos ataques e motivações. Dentre os membros ativos estão, por exemplo, os grupos *Japanese Honeynet Project*, *New Zeland Honeynet Project* e o *Brazilian Honeynet Project* [HONEYNET ALLIANCE, 2002].

3.11 Projeto Honeynet.br

Assim como o *Honeynet Project*, o *Honeynet.br Project* constitui um *Brazilian Honeypots Alliance*, que é um consórcio de *honeypots* distribuídos formados por diversos *honeypots* espalhados pelo país.

O objetivo é aumentar a capacidade de detecção de incidentes através da análise de dados do espaço da internet brasileira. Para tanto, são montadas e configuradas diversas redes compostas de *honeypots* de baixa interatividade (honeyd) que juntos formam uma bases de dados que é reunida e analisada para posteriormente serem disseminadas com a parceria do *Computer Security Incidente Response (CSIRTs)* [HONEYNET ALLIANCE, 2002].

No site do *Honeynet.br* estão disponíveis, além de ferramentas e artigos, o status periódico e atualizado dos dados estatísticos do tráfego das redes com *honeypots* que compõem o *Brazilian Honeypots Allaince*. Estes gráficos são divididos por categorias: portas, código de países e sistemas operacionais de origem como ilustrado na figura 3.11.

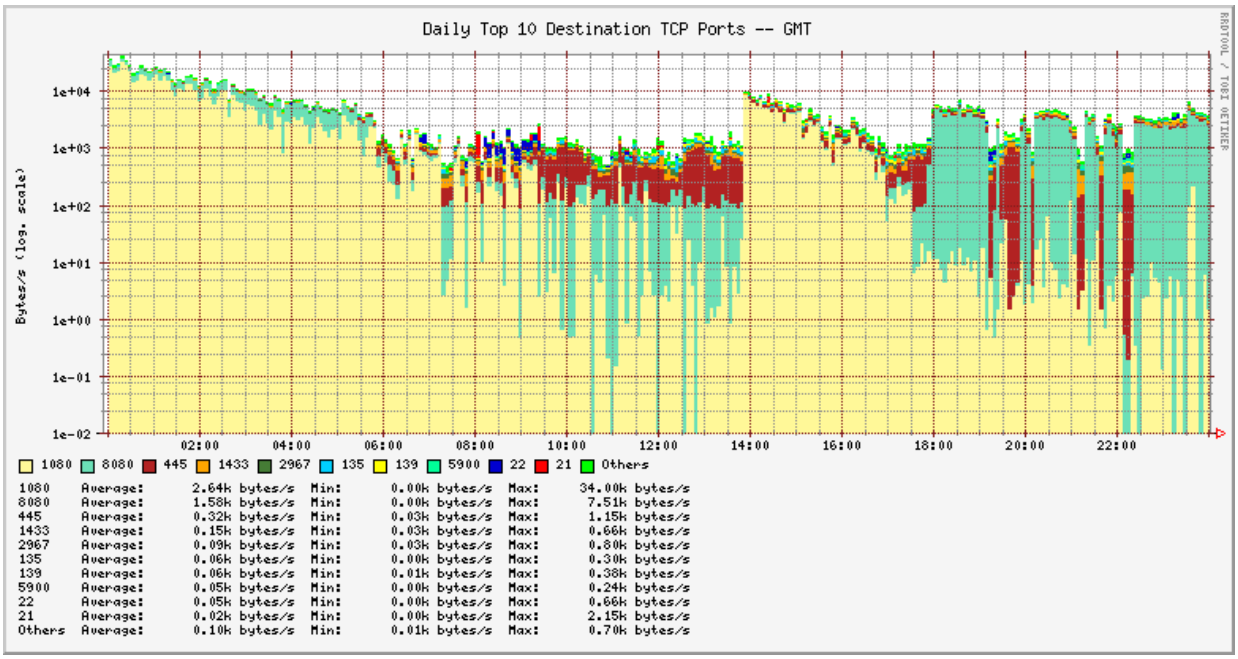

Figura 3.11 - Gráfico top 10 portas *Honeynet.br Alliance Group*.

4 PROPOSTA E MODELO DE SOLUÇÃO

Os extensos relatórios da mídia sobre a disseminação de softwares mal-intencionados através da internet aumentou significativamente o perfil das ameaças externas aos recursos de rede das organizações. Entretanto, algumas das maiores ameaças à infra-estrutura de qualquer organização vêm de ataques originados na rede interna. Os ataques internos com maior potencial de dano resultam das atividades de pessoas nas posições mais confiáveis. A análise das ameaças internas e externas levou muitas organizações a pesquisarem sistemas que monitoram as redes e detectam ataques. [MICROSOFT, 2009]

Sendo assim, neste modelo de projeto será implementada uma solução de *honeypot* na rede interna para analisar os ataques internos.

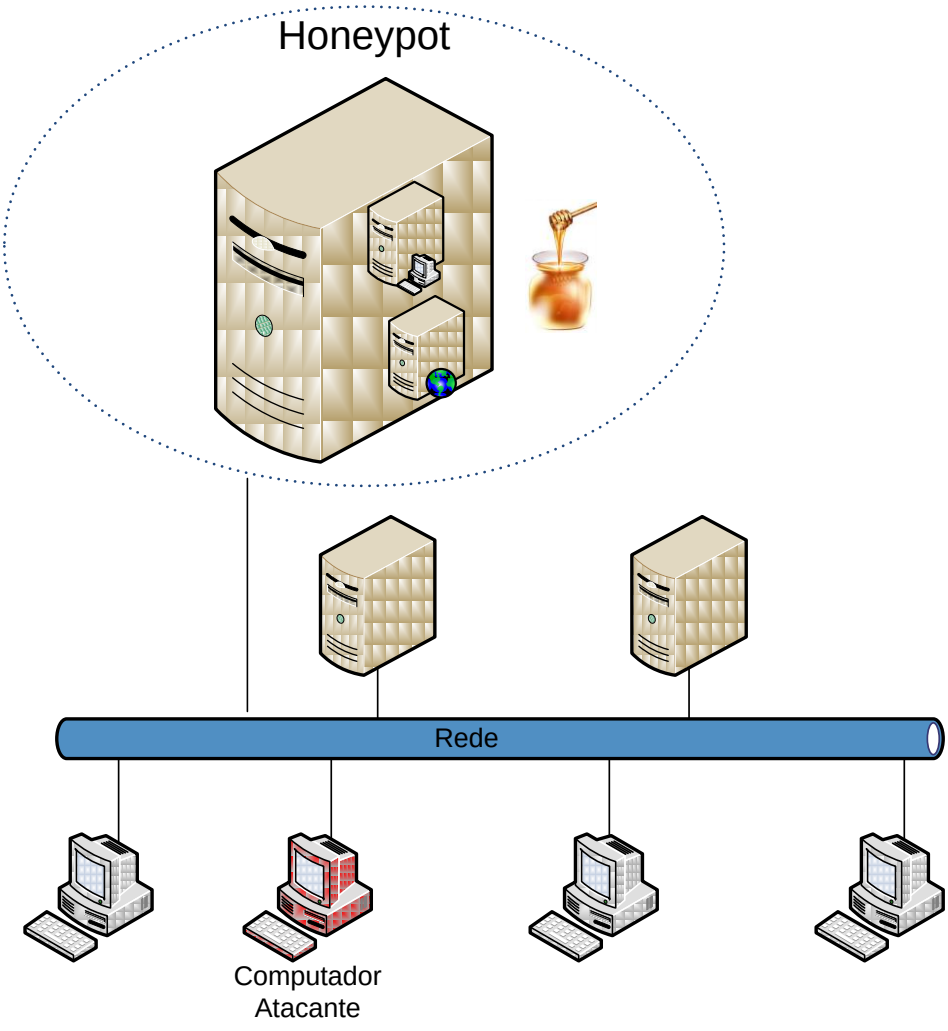

Figura 4.1 - Topologia da solução proposta.

FLUXO DO PROCESSO – MODELO DE IMPLEMENTAÇÃO

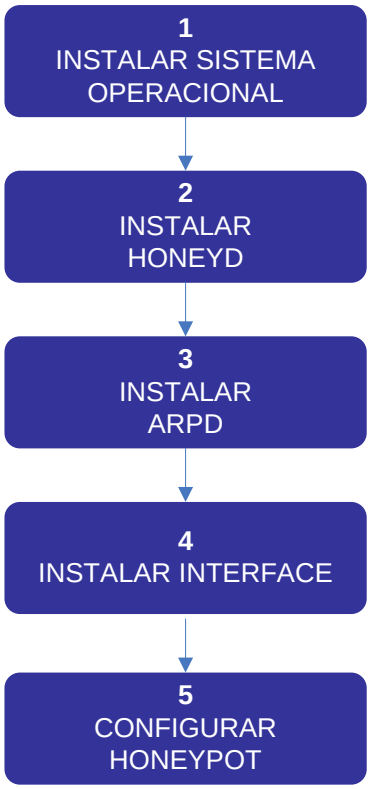

Figura 4.2 - Processo do modelo de implementação da solução proposta.

4.1 Sistema Operacional

Na instalação foi utilizado sistema operacional virtualizado, uma vez que possibilita testar em uma só máquina todos os serviços através do acesso do *host* ao sistema virtualizado. O Sistema Operacional escolhido para implementação foi o OpenBSD, devido ao alto nível de segurança que o mesmo apresenta e oferece suporte a diferentes plataformas.

O OpenBSD 4.5 foi instalado, utilizando-se na instalação o tipo de terminal VT220, configuração de teclado PC-AT e tabela de codificação de teclado “BR” (ABNT 2). O OpenBSD foi a primeira partição criada, com 8gb de espaço e com flag do tipo de partição no código hexadecimal A6 (identificação do sistema OpenBSD). As partições também foram configuradas como inicializáveis (flag de “*boot*” ativas).

Configuradas as partições, nos sistemas baseados no BSD Unix, também é preciso configurar os seus *slices*. *Slice* podem ser entendidos como um sub-particionamento. Dessa forma, a partição que foi definida acima ficará subdividida em espaços de tamanhos diferentes e são esses espaços que o sistema operacional utilizará como pontos de montagem.

Os *slices* utilizados foram os seguintes:

Slice	Tamanho	Ponto de Montagem
A	150 Mb	/
B	512 Mb	Swap
C	-	-
D	120 Mb	/tmp
E	80 Mb	/var
G	4 Gb	/usr
H	2 Gb	/home

Tabela 4.1 - Particionamento do OpenBSD

O dispositivo de disco rígido no OpenBSD é identificado como “wd”. Como nas máquinas utilizadas, o disco rígido encontra-se como Master do barramento primário da controladora IDE, mais conhecido como *Primary Master* e é o primeiro na contagem de dispositivos do *kernel* [OpenBSD FAQ, 2005]. Ou seja, no diretório de dispositivos, o arquivo de identificação do disco rígido no sistema atual é “wd0”.

Dessa forma, têm-se também os slices acima definidos como arquivos no diretório de devices:

Disco Rígido: /dev/wd0

Slice A: /dev/wd0a

Slice B: /dev/wd0b

Slice C: /dev/wd0c

Slice D: /dev/wd0d

Slice E: /dev/wd0e

Slice G: /dev/wd0g

Slice H: /dev/wd0h

Tem-se então que o *Slice* A é o “/”, ou seja, o *root* ou raiz do sistema. Por padrão, os *kernels* do OpenBSD são colocados na raiz do sistema, de forma que a primeira partição a ser lida, para que o sistema possa subir é o /. Dessa forma, fisicamente, o *Slice* A corresponde aos primeiros *bytes* do disco rígido.

O *Slice* B, no OpenBSD, é utilizado como Swap *Space* ou mais conhecido como memória virtual. O *Slice* C não pode ser utilizado, pois no OpenBSD ele representa a identificação da partição inteira. O *Slice* D, definido como /tmp é o diretório onde o sistema operacional cria, apaga e se utiliza do espaço para fazer operações que necessitem de arquivos temporários. O *Slice* E, ou o /var, é o espaço definido para ser utilizado com arquivos que variam, ou seja, *spools* de servidores de e-mail, arquivos de cachê, *logs* do sistema, etc. O *Slice* G é o /usr, onde ficam os arquivos de sistema, e que na atual topologia, foi definido como o maior dos *slices* porque para o projeto será necessário espaço para as informações coletadas dos ataques. Finalmente, o *Slice* H é o /home, onde ficam os arquivos de todos os usuários (exceto os do administrador do sistema). Todos os *Slices* (exceto B e C, por serem de sistema) foram formatados com o tipo de sistema de arquivos 4.2BSD.

4.2 Honeyd

O *honeyd*, foi utilizado, pois emula diversos serviços tornando-o um *honeypot* de baixa interatividade. Inicialmente podemos considerá-lo um *honeypot* de produção que visa à detecção de ataques, porém ele também possui aplicações específicas de pesquisa. Através de *scripts*, ele é capaz de simular diversos sistemas operacionais e são suportados os protocolos UDP, TCP e ICMP como ilustrado na figura 4.3.

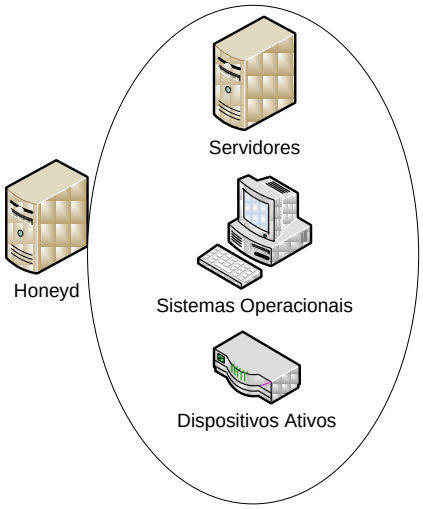

Figura 4.3 - Honeyd.

Baseado nas práticas de Software Livre, todo o código do *honeyd* é disponibilizado o que nos permite customizá-lo para fins específicos. Esta customização é de grande valor, pois diferentemente de outros *honeypots*, podemos emular serviços em qualquer porta TCP que quisermos.

O desenvolvimento do *honeyd* trouxe novos conceitos para as tecnologias associadas à *honeypots,* como por exemplo, ele não responde apenas a ataques direcionados ao próprio IP, pois pode assumir diversas identidades. É conceitualmente a aplicação de *honeypots* para toda a rede. Além disso, pode simular diversos Sistemas Operacionais ao mesmo tempo. Por fim, outro novo conceito é a capacidade do *honeyd* em emular além das aplicações, a pilha TCP, fazendo com que estes serviços emulados se comportem como verdadeiros Sistemas Operacionais.

O arquivo mais recente de instalação do *honeyd* encontra-se em: http://www.citi.umich.edu/u/provos/honeyd/honeyd-1.5c.tar.gz

4.3 ARPD

O *honeyd* não é capaz de direcionar para si todos os ataques, sendo assim ele deve identificar os endereços IP que não estão ativos na rede. Ele realiza isso em duas situações de configurações diferentes de ambiente de rede. A primeira corresponde à situação onde não existem sistemas ativos e sim apenas o *honeyd*.

Para tanto, ele utiliza o método *blackholing*. O funcionamento é análogo ao buraco negro, pois como sabemos que não existe serviço ativo, então certamente é um ataque, por conseqüência todo o tráfego é direcionado para o *honeypot*.

A segunda situação corresponde a uma topologia onde temos, além do *honeypot*, um ambiente de produção real. Nesse ambiente o *honeyd* utiliza uma forma mais complexa de direcionar este tráfego, através de outro programa ARPD, utilizando o método de ARP *spoofing* ou ARP *proxy*. Esta ferramenta atuando juntamente com o *honeyd* atende as requisições ARP através da monitoração dos endereços IP que não estão ativos.

Com as requisições ARP atendidas, ou seja, mesmo quando um endereço IP não existente é atacado, o *honeyd* assume a identidade de vítima e interage com o atacante [SPITZNER, 2002]. Após essa conexão, a porta selecionada pelo atacante é identificada e inicia-se a interação.

Como o *honeypot* proposto é para identificar ataques internos, existe a necessidade do ARPD para direcionar os ataques para o *honeyd*.

O arquivo mais recente de instalação do ARPD encontra-se em: http://www.citi.umich.edu/u/provos/honeyd/arpd-0.2.tar.gz

4.4 Interface gráfica

O *honeyd* é uma ferramenta excelente de coleta de dados dos ataques, porém, pode ser difícil obter uma visão geral do que realmente está acontecendo. A análise do volume de dados gerados consome tempo e não é uma tarefa simples. De posse desses dados capturados, cabe ao administrador fazer a análise e extrair informação útil, o que exige uma percepção boa para distinguir realmente a utilidade dela. É aconselhável e praticamente necessário o uso de ferramentas que permitam a visualização rápida destes dados através de tabelas e gráficos.

A idéia básica foi de colocar os dados gerados pelo *Honeypot* em um banco de dados para permitir consultas eficientes em uma grande quantidade de dados, utilizando uma interface gráfica de acesso via web, que poderá permitir:

- Consultar e visualizar os dados em um formato baseado em texto (text-based);

- Gerar gráficos para obter uma visão geral rápida.

No modelo proposto, será instalada uma interface gráfica baseada na ferramenta *HoneyView*. O arquivo de instalação da ferramenta *HoneyView* encontra-se em: http://sourceforge.net/projects/honeyview

4.5 Honeypot

Com as possibilidades de configuração e emulação de diversos sistemas operacionais através da criação de *hosts* virtuais é possível a criação de topologia de redes inteiras, formadas por roteadores, servidores e clientes, todos com características específicas.

O arquivo de configuração do *honeyd*, usualmente nomeado como honeyd.conf contém os *templates* que definem as características dos *honeypots*. Existem inúmeras possibilidades de criação. Neste modelo, optou-se por emular um servidor web, um roteador e um sistema operacional cliente. Feita esta escolha, a resposta a qualquer tentativa de acesso as portas escolhidas recai em uma das opções de *reset*, *open*, *block* e *script*.

A figura 4.4 apresenta a configuração do arquivo honeyd.conf.

```
### Roteador cisco
create router
set router personality "Cisco IOS 12.1(4) on a 2600 router"
set router default tcp action reset
set router default udp action reset
set router uid 32767 gid 32767
set router uptime 1327650
add router tcp port 23 "/etc/Honeypots/script s/router-telnet.pl"
add router tcp port 80 open

### winxppro
create winxppro
set winxppro personality "Microsoft Windows XP Professional"
set winxppro default icmp action open
set winxppro default tcp action reset
set winxppro default udp action reset
add winxppro tcp port 445 open
add winxppro udp port 445 open
add winxppro tcp port 23 "/etc/Honeypots/script s/cmdexe.pl -p winxp -l
/etc/lh/cmdexe"

### webserver
create winserver
set default personality "Windows 2003 Server SP1"
set default default tcp action reset
add default tcp port 110 "sh scripts/pop.sh"
add default tcp port 80 "perl scripts/iis-0.95/main.pl"
```

```
add default tcp port 25 block
add default tcp port 21 "sh scripts/ftp.sh"
add default tcp port 22 proxy $ipsrc:22
add default udp port 139 drop
set default uptime 3284460
```

Figura 4.4 - Configuração do arquivo honeyd.conf.

5 RESULTADOS DO PROJETO DE HONEYPOT

5.1 Cenário

Para o teste de implementação foi instalado o *honeypot* em uma máquina virtual, emulada em um servidor utilizando o VMWare ESXi 4.0, instalada conforme a figura 5.1.

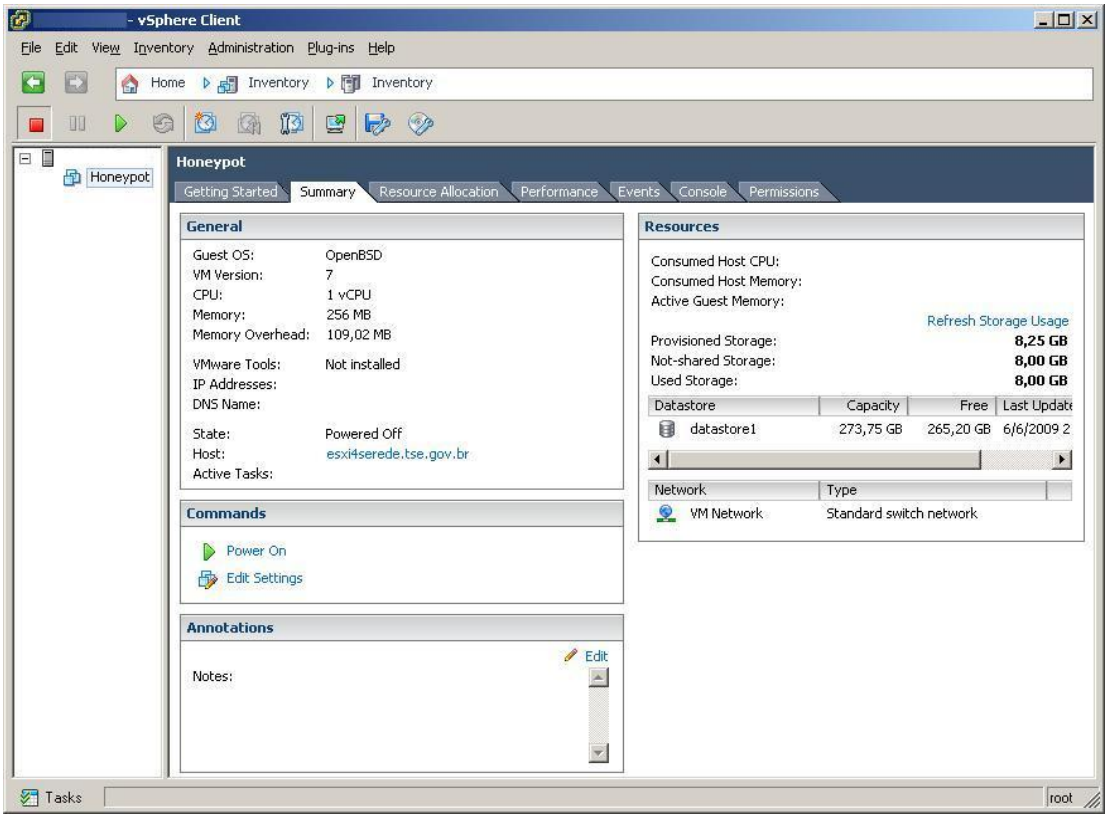

Figura 5.1 - Máquina virtual *honeypot.*

Configuração da máquina virtual:

- 1Processador
- 256 RAM
- 8 GB
- OpenBSD 4.5
- Honeyd 1.05c
- Arpd 0.2

O *honeypot* será instalado em uma rede local com mais de 1600 estações e por volta de 100 servidores. Por se tratar de uma instituição pública, será mantida a confidencialidade das informações, com a utilização de IPs e nomes fictícios.

A configuração dos *scripts* e do honey.conf foram executados de acordo com o modelo proposto. Foi atribuído um IP para a máquina que foi instalada o *honeypot* e um intervalo de IPs para as 3 máquinas emuladas pelo *honeyd*. O ARPD se encarregou de direcionar os acessos aos IPs não utilizados pela rede para o *Honeyd*. O cenário da implementação esta descrito na figura 5.2.

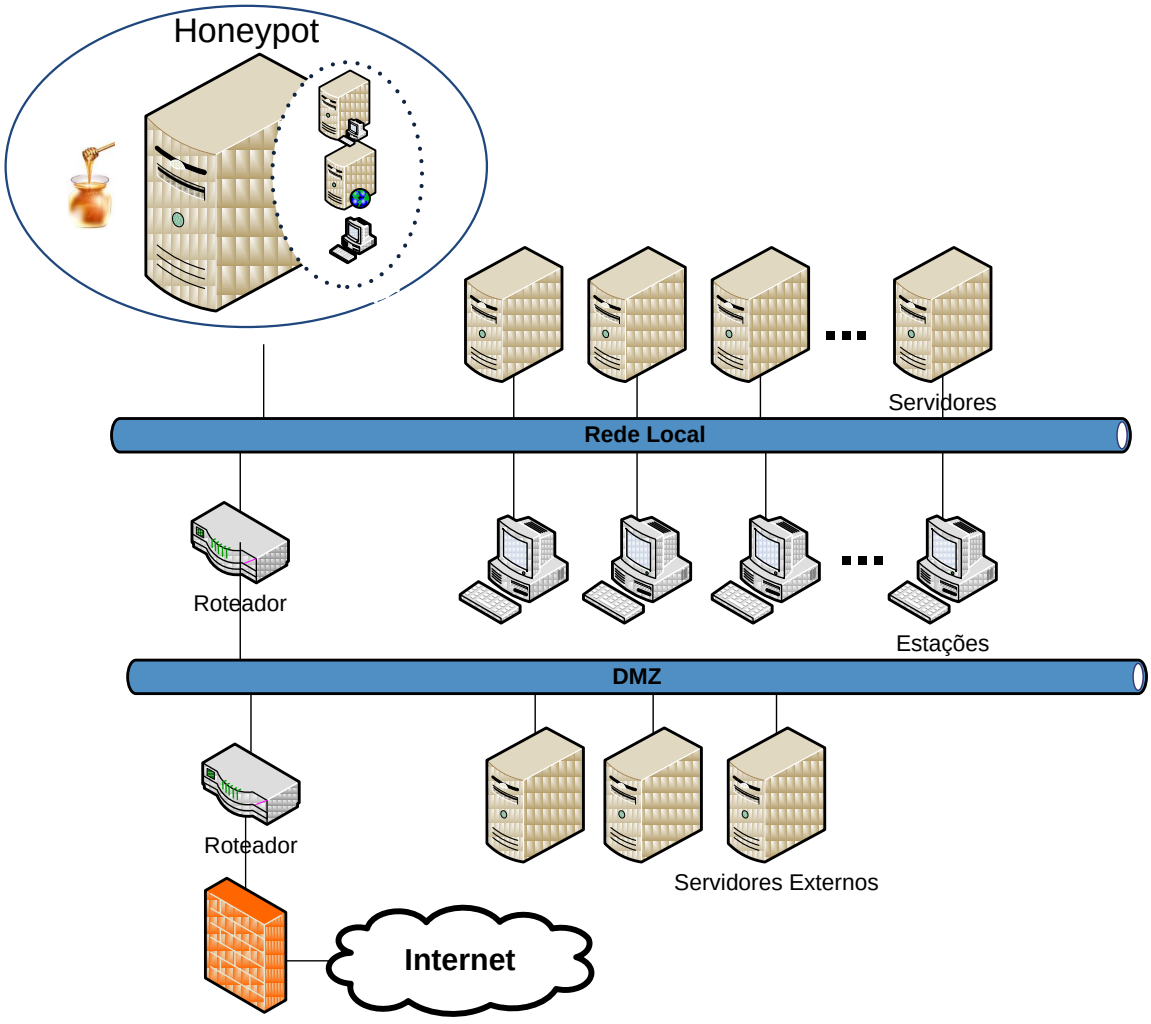

Figura 5.2 - Cenário da implementação.

Com a instalação do Sistema Operacional OpenBSD 4.5, com a instalação do *honeyd* e ARPD, falta a instalação da interface baseada no *HoneyView*.

5.2 Instalação da Interface Gráfica.

A instalação da interface gráfica não é uma tarefa trivial, uma vez que necessita de diversos pré-requisitos para executá-la, conforme listado abaixo:

- Versão do *Honeyd* com *patch* do *Honeyview*;
- Webserver com suporte a PHP e Mysql;
- Interpretador do PHP-CLI *Standalone* com suporte ao Mysql;
- Mysgl-dbms rodando.

5.2.1 Configuração do Servidor de Monitoração:

- Base de dados do *Honeyview* deve ser chamada de "master_hlog"
- Programar a cron-job para enviar a cada hora o arquivo chamado "hlog.gz" para o diretório "tmp" (este arquivo contem todo o trafego logado durante uma hora pelo honeyd)

5.2.2 Configurar Base de Dados:

- Mudar o diretório da base de dados para o diretório “bin”;
- Editar o arquivo "dbaccess.conf" para configurar a autenticação do dbms-server;
- Colocar a master_hlog no /init.db;
- Chamar a base de dados de "master_hlog" de DBMS.

5.2.3 Arquivos de configuração

- Mudar para o diretório "conf"

5.2.4 *Configurar a cron-job*

- Definir a periodicidade da transferência do arquivo “hlog.gz” para o diretório “tmp”.

5.2.5 *Configurar o cliente Web*

- Editar o arquivo “config.php” para permitir apenas os usuários que tenham permissão de leitura.

5.2.6 *Configurar o honeyd*

- Instalar a versão do *honeyd* com o *patch*;
- Transferir os scripts para o diretório "scripts" do *honeyd*

Depois de concluídas todas as configurações, a *honeyview* não funcionou, pois o *patch* disponibilizado para o *honeyd* era para a versão 0.5 e o *honeyd* instalado está na versão 1.5c. Sendo assim, foi necessário customizar o *patch* para a nova versão. A figura 5.3 mostra o primeiro teste da interface gráfica.

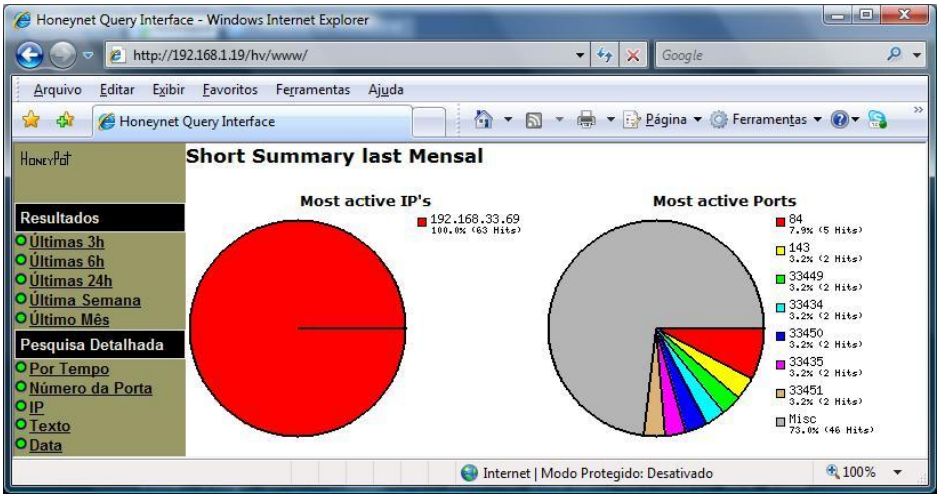

Figura 5.3 - Interface Gráfica primeiro teste.

Após a interface instalada, foi realizado o teste para verificar se o ARPD direcionaria o tráfego para o *honeyd*. Como visto anteriormente, o *honeyd* não foi capaz de direcionar para si os ataques.

```
>arpd 10.80.1.0/24
```

Com os dois sistemas funcionando corretamente, foi configurado o arquivo honeyd.conf, que contém os *hosts* emulados e portas abertas, associados aos respectivos IPs.

A execução do *honeyd* foi iniciada através da linha de comando:

```
>honeyd -d -f honeyd.conf -p /usr/local/share/honeyd/nmap.prints - l /etc/lh/honeyd.log 10.80.1.100
```

Onde:

- -d - executa o *honeyd* em primeiro plano;
- -f - indica onde está localizado o arquivo de configuração do *honeyd*;
- -p - indica onde está localizado o arquivo que contem as assinaturas dos sistemas operacionais emulados;
- - l - indica o caminho do arquivo de log. de tentativas de conexões em cada host virtual emulados pelos scripts.

Os *scripts* utilizados nessa solução de *Honeypot* são:

- iisemul8.pl – emula o servidor web iis (Internet Information Server);
- cmdexe.pl – emula a tela do prompt do DOS;
- router-telnet.pl – emula telnet.

Além do registro de conexão feito pelo *honeyd*, o *script* também registra os comandos executados na interação com o atacante, em arquivo separado. Esse arquivo é configurado na execução do *honeyd* é o caminho após o -l.

Depois de feito todos os testes de funcionalidade, a máquina foi colocada na rede para fazer a coleta de dados. O período de coleta de dados foi de 11 a 22 de maio de 2009.

5.3 Análise dos Resultados

5.3.1 Primeira Coleta de Dados

O período de coleta de dados foi de 11 a 22 de maio de 2009. A primeira análise feita é o número total de conexões. Como mostra a figura 5.4.

Quantidade de Acessos

	Número de acessos	%
TCP	90	23,87
ICMP	238	63,13
Total	377	100

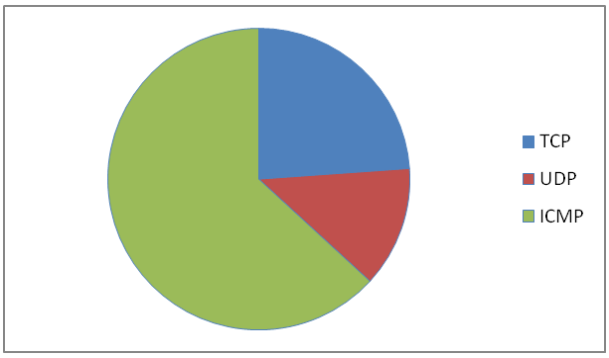

Figura 5.4 - Total de acessos ao *honeypot.* - primeira coleta de dados.

Foi detectada quantidade inferior de conexões do que era esperado. Ao analisar o porquê do resultado, foi considerada a localização do *honeypot* em uma rede interna, protegida por *Firewall* e IDS. Contanto, esse não foi o motivo da baixa quantidade de acessos. Com a idéia de fazer um *honeypot* emulando um roteador, um servidor web e uma estação cliente, conforme o modelo proposto, fez com que o *honeypot* se tornasse pouco atrativo. Assim, para que ele fosse mais atrativo, foram adicionados mais sistemas emulados. Abaixo é demonstrado o resultado da nova configuração feita no arquivo *honeyd.*

```
### Roteador cisco
create router
set router personality "Cisco 1601R router running IOS 12.1(5)"
set router default tcp action reset
set router default udp action reset
set router uid 32767 gid 32767
set router uptime 1327650
add router tcp port 23 "/etc/Honeyd/scripts/router-telnet.pl"
add router tcp port 80 open
### winxppro
create winxppro
set winxppro personality "Microsoft Windows XP Professional"
set winxppro default icmp action open
set winxppro default tcp action reset
set winxppro default udp action reset
add winxppro tcp port 445 open
add winxppro udp port 445 open
```

```
add winxppro tcp port 23 "/etc/honeyd/scripts/cmdexe.pl -p winxp -l
/etc/lh/cmdexe"

### webserver
create winserver
set default personality "Windows 2003 Server SP1"
set default default tcp action reset
add default tcp port 110 "sh scripts/pop.sh"
add default tcp port 80 "perl /etc/honeyd/scripts/iisemulator-0.95/iisemul8.pl"
add default tcp port 25 block
add default tcp port 21 "sh scripts/ftp.sh"
add default tcp port 22 proxy $ipsrc:22
add default udp port 139 drop
set default uptime 3284460

### Atrativos
create backdoor
set backdoor personality "Microsoft Windows XP Home Edition"
set backdoor default tcp action reset
set backdoor default udp action reset
set backdoor default icmp action open
# blaster
add backdoor tcp port 4444 "/etc/Honeyd/scripts/cmdexe.pl -p winxp –l /etc/lh/cmdexe"
# sasser
add backdoor tcp port 5554 "/etc/Honeyd/scripts/cmdexe.pl -p winxp –l /etc/lh/cmdexe"
add backdoor tcp port 9996 "/etc/Honeyd/scripts/cmdexe.pl -p winxp –l /etc/lh/cmdexe"
# dabber
add backdoor tcp port 8967 "/etc/Honeyd/scripts/cmdexe.pl -p winxp –l /etc/lh/cmdexe"
add backdoor tcp port 9898 "/etc/Honeyd/scripts/cmdexe.pl -p winxp –l /etc/lh/cmdexe"
# lovgate
add backdoor tcp port 20168 "/etc/Honeyd/scripts/cmdexe.pl -p winxp –l /etc/lh/cmdexe"
add backdoor tcp port 1720 "/etc/Honeyd/scripts/cmdexe.pl -p winxp –l /etc/lh/cmdexe"

### FreeBSD computer 4.7
create freebsd
set freebsd personality "FreeBSD 4.7-RELEASE (X86)"
set freebsd default icmp action open
set freebsd default tcp action reset
set freebsd default udp action reset
add freebsd tcp port 80 open
add freebsd tcp port 21 "sh /etc/Honeyd/scripts/ftp.sh"add freebsd udp
port 53 open
set freebsd uptime 3284460

### Linux 2.4.x computer
create linux
set linux personality "Linux 2.4.16 - 2.4.18"
set linux default tcp action reset
set linux default udp action reset
set linux default icmp action open
add linux tcp port 110 "sh /etc/Honeyd/script s/pop3.sh"
add linux tcp port 25 "sh /etc/Honeyd/script s/smtp.sh"
add linux tcp port 21 "sh /etc/Honeyd/script s/ftp.sh"
set linux uptime 3284460

### Mydoom vírus
create mydoom
set mydoom personality "Microsoft Windows XP Professional"
set mydoom default icmp action open
set mydoom default tcp action reset
```

```
set mydoom default udp action reset
add mydoom tcp port 445 open
add mydoom udp port 445 open
set mydoom uid 32767 gid 32767
add mydoom tcp port 1080 "/etc/Honeyd/scripts/mydoom.pl -l /etc/lh"
add mydoom tcp port 3127 "/etc/Honeyd/scripts/mydoom.pl –l /etc/lh/mydoom"
add mydoom tcp port 3128 "/etc/Honeyd/scripts/mydoom.pl –l /etc/lh/mydoom"
add mydoom tcp port 10080 "/etc/Honeyd/scripts/mydoom.pl –l /etc/lh/mydoom"


bind 10.80.1.200 router
bind 10.80.1.201 winserver
bind 10.80.1.202 winxpro
bind 10.80.1.203 backdoor
bind 10.80.1.204 backdoor
bind 10.80.1.205 freebsd
bind 10.80.1.206 linux
bind 10.80.1.207 mydoom
bind 10.80.1.208 mydoom
bind 10.80.1.209 winxppro
bind 10.80.1.210 winxppro
bind 10.80.1.211 winxppro
bind 10.80.1.212 winxppro
```

Figura 5.5 - Configuração do arquivo honeyd.conf.

Foi alterado a configuração do arquivo honeyd.conf do modelo proposto, conforme mostrado na figura 5.5 e colocado novamente na rede para uma segunda coleta de dados.

5.3.2 Segunda Coleta de Dados

O período da segunda coleta de dados foi de 25 de maio a 05 de junho de 2009. Com a nova configuração do *honeypot* o resultado mostrado foi de acordo com o esperado, onde o número total de conexões aumentou mais de 440% em relação a primeira coleta de dados, considerado um aumento muito significativo. O gráfico mostrado na figura 5.5 demonstra o resultado dos testes realizados. O maior número de acessos foi feito utilizando o protocolo ICMP, com quase 50% seguido dos acessos via TCP com 40,2%. O restante utilizando o protocolo UDP com 10,18%.

Quantidade de Acessos

	Número de acessos	%
TCP	675	40,20
UDP	171	10,18
ICMP	833	49,61
Total	1679	100

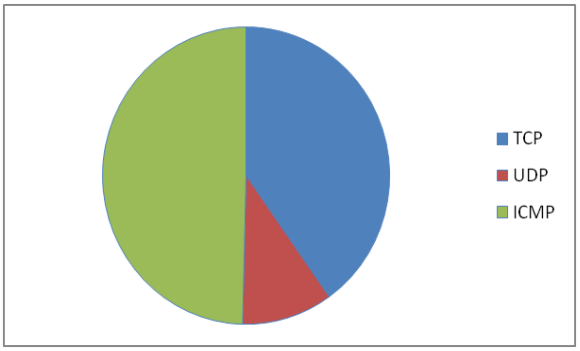

Figura 5.6 - Total de acessos ao *honeypot* - segunda coleta de dados.

Com o resultado da captura, foram analisados os recursos mais acessados, apresentando os seguintes resultados (representados pelo número da porta/ protocolo), conforme a figura 5.7:

Recursos mais Acessados	
Recursos	**Conexões**
8/ICMP	833
445/TCP	301
8080/TCP	210
6553/TCP	96
27520/UDP	38
4444/TCP	29
1026/UDP	15
1027/UDP	13
5900/TCP	5
6588/TCP	3

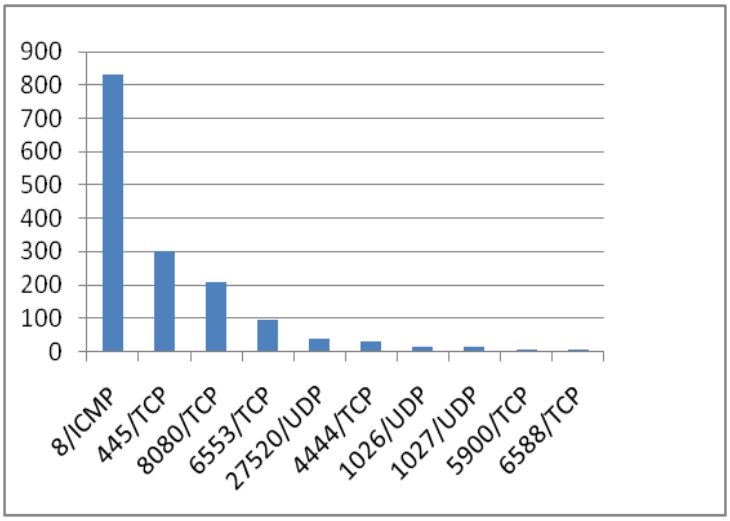

Figura 5.7 - Recursos mais acessados na segunda coleta.

Analisando as portas de acordo com a IANA (*The Internet Assigned Numbers Authority*):

- 8 – ICMP – ping;
- 445 – Microsoft-DS;
- 8080 – Http alternativa;
- 6553 – Não há serviços, programas, Trojans ou *worms* associados a esta porta;

- 27520 – Não há serviços, programas, Trojans ou *worms* associados a esta porta;
- 4444 – krb524;
- 1026 e 1027 UDP – Diversas mensagens de spam são enviadas através do Windows Messenger nesta porta;
- 5900 – VNC Server;
- 6588/TCP - Não há serviços, programas, Trojans ou *worms* associados a esta porta.

Quanto a análise dos IPs que fizeram tentativas de acesso ao *honeypot*, como o ambiente testado é uma rede local, as máquinas atacantes foram facilmente localizadas. Embasado nessas informações, foi possível já no primeiro momento tomar providências a respeito das máquinas atacantes. Os ataques direcionados a porta 445 foram efetuados por uma máquina servidora, que não possui antivírus instalado e estava contaminada com o *Worm Downad*. Os ataques direcionados a outras portas foram realizados por um *scanner* de rede, ou seja, por um usuário mal intencionado.

5.3.3 Análise Geral da Solução

A engenharia preza por soluções eficientes e eficazes e com o menor custo possível. Analisando por esta ótica, o modelo proposto foi eficiente, pois conseguiu solucionar o problema inicial de trazer de forma útil e objetiva as informações dos ataques das redes internas, segundo proposto pela solução. No entanto, com o modelo proposto inicialmente, conforme primeira amostra, não se obteve a eficácia desejada, uma vez que não foram geradas informações suficientes para análise.

Com a análise da primeira coleta de informações do *Honeypot*, foi identificada a necessidade de alteração do modelo para aumentar a geração das informações, de forma a obter o resultado esperado. Feitas as alterações, segundo resultados apresentados, comprova-se o cumprimento dos requisitos desejados para solução de projeto proposta.

De acordo com os resultados apresentados, conclui-se que os *honeypots* baseados em software livre se mostram de grande valor para desempenhar a busca de informações sobre os ataques feitos na rede, funcionando como uma ferramenta estratégica para a elaboração de contramedidas de segurança de baixo custo.

CONCLUSÃO

Um dos objetivos da engenharia é desenvolver trabalhos com a maior eficiência e utilizando o menor custo possível. Neste sentido, os resultados apresentados através da solução proposta, demonstraram que é possível utilizar sistemas operacionais baseados em software livre para se implementar uma segurança de rede pró-ativa através do *honeypot*. O baixo custo da solução provém não só do fato do software livre em si não causar ônus, mas também do fato desses tipos de *software* serem feitos para um grande número de plataformas diferentes, utilizando vários recursos, com novas tecnologias em uma máquina de baixo custo ou mesmo em ambientes virtuais como é o caso deste trabalho.

Do outro lado, temos as comunidades e empresas de segurança, investindo em novas soluções e correndo para corrigir as falhas, aprimorar os sistemas de antivírus e sistemas de segurança. Todo este trabalho é feito basicamente através do estudo das ferramentas utilizadas pelos invasores e também da análise dos ambientes comprometidos, o que nem sempre é uma tarefa simples. Além deste trabalho investigativo, é necessário a observação do comportamento dos invasores, com intuito de detectar as tendências e os motivos dos ataques, objetivo este que é atendido pelo foco de estudo deste trabalho, com a implementação da solução de *honeypot*.

A solução de *honeypot* mostrou grande eficiência na redução do risco à segurança da rede proposta, porém, sua implementação não é simples e pode permitir que um administrador despreparado exponha seu sistema a riscos desnecessários. A princípio, pode ser difícil obter uma visão geral do que realmente está acontecendo, uma vez que a análise do volume de dados gerados consome tempo e não é uma tarefa simples para o administrador de rede. No entanto, uma vez que um *honeypot* esteja em funcionamento, as vantagens para o administrador da rede vão desde a aquisição de provas e análise dos ataques até o aprendizado de novas ferramentas e formas de ataque.

No modelo proposto para capturar as informações dos ataques e dos atacantes a uma rede interna, as informações capturadas foram demonstradas de forma gráfica, permitindo uma rápida visualização dos dados através de tabelas e gráficos, facilitando a análise e agregando valor a segurança de rede. Além disso, a

exploração das vulnerabilidades foi focada, onde se observou maior possibilidade de riscos à rede.

Contudo, como todas as tecnologias de segurança a rede, o *honeypot* também apresentou alguns problemas, sendo o mais significativo relacionado ao seu limitado campo de visão, uma vez que só foi possível capturar as atividades direcionadas contra ele. Sendo assim, conclui-se que a solução de *honeypot* não se aplica como ferramenta de proteção direta à rede, sendo sua principal função a prevenção de ameaças e mitigação de riscos aos possíveis ataques, útil como ferramenta na prevenção de invasões a redes de computador e no estudo de novas técnicas e ferramentas de invasão. Por essa razão, o *honeypot* não substitui as ferramentas de segurança já existentes e deve ser implementado como uma tecnologia complementar de rede e de proteção contra intrusões para auxiliar na diminuição destas falhas e contornar as fragilidades de cada ferramenta.

Através do estudo apresentado, foi possível conhecer e compreender a importância do *honeypot* como uma ferramenta adicional ao trabalho desenvolvido pela comunidade de segurança, mas também, ressaltar que para se alcançar o real objetivo desta solução e evitar riscos em sua utilização, é necessário investimento em profissionais qualificados para a preparação e acompanhamento do ambiente.

Por fim, esse projeto é o primeiro passo para os administradores que tenham interesse em melhorar a segurança de suas redes sendo pró-ativos, buscando informações e controlando as vulnerabilidades de suas redes.

GLOSSÁRIO

AIX	Uma versão da IBM para o sistema operacional Unix que é executado em computadores IBM de médio porte.
BOT	Programa que, inclui funcionalidades de *worms*, sendo capaz de se propagar automaticamente através da exploração de vulnerabilidades existentes ou falhas na configuração de *softwares* instalados em um computador.
Bugs	É um erro no funcionamento comum de um software, também chamado de falha na lógica programacional.
Irix	Sistema Operacional baseado no Unix com o BSD desenvolvido pela Silicon Graphics (SGI).
Framework	Estrutura a partir da qual um software pode ser organizado e desenvolvido.
FreeBSD	Sistema Operacional livre do tipo Unix descendente do BSD desenvolvido pela Universidade de Berkeley.
Honeyd	É um *daemon* desenvolvido por Niels Provos para ser utilizado tanto em Windows quanto *nix.
Logs	Termo utilizado para descrever o processo de registro de eventos relevantes num sistema computacional.
NeXTStep	Sistema Operacional, lançado em 10 de setembro de 1989, pela NeXT, atualmente parte da Apple Computer.
Nmap	Software livre que realiza *port scan* desenvolvido pelo auto-proclamado *hacker Fyodor*.
Patch	Alteração feita em uma aplicação para consertar bugs ou adicionar funcionalidades.
Ping	Comando usado pelo protocolo ICMP para testar a conectividade entre equipamentos, desenvolvido para ser usado em redes com a pilha de protocolo TCP/IP (como a Internet).
SPAM	Abreviação em inglês de "*spiced ham*", é uma mensagem eletrônica não-solicitada enviada em massa.
Syslog	é um padrão criado pela IETF para a transmissão de mensagens de log em redes IP.
TELNET	Protocolo cliente-servidor usado para permitir a comunicação entre computadores ligados numa rede (exemplos: rede local / LAN, Internet), baseado em TCP.
Tracert	Significa rastreio de rota e consiste em obter o caminho que um pacote atravessa por uma rede de computadores até chegar ao destinatário.
Unix	Sistema Operacional portátil, multiusuário, originalmente criado por Ken Thompson, que trabalhava nos Laboratórios Bell (Bell Labs) da AT&T.
Unisys	Empresa mundial de serviços e soluções de Tecnologia da Informação.

REFERÊNCIAS

ABNT, Associação Brasileira de Normas Técnicas, “ISO / IEC 17799 –Tecnologia da Informação – Código de prática para a gestão da Segurança da Informação”, Brasil, 2001.

AZEVEDO, Tiago Souza. Honeypots - A segurança através do disfarce. Artigo, baseado na dissertação de mestrado Proposta e Avaliação de um modelo alternativo Baseado em Honeynet para Identificação de Ataques e Classificação de Atacantes na Internet. Honeypots: Tracking hackers. Rio de Janeiro, 2005. COHEN, F. Disponível em <http://www.ravel.ufrj.br/arquivosPublicacoes/honeynet.pdf.>, Acesso em 15/04/2009.

BIZSYSTEM, 2009 Disponível em <http://scans.bizsystems.net/paged_report.plx>, Acesso em 15/04/2009.

CERT.BR, Disponível em < http://www.cert.br/stats/incidentes/2009-jan-mar/tipos-ataque.html>, Acesso em 08/03/2009.

CHESWICK, W. R. *An Evening with Berferd in Which a Cracker is Lured, Endured, and Studied, Proceedings of the Winter* 1992 USENIX Conference, 1992.

CHESWICK, William R. *Firewalls* e Segurança na Internet. 2. ed. Porto Alegre: Bookman, 2005.

D’AVILA, Márcio d'Ávila. Phishing Scam - A fraude inunda o correio eletrônico. 28 de junho de 2004. Revisão 40, 2 de maio de 2008. Disponível em <http://www.mhavila.com.br/topicos/seguranca/scam.html>, Acessado em 16/05/2009.

Deception Toolkit, Disponível em <http://www.all.net/dtk/dtk.html>, Acesso em 08/04/2009.

FAGUNDES, L. Lenardo. Metodologia para avaliação de sistemas de detecção de intrusão. São Leopoldo, Brasil, 2002.

FRANCO, Lucio Henrique, BARBATO, Luiz Gustavo, MONTES, Antonio. Instalação e Uso de Honeypot de Baixa Interatividade. Centro de Pesquisas Renato Archer - CenPRA/MCT, Campinas – SP. *BackOfficer Friendly*, Disponível em < http://www.honeynet.org.br/presentations/hnbr-gts2004-01-tutorial.pdf>, Acesso em 08/04/2009.

FRANCO, L. H. and MONTES. A. Desvio de Tráfego Malicioso Destinado a Redes de Produção para uma Honeynet. *In Grupo de Trabalho em Segurança de Redes,* Rio de Janeiro, 2003.

G. A. Fink and B. L. Chappell and T. G. Turner and K. F. O'Donoghue, "*A Metric-Based Approach to Intrusion Detection System Evaluation for Distributed Real-Time Systems*", *Proceedings of the International Parallel And Distributed Processing Symposium (IPDPS'02)*, 2002.

HONEYNET, Disponível em <http://www.citi.umich.edu/u/provos/honeyd/>, Acesso em 08/04/2009.

Honeynet Research Alliance, Disponível em <http://www.honeynet.org/alliance/>, Acesso em 14/04/2009.

HONEYNET, P. *The Philippine Honeynet Project*, Disponível em http://www.philippinehoneynet.org, Acesso em 15/04/2009.

NEVES, G**.** Implementação de uma Honeynet: Requisitos, Metodologias e Riscos**.** Básicos", CERT.PT FCCN, 2005.

KFSensor, Disponível em <http://www.keyfocus.net>, Acesso em: 05.04.2009.

Mantrap, Disponível em <http://www.trtec.com.br/pages/mantrap.html>, Acesso em 06/03/2009.

MARCELO, Antônio; PITANGA, Marcos. Honeypots: a arte de iludir hackers. Rio de Janeiro: Brasport, 2003.

MCDONALD, J. Defeating Solaris/SPARC Non-Executable Stack Protection. 1999. Disponível em <http://www.thc.org/root/docs/exploit_writing/sol-ne-stack.html>, Acesso em 21/04/2009.

MICROSOFT, Guia de Planejamento de Monitoramento de Segurança e Detecção de Ataques. Disponível em <http://www.microsoft.com/brasil/security/guidance/attackdetection/default.mspx>, Acesso em 23/03/2009.

MIRKOVIC, J., DIETRICH, S., DITTRICH, D., REIHER, P. *Internet Denial of Service: Attack and Defense Mechanisms*. EUA, Prentice Hall PTR, 1ª Edição, 2004.

Netbaitinc, Disponível em <http://www.netbaitinc.com/>, Acesso em 05/03/2009.

NIELS PROVOS, T. H. Virtual *Honeypots: From Botnet Tracking to Intrusion Detection. Addison-Wesley Professional*, 2007.

NORTHCUTT, S.; NOVACK, J.; MCLACHLAN, D., Segurança e Prevenção em Redes –Trad. Marcos Vieira. ed. Berkley, 2001.

OLIVEIRA, Etienne, Tópicos Especiais em Redes II, Arquitetura de Segurança, Hackers, Tipos de Ataque. 01 p. a 05 p., 2003.

OpenBSD. *Supported Hardware,* 2005. Disponível em <http://www.openbsd.org/i386.html>, Acesso em 20/04/2009

OpenBSD FAQ. *Documentation and Frequently Asked Questions,* 2005. Disponível em <http://www.openbsd.org/faq/index.html>, Acesso em 20/04/2009.

PEIXOTO, Jarbas de Freitas. Honeynet: Um ambiente para Análise de Intrusão. 2002. Monografia (Bacharelado em Ciência da Computação). UNESP – Universidade do Estado de São Paulo, São Paulo 2002. Disponível em: http://www.unesp.br, Acesso em 05/03/2009.

PROJECT, H. *Know Your Enemy*, 2008. Disponível em <http://www.honeynetproject.org>, Acesso em 21/04/2009.

PROJECT, T. H. *Know Your Enemy: Revealing the Security Tools, Tactics, and Motives of the Blackhat Community, 2001*.

PROVOS, N. *Developments of the Honeyd Virtual Honeypot*. Disponível em http://www.honeyd.org>, Acesso em 23/04/2009.

RAVANELLO, Anderson Luiz; HIJAZI, Houssan Ali; MAZZORANA, Sidney Miguel. Honeypots e Aspectos Legais. Curitiba, 2004, Dissertação - Especialização em Redes e Segurança – Programa de Pós-Graduação em Informática Aplicada, Pontifícia Universidade Católica do Paraná, Curitiba - PR.

ROJAS, Gislaine Aparecida. *Análise de Intrusões através de Honeypots e Honeynets*. Dissertação, Americana, São Paulo, 2003.

ROOTKIT, 2005. Disponível em <http://www.linhadefensiva.org/2005/03/rootkit/>, Acesso em 25/03/2009.

ROQUE, Aparecida Kátia. O Projeto Honeynet, Conheça o seu Inimigo. Editora Makron Books, São Paulo, 2002.

SCHNEIER, Bruce. Segurança.com: segredos e mentiras sobre a proteção na vida digital. Rio de Janeiro: Campus, 2001.

SOLOMON, D. A. *THE WORLD OF BOTNETS*. Gadi Evron, 2002.

Smoke Detector, Disponível em <http://palisadesys.com/products/smokedetector>, Acesso em 08/03/2009.

SNORT, Disponível em <http://www.snort.org/>, Acesso em 15/03/2009.

SPECTER, *Instrusion Detector System*. Disponível em <http://www.specter.com/default50.htm>, Acesso em 06/03/2009.

SPITZNER, L. and Ranum, M. *Honeypots: Tracking Hackers*. In *SANS 2002 Annual Conference*, Orlando, Florida, USA, 2002.

STEIN, L. & STEWART, J. Securing Against Denial of Service Attacks. Disponível em <http://www.w3.org/Security/Faq/wwwsf6.html>, Acesso em 16/03/2009.

STOLL. Cliff. The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage, USA: Pocket Books, 2000.

SYMANTEC, 2009. Disponível em <http://www.symantec.com>, Acesso em 12/04/2009.

TANENBAUM, Andrew S. Redes de Computadores. 3. ed. Tradução Insight Serviços de Informática. Revisão Técnica: Nelson L. S.Fonseca. Rio de Janeiro: Campus, 1997.

Tiny Honeypot, Resource Consumption For the Good Guys, Disponível em <http://www.alpinista.org/thp/>, Acesso em 09/04/2009.

THE HONEYNET PROJECT. Know your Enemy: HONEYNETS.Disponível em <http://project.honeynet.org/>, Acesso em 05/03/2009.

The Honeynet Project., "Virtual Honeypots", Disponível em <http://niels.xtdnet.nl/papers/honeynet/>,. Acesso em 08/02/2009.

VAZ, Tiago B., Tássia C., Gorgonio A. (2004). Sistemas de Detecção de Intrusão Livres: suas limitações e uma arquitetura proposta sobre concentração de mensagens e correlacionamento de eventos, Disponível em <http://www.uefs.br/erbase2004/documentos/wticgbase/Wticgbase2004ArtigoIC005.pdf>, Acesso em 23/03/2009.

VERISSIMO, Paulo e Rodrigues, Luís. Distributed Systems For System Architects. © 2001 by Kluwer Academic Publishers and Paulo Veríssimo and Luís Rodrigues. Disponível em <http://www.navigators.di.fc.ul.pt/dssa/>, Acesso em 10/04/2009.

WILKINSON, M. GCIA *Practical for SANS Darling Harbour*, Disponível em mms://streamer.cbtnuggets.com/freevideos/security/ids.wmv, Acessado em 10/04/2009.