UNIVERSIDADE FEDERAL DE ALFENAS
INSTITUTO DE CIÊNCIAS EXATAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Lucas Tardioli Silveira

DETECÇÃO DE INTRUSÃO ATRAVÉS DE CONFIGURAÇÃO DE HONEYPOT DE BAIXA INTERATIVIDADE

Alfenas, 27 de Junho de 2011.

UNIVERSIDADE FEDERAL DE ALFENAS
INSTITUTO DE CIÊNCIAS EXATAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

DETECÇÃO DE INTRUSÃO ATRAVÉS DE CONFIGURAÇÃO DE HONEYPOT DE BAIXA INTERATIVIDADE

Lucas Tardioli Silveira

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Alfenas como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação.

Orientador: Prof. Flávio Barbieri Gonzaga

Alfenas, 27 de Junho de 2011.

Lucas Tardioli Silveira

DETECÇÃO DE INTRUSÃO ATRAVÉS DE CONFIGURAÇÃO DE HONEYPOT DE BAIXA INTERATIVIDADE

A Banca examinadora abaixo-assinada aprova a monografia apresentada como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação pela Universidade Federal de Alfenas.

Luiz Eduardo da Silva

Universidade Federal de Alfenas

Prof. Tomás Dias Sant'Ana

Universidade Federal de Alfenas

Prof. Flávio Barbieri Gonzaga (Orientador)

Universidade Federal de Alfenas

Alfenas, 27 de Junho de 2011.

À minha família, aos meus amigos e a todos que fazem parte da minha vida.

AGRADECIMENTO

Agradeço minha família por ter me apoiado nessa etapa da minha vida. Agradeço meus pais, Geraldo e Claudia, por sempre me apoiarem não importando a decisão que eu tomasse. Agradeço meu avô e minha avó, João e Solange, pela hospedagem que me deram no começo do curso. Agradeço meus irmãos Leonardo, Natália e Cecília por todas as vezes que eu voltei pra casa e me receberam com carinho.

Agradeço os "manolos" da Confraria dos Manolos (Porko, Didi, Tuti, Micka e Gustavo) por todas as vezes que reunimos, degustamos e harmonizamos diversos tipos de cervejas com diversos pratos diferentes.

Agradeço meus sogros Eliz e Muniz, que sempre arrumavam alguma forma de me "mimar".

Agradeço os meus colegas da faculdade, pelas conversas e experiências trocadas, que com certeza contribuiu muito para formação de todos.

Agradeço todos os professores pelo conhecimento passado, seja através das aulas ou através dos trabalhos infernais. Agradeço também meu orientador, Flávio, por ter me mostrado o caminho que deveria tomar para realizar esse trabalho.

Agradeço a minha namorada, Mickaela, que esteve sempre me ajudando a seguir em frente quando as coisas não estavam indo bem e sempre esteve por perto nos bons e nos maus momentos e que se não fosse por ela, todos esses quatro anos seriam bem piores.

Por fim agradeço a todos que contribuíram direta ou indiretamente para que esse trabalho fosse realizado.

"Se a gravidade é a força primordial que deu início ao Universo e que o mantém funcionando no mais perfeito e harmonioso caos, então chamo de Deus essa gravidade."

Lucas Tardioli Silveira (2011)

RESUMO

Devido ao crescente aumento da utilização das redes de computadores, as informações estão cada vez mais disponíveis na rede e em conseqüência disso cresce também técnicas para a invasão das redes. Para se evitar que invasões aconteçam é necessário estudá-las e entendê-las e para isso pode-se utilizar de um mecanismo denominado *honeypot,* que oferece um ambiente dedicado a ser sondado e invadido. Nesse ambiente, todas as ações do invasor são coletadas, sendo possível então ao administrador da rede a realização de um estudo e assim se preparar para os próximos ataques em servidores reais. Nesse trabalho foi desenvolvido inicialmente um *honeypot* no LaReS (Laboratório de Redes de computadores e Sistemas distribuídos), dando a impressão a quem rastreasse os servidores do BCC (Bacharelado Ciência da Computação) que os mesmos possuíam vulnerabilidades. Após esse passo, foi configurado uma *honeynet*, ou seja, uma rede local toda emulada, que pode atrair usuários internos que estejam mal intencionados. Uma análise das configurações e resultados obtidos são discutidos no decorrer do presente trabalho.

Palavras-Chave: *Honeypot*, *Honeynet*, Segurança, Invasão, Honeyd.

ABSTRACT

Due to the increasing use of computer networks, the information is increasingly available on the network and as consequence of that, increase techniques for the invasion of these networks. To prevent intrusions to happen it is necessary to study them and understand them. For that you can use a approach called honeypot that provides an environment dedicated to being probed, invaded and exploited. In this environment, all actions of the attacker is collected, become possible a study by the network administrator and so, prepare for the next attacks on real servers. In this work was initially developed a honeypot at LaReS(Laboratory of Computer Networks and Distributed Systems), giving the impression to those who crawl the server of the BCC(Computer Science Bachelor Degree) that it had vulnerabilities. After this step, was set up a honeynet, ie an entire emulated LAN, which can attract internal users who are malicious. An analysis of the settings and results are discussed in the course of this work.

Keywords: *Honeypot, Honeynet, Security, Invasion, Honeyd.*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIAÇÕES

ARP	*Access Resolution Protocol*
BCC	*Bacharelado em Ciência da Computação*
DHCP	*Dynamic Host Control Protocol*
DoS	*Denial of Service*
FTP	*File Transfer Protocol*
HIDS	*Host Intrusion Detection System*
HTML	*Hyper Text Markup Language*
HTTP	*Hyper Text Transfer Protocol*
IDS	*Intrusion Detection System*
IP	*Internet Protocol*
LaReS	*Laboratório de Redes de Computadores e Sistemas Distribuídos*
MAC	*Media Access Control*
NFS	*Network File System*
NIDS	*Network Intrusion Detection System*
NIS	*Network Information Service*
POP	*Post Office Protocol*
RFC	*Request For Comments*
RPC	*Remote Procedure Call*
MSRPC	*Microsof Remote Procedure Call*
SMTP	*Simple Mail Transfer Protocol*
SSL	*Secure Sockets Layer*
TCP	*Transfer Control Protocol*
UDP	*User Datagram Protocol*
UML	*User Mode Linux*

SUMÁRIO

1
Introdução

Com o advento da globalização, a tecnologia da informação é cada vez mais abrangente em nosso cotidiano. Um crescente número empresas utiliza soluções informatizadas para gerenciar e armazenar suas informações para que seus colaboradores usufruam dessas informações.

Nesse contexto, a informação é um recurso vital em todas as instituições, tendo influência em muitos aspectos do negócio e da própria sobrevivência da organização Albernaz (2001) apud Rosamann (2002, pág.3).

Para se assegurar que não ocorra o extravio dessas informações, pode-se utilizar mecanismos conhecidos na literatura que possuem um bom desempenho em relação à proteção dos sistemas.

Dentre as alternativas, o *firewall,* descrito em Neto (2004), é um programa que detém autonomia concedida pelo próprio sistema para determinar e disciplinar todo tipo de tráfego existente entre ele e outros *hosts*/redes. Um *firewall* é uma boa solução para barrar qualquer tráfego malicioso que possa vir a prejudicar o sistema e isolá-lo da rede externa.

Outra possibilidade para o auxílio da segurança, e também para os estudos dos ataques são os *honeypots;* Spitzner (2002) descreve um *honeypot* como sendo recursos computacionais dedicados a serem sondados, atacados ou comprometidos, num ambiente que permita o registro e controle dessas atividades. Os *honeypots* funcionam como armadilhas para os atacantes simulando um ambiente que não possui informações importantes, desviando a atenção para si próprio e assim protegendo os sistemas reais.

Observa-se então que a informação é um ativo importante das organizações e que sua segurança é essencial tanto para o retorno dos investimentos quanto para a continuidade dos negócios. Tendo isso em vista, mecanismos de proteção e estudo das atividades maliciosas devem ser tratados de forma criteriosa e madura

para que as pessoas possam utilizar suas informações de forma cada vez mais segura.

1.1 Justificativa e Motivação

A realização deste projeto é justificada pelo fato de que hoje em dia a informação é considerada o bem mais precioso dentro de uma instituição e que o extravio destas pode ocasionar graves transtornos podendo até haver algum prejuízo considerável e assim prejudicar a reputação de uma grande instituição.

A segurança da informação é fundamental para estabilidade e segurança da sociedade, pois, ao vivermos na era da informação, onde a dependência da tecnologia aumenta a cada dia, os riscos ligados às perdas de requisitos da segurança da informação como a confiabilidade, integridade e disponibilidade aumentam exponencialmente Alvarez (2010).

De acordo com Marciano e Lima-Marques (2006) com a proliferação da Internet e de redes corporativas, ao mesmo tempo em que introduz formas de fácil e rápida utilização dos recursos computacionais, expõe ainda mais a fragilidade e os riscos a que estão expostos os usuários e os sistemas.

Por esse motivo, o estudo de novas formas para prevenir e estudar esse tipo de ameaça vem sendo amplamente abordado com intuito de amenizar os males que crescem juntamente com o advento da tecnologia da informação.

Implantando um *honeypot* em uma organização, pode-se então simular um ambiente vulnerável atraindo a atenção dos invasores e com isso proteger os sistemas importantes.

1.2 Problematização

Devido à qualidade dos *honeypots* de monitorar a armazenar informações sobre as invasões que ocorrem, pode-se utilizar isso como um meio de levantamento de dados para então estudar as características do ataque e assim se preparar melhor futuramente. Observando-se isso, pergunta-se sobre a viabilidade de se implantar um *honeypot* de baixa interatividade na rede BCC Unifal-MG como um mecanismo de detecção de invasões e estudo de ataques. Mais do que oferecer mais segurança à rede do BCC, o presente trabalho possibilita também a coleta de dados que podem ser usados futuramente em outros trabalhos relacionados ao tema de segurança de redes.

1.3 Objetivos

1.3.1 Gerais

A configuração de um serviço de *honeypot* de baixa interatividade é o objetivo do presente trabalho (as diferentes classificações sobre *honeypots* são destacadas no Capítulo 2). As etapas de configuração e execução da mesma podem depois ser aproveitadas para auxiliar os mecanismos de detecção de invasão e estudar as formas utilizadas nos ataques, de modo a aprimorar a configuração de servidores.

1.3.2 Específicos

Este projeto tem como objetivos específicos os seguintes itens:

- Estudar e entender o funcionamento dos *honeypots*;
- Pesquisar ferramentas para implementação de *honeypots* existentes;
- Avaliar qual a mais condizente para solucionar o problema apresentado;

- Estudar configurações de servidores;
- Configurar uma pequena *honeynet* como adicional ao trabalho proposto.

2
Estudos Preliminares

2.1 Considerações Iniciais

Na literatura sobre *honeypots* são discutidas diversas formas sobre como eles são implantados em uma rede e suas características tais como: baixa e alta interatividade, real e virtual juntamente com as vantagens e desvantagens de cada tipo.

Nas próximas sessões será apresentada uma revisão destes termos citados, implementações já realizadas, um histórico sobre a segurança de redes e um pequeno informativo sobre sistemas operacionais, servidores e protocolos de redes que serão necessários para o entendimento deste trabalho.

2.2 Conceitos de Redes de Computadores

Será abordada nessa sessão uma breve revisão sobre algumas características básicas das redes de computadores. O entendimento desses conceitos são importantes para a explicação sobre o uso e configuração do *honeypot* apresentado no Capítulo 3.

2.2.1 Pilha TCP / IP

De acordo com Ulbrich e Della Valle (2009, pág.55) o conjunto de protocolos TCP/IP foi criado para propósitos acadêmicos e militares e é atualmente o padrão de fato.

Esse conjunto forma uma pilha de protocolos que é dividida em quatro camadas: camada de aplicação, camada de transporte, camada de Internet e camada de interface com a rede.

Cada uma das camadas citadas possuem internamente vários protocolos, que podem ser aplicados/substituídos de acordo com a característica da rede onde o mesmo será utilizado.

A camada de aplicação é responsável pela comunicação entre o protocolo destinado ao transporte e os aplicativos em execução, como DNS, FTP, HTTP entre outros. A camada de transporte cria a conexão virtual entre dois computadores que irão se comunicar. A camada de Internet é responsável pela organização e roteamento dos pacotes definindo seus endereços. E por fim, a camada de interface com a rede que é responsável pelo envio dos datagramas provenientes da camada de Internet.

Figura 1 - Pilha TCP / IP

A pilha TCP/IP possui diversos protocolos, cada um com suas funções específicas, mas os mais importantes pode-se dizer que são o TCP e o IP que dão o nome ao protocolo.

Dentre os inúmeros protocolos possíveis de serem utilizados no TCP/IP, serão detalhados a seguir os protocolos necessários para o entendimento do trabalho.

2.2.1.1 Protocolo IP

O protocolo IP (Internet *Protocol*), Protocolo de Internet, é encontrado na camada de Internet da pilha TCP/IP. É ele que torna possível a localização de computadores em uma rede como a Internet.

Atualmente é usada a versão 4 do protocolo IP, mas devido ao aumento dos equipamentos que utilizam essa protocolo estima-se que dentro de pouco tempo uma nova versão, o IPv6 substitua a versão atual. Como na construção desse trabalho o protocolo vigente ainda é o IPv4 então será abordada apenas essa versão.

O endereço IP possui 32 *bits* ou 4 *bytes,* eles são descritos na sua forma decimal utilizando um ponto para separar cada *byte* como mostra o exemplo a seguir: Considere o endereço 192.32.216.9, o número 192 é o número decimal equivalente aos oito primeiros *bits* do endereço; o número 32 é equivalente aos oito bits do segundo grupo de *bits* e assim por diante (Kurose e Ross, 2006, pág 260). O endereço exemplificado a cima também pode ser escrito em sua forma binária como: 11000000.00100000.11011000.00001001.

Cada interface de rede de um *host* deve ter um endereço IP associada a ela e esse endereço deve ser único em sua rede, pois é através dele que os outros *hosts* saberão onde encontrá-lo para assim se comunicar.

Na Figura 2 podemos ver um roteador com três interfaces, duas ligadas à sub-redes e uma ligada na Internet. A sub-rede da esquerda, ligado ao roteador através do IP 192.168.0.1 pode ser endereçada usando a notação /24, também conhecida como máscara de rede da seguinte forma 192.168.0.0/24, onde os 24 *bits* mais a esquerda são utilizados para identificar a sub-rede e os 8 últimos *bits* são utilizados para endereçar os *hosts* dentro dessa sub-rede. Ou seja, todo *host* que se conectar a essa sub-rede terá o seguinte endereço IP: 192.168.0.X (Kurose e Ross, 2006, pág. 361).

Na outra sub-rede segue o mesmo critério, só que agora seu endereço é 192.168.1.0/24, ou seja, todo *host* que fizer parte desta sub-rede terá os 3 primeiros dígitos do IP definidos por 192.168.1 e o último um número deverá ser um decimal de 8 *bits* que ainda não foi utilizado por algum outro *host*.

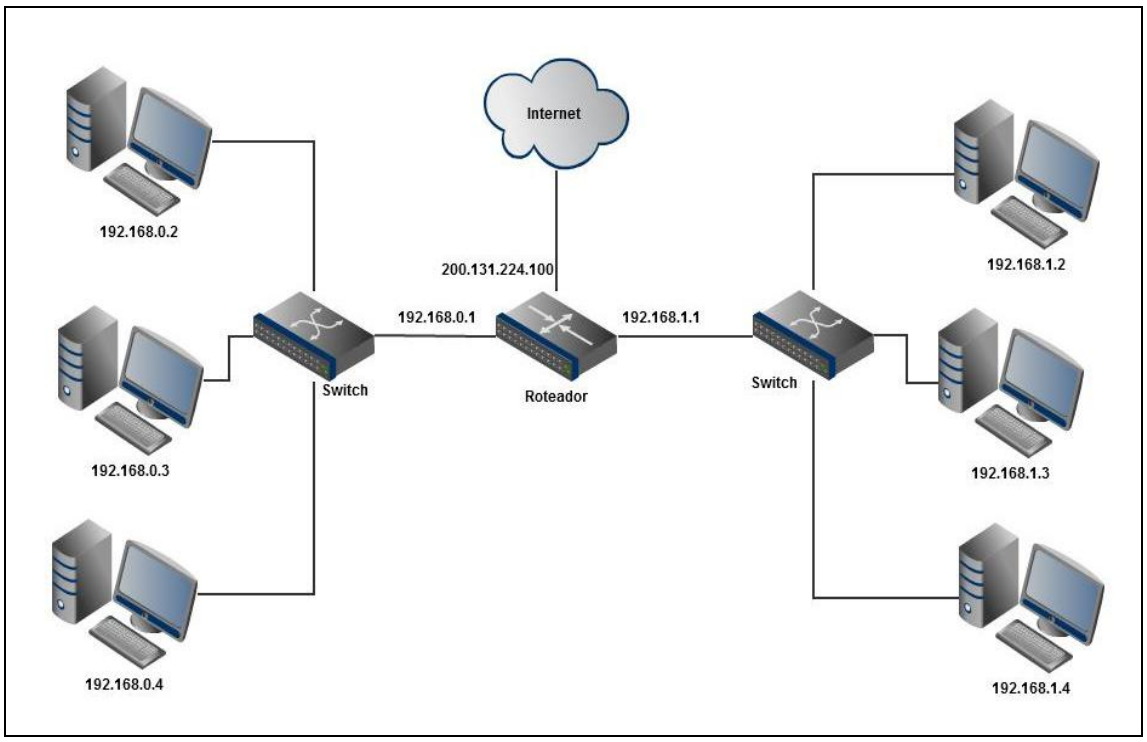

Figura 2 - Exemplo de sub-rede

Para este trabalho foi criado um *honeypot* e uma *honeynet* dentro da sub-rede do BCC. A topologia básica dessa rede é definida pela Figura 3.

A Figura 3 mostra uma conexão de uma rede externa nesse caso a Internet com do servidor do BCC, e ela sendo conectada nas suas sub-redes através de um *switch*. Em uma das sub-redes estão o *honeypot* e a *honeynet* que foram desenvolvidos nesse trabalho. Mais detalhes sobre essa topologia e implementação do *honeypot* e *honeynet* serão fornecidos no Capítulo 3.

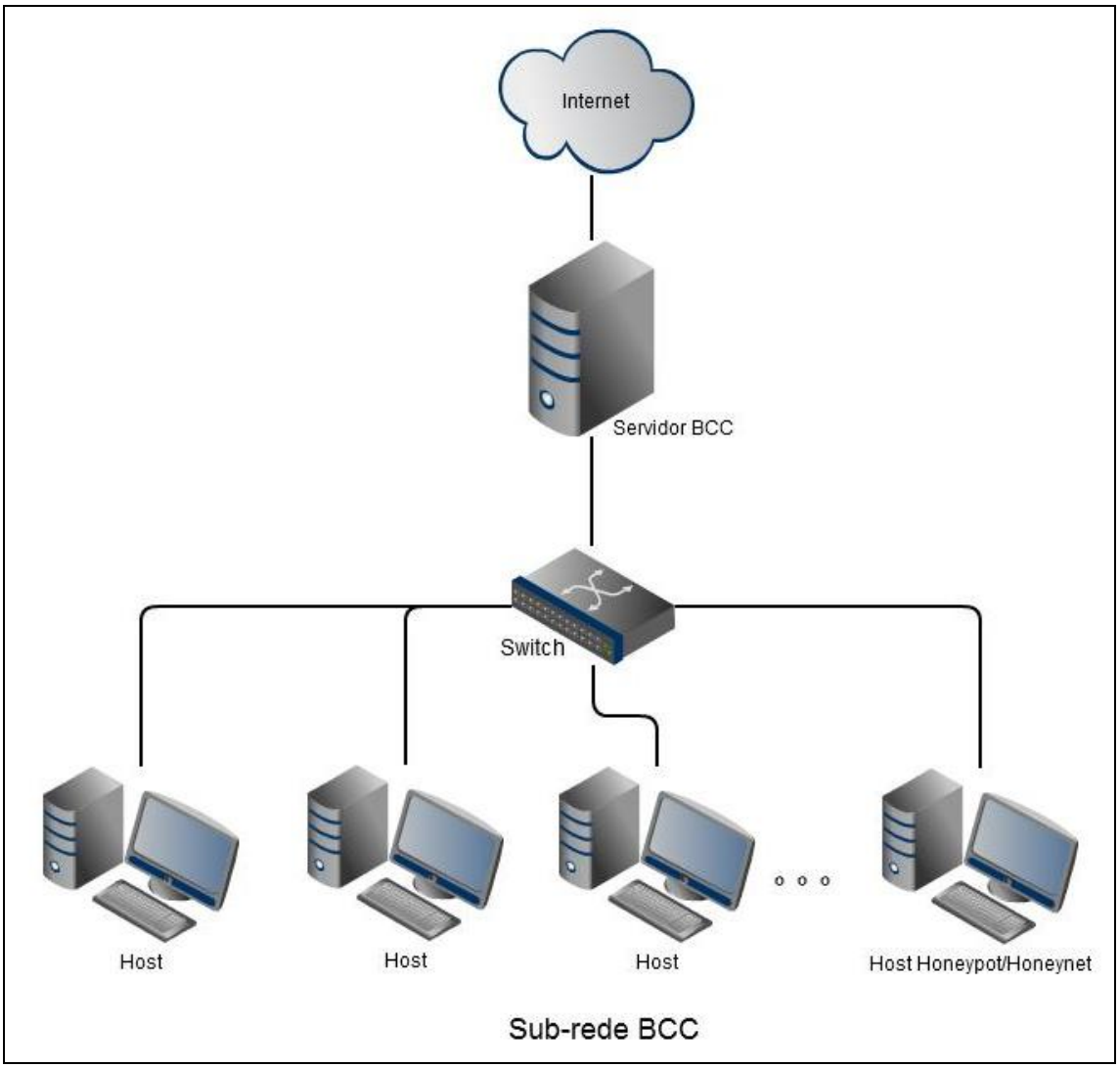

Figura 3 - Topologia básica BCC

2.2.1.2 Protocolo TCP e UDP

Como vimos na sessão passada, o protocolo IP provê uma comunicação entre dois *hosts* permitindo encontrá-los através de seu endereçamento. Entretanto, o protocolo IP é chamado de serviço não confiável devido ao fato de ser um protocolo que não da garantias de entrega dos dados. Para isso foram criados os protocolos da camada de transporte que amplia o serviço de entrega do protocolo IP (Kurose e Ross, 2006, pág. 146).

O protocolo TCP (*Trasmition Control Protocol*) – Protocolo de Controle de Transmissão é um protocolo que opera na camada de transporte e é usado para gerenciar a transferência de dados em uma rede. O TCP é um protocolo orientado a

conexão, o que significa que algum *host* que queira se comunicar com outro em uma rede ele terá que abrir um canal virtual de comunicação para que a troca de dados seja realizada.

Isso acontece através do *three-way handshake*, ou aperto de mão triplo, onde um cliente pede uma conexão ao servidor, o servidor por sua vez responde aceitando ou não e o cliente novamente envia uma mensagem confirmando a conexão.

Segundo Kurose e Ross (2006) para ampliar as funcionalidades do IP, o TCP possui métodos que garantem a transferência confiável de dados tais como: controle de fluxo, números de seqüência, reconhecimentos e temporizadores. Assim, ele assegura que os dados sejam entregues do processo remetente ao processo destinatário corretamente, em ordem. Mas para ampliar as funcionalidades de endereçamento existe o conceito de portas TCP.

Quando dois *hosts* precisam trocar informações em uma rede, é possível saber quem é quem através de seu endereço IP, mas em um único *host* podem estar rodando diversos serviços de rede que utilizam o mesmo IP, então a forma do protocolo TCP saber para quem entregar os pacotes é utilizando uma porta associada a algum serviço. Por exemplo, pode-se usar o navegador *web* para acessar sites da Internet e ao mesmo tempo baixar algum arquivo via FTP.

Quando a requisição chega ao servidor *web* ela deve ser encaminha para o serviço que cuida das requisições de páginas, geralmente esse serviço roda na porta 80. Então com o endereço do servidor mais o número da porta que roda o serviço é possível fazer essa requisição. Quando o servidor for responder, ele irá responder ao número de IP mais a porta que está associada ao navegador *web* do requisitante, enquanto outra porta relacionada ao seu cliente FTP recebe os dados de outro servidor. Dessa forma, com um mesmo endereço IP é possível rodar diversos serviços de rede sem que haja preocupações sobre quem irá receber os pacotes, pois o protocolo TCP cuida disso utilizando o conceito de portas.

O UDP (*User Datadram Protocol*) – É outro protocolo da camada de transporte junto ao TCP. O UDP é considerado um protocolo mais simples que o TCP, pois ele não faz todas as verificações de integridade, controle de congestionamento e entre outras que o TCP faz. O UDP não é um protocolo

orientado a conexão, isso significa que não ocorre o *three-way handshake* como ocorre com o TCP, ele simplesmente envia os dados sem nenhuma comunicação preliminar.

Por ser um protocolo mais simples e mais rápido que o TCP, o UDP é amplamente utilizado em aplicações onde é preferível a perda de pacotes do que todo o trabalho para a sua retransmissão, como por exemplo, aplicações de envio de vídeos pela Internet (*streaming)*, onde é melhor perder um pacote e perder a qualidade do vídeo e ainda sim continuar a transmissão, do que perder tempo reenviando o pacote e travando a transmissão por um longo tempo.

Para descobrir quais serviços devem ser receber os pacotes, o UDP também utiliza o conceito de portas para fazer a comunicação entre dois *hosts*.

De acordo com Kurose e Ross (2006), são 65536 portas TCP e UDP, começando em 1. As portas de 1 a 1024 são chamadas de portas baixas e são reservadas para aplicações já padronizadas, as portas acima de 1024 são chamadas de portas altas e são livres para serem utilizadas por qualquer aplicação. No site http://www.iana.org/assignments/port-numbers é possível encontrar uma lista com as portas e os serviços relacionados a elas.

2.2.2 MAC (*Media Access Control*)

O endereço MAC é encontrado na camada de interface com a rede, é o endereço físico do adaptador de um *host* (roteadores, *hubs* e etc...).

O endereço do MAC possui seis bytes de comprimento, o que possibilita 2^{48} endereços diferentes, tipicamente escritos em hexadecimal, com cada *byte* do endereço expresso com um par de números hexadecimal (Kurose e Ross, 2006, pág. 349).

Uma característica importante sobre MAC é que cada adaptador possui um endereço diferente, inserido na sua memória ROM pelo próprio fabricante. Os três primeiros *bytes* são utilizados para determinar o fabricante, os 3 *bytes* restantes é a faixa de endereços que o fabricante pode distribuir.

Conforme ilustrado na Figura 1, o protocolo IP explicado anteriormente, é o utilizado para endereçamento das máquinas em nível de camada de Internet,

enquanto que o endereço MAC é o utilizado em uma rede local (rede de pequena escala Tanenbaum (2008)) para endereçamento na camada de acesso à rede.

2.2.3 ARP (*Adress Resolution Protocol*)

Pelo fato de existirem protocolos na camada de Internet, como o IP, e protocolos na camada de interface com a rede como o MAC, se torna necessário um meio de conversão desses dois endereços. Quem é responsável por essa tradução é o protocolo ARP.

Quando um pacote chega a uma rede com seu IP de destino, deve-se descobrir qual *host* o pacote deverá se dirigir. Então o protocolo ARP age para descobrir qual é o MAC do *host* dono do IP que está endereçado no pacote. Para isso, o protocolo ARP pergunta a todos os *hosts* da rede quem possui tal IP, a interface possuidora desse IP responde ao protocolo ARP o seu endereço de MAC e o pacote é encaminhado a ela.

2.2.4 Servidores

Devido ao fato de que neste trabalho foram emulados comportamentos de alguns servidores nos *honeypots* e que alguns servidores foram utilizados para a configuração da topologia dos *honeypots* e *honeynets*, será realizada nessa sessão uma pequena explicação do funcionamento desses servidores.

2.2.4.1 Servidor DHCP (*Dynamic Host Control Protocol*)

Para que um *host* se conecte a rede é necessário atribuir a ele um endereço de IP. Essa tarefa pode ser realizada de duas formas: a primeira e mais trabalhosa, é o administrador da rede configurar manualmente o IP do hospedeiro, a segunda é através do protocolo DCHP.

Um servidor DHCP depois de configurado, possibilita aos hospedeiros obter um endereço IP automaticamente, bem como informações adicionais, como, máscara de sub-rede, endereço do *gateway* e endereço do servidor DNS local (Kurose e Ross, 2006, pág.266).

O DHCP funciona como uma espécie de reservatório de endereços IP, atribuindo temporariamente um endereço IP para um hospedeiro que acaba de se conectar e devolvendo o IP ao reservatório assim que o hospedeiro se desconectar.

O DHCP também pode ser configurado para armazenar qual endereço IP atribuir para determinada interface com determinado MAC e assim, devolver o mesmo endereço de IP sempre que o hospedeiro com esta interface se conectar a rede.

Esta característica se fez necessária neste trabalho devido ao fato de que nos *honeypots* que emulam servidores é importante que seus IP's sejam fixos para se obter um maior realismo. Sem essa característica, toda vez que o *honeypot* fosse ligado o servidor DHCP iria atribuir qualquer IP aleatório em seu repositório e cada vez que o *host* do *honeypot* recebesse um novo IP o *script* no servidor que faz o encaminhamento de portas precisaria ser modificado. Mais detalhes sobre o encaminhamento de portas são apresentados na Sessão 3.4.

2.2.4.2 Servidor HTTP

Um dos servidores emulados pelo *honeypot* desenvolvido nesse trabalho é o servidor HTTP (*Hyper Text Transfer Protocol*). O servidor HTTP é responsável por disponibilizar páginas, fotos, ou qualquer outro tipo de objeto ao navegador *web* do cliente. Ele também pode operar recebendo dados do cliente, processando e enviando o resultado para que o cliente possa tomar a ação desejada (como por exemplo em aplicações CGI's, banco de dados *web* e preenchimento de formulários) (da Silva, 2007, pág.197).

Os pedidos ao servidor HTTP se referem habitualmente a páginas HTML e são normalmente utilizados através de navegadores, como Internet Explorer, Mozilla Firefox entre outros. Tudo começa com a conexão entre o computador onde está instalado o servidor HTTP e o computador do cliente que irá acessar o servidor. Como na Internet não é possível prever a que hora se dará essa conexão, os servidores HTTP precisam estar disponíveis dia e noite. A partir daí é processado o pedido do cliente, e conforme as restrições de segurança e a existência da informação solicitada, o servidor devolve os dados.

Podemos citar alguns servidores HTTP tais como o Apache (http://www.apache.org/), que é atualmente o servidor HTTP mais utilizado no

mundo segundo Netcraft appud (Silva, 2007, pág.197) e o IIS (*Internet Information Services*) da *Microsoft* (http://www.iis.net/).

2.2.4.3 Servidor POP3

Saber como funciona o protocolo de recebimento de e-mails é importante dentro de uma rede onde um servidor POP3 está instalado. Muitos invasores aproveitam das falhas de segurança deste servidor para ler e-mails de outras pessoas e assim roubar informações.

O nome POP3 deriva do inglês *Post Office Protocol – Version* 3, mais detalhes sobre ele podem ser encontrados na RFC 1939.

O POP3 é um servidor para o recebimento de mensagens de e-mails para o cliente. Programas como o Thunderbird e o Outlook contatam o servidor POP3 através da porta 110 (por padrão) e baixam as mensagens utilizando um conjunto de comandos de texto, derivados do Telnet.

Originalmente, o POP3 é um protocolo tão inseguro quanto o Telnet,uma vez que a comunicação é realizada através de texto plano, ou seja, se os pacotes forem interceptados todas as mensagens serão passíveis de serem roubadas; os servidores atuais já suportam encriptação via SSL (o mesmo sistema de encriptação usado para acessar páginas seguras, via HTTPs), o que garante um bom nível de segurança (Morimoto, 2006, pág. 141).

2.2.4.4 Servidor SMTP

O SMTP (*Simple Mail Transfer Protocol*) é um servidor que roda por padrão na porta 25 e é utilizado para o envio de e-mails tanto do seu cliente local para o servidor SMTP de uma empresa quanto de um servidor SMTP para outro.

O protocolo SMTP é apenas para envio, o que significa que ele não permite que um usuário descarregue as mensagens de um servidor. Para isso, é necessário um cliente de email com suporte ao protocolo POP3 ou IMAP, que é o caso da maioria dos clientes atuais. Mais informações sobre o SMTP pode ser encontrada na RFC 821.

2.2.4.5 Servidor FTP

O FTP (*File Transfer Protocol*) é um dos protocolos de transferência de arquivos mais antigos e mesmo assim é muito utilizado atualmente, muitos invasores buscam brechas nesse serviço para transferir arquivos maliciosos para servidores ou conseguir acesso a alguns arquivos dos servidores.

Segundo Morimoto (2006, pág.138) existem dois modo onde o FTP opera: Modo Ativo e Modo Passivo. No modo ativo o cliente conecta com o servidor usando uma porta alta aleatória, por exemplo, 1029, e endereça os pacotes para porta 21 do servidor. O servidor responde ao cliente na porta seguinte, no caso 1030 para enviar os dados. O problema desse modo é que o cliente não consegue receber os arquivos se estiver utilizando uma conexão compartilhada, pois o servido responderia a porta 1030 do *gateway* não alcançando o cliente.

No modo passivo o cliente contata o servidor na porta 21, o servidor responde ao cliente em que porta alta aleatória ele deve se conectar, o cliente então se conecta ao servidor na porta especificada e o servidor responde enviando os dados.

Praticamente todos os clientes de FTP atuais utilizam o modo passivo por padrão, mas isso pode ser modificado dentro da configuração. Alguns poucos servidores de FTP não podem ser acessados em modo passivo, pois para isso é necessário que o administrador faça uma configuração de *firewall* mais cuidadosa, mantendo aberto um conjunto de portas altas.

Em resumo, no modo ativo o servidor precisa ter aberto apenas a porta 21, mas em compensação o cliente precisa acessar a *web* diretamente e ter um conjunto de portas altas abertas no *firewall*. No modo passivo, os papéis se invertem: o cliente não precisa ter portas abertas, mas o servidor sim.

2.3 Métodos de Segurança

Nesta sessão serão abordados alguns métodos que provêm à segurança paras as redes. Será apenas comentada a existência destes bem como algumas características superficiais, devido ao fato de que este trabalho possui um enfoque em *honeypots*.

2.3.1 *Firewalls*

De acordo com Neto (2004, pág.11) um *firewall* é um programa que detém autonomia concedida pelo próprio sistema para determinar e disciplinar todo tipo de tráfego existente entre ele e outros *hosts*/redes; salvo aonde o *firewall* é um componente denominado "*Firewall-in-a-box*", onde nesse caso, trata-se não tão somente de um software e sim de um agrupamento de componentes incluindo software e hardware, ambos projetados sob medida para compor soluções de controle perante o tráfego de um *host*/rede.

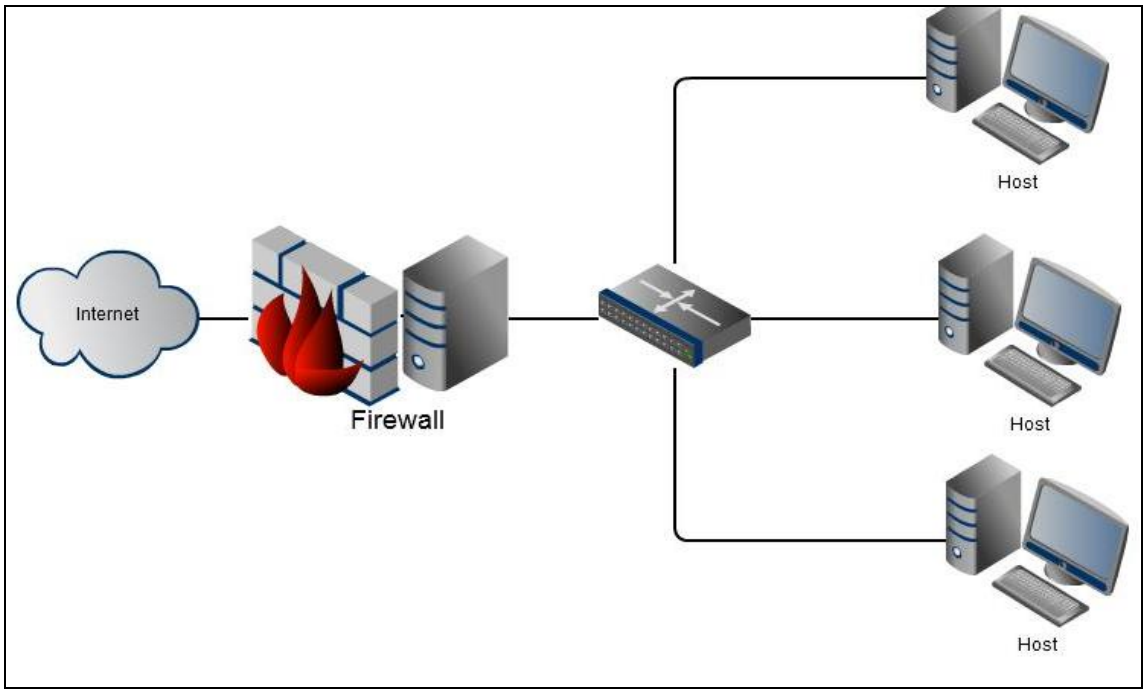

Figura 4 - Exemplo de um *firewall*

Os *firewalls* podem ser divididos em três classes: *firewall* filtro de pacotes, *firewall* NAT e *firewall* híbrido Neto (2004, pág. 12).

O *firewall* filtro de pacotes tem a tarefa de filtrar todos os pacotes direcionados a ele (*host firewall*) ou a rede em que este isola, ele também filtra todos os pacotes emitidos por ele e por sua rede mediante a consulta em um arquivo de configuração que possui regras previamente configuradas pelo administrador. Esta classe de *firewall* trabalha analisando os cabeçalhos (*headers*) dos pacotes que

trafegam na rede e realizando a análise no arquivo de configuração decidindo o destino dos pacotes: trafegar livremente pela rede ou ser barrado.

O *firewall* NAT (*Network Address Translator*, RFC 1631) possui o objetivo de alterar as rotas dos pacotes que passam por ele aplicando a tradução dos endereços. Essa tradução funciona da seguinte maneira: Existe um *host* na rede local denominado *host* A. Entre o *host* A e a Internet existe um *firewall* NAT isolando-o do mundo externo. Toda vez que o *host* A faz uma requisição a algum domínio da Internet, o *firewall* altera o IP do pacote para o seu próprio endereço de IP e armazena em uma tabela que uma requisição a determinado domínio pelo *host* A foi feita. Quando o domínio responder, ele não estará respondendo para o *host* A, mas sim para o *firewall* que então repassará a resposta para o *host* A. Dessa forma os *hosts* da rede interna ficam isolados da rede externa sendo protegidos pelo *firewall*.

O *firewall* híbrido é a mescla dos dois *firewalls* comentados anteriormente. Ao invés de separar as funcionalidades dos dois em classes diferentes, foram agrupadas todas essas funcionalidades em um só modelo provendo uma tradução de endereços e filtragem de pacotes aumentando a segurança das redes.

Os *firewalls* filtro de pacotes são amplamente utilizados em diversas redes. Uma solução para Linux que implementa esse tipo de *firewall* é o IPTABLES.

De acordo com Silva (2007), o IPTABLES é um *firewall* em nível de pacotes que funciona baseado no endereço/porta de origem/destino do pacote, prioridade e etc. Ele compara as regras escritas pelo administrador para verificar se os pacotes podem ou não prosseguir. Ele também pode ser usado para modificar e monitorar o tráfego da rede, fazer NAT, redirecionamento de pacotes, marcação de pacotes, modificar a prioridade de pacotes que chegam/saem do sistema, dividir o trafego entre máquinas, criar proteção *anti-spoofing*, DoS (*Denial-of-Service*) e etc.

Na rede do BCC, onde esse trabalho foi realizado, um *firewall* IPTABLES foi configurado para proteger a rede. Algumas configurações realizadas no IPTABLES foram feitas para que os *honeypots* fossem criados de forma que quando um atacante sondasse os servidores do BCC não percebesse que na realidade se trata de um *honeypot* emulando alguns serviços. No Capítulo 3 serão melhor detalhadas as configurações realizadas.

2.3.2 IDS (*Intrusion Detection System*)

Os IDS's são mecanismo que ouvem o tráfego na rede ou no *host* de maneira furtiva, para localizar atividades anormais ou suspeitas e permitindo assim ter uma ação de prevenção sobre os riscos de intrusão.

Um IDS possui diversas características e pode ser classificado de diversas formas.

Segundo o documento elaborado no RAVEL – Laboratório de Redes de Alta Velocidade (2000) uma forma de classificar é de acordo com o momento do ataque, onde o IDS pode enviar um alerta antes de o ataque ocorrer, durante, ou depois.

Outra forma é com relação ao modo de análise utilizada pelo analisador de eventos, que podem ser:

- Análise das assinaturas, onde o IDS funciona de forma parecida com antivírus;
- Análise estatística onde o IDS constrói modelos estatísticos do ambiente se baseando em fatores tais como duração média de uma sessão de Telnet, por exemplo, qualquer desvio de um comportamento normal pode ser identificado como suspeito;
- Sistema adaptativo onde o IDS começa por generalizar regras de aprendizado para o ambiente que está inserido e então determinar o comportamento dos usuários com o sistema. Depois do período de aprendizado, o sistema pode reconhecer determinados padrões como sendo acessos normais ou ataques. Uma rede neural pode ser uma escolha natural para tais sistemas.

A divisão mais clássica encontrada na literatura sugerida pela ICSA (2000) é sobre o sistema em que o IDS está agindo.

- IDS baseado em rede (NIDS);
- IDS baseado em *host* (HIDS);
- Verificador de integridade de arquivos;

O NIDS possui usualmente dois componentes: Sensores e Estação de Gerenciamento. Os sensores são colocados em seguimentos de redes distintos que

se desejam monitorar. A estação de gerenciamento possui uma interface gráfica que recebe os alarmes disparados pelos sensores avisando os operadores.

Apesar dos sensores serem colocados em máquinas específicas eles possuem a capacidade de monitorar grande parte da rede, devido que suas interfaces de rede são colocadas em modo promíscuo permitindo assim receber pacotes de todos os IP's e não apenas os pacotes destinados ao IP da interface do sensor.

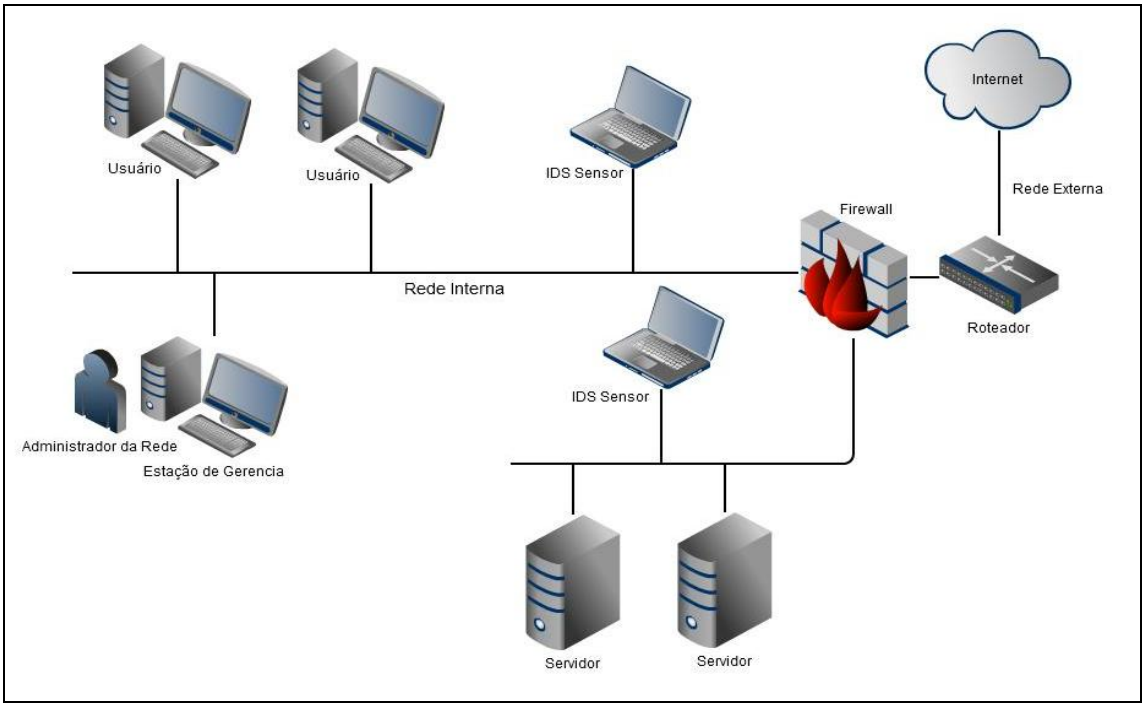

Figura 5 - Exemplo de um NIDS

O HIDS analisa os sinais de intrusão nos *hosts* nos quais estão instalados. Utilizam muito dos recursos dos *hosts*, verificando *logs* dos sistemas operacionais. Agem verificando comportamentos não usuais nas tarefas como tentativas de *login*, tentativas de acesso a arquivos e mudanças de privilégios.

Já os verificadores de integridades de arquivos analisam os arquivos procurando alguma alteração desde a última verificação. Eles utilizam funções *hash* para determinar se certo arquivo foi alterado e assim avisar o operador do IDS.

Os IDS's são ótimos mecanismos de detecção de intrusão, mas seu poder contra atividades maliciosas pode ser ainda maior se aliarmos sua capacidade de detecção junto com a capacidade de um *honeypot* de enganar os atacantes.

Como exemplo, pode-se criar um mecanismo de detecção utilizando IDS e quando este perceber uma anomalia na rede ele redireciona todo tráfego malicioso para um *honeypot* e assim proteger os reais sistemas.

Outra abordagem que pode ser realizada aliando-se *honeypots* com IDS é utilizando os *honeypots* para capturar os tráfegos maliciosos e a partir desses gerar assinaturas, e essas assinaturas podem ser enviadas à um IDS para auxiliá-lo na detecção de novas intrusões.

Mais informações sobre o funcionamento dos *honeypots* serão dadas na sessão 2.4 deste capítulo.

2.4 *Honeypots*

De acordo com Spitzner (2002) São recursos computacionais dedicados a serem sondados, atacados ou comprometidos, num ambiente que permita o registro e controle dessas atividades.

2.4.1 Histórico dos *Honeypots*

De acordo com HoneynetBR (2002) apud Duarte e Jabour (2004, pág. 2) o primeiro registro de implementação de um mecanismo para o acompanhamento de atividades maliciosas realizadas por invasores são de meados dos anos 80 quando Clifford Stoll tomou uma atitude peculiar ao ocorrer uma invasão no sistema da LBL (*Lawrence Berkeler Laboratory*), onde ao invés de fechar as portas e bloquear o atacante ele resolveu abrir e acompanhar o que ele estava fazendo. Este caso ocorreu durante um ano e com essa atitude conseguiram não só descobrir a localização do ataque, mas também o motivo e as redes que ele estava interessado.

Em 1997 Fred Cohen`s lança a primeira versão do DTK (*Deception Toolkit*) uma coleção de *scripts* em Perl e códigos em C aonde são emuladas diversas vulnerabilidades conhecidas para os sistemas Unix . Este foi primeiro *honeypot* livre disponível para a comunidade de segurança de redes (Spitzner, 2002).

Seguindo o DTK, em 1998 foi iniciado o desenvolvimento do *CyberCop Sting*, o primeiro *honeypot* comercial vendido ao público. Ele difere do DTK, pois roda em sistemas Windows NT e não em Unix, e introduziu um conceito de múltiplos sistemas virtuais rodando em apenas um *honeypot (*Spitzner, 2002).

Após esses projetos citados, diversos outros surgiram, mas em 1999 surgiu o Projeto *Honeynet*, como uma lista de discussão composta por um pequeno grupo de pessoas. O grupo percebeu que para analisar as informações sobre os ataques precisariam de mais pessoas e assim expandiu a lista que passou a contar com diversos especialistas da área. No ano 2000 o grupo mudou o nome para *Honeynet Project* e foi formalizado como uma organização não governamental. Ainda assim faltavam os recursos necessários para pesquisar e desenvolver múltiplas *honeynets*. Nesse momento foi formada a *Honeynet Research Alliance* que inclui mais de dez organizações no Brasil, Grécia, Índia, México, Irlanda e Estados Unidos (Duarte e Jabour 2004, pág.3).

2.4.2 *Honeypots* de Produção e Pesquisa

De acordo com Spitzner (2002), os *honeypots* podem ser divididos em dois grupos gerais, os *honeypots* de produção e os *honeypots* de pesquisa.

Spitzner (2002) diz que os *honeypots* de produção são utilizados nas organizações para mitigar o risco de segurança auxiliando outros mecanismos de segurança como *firewalls* e IDS. Eles são geralmente mais fáceis de serem implantados, pois necessitam de menos funcionalidades e isso acarreta em um menor risco. Entretanto, esse tipo de *honeypot* nos oferece menos informações sobre os ataques, pois podemos descobrir de onde os ataques vieram e quais métodos foram utilizados, mas não poderemos ter detalhes de como a ferramenta foi desenvolvida, ou como os atacantes se comunicaram e etc.

Já os *honeypots* de pesquisa são desenvolvidos para se obter informações sobre os atacantes. Não agrega valor diretamente a alguma organização. Seu principal objetivo é capturar todo tipo de informação possível proveniente de um ataque, como quem são os atacantes, como eles estão organizados, de onde ocorrem os ataques, que ferramentas são utilizadas e como são obtidas essas

ferramentas. Esse tipo de informação pode ajudar a entender melhor as ameaças para poder ter um melhor preparo para se proteger (Sptzner, 2002).

Como o nível de detalhamento desse tipo de *honeypot* é maior, pode-se perceber que o aumento da complexidade leva a um aumento dos riscos e recursos necessários necessitando de uma administração e de uma manutenção mais minuciosas.

Esta definição não precisa ser absoluta, pode ser considerada apenas como um guia para definir o propósito do *honeypot* que irá ser implantado. Um mesmo *honeypot* pode ser usado para produção e pesquisa, ele depende muito mais de como é usado do que de como é construído.

Um exemplo dado por Spitzner (2002) é sobre uma organização que possui um *honeypot* que sofreu uma invasão. Foi possível capturar e gravar cada atividade desenvolvida pelo atacante. Se uma organização usa como produção, é muito mais interessante detectar o ataque e bloqueá-lo, mas se uma organização usa como pesquisa é mais interessante descobrir as ferramentas utilizadas, de onde veio o ataque, e quais foram os interesses na invasão.

2.4.3 *Honeypots* de Baixa e Alta Interatividade

Os *honeypots* de baixa interatividade são na maioria das vezes mais fáceis de configurar, instalar e manter. São conhecidos por emular vários serviços em um único *honeypot*. Eles são desenvolvidos para oferecer respostas às requisições feitas pelos atacantes, mas nenhum serviço real está sendo acessado e nenhum sistema operacional poderá ser comprometido (em teoria).

Por serem mais simples eles oferecem menos informações, mas consequentemente possuem um menor risco. Geralmente o que se pode obter com os *honeypots* de baixa interatividade é o IP e a porta de onde vem ataque, a hora e a data do ataque, o IP e a porta de destino do ataque (Spitzner, 2002).

Esse tipo de *honeypot* é um ótimo recurso para se utilizar em proteção contra ataques conhecidos e ataques automatizados como *trojans* e *worms*.

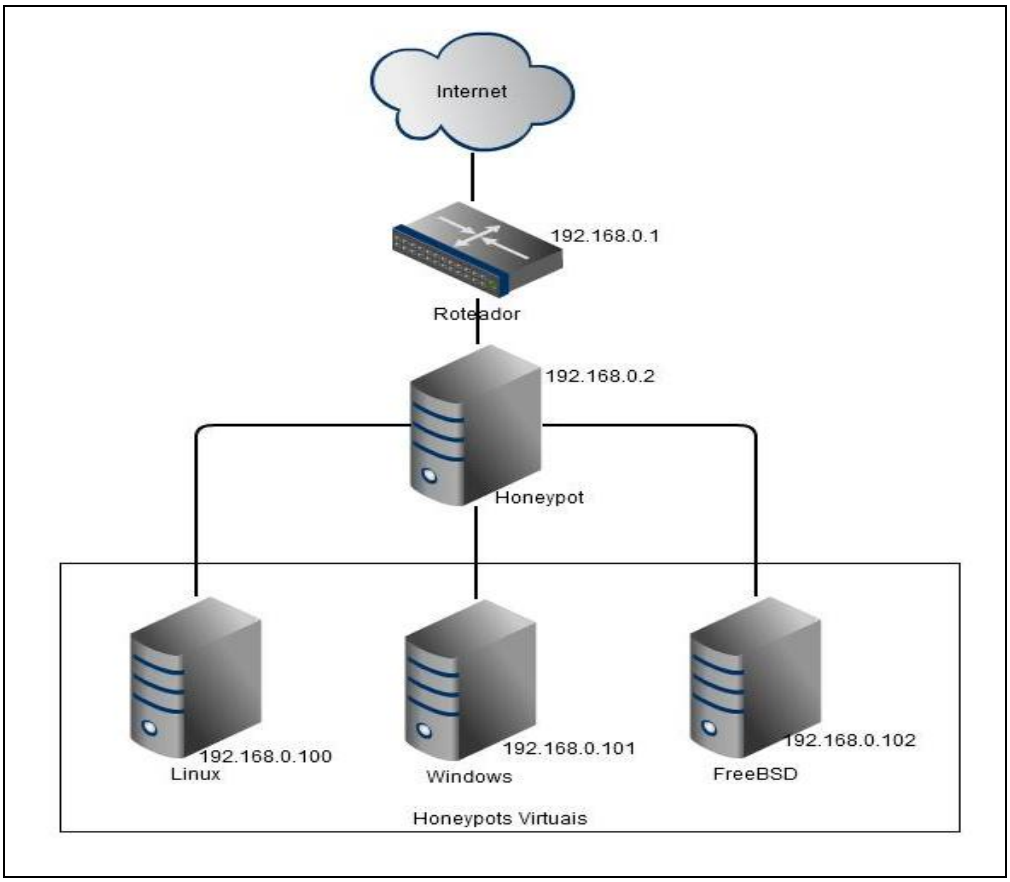

Figura 6 - Exemplo de *honeypot* de baixa interatividade

Já os *honeypots* de alta interatividade, oferecem todo o sistema para os atacantes interagirem. Isso significa que não existe nenhum serviço ou sistema operacional sendo emulado, tudo que é acessado pelo invasor é real (Provos, 2007).

Como o nível de liberdade que é dado ao atacante, aumenta-se muito o risco de se utilizar esse tipo de *honeypot*, mas por outro lado oferece uma vasta informação sobre o atacante possibilitando descobrir novas ferramentas, novas vulnerabilidades em sistemas operacionais, pode-se também descobrir mais sobre o comportamento dos adversários e estudar os seus comportamentos.

Sendo maior o risco, se o *honeypot* for comprometido, ele pode comprometer toda a rede que pode ser utilizada pelo atacante como base para novos ataques. Visto esse problema torna-se obrigatório configurar poderosos meios de contenção quando se utilizar esse tipo de *honeypot* para evitar que toda rede se torne uma rede "zumbi".

De acordo com Amorim (2007) redes zumbi são redes em que os atacantes utilizam vários computadores, infectados com *malwares* que fazem o papel de agente executando comandos de atacantes remotos. Com redes como essas, os atacantes podem programar ataques em massa em um determinado serviço utilizando uma rede zumbi com vários computadores para enviar várias mensagens ao mesmo tempo.

Neste trabalho foi escolhido um *honeypot* de baixa interatividade devido ao fato do local onde foi realizado os experimentos ser uma rede de uma instituição pública, onde os riscos de se utilizar um *honeypot* de alta interatividade não eram compensados por suas vantagens.

2.4.4 *Honeynets*

Segundo Honeynet Project (2004) uma *honeynet* nada mais é do que um tipo de *honeypot*. Podendo ser um *honeypot* de alta ou baixa interatividade, projetado para pesquisa e obtenção de informações dos invasores. É conhecido também como *honeypot* de pesquisa.

Uma *honeynet* normalmente contém um segmento de rede com *honeypots* de diversos sistemas operacionais e que fornecem diversas aplicações e serviços. Também contém mecanismos de contenção robustos, com múltiplos níveis de controle, além de sistemas para captura e coleta de dados, e para geração de alertas (CERT, 2010).

Quando comprometida a *honeynet* é utilizada para observar o comportamento dos atacantes e analisar ferramentas utilizadas a fim de realizar novas descobertas sobre vulnerabilidades e métodos de ataque.

De acordo com CERT (2010) as *honeynets* podem ser classificadas como *honeynets* reais e *honeynets* virtuais.

As *honeynets* reais utilizam recursos físicos reais para serem construídas, como um computador para cada *host*, um para o *firewall*, um para o IDS, um para repositório dos dados coletados e também *hubs* e *switches* caso seja necessário. Esse tipo de *honeynet* costuma ser caro, pois depende de muitos recursos físicos, tempo

para manutenção e mão-de-obra bastante qualificada, por outro lado oferece uma grande realidade para o atacante como nenhum outro tipo de *honeypot*.

As *honeynets* virtuais vem com a idéia de diminuir o numero de *hosts* físicos virtualizando os componentes em um único computador utilizando-se de softwares como VMware (*Virtual Infrastructure Software*) ou UML (*User Mode* Linux). As *honeynets* virtuais podem ser ainda divididas em mais dois tipos: Auto-Contenção e Híbridas. Na de auto-contenção todos os recursos são virtualizados incluindo o sistema de coleta de dados, mecanismo de contenção e geração de alertas. As híbridas virtualizam apenas os *honeypots*, os demais sistemas como o de contenção, geração de alerta e coleta de dados são dispositivos distintos.

Devido à dimensão desse projeto, seus propósitos e suas limitações, foi escolhida o modelo de *honeynet* virtual para ser implementada, onde foram utilizados diversos *honeypots* virtuais de baixa interatividade para compor a rede desenvolvida.

3Metodologia e Desenvolvimento

Nesse capítulo será explicado como foram construídos os *honeypots*, as *honeynets*. Também será apresentada a ferramenta que foi utilizada para construí-los e as configurações realizadas no *firewall* IPTABLES para que fosse possível montar a estrutura realizada nesse trabalho.

3.1 A Ferramenta Honeyd

Para o desenvolvimento do proposto nesse trabalho foi escolhida a ferramenta Honeyd desenvolvida por Provos (2007).

O Honeyd é um *framework* para o desenvolvimento de *honeypots* virtuais. Com o Honeyd é possível trabalhar com centenas de endereços IP, cada um respondendo por um sistema diferente e rodando diferentes serviços.

Algumas características do Honeyd são:

- **Emular centenas de *hosts* virtuais ao mesmo tempo:** O Honeyd possui a habilidade de emular centenas de *hosts* ao mesmo tempo. Um invasor pode interagir com cada um dos *hosts* via rede e encontrar comportamentos diferentes de cada *host* dependendo da forma que foram configurados.

- **Configuração de diferentes serviços via arquivo de configuração:** Através do arquivo de configuração, é possível configurar diversos programas para responder ao adversário quando uma nova conexão é estabelecida.

- **Emular sistemas operacionais no nível da pilha TCP/IP:** Essa característica permite o Honeyd enganar o Nmap e o Xprobe para que acreditem que é realmente um sistema real funcionando respondendo de acordo com o que foi configurado.

- **Emular diversas topologias de roteamento:** É possível configurar a latência de um roteador, a perda de pacotes e a largura de banda.

A única interação que o invasor tem com o Honeyd é via rede, o que significa que o invasor não pode ir até o computador e fazer o *login* através do teclado, pois não existe nenhum computador físico relacionado com os *honeypots* virtuais. Ao invés de simular todos os aspectos do sistema operacional, o Honeyd emula apenas a pilha de rede.

Devido ao fato do Honeyd ser um *honeypot* de baixa interatividade, espera-se que o invasor não obtenha o acesso total ao sistema mesmo que comprometa o serviço emulado.

Para ser capaz de enganar os atacantes, o Honeyd precisa enganar ferramentas de *fingerprint* como o Nmap. Para isso ele usa uma base de dados que são as mesmas utilizadas por essas ferramentas e sempre que um *honeypot* precisar enviar um pacote de rede ele procura em sua base de dados o *fingerprint* correspondente ao do sistema operacional que *honeypot* está configurado e assim simula o comportamento deste sistema.

A arquitetura do Honeyd é mostrada na Figura 7.

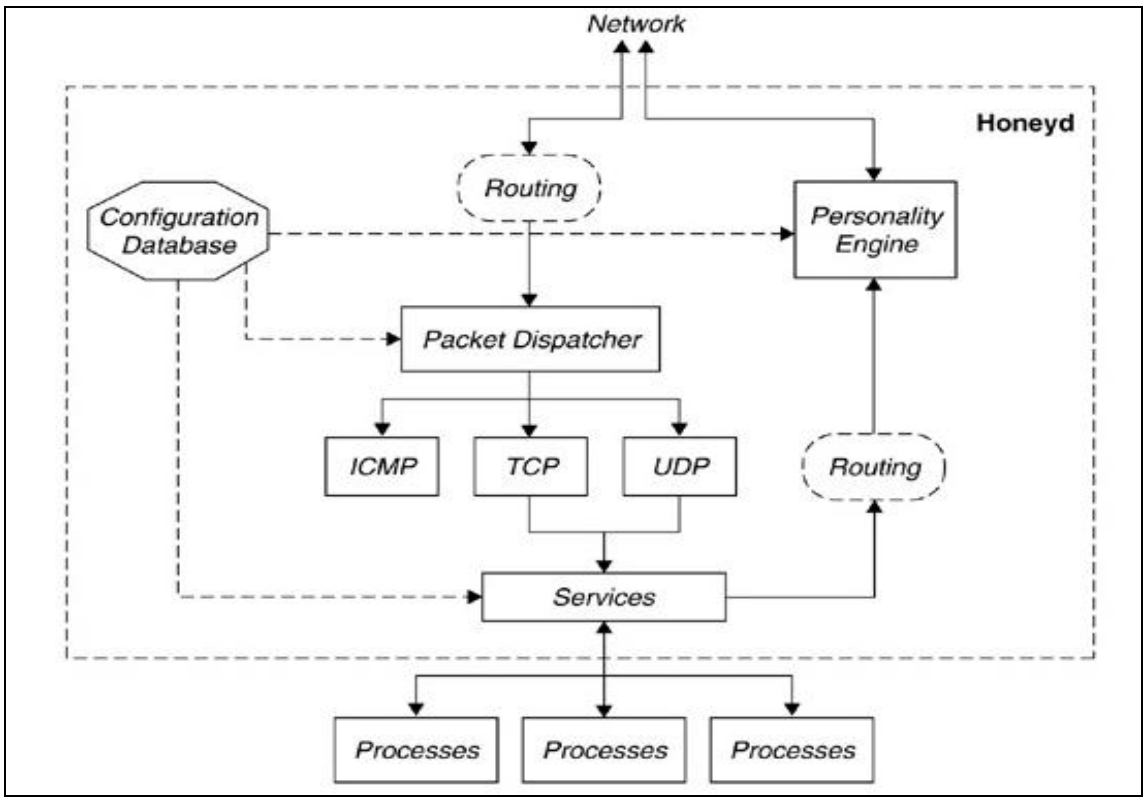

Figura 7 - Arquitetura Honeyd (Provos, 2007)

Todo tráfego é recebido pelo *Packet Dispatcher* (Despachante de pacotes), baseado em configurações específicas diferentes processos de serviços são criados para lidar com o tráfego. Cada pacote que é mandado de volta à rede é modificado pelo *Personality Engine* (Motor de personalidades) para corresponder as características do sistema operacional.

Para ser utilizado, um *honeypot* configurado através do Honeyd precisa ser encontrado dentro de uma rede. O Honeyd responde para a rede sobre todos os pacotes que pertencem aos seus *honeypots* virtuais, mas para fazer o Honeyd enxergar esses pacotes é preciso configurar a rede.

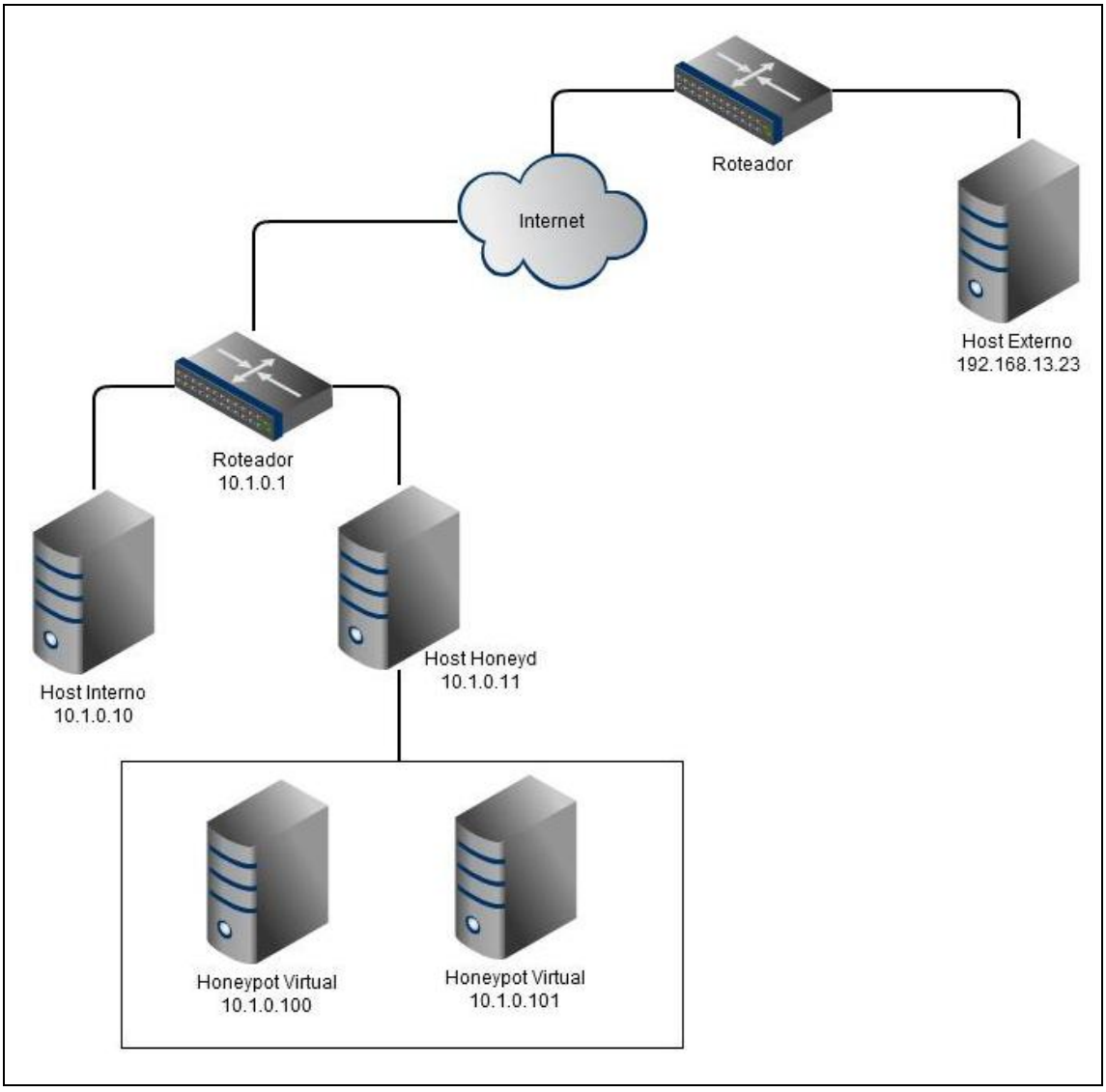

Figura 8 - Exemplo de uma rede com Honeyd

Para que os *honeypots* virtuais sejam alcançados é preciso que seja feito um mapeamento do endereço IP para o endereço de MAC, então um *host* que queira interagir com o *honeypot* deve fazer uma requisição ARP para realizar esse mapeamento.

Existem diversas formas de fazer essa configuração, como exemplo pode-se fazer um *Proxy* ARP para que o *host* do Honeyd seja responsável pelos IP's dos seus *hosts* virtuais. Ou então se pode fazer configurações no roteador e incluir os IP's dos *hosts* virtuais na tabela de encaminhamento.

Nas sessões que se seguem será mostrado como foram realizadas essas configurações nesse trabalho.

Para analisar posteriormente as informações sobre os ataques que possam vir a ocorrer ao *honeypot* emulado pelo Honeyd existem três tipos de *logs* que podem ser utilizados.

Um deles é o próprio *log* gerado pelo Honeyd que apresenta informações sobre as conexões que ocorreram, as horas que ocorreram e os IP's de origem e destino bem como as portas.

Outro é o *log* que armazena as informações sobre os *scripts* que são emulados no Honeyd onde se encontram informações sobre quando um *script* foi ativado devido uma conexão e mensagens de depuração para os desenvolvedores dos *scripts*.

E a última forma de *log* são os *logs* que os próprios *scripts* geram e armazenam diversas informações dependendo da criatividade e da habilidade de programação do desenvolvedor do *script*.

Devido à grande utilização do Honeyd para emular *honeypots* de baixa interatividade, foram desenvolvidas diversas ferramentas para aprimorar suas funcionalidades e com isso obter melhores resultados.

Uma dessas ferramentas é o Honeydsum desenvolvido pelo *Brazilian Honeynet Team* (HoneynetBR, 2011). Essa ferramenta varre o arquivo de *logs* criado pelo Honeyd e então gera uma página HTML com diversas estatísticas em forma de tabelas e gráficos.

No Capítulo 4 serão apresentados os resultados gerados por essa ferramenta.

3.2 Configuração do Honeyd

A ferramenta Honeyd é compatível com sistemas operacionais da família Unix. Também existem versões para Windows, mas não são estáveis para serem utilizadas. Neste trabalho foi utilizado o sistema operacional Linux Debian versão 5 (Lenny), onde a ferramenta Honeyd pode ser encontrada em seu repositório padrão.

Para instalar o Honeyd e todas as suas dependências foi utilizado o comando: *apt-get install honeyd*; O gerenciador de pacotes se encarrega de instalar as dependências necessárias.

Após a instalação podem ser encontradas duas pastas no sistema com os arquivos do Honeyd, uma localizada em /usr/share/honeyd e outra em /etc/honeypot. Em /usr/share/honeyd é onde se encontra a pasta chamada *scripts*, que contém diversos *scripts* de emulação de serviços prontos para serem utilizados. Na pasta /etc/honeypot, está o arquivo de configuração do Honeyd chamado honeyd.conf e as bases de dados nmap.prints, xprobe2.conf e nmap.assoc.

O arquivo de configuração é onde conseguimos definir como os *honeypots* funcionarão. Um exemplo de um arquivo de configuração básico (Código 1):

```
1. # Exemplo de um template simples
2. create template
3. set template personality "Microsoft Windows XP Professional SP1"
4. set template uptime 1728650
5. set template default tcp action reset
6. set template default udp action reset
7. set template ethernet “dell”
8. add template tcp port 80 "sh /usr/share/honeyd/scripts/win32/web.sh"
9. add template tcp port 22 proxy $ipsrc:22
10. bind 192.168.1.100 template
```

Código 1 – Exemplo de configuração do Honeyd

O comando “*create*” é utilizado para criar novos modelos de *honeypots*, nesse caso foi criado um modelo chamado “t*emplate*”.

O comando “*set*” é utilizado para mudar as características do *honeypot*, no exemplo a cima o comando “*set template personality "Microsoft Windows XP Professional SP1"”* diz para o Honeyd que o modelo chamado template emulará um Windows XP, para que na hora de chamar o motor de personalidades para enganar as ferramentas de *fingerprint* ele saiba qual sistema deve fingir ser, ou seja, é esse comando que determina o comportamento do *honeypot.* Para obter uma lista das personalidades que podem ser emuladas pode-se usar o comando Linux: *>Personalidades.txt grep “^Fingerprints” nmap.prints |less.* Assim será gerado um arquivo de texto com todas as possíveis personalidades que podem ser emuladas.

O comando “*set template ethernet “dell”* define que o *honeypot* terá uma interface de rede da Dell e um MAC com os três primeiros bytes representando a interface da Dell e os três últimos são gerados aleatoriamente.

O comando “*set*” também pode ser utilizado para mudar outras características. A linha “*set template uptime 1728650 ”*determina quanto tempo o sistema emulado deverá fingir que está ligado. As linhas “*set template default tcp*

action reset" e "*set template default tcp action reset"* diz qual a ação padrão o *honeypot* deverá tomar em uma conexão TCP ou UDP.

As ações permitidas no Honeyd são:

- *Open*: Todas as portas são abertas por padrão, o Honeyd irá agir como se todas as portas estivessem abertas, mas não existe nenhum serviço rodando nelas.

- *Block*: Todas as portas estão fechadas por padrão, ou seja, o Honeyd irá ignorar e descartar todos os pacotes e não irá responder nada.

- *Reset*: Significa que todas as portas estão fechadas por padrão. Se for uma porta TCP, o *honeypot* irá responder com um TCP RST para um pacote SYN e se for uma porta UDP o *honeypot* irá responder com uma mensagem ICMP para porta não alcançável.

O comando "*add*" também é utilizado para realizar algumas mudanças no *honeypot.* A linha "*add template tcp port 80 "sh /usr/share/honeyd/scripts/win32/web.sh""* diz que na porta TCP 80 está rodando um *script* de emulação de algum serviço, no exemplo está sendo emulado um servidor HTTP na porta 80. Diversos outros serviços em diversas outras portas podem ser configurados de acordo com o que for necessário.

O Honeyd também permite que se faça um *proxy* do tráfego que é capturado por ele, isso é feito como mostra a linha: *add template tcp port 22 proxy $ipsrc:22,* onde todo pacote que chegar na porta 22 será redirecionado para a porta 22 do IP de origem, ou seja, o tráfego será enviado de volta ao atacante.

O comando "*bind*" é utilizado para vincular um modelo de *honeypot* a um determinado endereço IP, no caso do exemplo o *honeypot "template"* está vinculado ao endereço de IP 192.168.1.100, mas diversos outros IP's podem ser vinculados a um modelo.

Existem diversos outros comandos possíveis de serem combinados no arquivo de configuração do Honeyd, no Anexo I se encontra a BNF (*Backus- Naur Form*) do arquivo de configuração.

Para executar o Honeyd é necessário chamá-lo em linha de comando através do comando: *honeyd 192.168.0.1/24* (a rede não necessariamente deve ser esta).

Diversas flags podem ser adicionada para determinar o comportamento de como o Honeyd irá rodar, abaixo são listadas e explicadas as principais flags.

- **-f configfile:** Usada para especificar onde encontrar o arquivo de configuração do Honeyd.

- **-i interface:** É possível especificar a interface que o Honeyd irá rodar. Por padrão ele roda na primeira interface instalada, mas se existir mais de uma interface instalada ela pode ser especificada através da flag –i. Ex: -i eth0 –i eth1.

- **-d debug:** Usada para rodar o Honeyd em modo de depuração. Diversas mensagens são mostradas no terminal sobre o que está ocorrendo no Honeyd.

- **-l logfile:** Com essa flag é possível definir onde será armazenado o arquivo de *log* do Honeyd. Essa flag é desabilitada por padrão.

- **-s servicelog:** Parecida com a anterior, é usada para definir onde será armazenado o *log* dos serviços que estão sendo emulados. Essa flag é desabilitada por padrão.

- **-p fingerprints:** É usada para dizer onde está o arquivo nmap.prints. Se for a instalação padrão do Honeyd ele pode ser encontrado em /etc/honeypot/nmap.prints e essa flag pode ser omitida.

- **-x prob:** É usada para especificar o caminho da base de dados do Xprobe.

3.3 Configuração de Serviços para Honeyd

Como foi dito anteriormente, é possível vincular um serviço a uma porta do *honeypot* que foi configurado. Existem diversos *scripts* que fazem isso e na maioria das vezes são *scripts* desenvolvidos por pelo próprio Niels Provos, criador do Honeyd e também por organizações como a *http://www.honeynet.org.br*.

Os *scripts* podem ser escritos em diversas linguagens de *script* tais como Shell Script, Perl e Python. Abaixo um exemplo de um *script* muito simples escrito em Shell Script (Código 2):

```
1. DATE=`date`
2. echo "$DATE: Started From $1 Port $2" >> /tmp/log
3.  echo SSH-1.5-2.40
4. while read name
5. do
6.       echo "$name" >> /tmp/log
7.       echo "$name"
8. done
```

Código 2 – Exemplo de *script*

Esse *script* mostra um cabeçalho SSH e então entra em loop armazenando em *log* e escrevendo na tela o que foi digitado pelo atacante. Com esse simples *script* é possível ter uma idéia de como desenvolver *scripts* mais elaborados que emule realmente um serviço real.

No Código 3 é ilustrado um exemplo de funcionamento de um protocolo de envio de e-mail, o SMTP. Mais informações sobre esse protocolo pode ser encontrada na RFC 821.

```
1. S: MAIL FROM:<Smith@Alpha.ARPA>
2. R: 250 OK
3. S: RCPT TO:<Jones@Beta.ARPA>
4. R: 250 OK
5. S: RCPT TO:<Green@Beta.ARPA>
6. R: 550 No such user here
7. S: RCPT TO:<Brown@Beta.ARPA>
8. R: 250 OK
9. S: DATA
10. R: 354 Start mail input; end with <CRLF>.<CRLF>
11. S: Blah blah blah...
12. S: ...etc. etc. etc.
13. S: <CRLF>.<CRLF>
14. R: 250 OK
```

Código 3 - Exemplo de funcionamento do SMTP

Observando o funcionamento do protocolo e seus comandos, é possível criar um *script* para emular o comportamento de um serviço SMTP como mostra o Código 4. Segue então um código em Python desenvolvido por Provos (2007) que emula parcialmente esse funcionamento:

```
1. #!/usr/bin/python
2. import re # para combinar o comando
3. import sys # ler do stdin
4. current_state = 'initial'
5. sender = ' '
```

```
6.  recipients = []
7.  for line in sys.stdin:
8.  if current_state == 'initial':
9.          res = re.match('mail from:(.*)', line, re.IGNORECASE)
10.         if not res:
11.                 print >>sys.stdout, '500 Syntax Error'
12.                 continue
13.         argument = res.group(1)
14.        # Analisando o argument para verificar se é um e-mail
15.        sender = argument
16.        current_state = 'need_recipient'
17.        print >>sys.stdout, '250 OK'
18. else:
19.         print >>sys.stdout, '500 Syntax Error'
20. elif current_state == 'need_recipient':
21.         res = re.match('rcpt to:(.*)', line, re.IGNORECASE)
22.         if res:
23.             argument = res.group(1)
24.             # Verificar se é um endereço de email --- you need to do that!
25.             recipients.append(argument)
26.             continue
27.    res = re.match('rcpt to:(.*)', line, re.IGNORECASE):
28.         if res:
29.              # Esperando dados agora. Diga ao usuário
30.             print >>sys.stdout, '354 Start mail input; end with
    <CRLF>.<CRLF>'
```

```
31.             current_state = 'getting_data'
32.             Continue
33.  print >>sys.stdout, '500 Syntax Error'
```

Código 4 – Exemplo de *script* SMTP, Provos (2007)

Alguns *scripts* criados para o Honeyd possuem um arquivo de configuração a parte, tornando possível definir alguns parâmetros sobre a emulação do serviço separado do código do *script*. Isso possibilita que os usuários do Honeyd configurem o comportamento de algum serviço sem ser necessário alterar o seu código.

Uma vantagem de se utilizar o Honeyd é sua flexibilidade em relação aos *scripts*. Caso os *scripts* já existentes não sejam suficientes para quem for utilizar, é possível adaptá-los, pois a maioria dos *scripts* disponíveis na *web* são softwares livres possibilitando sua modificação de acordo com as necessidades do usuário, e se adaptá-los não for suficiente, basta escrevê-los desde o começo.

3.4 Configuração do *Honeypot*

Nesta sessão será apresentada a construção do *honeypot* tais como as configurações necessárias nos servidores, o método utilizado para atribuir um IP ao *honeypot*, os serviços emulados, o modo como foi construído o arquivo de configuração do Honeyd, e a topologia construída.

Para o desenvolvimento do *honeypot* foi necessário entender a estrutura da rede onde ele foi implantado para que as configurações necessárias fossem realizadas, no caso o laboratório LaReS (Laboratório de Redes e Sistemas Distribuídos) que está na mesma sub-rede do BCC.

A Figura 9 representa uma visão alto nível de como a rede do LaReS está configurada. Pode-se observar dois servidores com os seguintes IP's: 200.131.224.100 e 200.131.224.101 respectivamente chamados de Gödel e Turing em

homenagem a Kurt Gödel e Alan Turing, duas grandes personalidades na área da computação.

Diversos serviços em cada um dos servidores estão em funcionamento para prover uma estrutura com controle de acesso e múltiplos usuários. Serviços como NIS+NFS e SAMBA implementando um sistema de arquivos distribuídos, *proxy* Squid para controle de acesso e *cache* de páginas *web* e serviço DHCP para distribuir IP's dinamicamente pela rede.

Também existe em funcionamento servidores HTTP como o Apache, servidores de banco de dados como PostgreSql e MySql, containers *web* e servidor de aplicações como Apache TomCat e GlassFish, servidor SSH, entre outros.

Para fazer a segurança, um *firewall* IPTABLES faz a filtragem de pacotes que entram e saem da rede. É através do IPTABLES que se torna possível modelar a topologia proposta.

O *honeypot* desenvolvido nesse trabalho tem como propósito capturar o tráfego malicioso que chega da rede externa ao BCC. Para conseguir isso é necessário um endereço de IP fixo para que atacantes de fora a rede possam alcançar o *honeypot* através da Internet.

O BCC possui dois endereços de IP fixos, ambos são utilizados por seus servidores, então para que fosse possível tornar o *honeypot* alcançável através da Internet foram necessárias configurações no *firewall* do servidor para que os endereços de IP fixo fossem aproveitados pelo *honeypot.*

Como o servidor Turing possui um endereço de IP na rede interna, devido ao fato de que ele está voltado para oferecer serviços internos, foram descobertos através de um escaneamento utilizando a ferramenta Nmap os serviços que rodam nele e com isso descobrir as portas que estão livres para assim emular serviços nessas portas a fim de enganar os atacantes que venham a escanea-lo através da Internet.

Após descobrir as portas livres foi possível escolher quais os serviços que poderiam ser emulados para então configurar os *honeypots*.

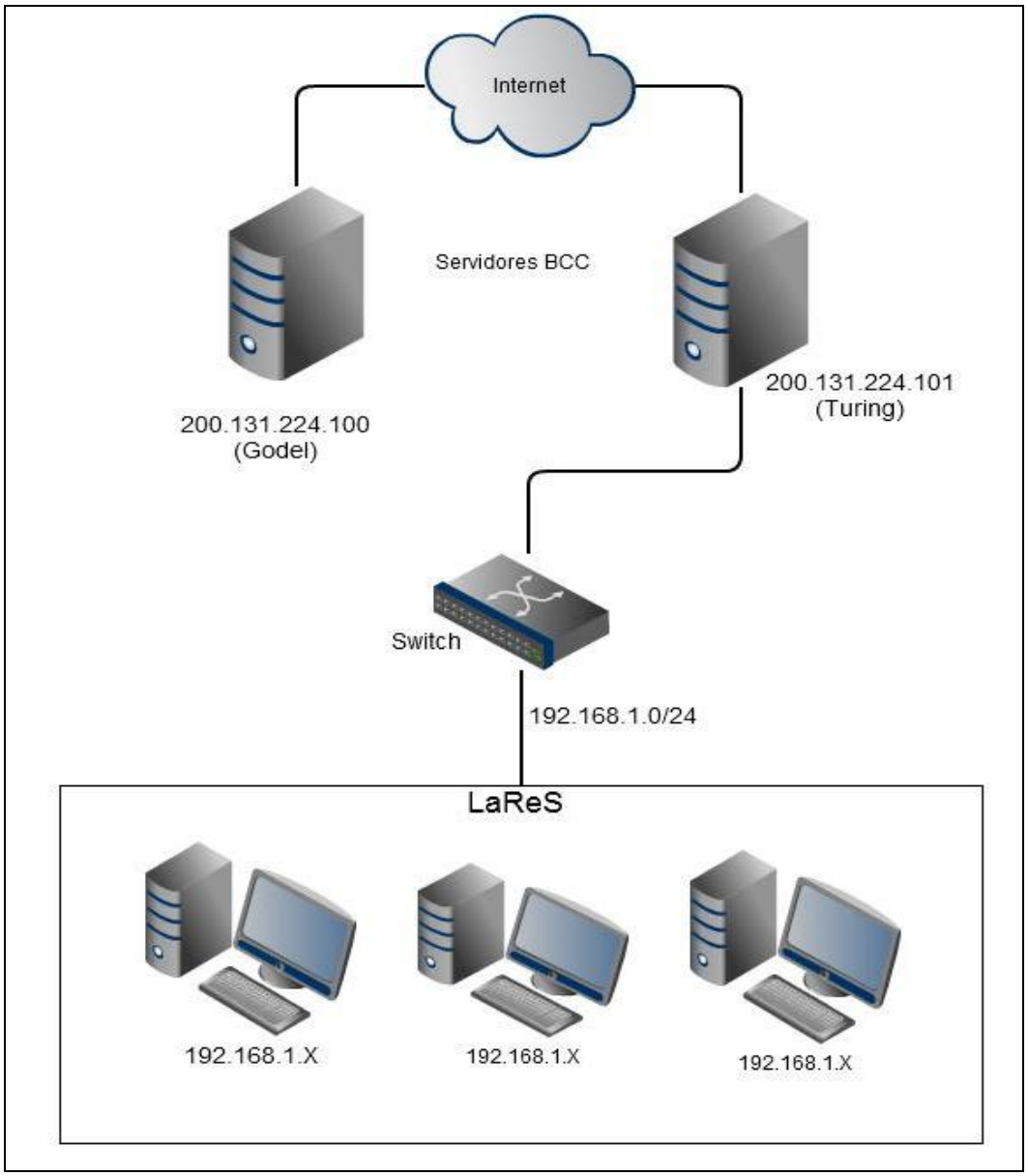

Figura 9 - Estrutura do LaReS

Foram escolhidos quatro serviços: um servidor HTTP na porta 1080, um servidor de envio de e-mails SMTP na porta 25, um servidor de recebimento de e-mails POP3 na porta 110 e um servidor FTP na porta 21.

O servidor HTTP foi emulado na porta 1080 do servidor Turing ao invés da porta padrão 80 porque esta porta já estava em utilização pelo servidor Apache que hospeda alguns sites do BCC. Mesmo não sendo na porta padrão não será perdida a função deste serviço, pois muitas instituições possuem servidores HTTP em

portas diferentes do padrão a fim de testar projetos em implantação ou então usar para serviços internos.

De qualquer forma, um atacante que vier a escanear as portas do servidor Turing irá se deparar com um servidor HTTP na porta 1080 e poderá começar uma tentativa de intrusão.

Para dar a impressão de que um servidor HTTP na porta 1080 do Turing também estava em funcionamento foi feito um encaminhamento de portas da porta 1080 do servidor Turing para porta 80 de um *host* virtual emulado pelo Honeyd.

A Figura 10 mostra como ficou configurada a rede.

Em um *host* físico na rede do LaReS com o endereço de IP 192.168.1.11 foi instalado a ferramenta Honeyd. Dentro do Honeyd foi configurado um *host* virtual com um sistema operacional Linux com o endereço de IP 192.168.1.15. Dentro deste *host* virtual está rodando um *script* que emula uma versão do servidor HTTP Apache na porta 80.

Esse encaminhamento de portas foi realizado através de uma configuração no IPTABLES dada pelos seguintes comandos:

iptables -I FORWARD -p tcp -d 192.168.1.15 --dport 80 -j ACCEPT
iptables -t nat -A PREROUTING -d 200.131.224.101 -p tcp --dport 1080 -j DNAT --to-destination 192.168.1.15:80

No primeiro comando está sendo dito ao IPTABLES para aceitar pacotes que sejam encaminhados a 192.168.1.15 porta 80 e no segundo comando está sendo feito um encaminhamento de todos os pacotes que chegarem destinados ao IP 200.131.224.101 porta 1080 para o *host* com endereço 192.168.1.15 porta 80 através de NAT.

Então se um atacante escanear as portas do servidor Turing irá encontrar um servidor HTTP Apache rodando na porta 1080 e poderá explorar as vulnerabilidades desse serviço para invadir o sistema. Como não se trata de um sistema real e sim de uma emulação provida pelo Honeyd, mesmo se o atacante comprometer o serviço, não ocorrerá nenhum problema ao servidor.

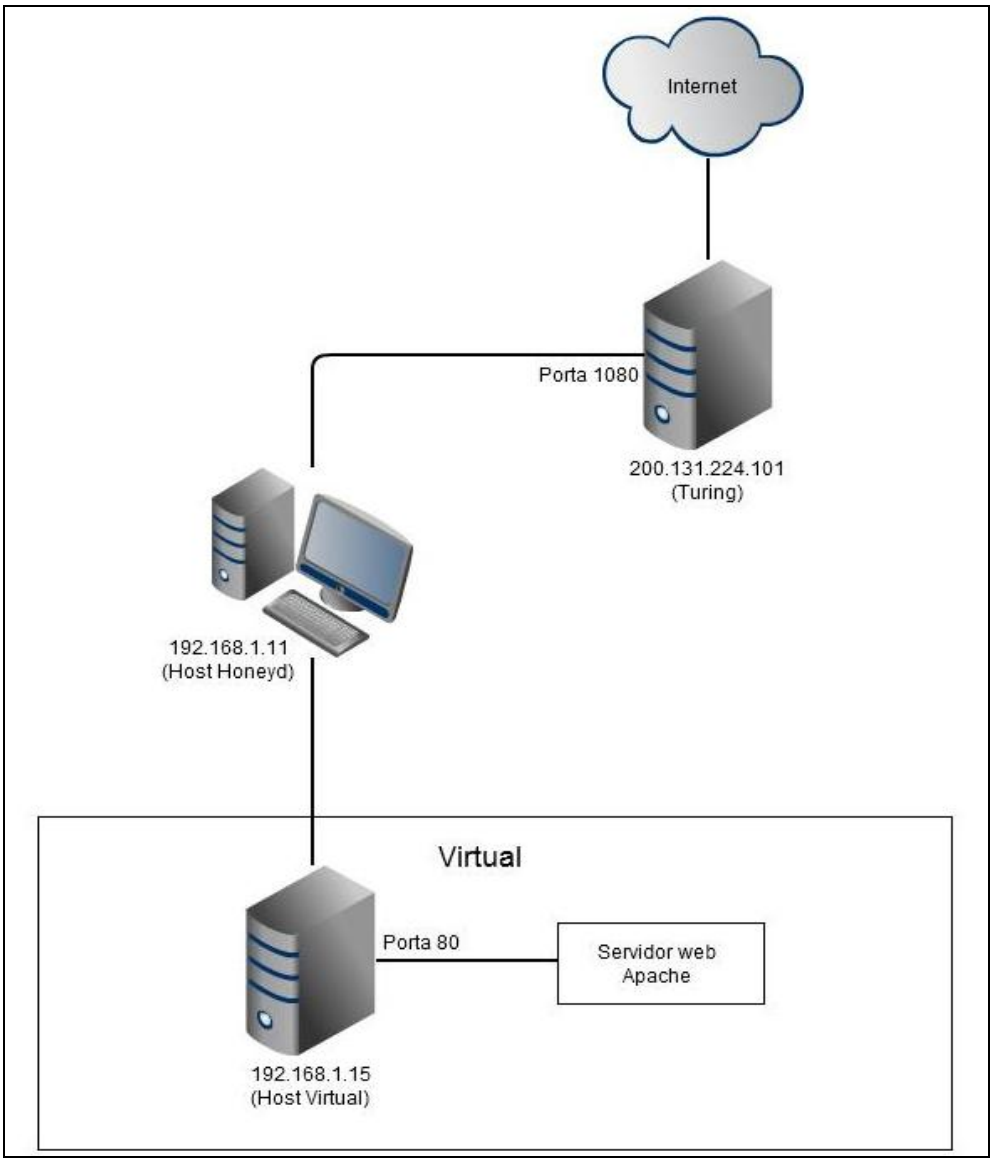

Figura 10 - *Honeypot* virtual emulando um servidor Apache

A Figura 11 mostra um escaneamento da porta 1080 do servidor Turing utilizando a ferramenta Nmap. Nesta figura podemos ver como um atacante enxerga o servidor, não conseguindo perceber que se trata de um serviço emulado que nem está rodando neste servidor.

```
nmap -sV -p 1080 200.131.224.101

Starting Nmap 5.51 ( http://nmap.org ) at 2011-04-28 13:42 Hora oficial do Brasil
Nmap scan report for 200.131.224.101
Host is up (0.036s latency).
PORT     STATE SERVICE VERSION
1080/tcp open  http    Apache httpd 1.3.23
Service Info: Host: bps-pc10.local.mynet

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.47 seconds
```

Figura 11 - Escaneamento da porta 1080 do servidor

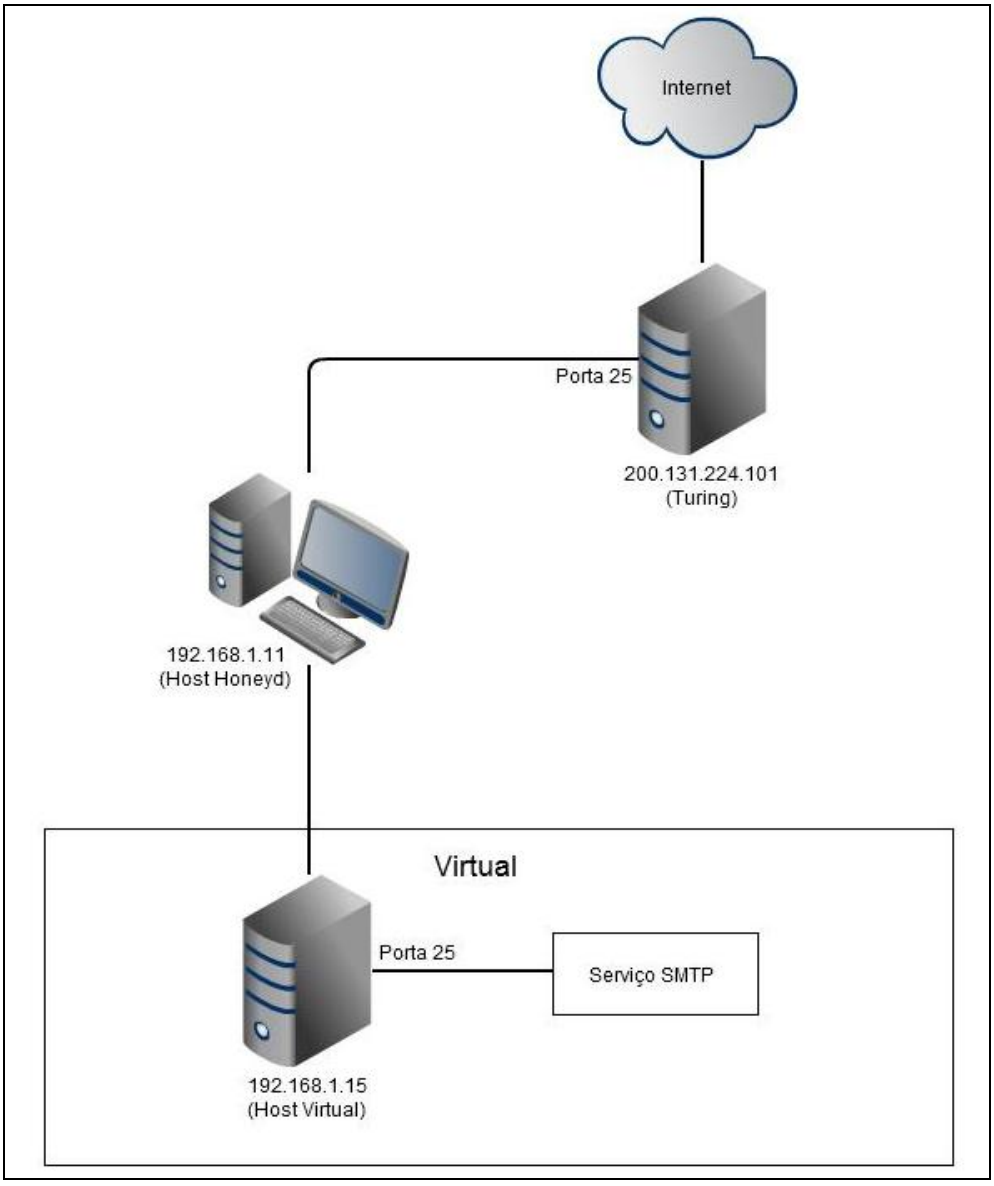

Figura 12 - *Honeypot* virtual emulando um servidor SMTP

Outro serviço que foi emulado pelo *honeypot* foi o de um servidor SMTP para o envio de e-mails. Com ele poderemos conseguir, caso houver, informações sobre o envio de SPAM e assim entender como funciona para então começar a desenvolver meios para evitá-los.

A configuração da rede é semelhante a do servidor HTTP que foi explicado a cima. Foi configurado um *script* para rodar na porta 25 do *host* virtual emulado pelo Honeyd e então foi feito um encaminhamento de portas da porta 25 do servidor Turing para a porta 25 do *host* virtual, como mostra a Figura 12.

Usando a ferramenta Nmap pode-se ver na Figura 13 que um serviço SMTP está funcionando na porta 25 do servidor Turing.

```
nmap -sV -p 25 200.131.224.101

Starting Nmap 5.51 ( http://nmap.org ) at 2011-04-28 14:04 Hora oficial do Brasil
Nmap scan report for 200.131.224.101
Host is up (0.038s latency).
PORT   STATE SERVICE VERSION
25/tcp open  smtp    Sendmail 8,12,2
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at http://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 19.66 seconds
```

Figura 13 - Escaneamento da porta 25 do servidor Turing

Para verificar a capacidade de realismo do *script* emulado, foram realizados alguns testes com esse serviço utilizando um cliente Telnet. Através do comando: *telnet 200.131.224.101 25,* foi possível estabelecer uma comunicação com o servidor como mostra a Figura 14.

```
220 Turing BCC ESMTP Debian Linux; Qui Abr 28 14:17:15 BRT 2011
Helo teste.br
250 Turing BCC Hello teste.br[187.86.109.92], pleased to meet you
mail from: <teste@teste.com>
250 2.1.0 <teste@teste.com>  ... Sender ok
RCPT TO: <teste@bcc.com.br>
553 sorry, that domain isn't in my list of allowed rcpthosts (#6.7.1)
quit
220 2.0.0 Turing BCC closing connection
Conexão ao host perdida.
```

Figura 14 - Teste em Telnet na porta 25

Neste teste o usuário se apresenta ao servidor e tenta enviar um email ao endereço teste@bcc.com.br, o servidor por sua vez não reconhece o comando e responde ao usuário com uma mensagem de erro, após isso o usuário encerra a sessão através do comando "*quit*".

Para a configuração do servidor de recebimento de e-mails, o POP3, também foram realizadas configurações de encaminhamento de portas através do IPTABLES. A Figura 15 mostra como ficou a rede após a configuração.

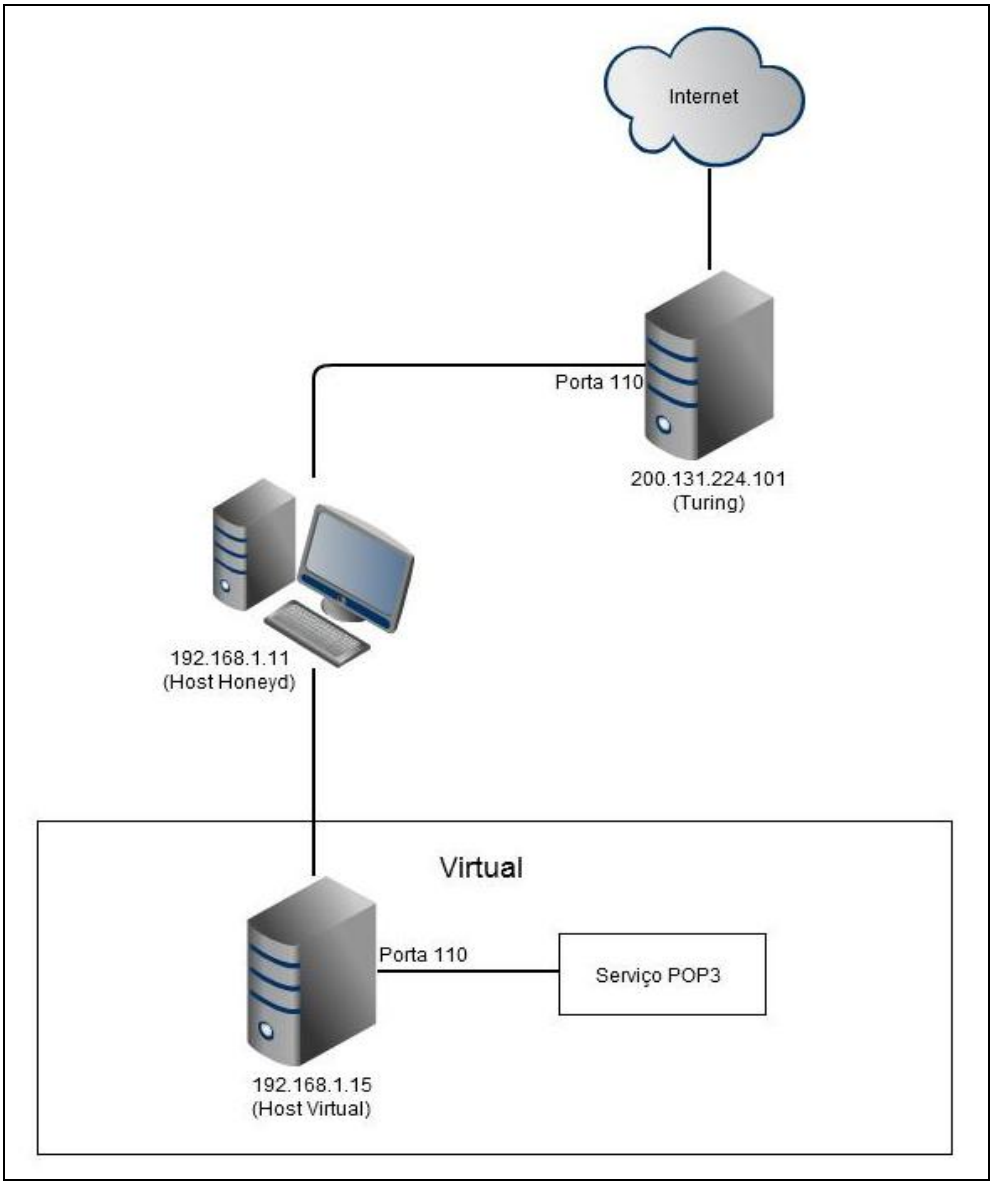

Figura 15 - *Honeypot* virtual emulando servidor POP3

O escaneamento de portas da Figura 16 confirma o funcionamento do *honeypot*. Com isso, poderemos saber quando algum invasor tentar acessar indevidamente os e-mails do servidor.

Figura 16 - Escaneamento da porta 110 do servidor Turing

Para confirmar o bom funcionamento do serviço emulado, também foram realizados testes através de um cliente Telnet (Figura 17) com o comando: *telnet 200.131.224.101 110*. Esse teste mostra um usuário conectando através do seu *login* e senha e então verificando se existem mensagens, como a caixa de emails está vazia o usuário encerra a sessão e o servidor se despede assim como os servidores reais.

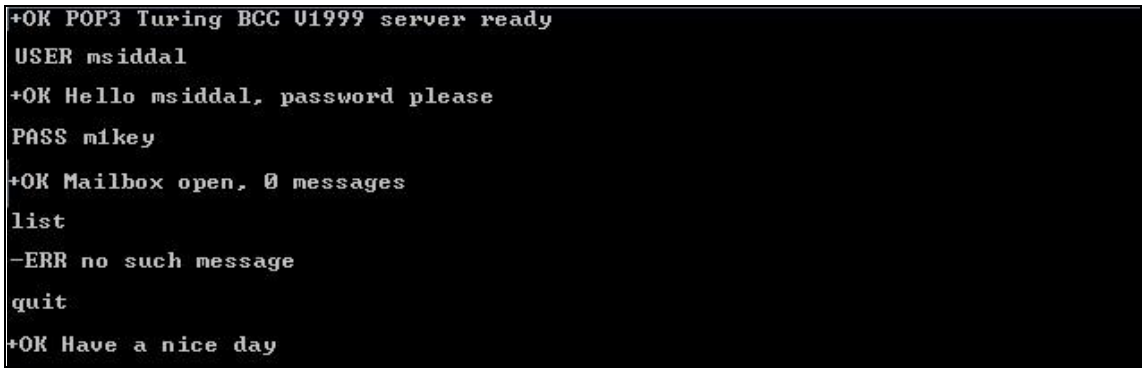

Figura 17 - Teste Telnet no servidor POP3

Por último, foi emulado um servidor FTP na porta 21 do servidor Turing. A Figura 18 mostra como ficou esquematizado o encaminhamento de portas.

Muitos invasores utilizam falhas dos servidores FTP para baixar suas ferramentas nos servidores, com esse *honeypot* será possível monitorar o servidor caso haja alguma tentativa de exploração de brechas.

Como nos outros serviços foi necessário fazer testes antes de colocar esse serviço *online* para verificar seu realismo e não entregar pro atacante que se trata de um *honeypot*.

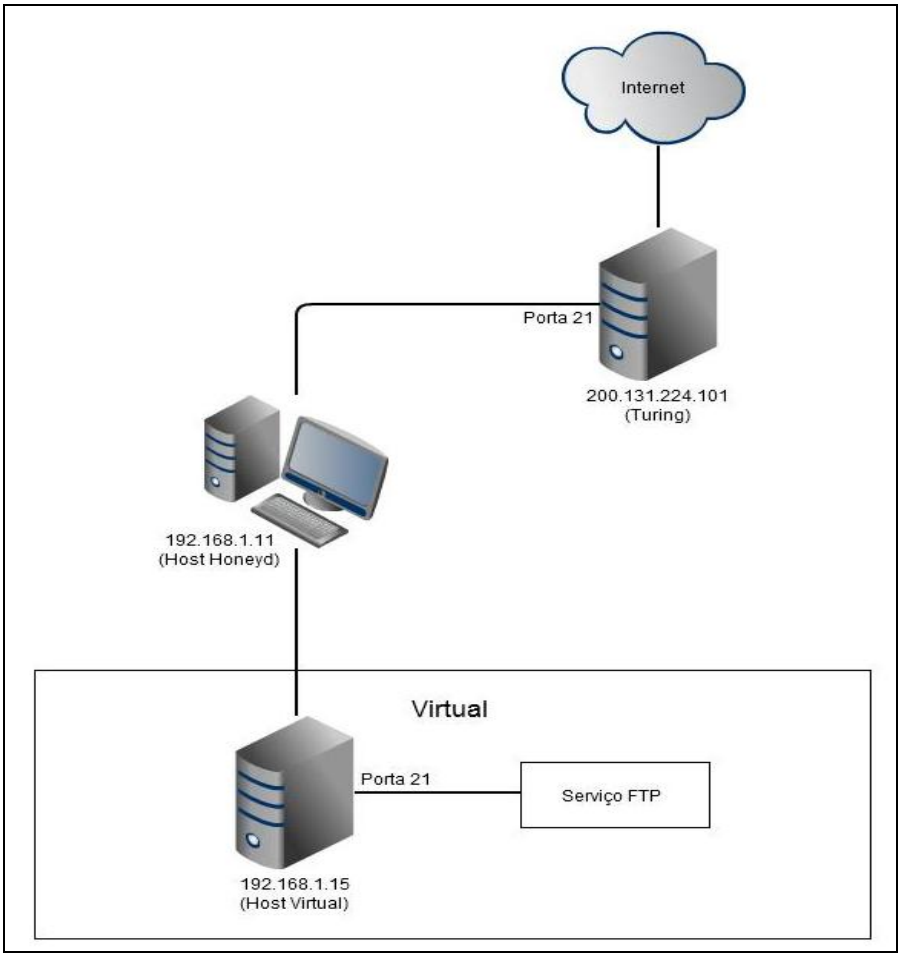

Figura 18 - *Honeypot* emulando servidor FTP

No teste realizado, o usuário tenta se conectar através de um usuário anônimo e o servidor aceita a conexão respondendo com algumas mensagens. Após isso o usuário utiliza o comando "*help*" e o servidor lista os comandos que podem ser usados como mostra a Figura 19.

Os *scripts* utilizados para essas emulações foram modificados para mostrar mensagens personalizadas de acordo com o servidor.

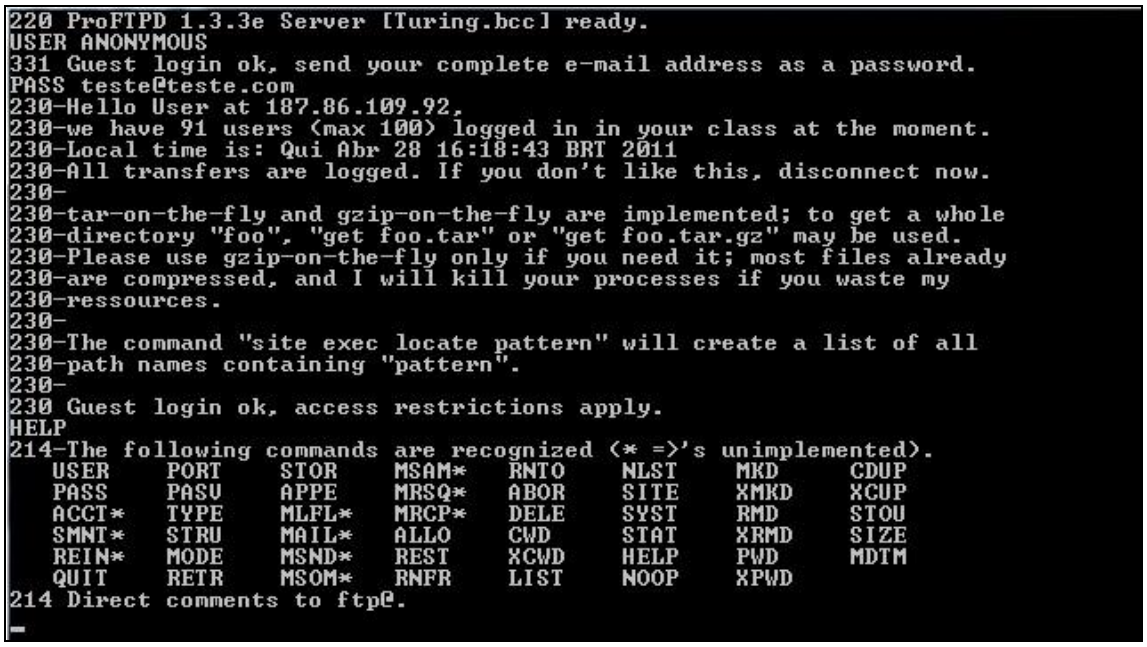

Figura 19 - Teste Telnet no servidor FTP

Para que o encaminhamento de portas funcione corretamente, é necessário garantir que o IP do *honeypot* onde está rodando os serviços nunca mude. Como na rede do LaReS os IP's são distribuídos através de um servidor DHCP, toda vez que o Honeyd fosse reiniciado, o servidor poderia dar um IP diferente para o *host* virtual. Para isso não ocorrer foram realizadas algumas configurações específicas no servidor e no arquivo de configurações do Honeyd.

O servidor DHCP possui uma funcionalidade que permite entregar sempre o mesmo endereço de IP para um determinado *host*, mas para isso é necessário saber o endereço de MAC do *host* e então realizar as devidas modificações.

No Honeyd como foi dito anteriormente é possível configurar uma interface de rede para cada *host* e atribuir um endereço de MAC específico a ele.

O Código 5 mostra como ficou o arquivo de configuração do Honeyd.

```
1. create honeypot
2. set honeypot personality "Linux kernel 2.4.16 – 2.4.18"
3. set honeypot default tcp action reset
4. set honeypot default udp action reset
5. set honeypot default icmp action open
6. set honeypot uptime 1110514
7. set honeypot uid 32767 gid 32767
8. add honeypot tcp port 80 "sh
   /usr/share/honeyd/scripts/unix/linux/suse8.0/apache.sh
9. add honeypot tcp port 110 "sh
   /usr/share/honeyd/scripts/unix/general/pop/emulate-pop3.sh
10. add honeypot tcp port 25 "sh
    /usr/share/honeyd/scripts/unix/general/smtp.sh
11. add honeypot tcp port 26 "sh
    /usr/share/honeyd/scripts/unix/linux/suse8.0/sendmail.sh
12. add honeypot tcp port 21 "sh
    /usr/share/honeyd/scripts/unix/linux/suse8.0/proftpd.sh
13. dhcp honeypot on eth0 ethernet "00:1e:4c:ca:1a:3b"
```

Código 5 – Arquivo de configuração final do Honeyd

Da linha 1 à linha 7 são os comandos de configurações básicas do Honeyd, como o nome, a personalidade emulada, o comportamento padrão e etc. Da linha 8 à linha 12 são os comandos que relacionam uma porta a um serviço específico. Já na linha 13 é mostrada a configuração necessária para se obter um IP fixo.

Primeiro é dito ao Honeyd que o endereço IP será obtido através de um servidor DHCP, depois é dito a interface na qual os pedidos DHCP deverão ser enviados e por fim é atribuído um endereço de MAC para essa interface.

Depois de configurar o endereço de MAC é possível modificar o servidor DHCP para enviar sempre o mesmo endereço de IP para esse MAC. A configuração é feita adicionando as seguintes linhas no arquivo de configuração do servidor DHCP (Código 6).

```
host honeypot {
hardware ethernet 00:1e:4c:ca:1a:3b;
fixed-address 192.168.1.15;
}
```

Código 6 – Configuração de IP fixo no DHCP

Após essas configurações o *honeypot* sempre irá receber o mesmo endereço de IP não importando quantas vezes o Honeyd for reiniciado.

3.5 Configuração da *Honeynet*

Como um adicional ao trabalho proposto, foi construído além do *honeypot*, uma pequena *honeynet* dentro da rede do LaReS. Nesta sessão será mostrada a forma como foi configurada essa *honeynet* utilizando a ferramenta Honeyd para emular diversos *hosts*. Será mostrado também a configuração de dois *hosts* infectados com os vírus MyDoom e Kuang2, que são *scripts* criados pela comunidade do Honeyd.

O propósito desta *honeynet* é ampliar a rede do LaReS com mais alguns *hosts* Linux e Windows e assim poder monitorar eventuais ataques vindos de dentro da rede.

Para atrair prováveis invasores, foram configurados dois *honeypots* virtuais cada um infectado por um vírus diferente. Um está emulando uma infecção pelo vírus MyDoom e outra está emulando uma infecção pelo vírus Kuang2. Esses dois vírus funcionam como espiões dentro da máquina hospedeira permitindo que o invasor tenha acesso remoto ao disco podendo baixar e colocar arquivos para então usá-los quando for necessário.

Além dos *hosts* infectados, também foram criados mais três *honeypots* virtuais rodando o sistema operacional Windows e mais cinco *honeypots* rodando Linux. Os *honeypots* rodando Windows possuem algumas portas abertas, essas, são portas que geralmente o sistema operacional deixa aberta para sua utilização.

Para descobrir quais são essas portas foi realizado um escaneamento de portas nas máquinas que rodam Windows dos laboratórios do BCC para assim ter uma maior realidade.

As portas escolhidas foram: a porta 135 que é utilizada pelo serviço MSRPC (Microsoft *Remote Procedure Call*), a porta 139 que é utilizada pelo serviço NETBIOS-SSN e a porta 445 que é utilizada pelo serviço MICROSOFT-DS.

Para os *hosts* Linux foi realizado o mesmo procedimento, e foi encontrada apenas a porta 111 aberta que é utilizada pelo serviço RPCBIND.

Diferente do *honeypot* descrito na sessão anterior, para esses *honeypots* não foram necessárias configurações adicionais de encaminhamento de portas, pois como esses *honeypots* devem seguir o mesmo funcionamento dos computadores físicos da rede, os endereços de IP são adquiridos dinamicamente através do DHCP. Então cada vez que o Honeyd for reiniciado, os *honeypots* virtuais receberão um endereço de IP diferente da mesma forma que ocorre quando um computador é real é ligado.

Os Códigos 7 e 8 mostram como ficaram os arquivos de configuração para os *hosts* infectados com MyDoom e Kuang2.

```
1. #CRIANDO UM HOST WINDOWS INFECTADO COM O MYDOOM
2. create hostWindowsMyDoom
3. set hostWindowsMyDoom personality "Microsoft Windows XP Professional"
4. set hostWindowsMyDoom default tcp action reset
5. set hostWindowsMyDoom default udp action reset
6. set hostWindowsMyDoom default icmp action open
7. set hostWindowsMyDoom ethernet "dell"
8. add hostWindowsMyDoom tcp port 135 open #MSRPC
```

```
9.  add hostWindowsMyDoom tcp port 139 open #NETBIOS-SSN
10. add hostWindowsMyDoom tcp port 445 open #microsoft-ds
11. add hostWindowsMyDoom tcp port 4444
    "/usr/share/honeyd/scripts/mydoom/mydoom.pl -l
    /usr/share/honeyd/logs/mydoom
12. dhcp hostWindowsMyDoom on eth0
```

Código 7: Arquivo de configuração para o *host* infectado com MyDoom

```
#CRIANDO HOST WINDOWS INFECTADO COM O KWANG2
create hostWindowsKwang
set hostWindowsKwang personality "Microsoft Windows XP Professional"
set hostWindowsKwang default tcp action reset
set hostWindowsKwang default udp action reset
set hostWindowsKwang default icmp action open
set hostWindowsKwang ethernet "dell"
add hostWindowsKwang tcp port 135 open #MSRPC
add hostWindowsKwang tcp port 139 open #NETBIOS-SSN
add hostWindowsKwang tcp port 445 open #microsoft-ds
add hostWindowsKwang tcp port 17300 "/usr/share/honeyd/scripts/kuang2.pl -
f /usr/share/honeyd/scripts/kuang2.conf"
dhcp hostWindowsKwang on eth0
```

Código 8: Arquivo de configuração para o *host* infectado com Kuang2

A novidade desses arquivos de configuração mostrados nos códigos 7 e 8 em relação aos outros é a linha 6, onde atribuímos uma interface de rede da marca Dell ao invés de definir manualmente um endereço de MAC para o *host*.

Nas linhas 7, 8 e 9 do Código 7 são abertas as portas dos serviços básicos do Windows. Na linha 10 é configurado um *script* que emula o MyDoom na porta 4444

e através da flag –l foi dito o local que será armazenado o arquivo de *log*. Por fim, na linha 11 é dito ao Honeyd para obter um endereço de IP para o *honeypot* virtual através do DHCP.

O Código 8 é semelhante ao 7 só que ao invés de emular o vírus MyDoom na porta 4444 ele emula um vírus Kuang2 na porta 17300. A forma de definir o local do arquivo de *log* do Kuang2 também é diferente e é feito através de um arquivo de configuração do próprio *script* e não no arquivo de configuração do Honeyd(Código 9).

```
1. ### kuang2.conf -- kuang2.pl configuration file.
2. # logdir -- directory where logfile and uploaded files are kept.
3. logdir = /usr/share/honeyd/logs/kuang2/
4. # values we report back to the kuang2 client.
5. num_drives = 1
6. computer_name = BCC D304-09
7. # maximum allowed upload file size, in bytes.
8. max_upload_size = 15728640
9. # timeout value, in seconds.
10. timeout = 30
```

Código 9: Arquivo de configuração do *script* do Kuang2

O Código 9 mostra o arquivo de configuração do *script* que emula o Kuang2. Na linha 3 é definido o caminho do arquivo de *log*, na linha 5 é definido o número de drives que terá o *host* emulado, na linha 6 é definido o nome do *host*, na linha 8 é definido a quantidade máxima de *upload* que o atacante poderá fazer e por fim na linha 10 é definido o tempo de *timeout* do serviço.

O Código 10 mostra a configuração dos três *hosts* Windows sem nenhuma infecção. O arquivo de configuração é semelhante ao dos *hosts* infectados, mas não possuem *scripts* emulando algum serviço malicioso.

```
1. #CRIANDO 3 HOSTS WINDOWS
2. create hostWindows
3. set hostWindows personality "Microsoft Windows XP Professional"
4. set hostWindows default tcp action reset
5. set hostWindows default udp action reset
6. set hostWindows default icmp action open
7. set hostWindows ethernet "dell"
8. add hostWindows tcp port 135 open #MSRPC
9. add hostWindows tcp port 139 open #NETBIOS-SSN
10. add hostWindows tcp port 445 open #microsoft-ds
11. dhcp hostWindows on eth0
12. dhcp hostWindows on eth0
13. dhcp hostWindows on eth0
```

Código 10: Arquivo de configuração dos *hosts* Windows

O Código 11 mostra o arquivo de configuração dos 5 *hosts* Linux, onde na linha 5 é aberta a porta do RPCBIND na linha 6 é definida um interface Dell para o *honeypot* e da linha 7 a 11 é dito ao Honeyd para obter 5 endereços de IP através do DHCP e vinculá-los com o *honeypot* criado.

```
#CRIANDO 5 HOSTS LINUX
1. create hostLinux
2. set hostLinux personality "Linux Kernel 2.4.20"
3. set hostLinux default tcp action reset
4. set hostLinux default udp action reset
5. add hostLinux tcp port 111 open #RPCBIND
6. set hostLinux ethernet "dell"
```

```
7. dhcp hostLinux on eth0
8. dhcp hostLinux on eth0
9. dhcp hostLinux on eth0
10. dhcp hostLinux on eth0
11. dhcp hostLinux on eth0
```

Código 11: Arquivo de configuração dos *hosts* Linux

Após implantada a *honeynet* é possível obter alguns dados sobre o tráfego na rede interna sendo possível descobrir se houve algum tipo de escaneamento na rede em busca de brechas, e se forem encontrada as brechas propositalmente abertas neste trabalho é possível saber como o atacante agiu para explorá-las.

4
Resultados Obtidos

Com a utilização de um *honeypot* e de uma *honeynet* implantados na rede do BCC foram obtidos diversos resultados que serão apresentados e explicados nesse capítulo.

4.1 Resultados Obtidos com o *Honeypot*

Para realizar o experimento e obter resultados realistas, no dia 27 de Abril, o *honeypot* foi colocado em funcionamento no servidor Turing e assim pôde ser acessado por qualquer pessoa através de Internet. A partir desse dia, nenhuma conexão para testes no *honeypot* foi realizada evitando assim a contaminação dos dados.

No dia 28 de Abril foi detectada uma grande quantidade de conexões nos serviços emulados pelo Honeyd, mas nada foi feito senão a geração dos arquivos de *log* dos serviços que estavam sendo sondados.

Após 10 dias foi observado que os *logs* dos serviços diziam que não havia mais espaço em disco para escrever os dados capturados (Figura 20). Assim se fez necessário a desativação do *honeypot* para então recolher os dados obtidos e assim continuar com o experimento.

```
|/usr/share/honeyd/scripts/unix/general/pop/emulate-pop3.sh: line 74: echo: write error: Não há espaço disponível no dispositivo|
|/usr/share/honeyd/scripts/unix/general/pop/emulate-pop3.sh: line 49: echo: write error: Não há espaço disponível no dispositivo|
|/usr/share/honeyd/scripts/unix/general/pop/emulate-pop3.sh: line 49: echo: write error: Não há espaço disponível no dispositivo|
|/usr/share/honeyd/scripts/unix/general/pop/emulate-pop3.sh: line 49: echo: write error: Não há espaço disponível no dispositivo|
|/usr/share/honeyd/scripts/unix/general/pop/emulate-pop3.sh: line 49: echo: write error: Não há espaço disponível no dispositivo|
|/usr/share/honeyd/scripts/unix/general/pop/emulate-pop3.sh: line 49: echo: write error: Não há espaço disponível no dispositivo|
|/usr/share/honeyd/scripts/unix/general/pop/emulate-pop3.sh: line 49: echo: write error: Não há espaço disponível no dispositivo|
```

Figura 20 - *Log* dos *scripts*

4.1.1 Análise dos *Logs* Gerados pelo Servidor POP3

Analisando os *logs* gerados pelos serviços emulados foi notado que um ou alguns atacantes fizeram uma tentativa de acesso massiva no servidor POP3 aqui

emulado, e com isso foi gerado centenas de arquivos de *log* com tamanhos acima de 100MB excedendo a capacidade do disco rígido do *host* em que se encontra o Honeyd.

Após os dados serem limpos manualmente, somente o necessário foi mantido nos arquivos e assim foi possível verificar que se tratava de um ataque dicionário, onde o atacante constrói uma lista com prováveis palavras para *login* e senha e então testam todas a fim de obter acesso ao serviço.

As palavras usadas na tentativa de *login* que foram encontradas nos arquivos de *log* eram palavras básicas dentro de um meio coorporativo e nomes comumente encontrados no mundo como mostram as Figuras 21, 22, 23 e 24.

```
Connection from 209.200.52.196 to 57523 started at Sex Abr 29 12:11:24 UTC 2011
2011-04-29 12:11:24 RESP: 2011-04-29 12:11:24 CMD: >USER adam
 <
2011-04-29 12:11:25 RESP: +OK Hello adam, password please
2011-04-29 12:11:26 RESP: 2011-04-29 12:11:26 CMD: >PASS adam
 <
2011-04-29 12:11:27 RESP: -ERR Bad login
```

Figura 21 - Tentativas de acesso com nomes

```
Connection from 24.187.210.198 to 40444 started at Sáb Abr 30 01:30:35 UTC 2011
2011-04-30 01:30:35 RESP: 2011-04-30 01:30:35 CMD: >USER mysql
 <
2011-04-30 01:30:37 RESP: +OK Hello mysql, password please
2011-04-30 01:30:37 RESP: 2011-04-30 01:30:37 CMD: >PASS mysql <
2011-04-30 01:30:39 RESP: -ERR Bad login
```

Figura 22 - Tentativas de acesso com a palavra mysql

```
Connection from 209.200.52.196 to 55434 started at Sex Abr 29 12:06:39 UTC 2011
2011-04-29 12:06:39 RESP: 2011-04-29 12:06:39 CMD: >USER oracle
 <
2011-04-29 12:06:40 RESP: +OK Hello oracle, password please
2011-04-29 12:06:40 RESP: 2011-04-29 12:06:40 CMD: >PASS zxcvbn
 <
2011-04-29 12:06:41 RESP: -ERR Bad login
```

Figura 23 - Tentativas de acesso com a palavras Oracle

```
Connection from 209.200.52.196 to 53470 started at Sex Abr 29 12:10:48 UTC 2011
2011-04-29 12:10:48 RESP: 2011-04-29 12:10:48 CMD: >USER backup
 <
2011-04-29 12:10:49 RESP: +OK Hello backup, password please
2011-04-29 12:10:49 RESP: 2011-04-29 12:10:49 CMD: >PASS backup
 <
2011-04-29 12:10:50 RESP: -ERR Bad login
```

Figura 24 - Tentativas de acesso com a palavra *backup*

Ainda com relação às Figuras 21, 22, 23 e 24, pode-se ver na primeira linha dos arquivos de *log* a presença do endereço de IP de quem está se conectando com o serviço. Assim utilizando alguma ferramenta para descobrir a localização geográfica de endereços IP, no caso foi utilizada a ferramenta disponível no site http://www.ip2location.com/, foi possível descobrir a localização desses IP's e então tentar tirar alguma informação desses dados.

As Figuras 25 a 28 mostram informações obtidas sobre a localização de alguns endereços IP's que tentaram invadir o servidor POP3.

IP Address	Country	Region	City	Latitude/ Longitude	ZIP Code	Time Zone
24.104.158.13	UNITED STATES	ILLINOIS	FOX LAKE	42.396687 -88.183696	60020	-06:00

Net Speed	ISP	Domain
DSL	COMCAST TELECOMMUNICATIONS INC	COMCASTBUSINESS.NET

IDD Code	Area Code	Weather Station
1	847	USIL0423 - FOX LAKE

MCC	MNC	Mobile Brand
-	-	-

Figura 25 - Localização do IP 24.104.158.13

IP Address	Country	Region	City	Latitude/ Longitude	ZIP Code	Time Zone
94.76.222.170	UNITED KINGDOM	-	-	54.166997 -4.482106	-	+00:00

Net Speed	ISP	Domain
COMP	POUNDHOST CUSTOMER ALLOCATION	AS29550.NET

IDD Code	Area Code	Weather Station
44	-	UKXX0233 - ESKMEALS

MCC	MNC	Mobile Brand
-	-	-

Figura 26 - Localização do IP 94.76.222.170

IP Address	Country	Region	City	Latitude/ Longitude	ZIP Code	Time Zone
210.77.75.173	CHINA	BEIJING	BEIJING	39.9 116.413	-	+08:00

Net Speed	ISP	Domain
DSL	ANHUI INFORMATIONG CENTER	-

IDD Code	Area Code	Weather Station
86	-	CHXX0008 - BEIJING

MCC	MNC	Mobile Brand
-	-	-

Figura 27 - Localização do IP 210.77.75.173

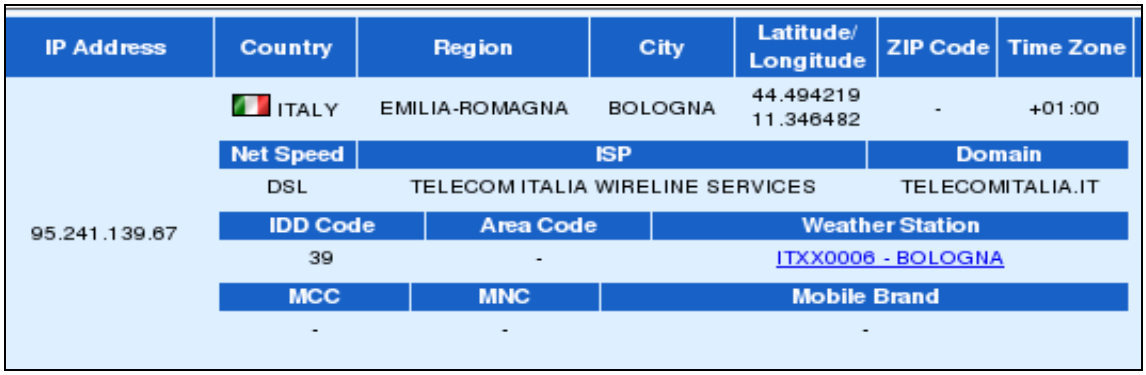

Figura 28 - Localização do IP 95.241.139.67

A Tabela 1 mostra com mais detalhes todos os endereços de IP e suas localidades.

Tabela 1 - Localidades de IP's que acessaram o servidor POP3

Endereço de IP	País	Região	Cidade
24.104.158.13	Estados Unidos	Illinois	Fox Lake
24.187.210.198	Estados Unidos	New York	Greenwood Lake
71.249.229.215	Estados Unidos	New York	New York
94.76.222.170	Reino Unido	-	-
95.241.139.67	Itália	Emilia-Romagna	Bologna
201.39.38.252	Brasil	-	-
208.115.238.196	Estados Unidos	Texas	Dallas
209.200.52.192	Estados Unidos	New Jersey	Ramsey
210.77.75.173	China	Beijing	Beijing
12.69.74.175	Estados Unidos	Texas	Richardson
200.179.255.69	Brasil	-	-
218.61.35.151	China	Beijing	Beijing

Observando a Tabela 1 pode-se chegar a duas informações sobre o atacante: Ou são vários atacantes ao redor do mundo, ou é apenas um atacante utilizando computadores infectados ao redor do mundo para assim esconder seu rastro.

4.1.2 Análise de *Logs* Gerado pelo Servidor SMTP

Após analisados os *logs* gerados pelo servidor POP3 foram analisados os resultados obtidos com o servidor de envio de email SMTP.

Ao contrário do POP3, o servidor SMTP não teve tantos acessos, mas teve o suficiente para perceber que estava sendo sondado por algum atacante.

As Figuras 29, 30, 31 e 32 mostram alguns pedaços dos arquivos de *log* gerados pelo servidor SMTP.

```
Qui Mai  5 18:09:08 BRT 2011: SMTP started from 118.167.10.26 Port 2872
HELO 200.131.224.101
MAIL FROM: <8888@163.com>
RCPT TO: <superedm001@yahoo.com.tw>
```

Figura 29 - *Log* SMTP 1

```
Dom Mai  8 00:24:57 BRT 2011: SMTP started from 114.44.100.241 Port 4292
HELO 200.131.224.101
MAIL FROM: <z2007tw@yahoo.com.tw>
RCPT TO: <vkihwpdh@yahoo.com.tw>
```

Figura 30 - *Log* SMTP 2

```
Dom Mai  8 03:47:55 BRT 2011: SMTP started from 118.161.240.189 Port 1135
HELO 200.131.224.101
MAIL FROM: <hi7188s.pp5975@msa.hinet.net>
RCPT TO: <zz@mail2000.com.tw>
```

Figura 31 - *Log* SMTP 3

```
Sáb Mai  7 11:24:27 BRT 2011: SMTP started from 138.199.70.129 Port 1225
EHLO windows
Sáb Mai  7 11:24:40 BRT 2011: SMTP started from 138.199.70.129 Port 1934
Sáb Mai  7 11:24:40 BRT 2011: SMTP started from 138.199.70.129 Port 1940
Sáb Mai  7 11:24:40 BRT 2011: SMTP started from 138.199.70.129 Port 1939
Sáb Mai  7 11:24:40 BRT 2011: SMTP started from 138.199.70.129 Port 1936
Sáb Mai  7 11:24:40 BRT 2011: SMTP started from 138.199.70.129 Port 1938
Sáb Mai  7 11:24:40 BRT 2011: SMTP started from 138.199.70.129 Port 1942
Sáb Mai  7 11:24:40 BRT 2011: SMTP started from 138.199.70.129 Port 1946
Sáb Mai  7 11:24:40 BRT 2011: SMTP started from 138.199.70.129 Port 1944
EHLO windows
EHLO windows
EHLO windows
EHLO windows
EHLO windows
EHLO windows
EHLO windows
EHLO windows
AUTH LOGIN
AUTH LOGIN
AUTH LOGIN
AUTH LOGIN
```

Figura 32 - *Log* SMTP 4

Pode-se ver claramente que alguém está tentando acessar o servidor emulado pelo *honeypot* para enviar emails para alguns destinatários.

Utilizando uma ferramenta para localização de endereços IP's foi possível descobrir algumas informações sobre quem estava sondando o servidor e com isso

tirar algumas conclusões. As Figuras 33 e 34 ilustram os dados obtidos sobre dois IP's que sondavam o *honeypot*.

IP Address	Country	Region	City	Latitude/ Longitude	ZIP Code	Time Zone
118.161.240.189	TAIWAN	T'AI-PEI	TAIPEI	25.017 121.45	-	+08:00

Net Speed	ISP	Domain
DSL	CHUNGHWA TELECOM DATA COMMUNICATION BUSINESS GROUP	CHT.COM.TW

IDD Code	Area Code	Weather Station
886	-	TWXX0021 - TAIPEI

MCC	MNC	Mobile Brand
-	-	-

Figura 33 - Localização do IP 118.161.240.189

IP Address	Country	Region	City	Latitude/ Longitude	ZIP Code	Time Zone
138.199.70.129	NETHERLANDS	NOORD-HOLLAND	AMSTERDAM	52.373801 4.890935	-	+01:00

Net Speed	ISP	Domain
DSL	CP-NET-SUPERNEWS	-

IDD Code	Area Code	Weather Station
31	-	NLXX0002 - AMSTERDAM

MCC	MNC	Mobile Brand
-	-	-

Figura 34 - Localização do IP 138.199.70.129

Tabela 2 - Localidades de IP's que acessaram o servidor SMTP

Endereço de IP	País	Região	Cidade
12.69.74.175	Estados Unidos	Texas	Richardson
144.44.100.241	Taiwan	T'ai-Pei	Taipei
118.161.240.189	Taiwan	T'ai-Pei	Taipei
118.167.10.26	Taiwan	T'ai-Pei	Taipei
138.199.70.129	Holanda	Noord-Holland	Amsterdam
200.179.255.69	Brasil	-	-
218.61.35.151	China	Beijing	Beijing

Observando a Tabela 2, nota-se que dois novos países apareceram na lista, Taiwan e Holanda. Mas ainda assim continuam aparecendo servidores iguais aos utilizados no ataque ao POP3, que é o caso do servidor 12.69.74.175 encontrado no Texas nos Estados Unidos, do servidor 200.179.255.69 encontrado no Brasil e do servidor 218.61.35.151 encontrado em Beijing na China.

A partir desses fatos pode-se perceber a ligação entre os dois ataques e levar em consideração o fato de que foram feitos pelo mesmo ou pelos mesmos atacantes.

4.1.3 Análise de *Logs* Gerado pelo Servidor FTP

Um dos servidores emulados pelo *honeypot* é um servidor FTP, pois esse tipo de servidor é muito visado em tentativas de ataque devido ao fato de que uma vez invadido o atacante consegue acessar arquivos e enviar arquivos para o servidor.

Ao mesmo tempo em que os servidores SMTP e POP3 estavam sob tentativa de invasão, também ocorria um ataque dicionário no servidor FTP e com isso foi gerado um arquivo de *log* com a lista de *login* e senha utilizada pelo atacante.

O arquivo de *log* registrou 18866 combinações diferentes de *login* e senha para tentar obter acesso ao servidor. Entre essas combinações foram encontrados nomes de pessoas de diversos países, senhas em branco, senhas numéricas seqüenciais e repetidas, senhas com caracteres especiais, nome de animais, nomes de personagem, entre outros, como mostra a Figura 35.

```
USER Administrator     USER Administrator
PASS !@#$!@#$          PASS Hobbit
USER Administrator     USER Administrator
PASS nakamori          PASS Hawkeye
USER anthony           USER Administrator
PASS password          PASS apache
USER anthony           USER Administrator
PASS 123456            PASS beer
USER Administrator     USER Administrator
PASS noboru            PASS bruce
USER Administrator     USER Administrator
PASS popeye            PASS 888888
USER Administrator     USER carol
PASS Beaver            PASS password
USER Administrator     USER carla
PASS BigBird           PASS 123456
USER career
PASS
```

Figura 35 - Tentativas de *login* no servidor FTP

Da mesma forma que os outros arquivos de *log* apresentados até o momento, o arquivo de *log* do servidor FTP também armazena informações sobre os endereços de IP's utilizados pelos atacantes. Assim é possível fazer uma análise e tirar algumas conclusões sobre o ataque.

Nas Figuras 36 e 37 são observadas algumas das localidades dos IP's utilizados para tentar invadir o servido FTP.

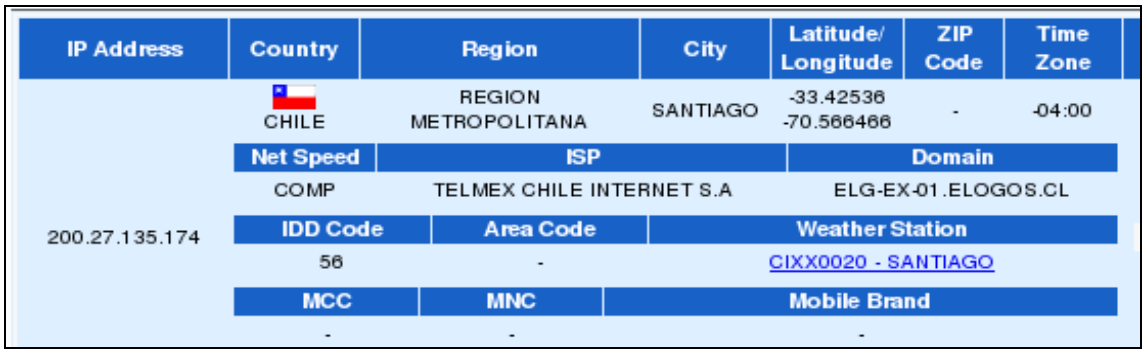

Figura 36 - Localização do IP 200.27.135.174

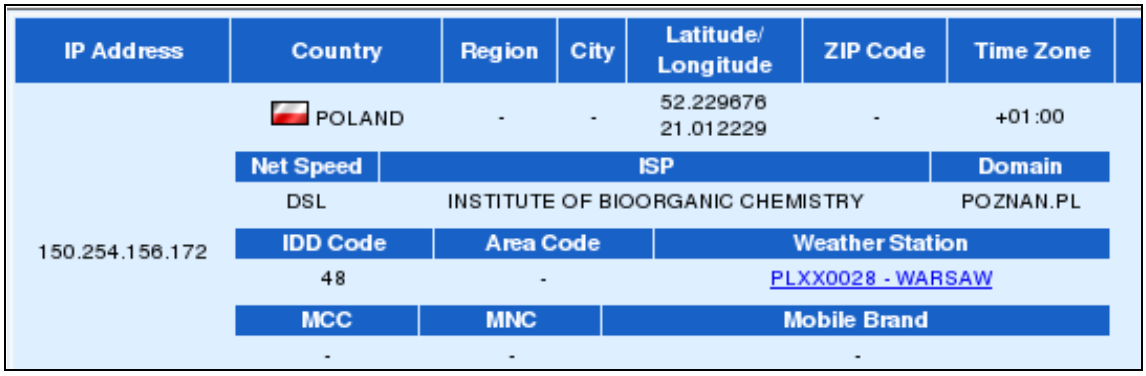

Figura 37 - Localização do IP 150.254.156.172

A Tabela 3 mostra com detalhes todos os endereços de IP's utilizados para esse ataque.

Tabela 3 - Localidades de IP's que acessaram o servidor FTP

Endereço de IP	País	Região	Cidade
200.27.135.174	Chile	Region Metropolitana	Santiago
150.254.156.172	Polônia	Wielkopolskie	Poznan
124.104.75.122	Filipinas	Benguet	Philippine
109.74.135.130	Rússia	Moskva	Moscow
74.208.155.21	Estados Unidos	Flórida	Miami
222.110.111.173	Coréia do Sul	Seoul-t'ukpyousi	Seoul
178.140195.8	Rússia	-	-

No caso do servidor FTP, não foi encontrada nenhuma correspondência de endereços de IP comparado com os outros ataques. Ao contrário disso, novos

endereços de IP de novos países foram adicionados à lista. Com base nessas informações não se pode dizer que se trata de um mesmo atacante, mas também não descarta essa possibilidade, pois os ataques nos servidores SMTP, POP3 e FTP ocorreram na mesma janela de tempo.

4.1.4 Análise de *Logs* Gerado pelo Servidor HTTP

O servidor HTTP emulado na porta 1080 do servidor do BCC também foi acessado, mas diferente dos outros servidores poucas conexões foram realizadas.

Qualquer pessoa que tentasse se conectar com esse servidor iria gerar uma linha no arquivo de *log* mesmo se não tivesse nenhuma intenção de invasão. Mas devido à premissa de ninguém saber da existência desse servidor HTTP rodando na porta 1080, os *logs* gerados podem ser tratados como *logs* de uma sondagem no servidor.

As Figuras 38 e 39 ilustram alguns dos endereços de IP que acessaram o servidor HTTP emulado.

IP Address	Country	Region	City	Latitude/ Longitude	ZIP Code	Time Zone
59.120.52.54	TAIWAN	TAI-PEI	TAIPEI	25.017 121.45	-	+08:00
	Net Speed	**ISP**				**Domain**
	DSL	CHTD CHUNGHWA TELECOM CO. LTD				CHT.COM.TW
	IDD Code	**Area Code**		**Weather Station**		
	886	-		TWXX0021 - TAIPEI		
	MCC	**MNC**		**Mobile Brand**		

Figura 38 - Localização do IP 59.120.52.54

IP Address	Country	Region	City	Latitude/ Longitude	ZIP Code	Time Zone
218.76.65.98	CHINA	HUNAN	HUNAN	28.716667 118.833333	-	+08:00
	Net Speed	**ISP**				**Domain**
	DSL	CHINANET-HN JISHOU NODE NETWORK				-
	IDD Code	**Area Code**		**Weather Station**		
	86	-		CHXX0463 - LINHAI		
	MCC	**MNC**		**Mobile Brand**		

Figura 39 - Localização do IP 218.76.65.98

A Tabela 4 apresenta informações sobre os quatro endereços de IP responsáveis pela sondagem.

Tabela 4 - Localidades de IP's que acessaram o servidor HTTP

Endereço de IP	País	Região	Cidade
59.120.52.54	Taiwan	T'ai-Pei	Taipei
218.76.65.98	China	Hunan	Hunan
61.180.241.61	China	Heilongjiang	Harbin
219.246.184.76	China	Gansu	Lanzhou

Nenhum dos quatro endereços de IP apareceram nos outros ataques, mas do mesmo modo do servidor FTP, pode-se presumir que se trata de apenas um atacante devido ao fato da janela de tempo do ataque ser igual para todos os servidores.

4.1.5 Análise dos *Logs* Gerados pela Ferramenta Honeydsum

Como foi dito na sessão 3.1, a ferramenta Honeydsum gera dados estatísticos em páginas HTML com diversas tabelas e gráficos através do *log* criado pelo Honeyd.

Nesta sessão serão apresentados alguns dados obtidos pelo Honeydsum e uma análise será feita a partir desses dados.

Na Tabela 5, são apresentados todos os IP's que acessaram o *honeypot* por ordem de quantidade de conexões realizadas. O Honeyd registrou 47 IP's de origens diferentes e de todos esses, 4 se destacam por um número muito alto de conexões que são: 461 conexões do IP 218.61.35.151 que se localiza na China e aparece tanto nos *logs* do servidor POP3 quanto nos *logs* do servidor SMTP. 431 conexões do IP 71.249.229.215 que se localiza nos Estados Unidos e aparece apenas nos *logs* do POP3. 428 conexões do IP 208.115.238.196 que também se localiza nos Estados Unidos e só aparece nos *logs* do POP3 e por fim 363 conexões do IP 94.76.222.110 localizado no Reino Unido e também só aparece nos *logs* do POP3.

Na Figura 40 é apresentado o gráfico gerado pelo Honeydsum ilustrando a situação descrita a cima.

Tabela 5 - Top 50 *hosts* de origem gerado pelo Honeydsum

Top 50 Source Hosts		
Rank	**Source IP**	**Connections**
1	218.61.35.151	461
2	71.249.229.215	431
3	208.115.238.196	428
4	94.76.222.170	363
5	24.187.210.198	48
6	210.77.75.173	46
7	109.74.135.130	29
8	200.179.255.69	28
9	192.168.1.254	25
10	95.241.139.67	17
11	59.120.52.54	13
12	138.199.70.129	12
13	77.240.189.46	7
14	192.168.1.229	7
15	219.146.225.147	5
16	201.39.38.252	4
17	196.38.40.108	4
18	12.69.74.175	3
19	65.202.200.36	3
20	218.76.65.98	3
21	213.229.84.203	2
22	200.27.135.174	2
23	61.164.40.151	2
24	187.86.108.180	2
25	211.95.73.79	2
26	204.152.213.194	2
27	202.21.176.57	2
28	118.168.138.234	2
29	114.44.100.241	1
30	150.254.156.172	1
31	74.208.155.21	1
32	216.244.76.102	1
33	178.140.195.8	1
34	222.124.213.98	1

35	118.167.10.26	1
36	124.104.75.122	1
37	118.161.240.189	1
38	122.182.5.170	1
39	178.18.129.8	1
40	222.110.111.173	1
41	219.246.184.76	1
42	193.226.53.118	1
43	83.145.198.52	1
44	210.188.204.246	1
45	219.238.133.20	1
46	61.180.241.61	1
47	173.236.245.172	1

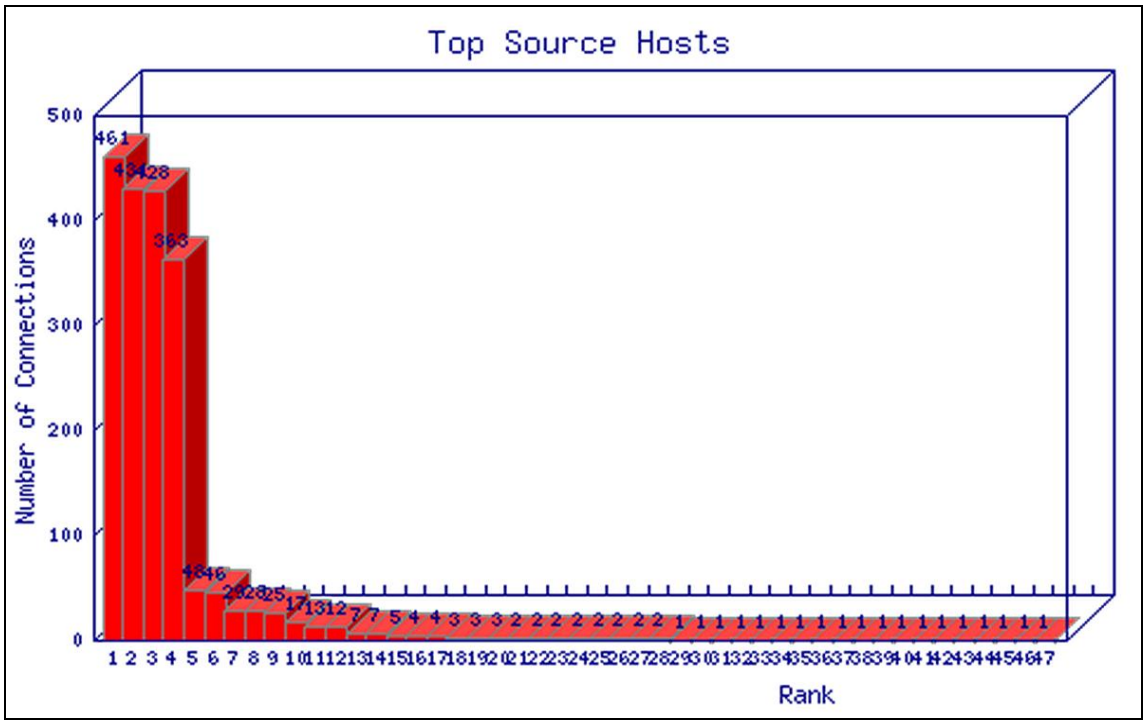

Figura 40 - Gráfico gerado pelo Honeydsum, *Hosts* x Conexões

Na Tabela 6, gerada pelo Honeydsum, mostra os recursos mais acessados do *honeypot*. Pode-se notar que o recurso mais acessado é o POP3 emulado na porta 110 explicando a quantidade de *logs* gerados por esse servidor. Existem também algumas conexões ICMP nas portas 80, 8, 5 e 22 que são geradas devido ao tráfego comum da rede em que se encontra o *honeypot* e trata-se apenas de ruídos.

Tabela 6 - Tabela dos recursos mais acessados

Top 10 Accessed Resources		
Rank	**Resource**	**Connections**
1	110/tcp	1826
2	25/tcp	56
3	21/tcp	43
4	80/icmp	18
5	8/icmp	14
6	1080/tcp	12
7	5/icmp	2
8	22/icmp	1

A Figura 41 ilustra o gráfico resultante dos dados apresentados na tabela 6 permitindo ver de forma ilustrativa a grande diferença da quantidade de conexões ocorridas no servidor POP3 com os outros.

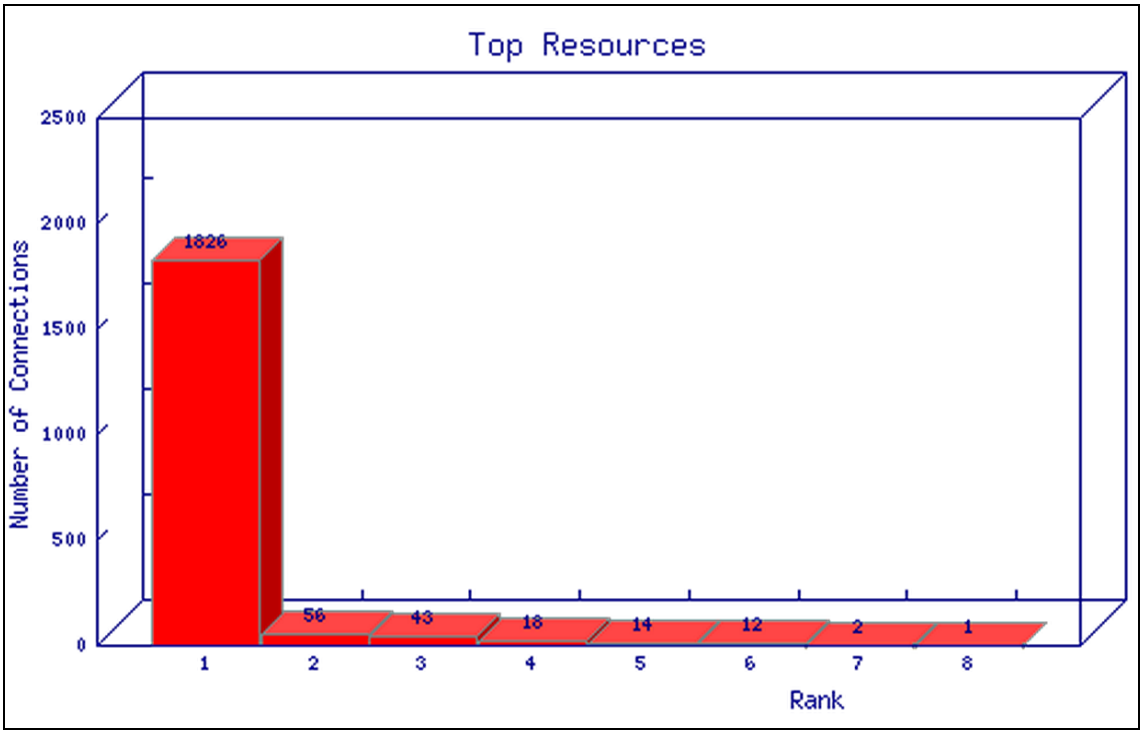

Figura 41- Gráfico gerado pelo Honeydsum, Conexões x Recursos

Na Tabela 7 é apresentada a quantidade de conexão que ocorreu em cada hora do dia. Pode-se ver uma maior atividade no período da madrugada, como 431 conexões à 1 da manhã e 462 conexões as 3 da manhã.

Tabela 7 - Quantidade de conexões realizadas em cada hora

Connections per Hour	
Hour	**Connections**
00:00	11
01:00	431
02:00	1
03:00	462
04:00	3
05:00	0
06:00	4
07:00	4
08:00	25
09:00	20
10:00	55
11:00	119
12:00	261
13:00	6
14:00	5
15:00	40
16:00	2
17:00	31
18:00	230
19:00	206
20:00	0
21:00	3
22:00	52
23:00	1

Na Figura 42 são ilustrados os dados representados na tabela. Dessa forma pode-se ver que a maior parte das conexões ocorreram no período da noite, contando que a noite começa as 18 horas e termina as 5.

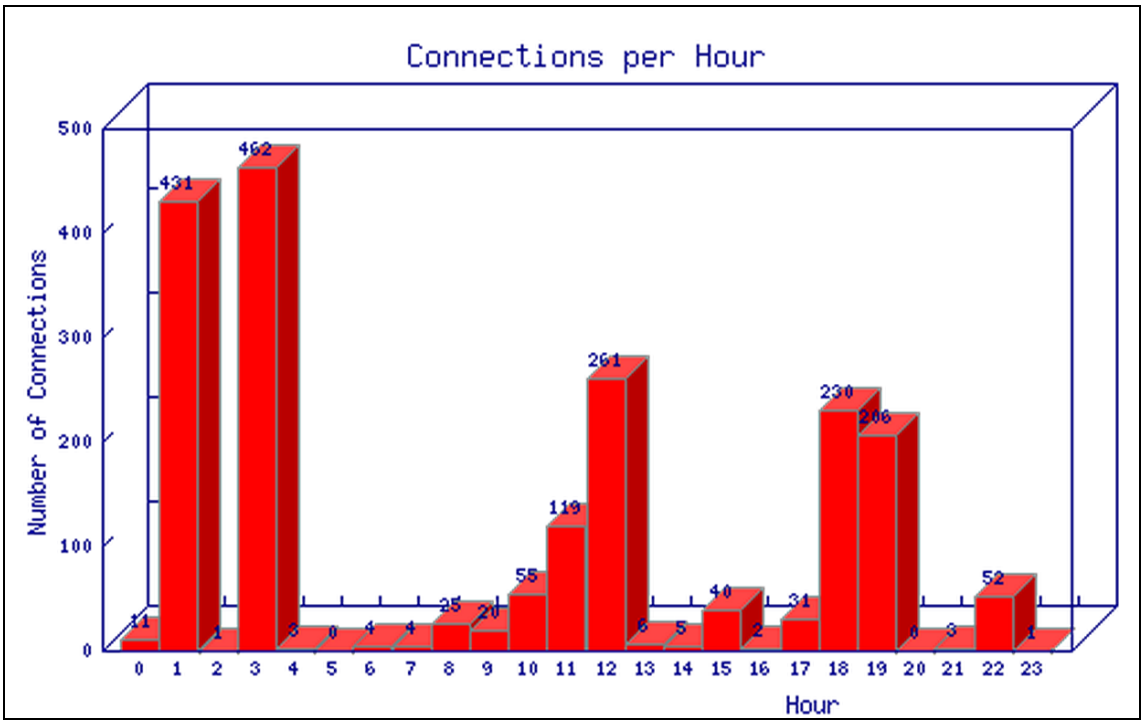

Figura 42 - Gráfico de conexões por hora

4.2 Resultados Obtidos com a *Honeynet*

No mesmo dia em que o *honeypot* foi colocado em funcionamento para se obter informações sobre o tráfego que chegava ao servidor do BCC a *honeynet* também foi colocada em funcionamento para obter informações sobre a rede interna do BCC.

Diferente dos resultados do *honeypot,* a *honeynet* não gerou nenhum *log* com informações sobre tráfego malicioso.

Antes do experimento era esperada uma maior quantidade de tentativas de invasões vindo da rede interna, pois mais de 150 alunos do curso de Ciência da Computação e mais algumas dezenas de alunos de outros cursos como Física e Matemática utilizam os computadores disponíveis na rede do BCC.

Os arquivos de *log* para os vírus emulados ficaram em branco, mostrando que não ouve nenhuma tentativa de se explorar essas brechas.

Os *logs* dos serviços do Honeyd só apresentaram mensagens informando que o serviço estava online.

O *log* principal do Honeyd apresentou umas poucas conexões que são geradas pelo tráfego normal da rede. Essas informações foram trabalhadas utilizando a ferramenta Honeydsum e serão apresentadas a seguir.

Tabela 8 - Quantidade de conexões

HONEYPOT'S CONNECTIONS	
Connection Counter	
Total	54
TCP	18
UDP	10
ICMP	26

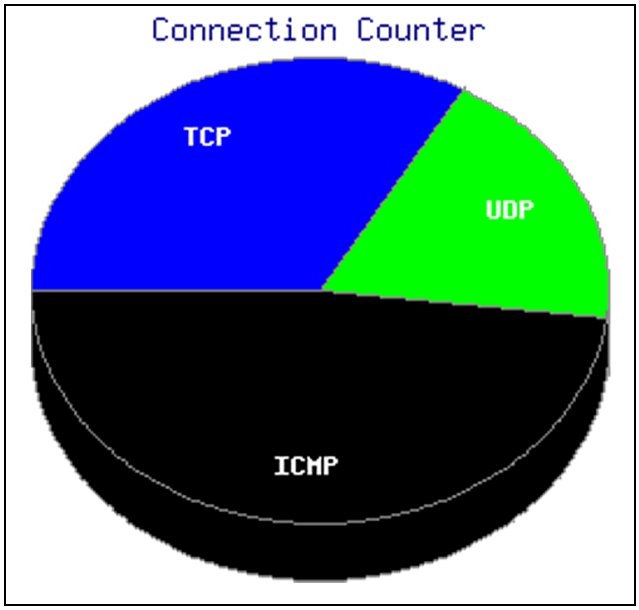

Figura 43 - Gráfico gerado pelo Honeydsum, tipos de conexões

A Tabela 8 e a Figura 43 mostram a quantidade de conexões TCP, UDP e ICMP que a *honeynet* teve durante o período de atividade. Pode-se observar que a maioria foram conexões ICMP que é bastante utilizado para informações sobre os *hosts* de uma rede.

Tabela 9 - Top 10 recursos acessados

Top 10 Accessed Resources		
Rank	**Resource**	**Connections**
1	8/icmp	26
2	856/tcp	15
3	137/udp	8
4	43948/tcp	3
5	711/udp	2

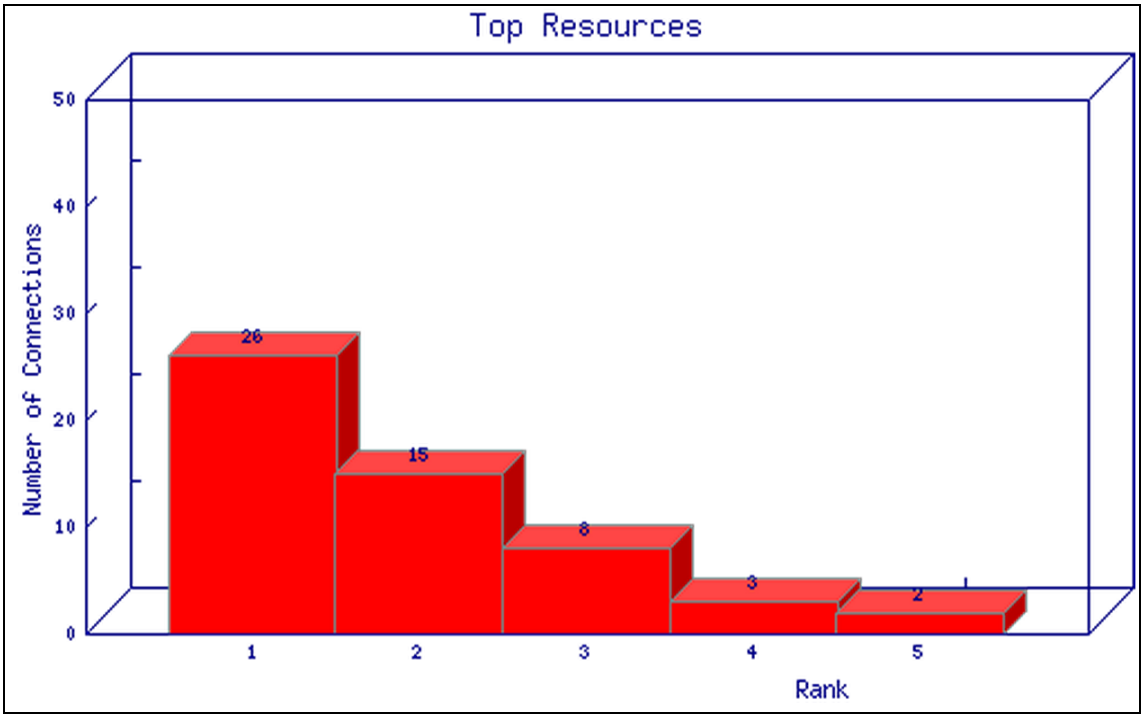

Figura 44 - Gráfico gerado pelo Honeydsum, 10 recursos mais acessados

Na Tabela 9 e na Figura 44 mostram as portas mais acessadas nos *hosts* da *honeynet* virtual. A porta 8 foi a mais acessada com 26 conexões ICMP e a porta 711 foi a menos acessada com 2 conexões UDP.

Tabela 10 - Conexões por hora da *honeynet*

Connections per Hour	
Hour	**Connections**
00:00	0
01:00	0
02:00	0
03:00	0
04:00	0
05:00	0

06:00	0
07:00	0
08:00	0
09:00	0
10:00	0
11:00	0
12:00	0
13:00	0
14:00	2
15:00	16
16:00	25
17:00	0
18:00	0
19:00	0
20:00	1
21:00	0
22:00	10
23:00	0

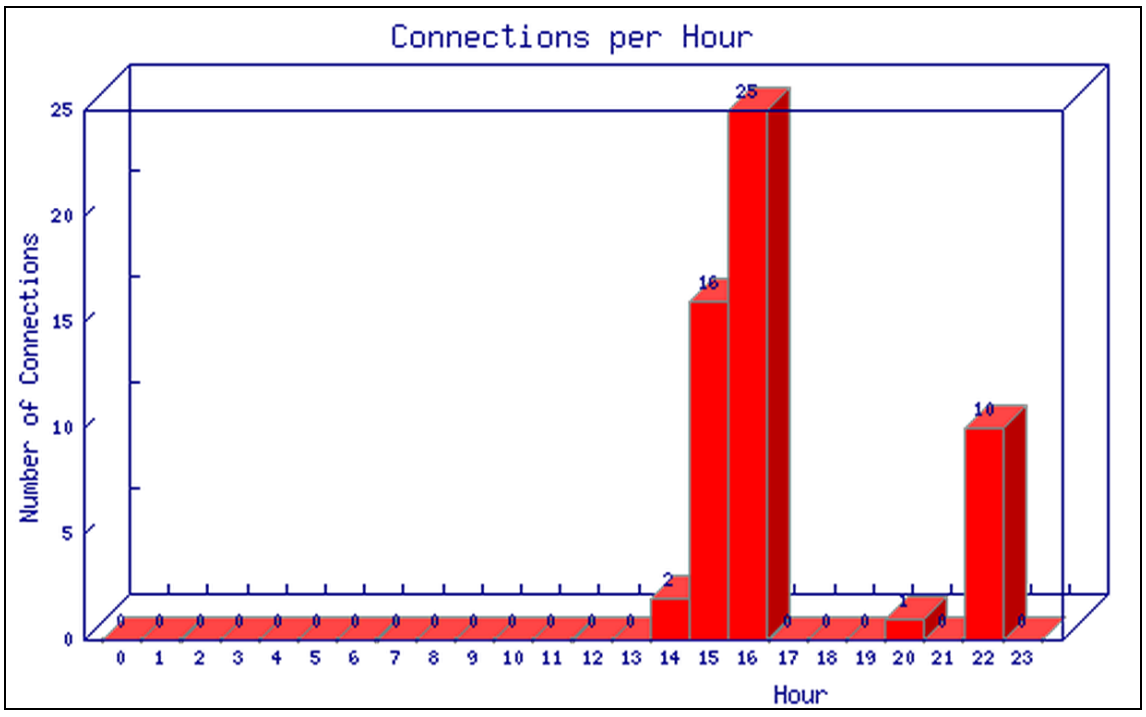

Figura 45 - Gráfico gerado pelo Honeydsum, Conexões x Hora

Por fim, a Tabela 10 e a Figura 45 ilustram a quantidade de conexões que ocorreram em cada hora do dia mostrando uma maior atividade no período da tarde das 14 horas às 16 horas.

5 Conclusões e Propostas Futuras

Neste trabalho foi proposto uma implementação de um *honeypot* de baixa interatividade e uma pequena *honeynet* dentro da rede do BCC com o intuito de realizar uma análise no tráfego gerado internamente e externamente, para então detectar tentativas de intrusão e analisar como foi realizado esse processo.

O *honeypot* implementado através de encaminhamentos de pacotes, proporcionando um alcance do tráfego externo aos serviços emulados pelo Honeyd obteve uma grande quantidade de dados gerados por tentativas de invasão.

Com esses dados foi possível descobrir algumas características do ataque, como os locais de onde partiram os ataques e os horários que os ataques ocorreram.

Outra questão que se deve levar em conta é o fato de que hoje em dia existem diversas ferramentas automatizadas para realizar alguns tipos de ataques, onde é possível um *cracker* programar uma certa faixa de endereços IP`s e então disparar esses ataques em todos os serviços que forem encontrados nos servidores que estão contidos nessa faixa de endereçamento.

Com os resultados obtidos nesse ataque pode-se supor que se tratou de apenas um atacante provavelmente utilizando alguma ferramenta de ataques automatizados devido à janela de tempo e a utilização dos mesmos servidores de origem para realizar ataques em serviços diferentes. Provavelmente o ataque partiu de um *cracker* que possui diversos *hosts* pelo mundo como escravos para então realizar ataques sem ser descoberto.

Existem formas para rastreá-lo, mas isso implicaria em conseguir contato com os administradores dos servidores de onde partiram os ataques, conseguir os *logs* dos horários em que os ataques ocorreram, analisar esses *logs* para então descobrir quem foi o invasor, ou então descobrir que os ataques partiram de outros servidores e assim repetir essa tarefa diversas vezes até chegar a alguma pessoa.

Provavelmente os *logs* foram apagados pelo invasor, então dificilmente ele seria encontrado. Dessa forma, se torna muito difícil capturar e punir pessoas que tentam roubar informações de outros servidores.

Por isso, verifica-se a importância de se pesquisar meios para proteger os servidores e estudar o modo como os atacantes trabalham, porque assim pode-se otimizar os formas de proteção e evitar que dados sejam roubados.

Para a *honeynet* poucos dados foram capturados, mas isso não tira seu valor. A rede do BCC está em expansão e cada dia mais pessoas irão utilizar seus recursos tornando cada vez mais difícil saber quem está acessando. Com uma *honeynet* seria possível capturar dados valiosos caso alguém de dentro da rede tentasse invadir outros *hosts* e assim tomar as devidas atitudes para aumentar a proteção do sistema.

Fazendo uma última análise quanto aos resultados obtidos pela *honeynet,* foi verificado que alguns dias da semana, no período da tarde, ocorrem aulas da disciplina de Sistemas Distribuídos e Programação Paralela, o que talvez explicasse o maior tráfego no período da tarde. Isso mostra que é possível capturar bastante tipo de tráfego utilizando um *honeynet* em uma rede interna.

A ferramenta Honeyd, se mostrou muito poderosa e flexível permitindo a criação e adaptação de serviços tornando-os mais reais de acordo com o ambiente de execução.

Para propostas futuras é possível que se trabalhe com o Honeyd de uma forma mais avançada, pois muitos recursos dessa ferramenta não foram utilizados devido à dimensão do trabalho como, por exemplo, a configuração de *honeypots* em tempo de execução onde pode-se configurar uma rede virtual completamente espelhada a uma rede real, ou seja, se um *host* for ligado na rede real, o Honeyd cria em tempo de execução um *host* virtual idêntico ao que foi ligado na rede real e caso um *host* da rede real for desligado, o Honeyd desliga seu correspondente na rede virtual.

Outra proposta que pode ser implementada é a utilização de uma ferramenta que trabalha em conjunto com o Honeyd, chamada de Honey Comb, que gera assinaturas de ataques a partir dos *logs* do Honeyd e então poderia desenvolver um meio de enviá-las para um IDS para aumentar seu poder de detecção.

Diversas implementações diferentes podem ser feitas dependendo da criatividade de quem está desenvolvendo. O ponto importante a se destacar é que os *honeypots* se mostram ferramentas bastante úteis para proteger os sistemas e para estudar os meios de ataque que estão sendo utilizados, aprendendo a se defender com os próprios atacantes.

6 Referências Bibliográficas

ALVAREZ T, Guerra e Defesa Cibernética [Internet]. Rio de Janeiro: SegInfo. Disponível em: <http://www.seginfo.com.br/guerra-e-defesa-cibernetica/>. Acesso em: 28 Set. 2010.

AMORIN, Rodrigo Diego Melo, Ataques Denial-of-Service. Centro de Informática UFPE. 2007 [Consult. 2011/04/18]. Capítulo 2 – Ataque DoS. Disponível em WWW: URL: <http://www.cin.ufpe.br/~fab/cursos/metodologia-graduacao/2006-2/monografias/rodrigo-diego.doc>

CERT, Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil. Disponível em: <www.cert.br/docs/whitepapers/honeypots-honeynets>.Acesso em: 12 Nov. 2010.

DIAS, Cláudia. Segurança e auditoria da tecnologia da informação. Rio de Janeiro: Axcel,2000. 218 p.

DUARTE, Otto C. M. B, JABOUR, E. C. M. G. Honeynets: Invasores, Ferramentas, Técnicas e Táticas. Rio de Janeiro: COPPE-Poli,GTA. 2004.

HONEYNET, Project The, KnowYour Enemy: Learning About Security Threats 2nd ed. Boston: Addison Weasley Professional. 2004. 800 p.

HONEYNETBR, Brazilian Honeynet Team, Disponível em: http://www.honeynet.org.br. Acesso em 20 de Abril de 2011.

ICSA, Internet Consortium Security Agency.Disponível em: <http://www.icsa.net>. Acesso em: 20 Set. 2010.

KUROSE, James F., ROSS, Keith W.,Redes de Computadores e a Internet: Uma abordagem Top-Down. 3.ed. São Paulo: Pearson Addison Weasley, 2006. 634 p.

MARCIANO, João Luiz e LIMA-MARQUES, Mamede. O enfoque social da segurança da informação. Ci. Inf. [online]. 2006, vol.35, n.3, pp. 89-98. ISSN 0100-1965. doi: 10.1590/S0100-19652006000300009.

MORIMOTO, Carlos E., Linux: Redes e Servidores. 2 ed. GHD Press e Sul Editores, 2006. 448 p.

NETO, Urubatan, Dominando Linux Firewall Iptables.1ed. Rio de Janeiro: Editora Ciência Moderna LTDA, 2004. 98p.

RAVEL. Laboratório de Redes de Alta Velocidade; Integrante do Programa de Engenharia de Sistemas e Computação - PESC da COPPE/UFRJ. Disponível em: < http://www.ravel.ufrj.br>. Acesso em 25 Set. 2010.

RFC, Request For Comments, Disponível em < http://www.ietf.org/rfc.html>. Acesso em : 20 Set. 2010

ROSEMANN, Douglas: Software Para Avaliação da Segurança da Informação de uma Empresa Conforme a norma NBR ISO/IEC 17799. 2002. 102 f. Trabalho de Conclusão de Curso – Universidade Regional de Blumenau, Blumenau. 2002.

SILVA, Gleydson Mazioli da, Guia Foca GNU/ Linux 2007. Disponível em:<www.guiafoca.org>. Acesso em 09/11/2010.

SPITZNER L, Honeypots: Traking Hackers.1 ed. Boston: Addison Wesley, 2002, 480 p.

PROVOS, Niels, From Botnet Tracking to Intrusion Detection. 1 ed. Boston Assison Weasley Professional, 2007, 480 p.

ULBRICH, Henrique C., DELLA Valle, James, Universidade Hacker. 6 ed. São Paulo: Digerati Books, 2009. 352 p.

TANENBAUM, Andrew S., Cmputer Networks. 4 ed. Editora Campus, 2008, 912 p.

7Anexos

7.1 Anexo I

```
config = creation | addition | delete | binding | set |
       annotate | route [config] | option
creation= "create" template-name | "create" "default" |
  "dynamic" template-name
addition= "add" template-name proto "port" port-number action |
  "add" template-name "subsystem" cmd-string ["shared"] ["restart"] |
  "add" template-name "use" template-name "if" condition
delete= "delete" template-name |
  "delete" template-name proto "port" port-number
binding = "bind" ip-address template-name |
  "bind" condition ip-address template-name |
  "bind" ip-address "to" interface-name |
  "dhcp" template-name "on" interface-name ["ethernet" cmd-string] |
  "clone" template-name template-name
set = "set" template-name "default" proto "action" action |
  "set" template-name "personality" personality-name |
  "set" template-name "personality" "random" |
  "set" template-name "ethernet" cmd-string |
  "set" template-name "uptime" seconds |
  "set" template-name "droprate" "in" percent |
  "set" <template-name> "maxfds" <number> |
  "set" template-name "uid" number ["gid" number] |
  "set" ip-address "uptime" seconds
annotate= "annotate" personality-name [no] finscan |
  "annotate" personality-name "fragment" ("drop" | "old" | "new")
route = "route" "entry" ipaddr |
  "route" "entry" ipaddr "network" ipnetwork |
  "route" ipaddr "link" ipnetwork |
  "route" ipaddr "unreach" ipnetwork |
  "route" ipaddr "add" "net" ipnetwork \\
             "tunnel" ipaddr(src) ipaddr(dst) |
  "route" ipaddr "add" "net" ipnetwork ipaddr \\
             ["latency" number"ms"] ["loss" percent] \\
             ["bandwidth" number["Mbps"|"Kbps"] \\
             ["drop" "between" number "ms" "-" number "ms" ]
proto = "tcp" | "udp" | "icmp"
action = ["tarpit"] ("block" | "open" | "reset" | cmd-string | \\
  "internal" cmd-string \\
  "proxy" ipaddr":"port )
condition = "source os =" cmd-string |
  "source ip =" ipaddr | "source ip =" ipnetwork |
  "time " timecondition
```

O anexo I ilustra a BNF (*Backus Naur Form*) do arquivo de configuração do Honeyd. Essa BNF foi tirada do livro: *Virtual Honeypots: From Botnet Tracking to Intrusion Detection*, escrito por Niels Provos, criador do Honeyd.