UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ – UTFPR
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

RODRIGO BARON

# IMPLEMENTAÇÃO DE UM GATEWAY DE SEGURANÇA COM SISTEMA OPERACIONAL DEDICADO

TRABALHO DE DIPLOMAÇÃO

MEDIANEIRA
2013

RODRIGO BARON

# IMPLEMENTAÇÃO DE UM GATEWAY DE SEGURANÇA COM SISTEMA OPERACIONAL DEDICADO

Trabalho de Diplomação apresentado à disciplina de Trabalho de Diplomação, do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas – da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Prof. MSc. Paulo Lopes de Menezes.

MEDIANEIRA

2013

Ministério da Educação
**Universidade Tecnológica Federal do Paraná**
Diretoria de Graduação e Educação Profissional
Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

## TERMO DE APROVAÇÃO

**Implementação de um gateway de segurança com sistema operacional dedicado**

Por

**Rodrigo Baron**

Este Trabalho de Diplomação (TD) foi apresentado às 08:20 h do dia 27 de março de 2013 como requisito parcial para a obtenção do título de Tecnólogo no Curso Superior de Tecnologia em Analise e Desenvolvimento de Sistemas, da Universidade Tecnológica Federal do Paraná, *Campus* Medianeira. Os acadêmicos foram argüidos pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. MSc. Paulo Lopes de Menezes
UTFPR – *Campus* Medianeira
(Orientador)

Prof. Dr. Neylor Michel
UTFPR – *Campus* Medianeira
(Convidado)

Prof. MSc. Cesar Angonese
UTFPR – *Campus* Medianeira
(Convidado)

Prof. MSc. Juliano Rodrigo Lamb
UTFPR – *Campus* Medianeira
(Responsável pelas atividades de TCC)

A folha de aprovação assinada encontra-se na Coordenação do Curso.

## *RESUMO*

Neste trabalho foi implementado um estudo de caso de um gateway de segurança baseado no sistema operacional OpenBSD com instalação do IDS/IPS snort; configurações de firewall, VPN, load balance e desenvolvimento de scripts de automatização visando torna-lo otimizado e especializado na proteção de redes de computadores.

Foram abordados aspectos sobre a segurança da informação tais como: custos de projetos envolvidos; dificuldades de aplicação; metodologia de analise. Foram abordados assuntos relacionados a tipos de ameaças virtuais, projetos com foco em segurança da informação, metodologia de avaliação da segurança de redes de computadores e boas praticas de configurações.

Foram feitos pesquisas do surgimento de vírus, hackers e sistemas de defesas como o packet filter, snort e o projeto OpenBSD.

***ABSTRACT***

*In this paper we implemented a case study of a security gateway based on the OpenBSD operating system installation with IDS / IPS snort, firewall, VPN, load balance and development of automation scripts in order to make it optimized and specialized in protecting computer networks.*

*Aspects of information security such as: project costs involved; implementation difficulties; analysis methodology. There were issues relating to types of cyber threats, projects focused on information security evaluation methodology of security of computer networks and best practices settings.*

*Searches were made of the emergence of viruses, hackers and systems defenses as packet filter, snort and OpenBSD project.*

## *LISTAS*

| | | |
|---|---|---|
| BSD | - | Berkeley Software Distribution |
| DDOS | - | Distribuid Denial Of Service; |
| DOS | - | Denial Of Service; |
| IDS | - | Intrusion Detection System; |
| IE6 | - | Internet Explorer 6. |
| NAT | - | Network Address Translation; |
| NIDS | - | Network Intrusion Detection Sistem; |
| OISSG | - | Open Information Systems Security Group |
| OSSTMM | - | Open Source Security Testing Methodology Manual |
| POC | - | Proof Of Concept; |
| SSL | - | Secure Sockets Layer; |
| VPN | - | Virtual Private Network; |

*SUMÁRIO*

# 1 INTRODUÇÃO

Devido ha necessidade de compartilhar informações privadas entre indivíduos surgiu a necessidade de criar-se métodos para prover segurança: autorizações, protocolos, criptografia, etc. Um indivíduo mal intencionado procura entender o funcionamento destes métodos de segurança, descobrir suas vulnerabilidades e desenvolver técnicas para violar as informações.

Perceber-se como é a evolução da tecnologia de segurança da informação onde observa-se que os sistemas estão cada vez mais sofisticados e da mesma forma os métodos para burlam os mesmos.

Desta forma, criar e manter redes e sistemas com um grau elevado de segurança da informação está se tornando cada vez mais complexo já que a segurança é implementada através de diversas ferramentas e tecnologias. A vulnerabilidade de um sistema pode comprometer a organização inteira.

## 1.1 OBJETIVO GERAL

Este TCC tem como objetivo principal o estudo das técnicas de segurança de redes e tem seus objetivos descritos a seguir.

Adaptar um sistema operacional baseado em software livre, para atender as necessidades de segurança básicas que uma rede deve possuir.

## 1.2 OBJETIVOS ESPECÍFICOS

- Apresentar conceitos e importância da segurança da informação;
- Apresentar as principais técnicas utilizadas pelos crackers;
- Apresentar o sistema operacional OpenBSD;
- Implementar um gateway de segurança com scripts, rotinas e interface de administração que sejam amigáveis ao usuário;
  Apresentar os resultados obtidos.

## 1.3 JUSTIFICATIVA

Desde o surgimento da telefonia e das redes de computadores consegue-se compartilhar informações estando em lugares distintos, para isso basta ter uma conexão entre os pontos. Desde então, indivíduos mal intencionados denominados crackers (Capitulo 2.5) tentam encontrar formas de obter estas informações para beneficio próprio.

Crackers são indivíduos com alto grau de conhecimento em determinadas tecnologias que procuram por falhas em sistemas e tentam se beneficiar. São comumente confundidos com Hackers pela mídia. Hacker é o profissional de segurança da informação que após encontrar uma falha comunica o responsável. Existem hierarquia e subdivisões destes indivíduos baseado em seus conhecimentos e objetivos.

Após encontrar uma determinada falha os crackers desenvolvem uma ferramenta para a exploração, essa ferramenta é chamada de exploit. Exploits são softwares cujo objetivo é explorar vulnerabilidades, isto é, tirar proveito de determinada falha para propósitos maléficos. É comum encontrar exploits que lhe permite executar comandos privilegiados, capturar informações de banco de dados ou de arquivos do sistema.

Com o surgimento da Internet milhões de pessoas passaram a compartilhar informações, no laser, estudo ou trabalho se “reunindo” virtualmente, da mesma forma grupos de crackers se reúnem estudando o funcionamento das organizações, sistemas e até mesmo do comportamento das pessoas envolvidas (conhecida como engenharia social) tentando

descobrir falhas e desenvolver exploits para invadir seus sistemas ou obtendo informações como senhas de conta bancária e outras informações sigilosas.

Como as falhas são constantes em muitas empresas e organizações, criou-se um *Cyber Black Market* (Mercado negro virtual) onde a venda de exploits, espionagem e alterações de dados são os produtos.

Existem várias técnicas para combater ataques de crackers, mas infelizmente são poucos os que utilizam, devido a complexidade de implementa-las corretamente ou pelo desinteresse da organização. Este trabalho tem a proposta de preparar um sistema operacional dedicado baseado no OpenBSD (sistema operacional que tem seu principal objetivo a segurança) para ser utilizado como um *gateway* seguro.

Gateway seguro é um sistema que tem softwares de segurança como firewall, VPS e IDS atuando como *gateway*, que pode resolver os problemas de segurança mais críticos de uma rede, com interface administrativa de simples configuração.

## 1.4 ESTRUTURA DO TRABALHO

Neste trabalho, a primeira parte aborda o desenvolvimento teórico das técnicas geralmente utilizadas por crackers.

Será desenvolvido um referencial teórico sobre o OpenBSD, suas vantagens, desvantagens e será mostrado o por que da sua escolha como sistema base.

As implementações utilizadas no sistema deverão satisfazer os seguintes serviços:

- Gateway;
- Firewall;
- Virtual Private Network (VPN) + Secure Socket Layer (SSL);

- Load Balance;
- Intrusion Prevention System (IPS);
- Intrusion Detection System (IDS);
- Honeypot;
- Failover;
- Scripts de configuração e interface administrativa.

Na primeira fase serão abordados os referenciais teóricos; na segunda fase será feito um estudo do código fonte do OpenBSD; na terceira fase será analisado e elaborado o projeto; na quarta fase será analisado possíveis melhorias; na quinta fase será a instalação dos serviços e desenvolvimento de scripts e interface de configuração; E por ultimo, aplicação dos testes de segurança (*Pentest*), Performance e estabilidade (stress).

# 2 REVISÃO BIBLIOGRAFICA

## 2.1 SEGURANÇA DA INFORMAÇÃO

"Os incidentes de segurança da informação vêm aumentando consideravelmente ao longo dos últimos anos e assumem as formas mais variadas, como, por exemplo: infecção por vírus, acesso não autorizado, *DOS*[1] e etc.." (Promon, 2006).

Juntamente com a difusão da Internet e o processo muitas vezes custoso para correção de vulnerabilidades são os principais fatores do aumento de ataques[2] em servidores.

Por este motivo, as corporações estão tratando a segurança com mais rigor e buscado terceirizar o processo de testes de segurança (Promon, 2006).

Pete (Herzog, Pete, 2010) diz em 2010 que segurança é uma separação entre o ativo que queremos proteger e a ameaça. Existem três formas para criar está separação:

1. Criar uma barreira física ou lógica entre eles;
2. Deixar a ameaça ineficaz (inofensivo);
3. Destruir a ameaça.

Para analisar o estado de segurança do ativo devemos avaliar o controle que possuímos das atividades e dos processos do mesmo (Herzog, Pete, 2010). Nesta analise utiliza-se 3 (três) elementos fundamentais:

- Visibilidade: E o elemento que encoraja o roubo. É visto como uma oportunidade pela a ameaça, aonde pode ser avaliado os benefícios e a diminuição do risco;

[1] *Denial Of Service* - Termo em inglês para negação de serviço.

[2] Ataque, diz respeito a uma tentativa de invasão de sistema computacional.

- Acesso: Como a segurança é a separação de uma ameaça de um ativo, devemos ter a capacidade de interagir com o ativo. Se houver uma interação direta com o ativo o acesso pode ser feito de inúmeros lugares diferente, aumentando a eficácia da ameaça;

- Confiança: A medida de confiança pode ser considerada um furo de segurança por considerar uma autenticação um meio para avaliar autenticidade. A utilização de métrica de confiança permitira medir quão valida é a relação de confiança (uma métrica usada é a permissão de acesso ao ativo apenas de um lugar especifico com uma autenticação especifica);

A segurança impacta diretamente nos princípios da informação: Confidencialidade, disponibilidade e integridade. Mas ela tem suas limitações, os mecanismos de proteção podem trazer a incapacidade ou indisponibilidade de realizar o trabalho. Estas limitações trazem problemas para manter a separação entre a ameaça e o ativo;

O conceito de segurança da informação tem sido apresentado sob diferentes formas por diferentes autores. Apenas para citar alguns exemplos, Promon (Promon, 2006) apresenta a segurança da informação como um caráter indisciplinar sendo o elemento central às atividades de apresentação e organização da informação. Para Anderson (Kent, Anderson, 2008), segurança da informação é um modelo de negocio em uma perspectiva tecnológica.

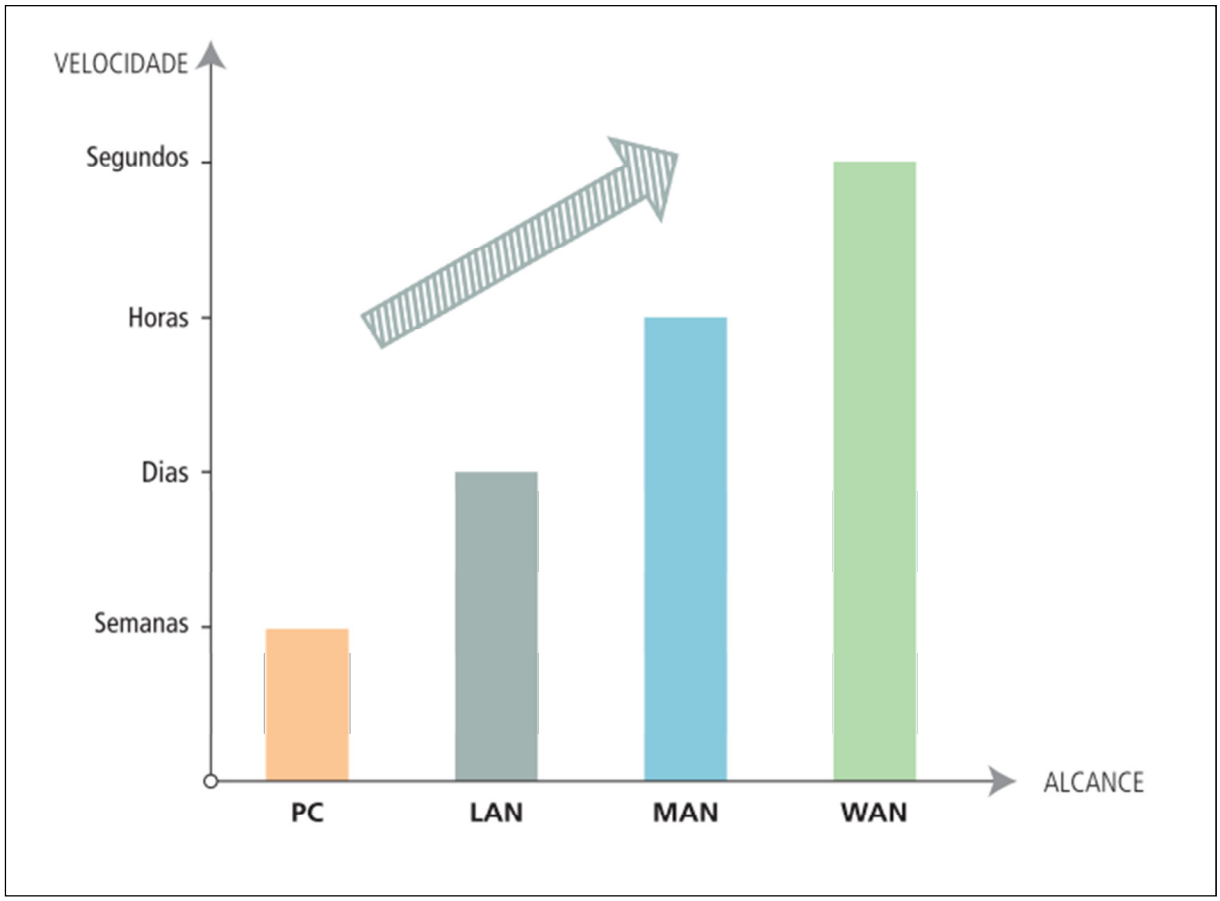
## *2.2 AMEAÇAS VIRTUAIS*

Uma das definições apresentadas para ameaça virtual é “evento ou atitude indesejável”, que compromete de alguma forma algum recurso. Muitos gestores alegam reconhecer que a segurança da informação é um fator importante em qualquer tipo de sistema, mas nem sempre é tratada com esta relevância (Segurança da Informação – uma abordagem social).

A tendência é que as ameaças à segurança continuam a crescer em ocorrência, velocidade e sofisticação (**Figura 2.1**).

"Não são apenas as ameaças externas que representam riscos a uma corporação. Os próprios funcionários representam alto risco quando mal-intencionados ou quando não conscientes dos riscos envolvidos na manipulação da informação" (Promon, 2006).

**Figura 2.1 – Crescimento de ameaças (Fonte: Promon, 2006)**




### 2.2.1 Tipos de Ameaças Virtuais

Os principais tipos de ameaças são feitos por códigos maliciosos tais como vírus, *worms*, *trojans* e outros que vêm trazendo prejuízos para as corporações (Promon, 2006), sendo elas as mais comuns:

#### *2.2.1.1 Vírus*

Um vírus de computador é um software que contem códigos maliciosos, infecta o computador e se autorreplica para outras aplicações. O vírus infecta uma determinada aplicação e fica dependente da aplicação para continuar o seu processo de disseminação. O primeiro vírus criado de fato foi em 1981 infectando disquetes da Apple II. O vírus se chamava **Elk Cloner** e mostrava na tela a seguinte mensagem:

"It will get on all your disks
It will infiltrate your chips
Yes it's Cloner!

It will stick to you like glue
It will modify ram too
Send in the Cloner!"
(Securimetric, 2001).

#### *2.2.1.2 Worms*

Um worm [3] de computador é um programa malicioso que se autorreplica. A diferença de um worm e de um vírus é que o worm não precisa de um programa hospedeiro, é um programa completo que normalmente se propaga pela rede de computador com maior velocidade e eficiência que o vírus (Weaver, Paxson, Staniford, Cunningham, 2005).

[3] Verme, referência a um programa de computador malicioso.

#### *2.2.1.3 Trojans*

Trojans (Trojan Horses), são aplicativos maliciosos que corrompem arquivos importante e instala um adware[4], spyware[5], keyloggers[6] e envia uma imagem da tela, arquivos pessoal e experiências de aplicativos da internet (usuários, senhas, sessões) para uma pessoa maliciosa (Weaver, Paxson, Staniford, Cunningham, 2005).

### 2.2.2 Ataques

Um ataque corresponde à concretização de uma ameaça, não necessariamente bem-sucedida. Uma vez que a geração de ataques é originada por pessoas, ainda que com o uso de recursos computacionais, a sua prevenção acaba sendo complexa por meios automatizados.

Diversos tipos de ataques em redes produzem resultados sobre a informação de forma padronizada. A mudança de rotas de acesso em redes e a falsificação de endereço de servidores computacionais são exemplos deste tipo de ataque. Desta forma, ferramentas de detecção e prevenção buscam por resultados compatíveis ao de um ataque, como é o caso de um software antivírus procurando por padrões de resultado de um vírus.

## 2.3 ABORDAGEM PARA SEGURANÇA DA INFORMAÇÃO

Uma das frases mais citadas no contexto da segurança da informação é que “uma corrente é tão resistente quanto seu elo mais fraco” (Marciano, João Luiz, Lima-Marques, Mamede, 2006). Sendo comum definir o usuário como o elo mais fraco da corrente da segurança da informação. Este capaz de infligir as regras de segurança.

---

[4] Programa que instala outros programas maliciosos

[5] Programa que recolhe automaticamente informações do computador infectado

[6] Programa malicioso que registra tudo o que é digitado e imagens da tela do computador e envia para a pessoa que o criou

Um dos aspectos que se tem mostrado extremamente relevante é o custo da segurança da informação. João Luiz e Mamede (Marciano, João Luiz, Lima-Marques, Mamede, 2006) mostram o custo relativo para a correção de falhas de segurança em softwares em cada estágio do processo de desenvolvimento, conforme ilustra a **Tabela 1**.

| **Estágio** | **Custo relativo** |
|---|---|
| **Projeto** | 1,0 % |
| **Implantação** | 6,5 % |
| **Testes** | 15,0 % |
| **Manutenção** | 100,0 % |

**Tabela 1: Custo relativo da segurança no desenvolvimento de software (Fonte: Marciano, João Luiz,. Lima-Marques, Mamede, 2006).**

Naturalmente, a segurança da informação tem custos. Contudo, sua ausência tem um custo ainda maior, seja econômico, seja social, na figura de uma imagem negativa perante o público.

A aplicação da segurança da informação em perspectiva da tecnologia da informação, está geralmente ligada as ameaças ou vulnerabilidades identificadas. Mesmo o reconhecimento destas ameaças e vulnerabilidades não seja trivial, após a ocorrência de um ataque, estes ataques não são devidamente relatados.

O impacto direto de incidentes de segurança nos resultados dos negócios faz com que as corporações mudem suas perspectivas tradicionais sobre segurança e abordam perspectivas de uma forma mais financeira encarando como um investimento que auxiliam a organização a atingir seus objetivos em negócio (**Figura 2.2**) (Promon, 2006).

**Figura 2.2 - Mudança de abordagem (Fonte: Promon, 2006).**


## 2.4 VULNERABILIDADES

Pode-se definir vulnerabilidade como uma falha em um sistema que permite usá-lo de uma forma não prevista pelo projetista (ANLEY, 2007). Ou seja, uma vulnerabilidade representa uma potencial falha, sendo um elemento relacionado à informação que é possível ser explorada por alguma ameaça, classifica-se desde um servidor ou sistema computacional até um usuário ou gestor de informações sensíveis como senhas.

Segundo (Malerba, César, 2010), pode-se dizer que vulnerabilidades surgem devido a 3 causas básicas:

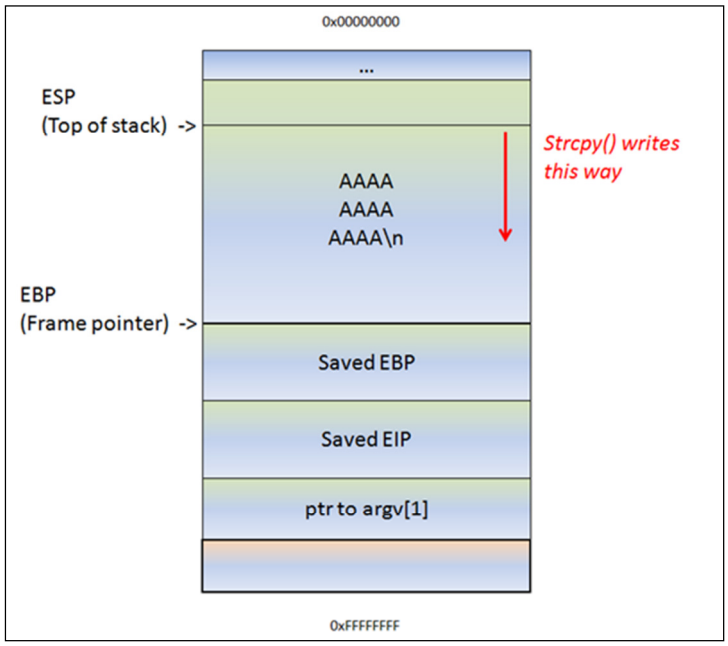
- Erro de implementação;
- Falha de design;
- Erros de configuração ou infraestrutura do sistema.

Falhas no controle de memoria de um sistema podem acabar resultando em vulnerabilidades gravíssimas e de difícil prevenção, pois dados e instruções não são diferenciados na memória (Malerba, César, 2010).

Como exemplo do funcionamento mais comum em vulnerabilidade será utilizado o *layout* básico da memória de um processador Intel x86[7] em um Sistema Operacional Windows, demonstrando uma vulnerabilidade de *buffer overflow* na função strcpy[8] da linguagem C.

A **Figura 2.3** mostra registradores (ESP, EBP e endereços de retorno EIP) em uma execução normal sendo este o estado do layout da memoria após ocorrer buffer overflow. Este tipo de vulnerabilidade pode ser explorado sobrescrevendo o endereço EIP (**Figura 2.4**) de retorno com um valor onde se encontra o código malicioso (Eeckhoutte, Peter Van, 2009).

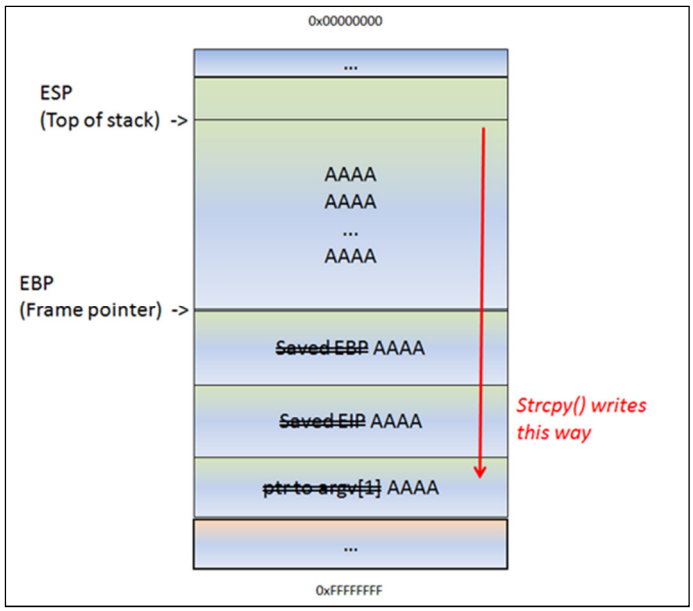
**Figura 2.3 – Memoria antes de ocorrer overflow (Fonte: Eeckhoutte, Peter Van, 2009).**




[7] Arquitetura de processadores de computadores baseados no Intel 8086.

[8] Função de copia de strings em memoria.

**Figura 2.4 – Memoria depois de overflow (Fonte: Eeckhoutte, Peter Van, 2009).**




Softwares modernos e populares são os maiores alvos de exploração apartir de uma falha. Grandes empresas como Google, Facebook e Apple pagam por vulnerabilidades encontradas em seus produtos. Como o Google, em 2012 pagou 1.000.000,00 U$ (dólares) por vulnerabilidade no Google Chrome (Computerworld, 2012).

## *2.5 HACKERS E CRACKERS*

Hacker é um termo com várias definições.

Termo Hacker para o dicionário Priberam da Linguagem Portuguesa (Priberam Informática, 2012) - “Individuo que utiliza ilegalmente a sistemas computacionais”.

Termo Hacker para o dicionário Aurélio (Dicionário do Aurélio, 2013) - “Programador com gênio para dominar e alterar programas e equipamentos de computação e

teleprocessamento, e capaz de invadir à distancia outros computadores, utilizando ilegalmente os recursos do modem".

De forma geral, Hacker é alguém que invade um sistema de computador ou consegue ver arquivos pessoais sem autorização. Nos primeiros anos de utilização do termo hacker, a maioria dos hackers eram estudantes e geeks [9] de computador que tinham como perfil solitário brilhante (McCarthy, Linda,. Weldon-Siviy, 2010).

Também denominado de hacker o adolescente imaturo que usava fáceis técnicas para explorar vulnerabilidades de segurança na Internet, e que após anos passou a ser denominado de Script kiddies[10].

Este é o mesmo significado do termo Cracker: Programadores mais experientes gostam de se chamar de hacker pela utilização de métodos não tradicionais que sobrepõem bug funcionais (IE6).

Nos últimos anos, foram criados os termos: Black Hats, Gray Hats e White Hats. Desta forma o Hacker e distinguido pelas suas ações.

### 2.5.1 Black Hats

Black Hat ("Chapeu Preto") seria o equivalente ao "Cracker" porem, "Cracker" é associado a um tipo bem especifico de pessoa, aquelas que desenvolvem formas de quebrar códigos de softwares e/ou sistemas (Marques, Eduardo, 2010).

Também são conhecidos como Hackers mal intencionados, isto é, criam e enviam vírus, worms[11] e exploits. Invadem sistemas de computador, roubam dados e basicamente, cometem crimes eletrônicos.

---

[9] Uma gíria inglesa que define pessoas obcecadas com tecnologia

[10] Hacker inexperiente

### 2.5.2 Gray Hats

É o Hacker que não rouba, mas invadem sistemas de computador sem autorização. Afirmam que a invasão é apenas para conhecimento, um treinamento em segurança da informação (Rouse, Margaret, 2007).

### 2.5.3 White Hats

É o termo usado para especialistas em segurança da informação. Costumam utilizar as mesmas ferramentas que os Black Hats utilizam, mas de forma ética ( *Ethical Hacking* [12] ). Criam POCs [13] para provar os riscos das falhas encontradas.

## 2.6 PENTESTING

Pentesting[14] (Penetration testing) é uma metodologia designada a avaliar o nível de segurança de uma rede de computadores, aplicações e sistemas (Open Information Systems Security Group, 2005). Ela consiste em três passos:

1. **Planejamento e avaliação** – Esta fase consiste em montar um projeto e definir escopos;
2. **Assesments** – Esta é a fase que ocorre os testes de segurança propriamente dita;
3. **Reporting** - Após completar os testes de segurança definidas no escopo é feito um relatório descrevendo os detalhes dos resultados do processo.

---

[11] Do português verme, programa auto-replicante

[12] Processo de utilização de ferramentas e metodologias de segurança para testar e melhorar a segurança

[13] Do inglês, Proof of concept (Prova de conceito)

[14] Do inglês – Teste de intrusão, uma metodologia de avaliação da segurança de um sistema.

O foco deste projeto se encaixa na fase de Assesments, por ser a fase em que é realizado os testes de segurança.

### 2.6.1 Assesments

O processo de teste apresenta grande importância por que pode apresentar informações priorizadas do sistema a ser testado. Este processo consiste nos seguintes passos **Figura 2.5**:

1. Aquisição de informações;
2. Mapeamento da rede;
3. Identificação de vulnerabilidades;
4. Teste de intrusão;
5. Ganhando acesso e elevação de privilégios;
6. Enumeração de possíveis vitima;
7. Comprometimento de usuários remotos;
8. Garantindo acesso;
9. Cobrindo rastros.

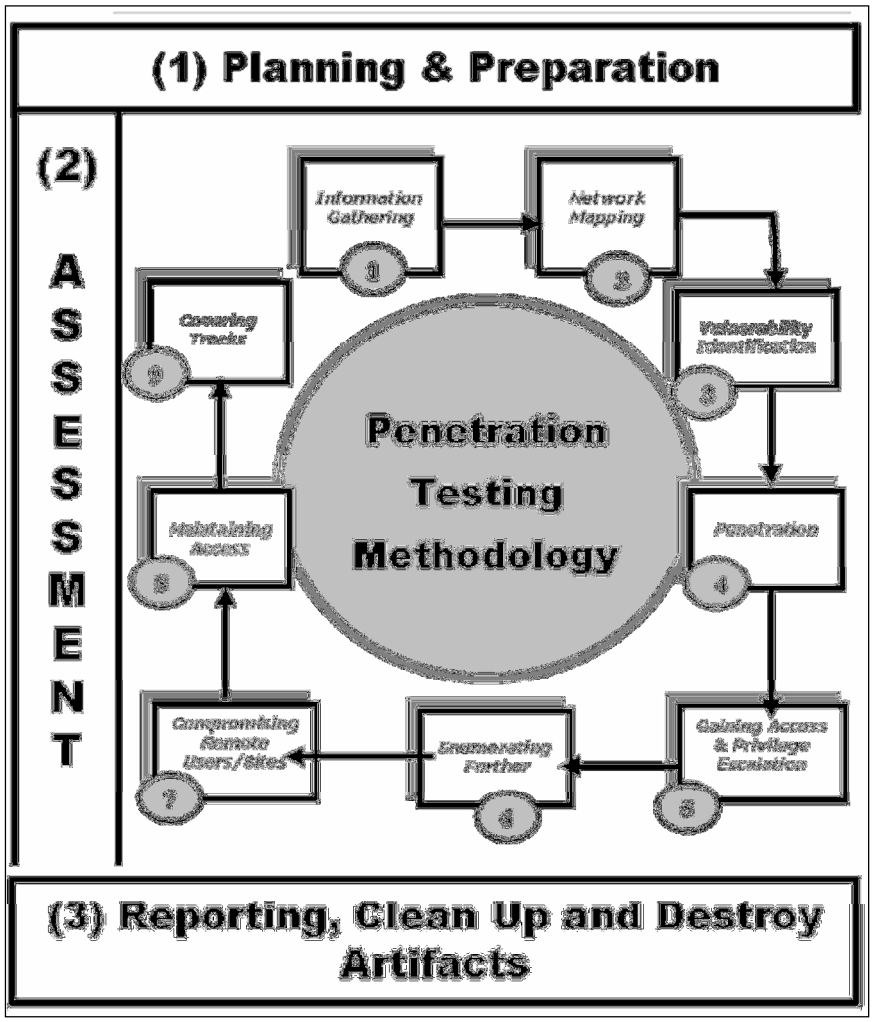
**Aquisição de informação**

Nesta etapa é usada principalmente a Internet para procurar informação sobre o alvo usando varias técnicas e métodos. Deve ser entendido o processo e os recursos que o alvo utiliza: Serviços de cartão de crédito entre outros serviços terceirizados ou internamente utilizados.

Para adquirir informações nem sempre é preciso ter algum contato com o sistema alvo. Muitas informações podem ser adquiridas de fontes publicas na internet e/ou organizações que publicam características e relatório de outras organizações.

A **Figura 2.5** mostra o ciclo de vida de um teste de intrusão.

**Figura 2.5 - Sumária de execução de teste de intrusão (Fonte: Open Information Systems Security Group, 2005).**




Está é uma etapa extremamente importante e requer um grande investimento de tempo, pois identifica pontos críticos que muitas vezes é suscetível a ataques. (Open Information Systems Security Group, 2005).

**Mapeamento de rede (*Scanning*)**

Seguindo a primeira etapa do teste de intrusão, quando todas as possíveis informações sobre o alvo são adquiridas, uma técnica chamada "footprint"[15] é utilizada para avaliar os recursos da rede em questão das informações obtidas na primeira etapa. Muitas ferramentas e aplicações podem ser utilizadas nesta etapa para:

1. Encontrar hosts;
2. "Varrer" portas de rede;
3. Mapear o perímetro de rede (roteadores, firewalls);
4. Identificar serviços críticos;
5. Identificar os sistemas operacionais.

**Identificação de vulnerabilidades**

Após obter um mapeamento completo, procura-se falhas nos serviços encontrados, isto é, após a identificação dos serviços e do sistema operacional, pode-se procurar por falhas conhecidas (CVE) ou até mesmo exploits prontos para serem usados. No caso de sistemas críticos uma auditoria do(s) códigos/aplicativos dos serviços encontrados pode ser realizada.

**Teste de intrusão**

Após a identificação de vulnerabilidade, é comum nesta etapa desenvolver um *POC* (*Proof Of Concept*, prova de conceito) e/ou uma ferramenta para auxiliar a exploração.

**Ganhando acesso e elevação de privilégios**

Ganhando acesso refere-se a obter acesso há contas privilegiadas utilizado contas não privilegiadas. Existem varias formas de se obter acesso, entre elas está:

1. Descobrindo combinação de usuário e senha (bruta force);

[15] Do inglês – Impressão digital

2. Descobrindo pontos de acesso com senhas “em branco” ou com senhas padrões, exemplo: usuário: admin; senha: admin;
3. Descobrindo serviços públicos que permitem operações privilegiadas;

Elevação de privilégios refere-se ao sistema que está comprometido de forma intermediaria, isto é, após a etapa de teste de intrusão foi identificado uma vulnerabilidade e está vulnerabilidade foi utilizada para ganhar acesso a contas com privilégios intermediários. Está conta é possibilita mapear serviços e aplicativos locais, e buscar por vulnerabilidade para obter acesso privilegiado (SYSTEM, ROOT).

**Enumeração de possíveis vitima**

Uma avaliação para descobrir senhas de usuário, geralmente criptografados, e por fim enumerando os usuários que ficaram com as suas credencias do sistema expostas. Geralmente as técnicas utilizadas nesta etapa são:

1. Obter arquivos de senhas para crackear (quebrar) em modo off-line;
2. Obter senhas utilizando sniffers de rede;
3. Obter cookies com isso explorar sessões para obter senhas.

**Comprometimento de usuários remotos**

“Uma maquina com uma falha simples pode comprometer a rede.” (Open Information Systems Security Group, 2005).

E normalmente a comunicação entre um usuário/site remoto com uma rede corporativa é provida de autenticação e criptografia usando tecnologias como VPN, para assegurar que os dados que trafega sobre a rede são confiáveis. Mas, isso não garante que a comunicação entre os dois pontos pode ser comprometida. Nesta etapa é avaliado como o usuário utiliza os serviços corporativos remotamente.

**Garantindo acesso**

Após avaliar como o usuário utiliza os serviços da corporação remotamente pode-se acessar a maquina "se escondendo" (covert channels[16]) através da comunicação deste serviço ou instalar um software que abre um serviço de acesso remoto (backdoor[17]).

**Cobrindo rastros**

Todos os sistemas gravam informações (log, histórico e etc..) sobre as ações ocorridas no sistema. Esta ultima etapa resume-se destruir as informações que o sistema coletou sobre as etapas anteriores.

## 2.7 EXPLOIT

Em segurança da informação exploit é um software que se aproveita de vulnerabilidades de outros softwares. Isto é, faz com que o software alvo seja inoperável (DOS[18]), faz com que o software alvo execute códigos maliciosos ou se beneficia do software alvo. Existem duas categorias de exploit:

---

[16] Serviço comprometido que faz o uso de criptografia.

[17] Software que permite acessar um computador comprometido sem o conhecimento do administrador.

[18] *Denial of Service* do inglês, negação de serviço.

**Local** – Explora uma vulnerabilidade do sistema para elevar o nível de privilegio no sistema ou ganhar acesso a dados de outros usuários;

**Remote** – Explora uma vulnerabilidade em um sistema remoto para ter resultando em acesso ao sistema.

## *2.8 BSD*

Quando AT&T criou o UNIX por volta de 1970 o UNIX se propagou nas corporações de telefonia, mas a AT&T não conseguia transformá-lo em um produto comercial porque na época o UNIX não se apresentava muito bem comparado a os demais sistemas operacional. Como tal, a AT&T licenciou o software UNIX e seu código fonte para as universidades. Inicialmente isso foi uma boa jogada comercial para a AT&T, ganhava dinheiro com as licenças e cientistas da computação. Como as universidades não mantinham altos custos com licenças, os alunos do *Computer Science Research Group* (CSRG, grupo de pesquisa da ciência da computação) da Universidade da Califórnia, Berkley, conseguiram entender como o sistema operacional funcionava. Em 1980 CSRG criou a licença BSD (*Berkeley Software Distribuition*) e começou distribuir o sistema operacional compatível com UNIX sobre a licença BSD para qualquer pessoa. (Lucas, Michael W., 2003).

A licença BSD sofreu várias modificações desde seu surgimento, na sua versão atual a licença permite que o software possa ser incorporado a produtos proprietários.

Desde então, surgiu derivações da distribuição do sistema operacional BSD sendo as principais FreeBSD, NetBSD e OpenBSD.

### 2.8.1 FreeBSD

FreBSD é a primeira distribuição derivada do BSD (também conhecido como 386BSD) que foi distribuída através de CDROM na sua versão 1.0 em 1993. Com o intuito de ser o mais robusto sistema operacional de rede (utilizado para prover serviços de rede). Os maiores clientes que usaram o FreeBSD:

- YAHOO! – Com mais de meio bilhão de páginas acessadas por dia;
- HOTMAIL – Serviço da Microsoft que disponibiliza e-mails gratuitos com mais de milhões de usuários.

O FreeBSD é tecnicamente mais simples, isso graças ao sistema de pacotes chamado **ports** (que foi implementado no NetBSD e OpenBSD). Seu objetivo é prover um sistema operacional de auto desempenho e simples de usar.

### 2.8.2 NetBSD

O NetBSD seguiu a linha de desenvolvimento do FreeBSD, porem, o FreeBSD é especialmente utilizado em plataformas i386 ou qualquer outra superior da Intel. O NetBSD tem o objetivo de ser portável para qualquer plataforma (SPARC, Intel, Compaq's Alpha, Apple Macintosh e etc ...), alta estabilidade e especialmente utilizado para pesquisas.

### 2.8.3 Projeto OpenBSD

O projeto OpenBSD foi fundado por Theo de Raadt, que foi desenvolvedor do NetBSD por vários anos. Após um desacordo com outros desenvolvedores do NetBSD sobre como o sistema deve ser desenvolvido. Theo fundou o projeto OpenBSD, uma distribuição derivada do NetBSD com foco em segurança da informação.

Atualmente o sistema operacional OpenBSD é baseado no sistema BSD com uma atenção fanática por segurança, usabilidade e liberdade. É suportado pelas arquiteturas mais conhecidas (Intel, AMD, Spark) e tem uma ótima documentação.

A cada release do OpenBSD é feita uma severa auditoria e correção preventiva do seu código à procura de erros de programação e principalmente à procura de vulnerabilidades. Quando uma falha é encontrada, imediatamente é aberta uma chamada para os desenvolvedores reproduzir a falha e corrigirem o mais rápido possível e por fim disponibilizar o patch de correção.

O sistema operacional OpenBSD implementa recursos de proteção à memória e ao vazamento de informações, com funções que não permitam execuções de códigos na memória manipulada por softwares. Implementa também recursos de randomização de alocação de memória impossibilitando alocação de memoria não previsível. Também é implementado politicas de segurança como "o mínimo necessário" (Promon, 2006), uma politica aonde o menor numero de recursos e serviços vem habilitado por padrão, os demais serviços devem ser habilitados manualmente; Separação de privilégios de serviços do sistema e "enjaulamento" nativo dos mesmo, fazendo com que serviços críticos e expostos à internet mesmo sendo vulneráveis não permitam o acesso de root.

Além do sistema operacional OpenBSD o projeto OpenBSD tem outros projetos com destaque em segurança da informação: OpenSSH, OpenSSL e Packet Filter.

- OpenSSH – um software que roda como serviço que providencia sessões criptografadas utilizando o protocolo ssh fornecendo acesso a uma shell segura através da rede;
- OpenSSL – um implementação dos protocolos ssl e tls que fornece vários esquemas de criptografia entre elas as mais importantes são: AES, MD5, RSA;
- Packet Filter – um firewall de alta segurança que lhe fornece o controle total sobre o trafego na rede.

## 2.9 FIREWALL

Um firewall basicamente é um software que impede usuários não autorizados obtenha acesso a uma rede ou computador. Considera um firewall uma barreira que verifica informações aplicando regras que podem permitir ou não que essas informações passem.

### 2.9.1 Packet Filter

*Packet Filter* é um firewall originalmente desenvolvido durante um evento do OpenBSD em 2001 por Daniel Hartmeier e outros desenvolvedores do OpenBSD. O resultado foi para o sistema base do OpenBSD 3.0 com o nome de NetFilter. Porem, o NetFilter não estava sob a licença BSD e foi removido do sistema base. Neste mesmo ano, Daniel Hartmeier desenvolveu uma versão melhorada do NetFilter, que aplicava proteções na stack, desta vez sob a licença BSD. Este então ficou com o firewall padrão do OpenBSD 3.0 (Openbsd, 2012).

O Packet Filter é um firewall "*Statefull*" (firewall de estado), que consiste em criar uma seção para cada conexão, desta forma possui a capacidade de identificar o protocolo dos pacotes transmitidos e "prever" as respostas legítimas. Opera inspecionando pacotes, protocolos, conexões, portas e serviços. Também opera interfaces de rede, endereço de origem e de destino, baseando de onde o pacote veio e para onde vai, e em qual porta de destino. Desta forma é determinado o que fazer com o pacote. Isto é chamado de "filtro em nível de aplicação" (Promon, 2006).

Até 2001 o Packet Filter passou por varias revisões, entre elas, a maioria foi para adicionar novas funcionalidades e as demais para maior estabilidade. O Packet Filter se encontra no OpenBSD 5.1 de uma forma estável capaz de:

- Classificar pacotes baseado em família de endereços, porta de origem e de destino, tipo de pacote e endereço de origem e destino.
- Redirecionar o trafico de pacotes para host de destino que não esteja diretamente conectado (NAT).
- Fazer *Load Balacing*.
- Fácil manutenção.

O packet filter se diferencia dos demais firewalls por ser fácil de configurar, rápido e estável. Possui arquivos de configurações e consegue operar nas 5 camadas do protocolo TCP/IP.

### 2.9.2 NAT

Firewalls modernos possuem implementações de NAT (*Network Address Translation*), conceito sobre rede interna e rede externa (RFC 1918, 1996).

Neste contexto NAT é uma solução temporária para a comunicação de hosts que não tem acesso direto a Internet, consiste em duas partes. A primeira é um mecanismo designado a reescrever endereços de pacotes. A outra parte designa a reencaminhar estes pacotes para o endereço que foi sobrescrito.

### 2.9.3 Gateway

Gateway é um sistema operacional intermediário que é responsável por outros sistemas operacionais, isto é, todo o trafego de dados destinado a um determinado sistema operacional que o gateway é responsável, todos os dados são passados e avaliados pelo gateway. Quando se trata de gateway, a perspectiva de segurança é outra. Deve ser avaliado para quem é permitido determinado tipo de trafego de dados.

## 2.10 VPN E SSL

VPN (Virtual Private Network, Rede Privada Virtual) é uma rede privada normalmente utilizada por organizações construída em cima de uma rede publica. VPNs não

são redes seguras, porem pode ser utilizado com tunelamento e criptografia que fornecem confidencialidade, autenticação e integridade que garantem privacidade e segurança.

SSL (Secure Socket Layer, socket em camada de segurança) é uma das criptografias mais usadas atualmente na Internet. Trabalha em nível de aplicação criptografando os segmentos que passam da camada de Aplicação para a camada de transporte do modelo OSI. A criptografia utilizada para a VPN foi IPSEC/EAP.

## 2.11 IDS

IDS (*Intrusion Detection System*) são técnicas usadas para detectar atividades maliciosas. Existem dois tipos de IDS: Network IDS (NIDS) e Host IDS (HIDS):

- NIDS – sistema de detecção de intrusão que captura pacotes de dados da rede e faz uma analise;
- HIDS - sistema de detecção de intrusão que captura log de aplicativos procurando por atividades maliciosas.

IDS pode trabalhar de duas formas: baseado em assinaturas de códigos maliciosos ou anomalias encontradas na rede. Quando utiliza-se da técnica de detecção por assinaturas, é gerado logs e alertas, quando for baseado em anomalias o IDS consulta a regra para cada situação.

### 2.11.1 O IDS SNORT

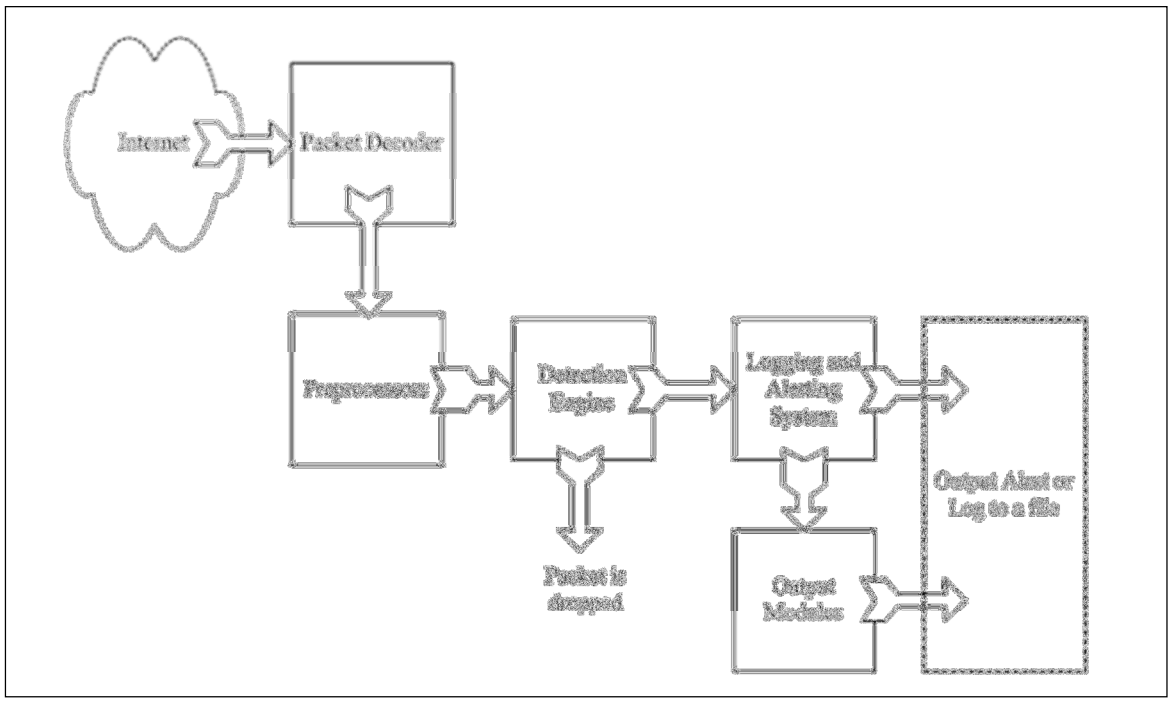
SNORT é um sistema *open source* (GNU GPL) que implementa as funcionalidades de IDS. Com ele é possível implementar regras de segurança para pacotes contendo assinaturas maliciosas ou anomalias na rede.

Segundo Rehman, Rafeeq Ur (*Intrusion Detection Systems with* SNORT), o SNORT é logicamente dividido em múltiplos componentes. Cada componente é feito para detectar um tipo particular de ataque. Abaixo segue a lista de componentes do SNORT:

- Packet Decoder;
- Preprocessors;
- Detecting Engine;
- Loggin and Alerting System;
- Output Modules.

A **Figura 2.6** mostra como são organizado estes componentes.

**Figura 2.6 – Arquitetura modular snort (Fonte: Bukac, Vít, 2010).**




- Packet Decoder - O **Packet Decoder** captura os pacotes de todas as interfaces de rede conectadas e prepara para ser processada pelo componente **Preprocessors**.

- Preprocessor - O **Preprocessor** organiza e modifica os pacotes de dados antes de passar pelo componente **Detection Engine**. A modificação dos pacotes de dados é feita em busca de anomalias nos cabeçalhos dos pacotes.

- Detection Engine - O **DetectionEngine** é componente mais importante para o SNORT. Ele é responsável por detectar atividades maliciosas nos pacotes de dados, para isso, o componente consulta as regras configuradas. Cada regra é lida em uma estrutura interna, se alguma regra for aplicada ao pacote, determinada ação é tomada. Porem na maioria dos casos o pacote é descartado.

- Loggin and Alerting System - Dependo dos resultados do componente **DetectionEngine** é gerado logs e alertas. Os logs são arquivos de texto simples com todas as informações capturadas do componente **DetectionEngine**. Todos os arquivos de log gerado pelo snort se encontra no diretório **/var/log**.

- Output Module - São como plug-ins, que captura os logs gerados são formatados para uma melhor compreensão, cada arquivo de log tem uma formatação diferente.

## 2.12 HONEY POTS

**Honey Pots** são sistemas usados para enganar Hackers mostrando deliberadamente vulnerabilidades conhecidas. Porem estas vulnerabilidades são falsas. Desta forma quando um Hacker encontra um **Honey Pot**, ele gasta um tempo considerável para tentar invadir. Durante este tempo, podemos registrar suas atividades e avaliar o que pode ser melhorado na segurança do sistema (*Intrusion Detection Systems with SNORT*).

# 3 MATERIAL E MÉTODOS

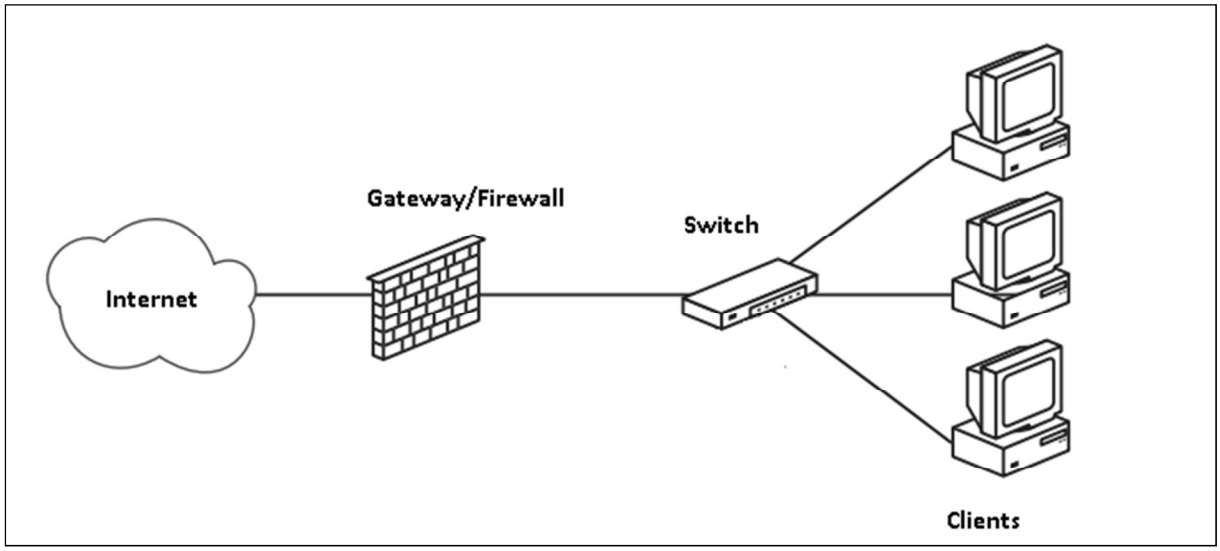
## 3.1 APRESENTAÇÃO DO ESTUDO DE CASO

No estudo de caso foi utilizado o OpenBSD para implementar o gateway de segurança desenvolvido com foco voltado para a segurança, tornando-o altamente indicado para este tipo de solução. Com o objetivo de utilizá-lo como um gateway seguro e implementar características de segurança para uma rede de computadores, foram feitas algumas configurações que serão apresentadas adiantes.

Para este estudo de caso foram criados três cenários como descrito a seguir:

**Cenário 1** – Neste cenário foi criada uma estrutura conforme ilustrada na **Figura 3.1**, que apresenta uma topologia de rede com ou mais clientes interligados através de um switch, e isolados da Internet através de um gateway/firewall, posição posteriormente assumida pelo gateway de segurança deste projeto, a fim de dar proteção aos clientes deste cenário, ao mesmo tempo em que restringe o acesso dos mesmos a apenas alguns serviços.

**Figura 3.1 – Modelo de cenário de rede, cenário 1.**



**Cenário 2** – Neste cenário foi criada uma estrutura conforme ilustrada na **Figura 3.2**, similar à estrutura apresentada no **cenário 1**, porem esta topologia de rede apresenta servidores conectados ao switch de clientes. Desta forma, o gateway/firewall assume maiores responsabilidades que além de isolar e dar proteção aos clientes o gateway seguro deve dar disponibilidade de serviços que os servidores apresentam de uma forma que não comprometa a segurança dos clientes.

**Cenário 3** – Neste cenário foi criada uma estrutura conforme ilustrada na **Figura 3.3**, esta apresenta uma topologia que interliga o gateway/firewall com dois switches de rede. O primeiro switch com um ou mais clientes conectados e fisicamente separados da Internet, e o segundo switch com um ou mais servidores conectados, porem, este disponível ao acesso da Internet e dos clientes com determinadas restrições. O modelo de topologia de rede apresentado neste cenário é considerado ideal na perspectiva de segurança (que é o foco deste trabalho), pois os clientes estão fisicamente separando dos servidores e interligado ao gateway seguro aumentando a ineficácia de uma possível ameaça, como descrito anteriormente (**Capitulo 6.1.1**).

Neste trabalho para todos os estudos de caso as configurações serão realizadas de forma manual, porem apenas será apresentado o menu inicial de forma rápida a seguir.

**Figura 3.2 – Modelo de cenário de rede, cenário 2.**

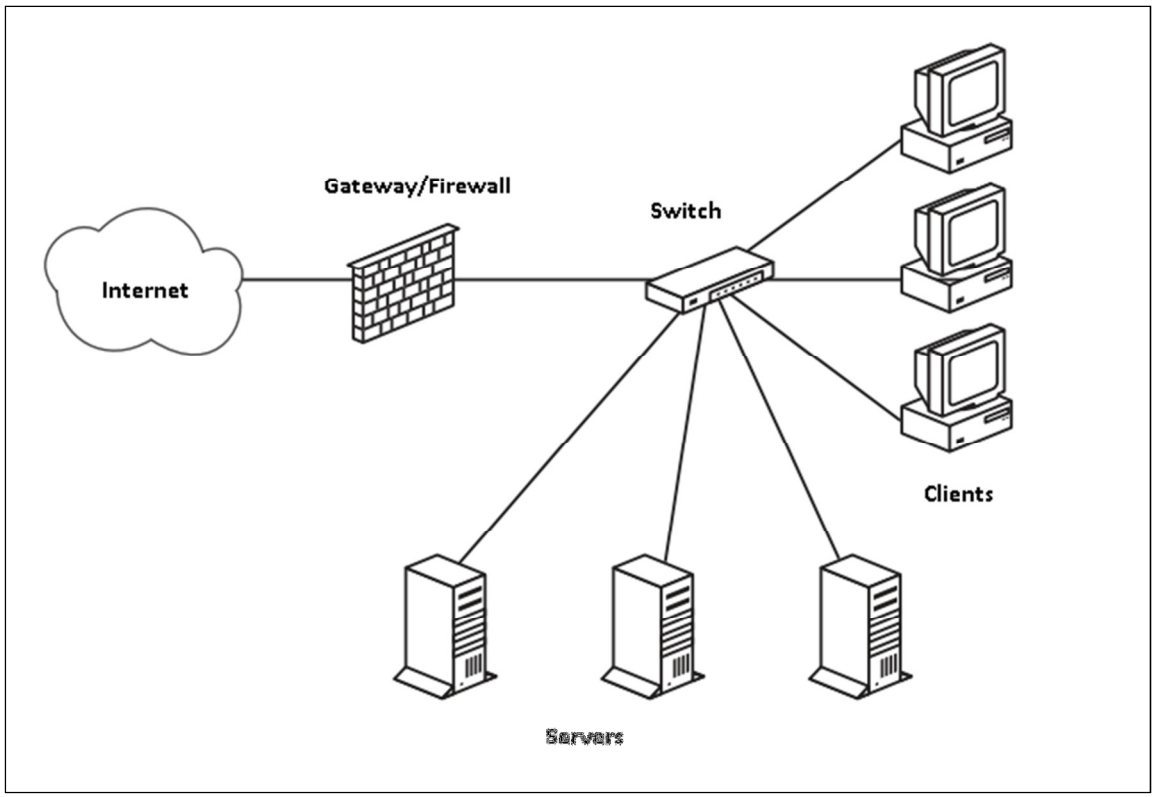
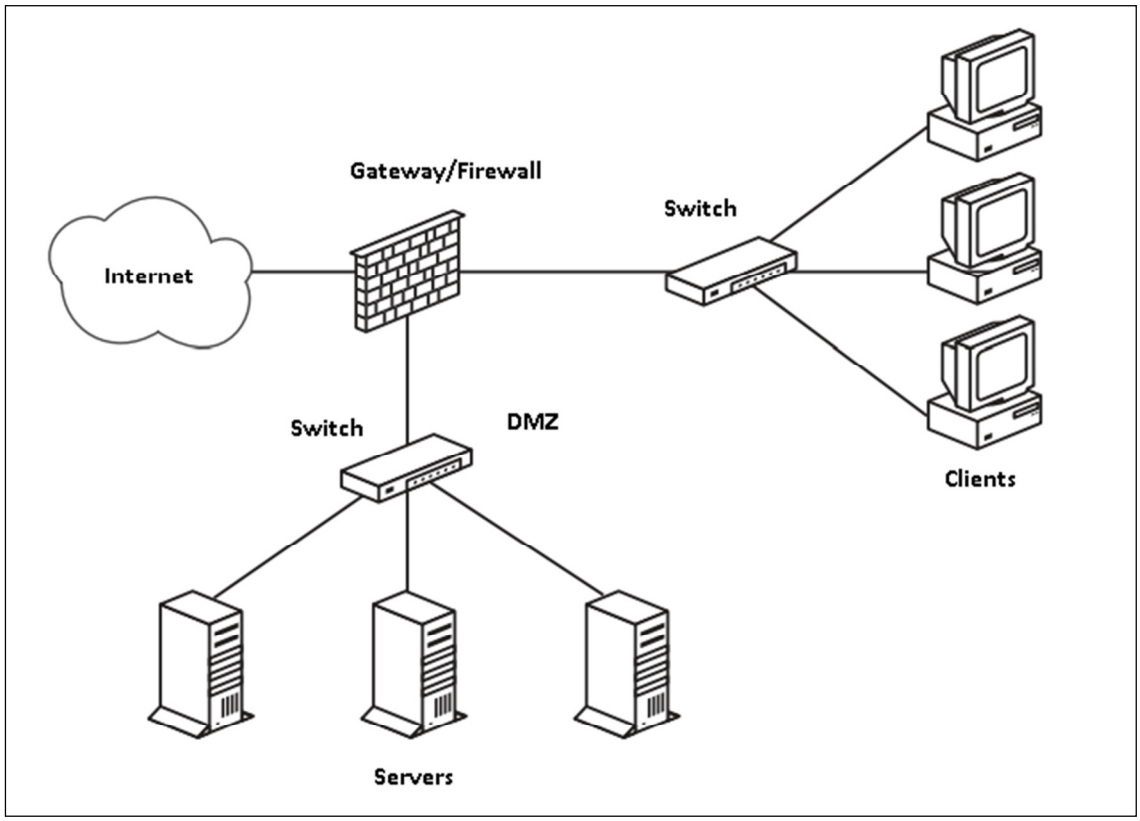



**Figura 3.3 – Modelo de cenário de rede, cenário 3.**

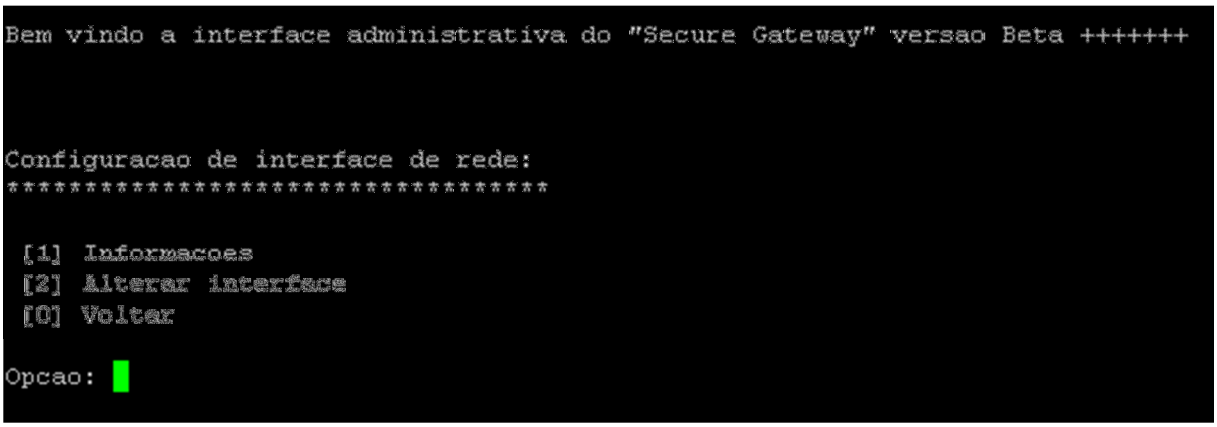


Ao iniciar o gateway de segurança e autenticar. O menu principal será apresentado (**Figura 3.4**).

**Figura 3.4 – Menu principal do gateway seguro.**


Inicialmente o usuário apenas devera configurar as interfaces de rede (**Figura 3.5**).

**Figura 3.5 – Menu de configuração de interface de rede do gateway seguro.**



## 3.2 CONFIGURAÇÕES DO GATEWAY

Na inicialização do gateway de segurança as configurações serão carregadas automaticamente para atuar como gateway. Seguindo as configurações no arquivo **/etc/sysctl.conf** (**Listagem 3.1**) do sistema OpenBSD.

**Listagem 3.1 – Configuração sysctl.conf para fazer com que o sistema atue como gateway.**

```
# /etc/sysctl.conf

net.inet.ip.forwarding=1
net.inet6.ip6.forwarding=1
```

O modulo do Kernel *net* do OpenBSD suporta reencaminhar pacotes entre interfaces sendo eles pacotes de IPV4 ou IPV6 (Hansteen, Peter N.M, 2008). Este reencaminhamento é habilitado ao setar para "1" os valores da **Listagem 3.1**.

## 3.3 CONFIGURAÇÕES DO FIREWALL

Na inicialização o gateway de segurança carrega as configurações do Packet Filter que é especificado no arquivo **/etc/rc.conf.local** e apresentado na **Listagem 3.2**. Observa-se que na linha 4 é especificado o arquivo que contem as regras, este arquivo que pode ser visto na **Listagem 3.3** contem *anchors* (ancoras) para outros arquivos em específico. Cada arquivo especifico representa as regras dos estudos de caso que serão apresentados adiante.

**Listagem 3.2 – Configuração rc.conf.local para carregar o Packet Filter.**

```
#/etc/rc.conf.local

pf=YES
pf_rules=/etc/pf.conf
```

**Listagem 3.3 – Arquivo de regras principal do Packet Filter.**

```
ext_if = "em0"
int_if = "em1"

anchor firewall
load anchor firewall from "/etc/rules/anchor-firewall"
anchor vpn
load anchor vpn from "/etc/rules/anchor-vpn"
anchor loadbalance
load anchor loadbalance from "/etc/rules/anchor-loadbalance"
anchor snort
load anchor snort from "/etc/rules/anchor-snort"
anchor failhover
load anchor failhover from "/etc/rules/anchor-failhover"
```

As regras devem ser criadas com cautela, pois erros de firewall podem causar indisponibilidade de serviços ou permissões indevidas. Para melhor entendimento, quando for escrever as regras tenha como boa pratica mapear as interfaces de rede, os acessos permitidos e/ou negados. As macros foram feitas para este objetivo.

**Listagem 3.4 – Exemplo de utilização de macros.**

```
ext_if = "em0"
permitido = "{ 192.168.2.1 , 192.168.2.10 }"
```

Macros podem armazenar uma string simples ou uma string que contem uma lista (**Listagem 3.4**). Uma boa pratica é mapear apenas interface de rede, interface de rede virtual ou portas. Para endereços IP e endereços de rede a boa pratica é utilizar tabelas (**Listagem 3.5**).

**Listagem 3.5 – Exemplo de utilização de tabelas.**

```
table <permitido> {192.168.2.1, 192.168.2.10 }
```

"Uma tabela é usada para armazenar um grupo de endereços IPv4 e/ou IPv6. Pesquisas em tabelas são muito rápidas e consumem menos memória e tempo de processamento do que as listas. Por essa razão, uma tabela é ideal para armazenar uma grande quantidade de endereços, já que o tempo de pesquisa em uma tabela com 50.000 endereços é apenas ligeiramente maior do que em uma contendo 50 endereços." (Openbsd, 2012).

**Listagem 3.6 – Sintaxe de regras do Packet Filter.**

```
action [direction] [log] [quick] [on interface] [af] [proto protocol] \
   [from src_addr [port src_port]] [to dst_addr [port dst_port]] \
   [flags tcp_flags] [state]
```

De forma geral e simplificada, a sintaxe para regras de filtragem é ilustrada na **Listagem 3.6**. Correspondendo a:

- **ação** – É a ação executada em pacotes que correspondem à regra;
- **direção** – A direção em que o pacote está se movendo em uma interface;
- **log** – Especifica que o pacote deve ser registrado;
- ***quick*** – corresponde a uma regra que é considerada final, as demais regras serão ignoradas;
- **interface** – nome ou grupo de interface de rede;
- **fam_de_end** – A família de endereços de pacotes (ipv4, ipv6);
- **protocolo** – O protocolo da camada 4 do modelo TCP/IP pacotes;
- **end_de_or** – É o endereço de origem do pacote, pode ser especificado um IP ou uma rede;
- **end_de_dest** – É o endereço de destino do pacote, pode ser especificado um IP ou uma rede;
- **porta_de_or** – É a porta de origem/destino no cabeçalho da Camada 4 do Modelo TCP/IP (um numero de 1 à 65535);
- **sinalizadores_tcp** – Especifica os sinalizadores que estão definidos no cabeçalho TCP;
- **estado**: Especifica se a informação de estado deve ser mantida em pacotes que correspondem à regra.

No firewall do **segure-gateway**, por padrão as regras são carregadas com as configurações de uma rede segura do primeiro cenário. Neste cenário todo e qualquer tipo de pacote de rede que passa pela interface de rede WAN é bloqueada e apenas alguns serviços de rede na entrada LAN é permitido:

- SSH;
- SMTP;
- DOMAIN;
- WWW;
- POP3;
- AUTH;
- HTTPS;
- POP3S

Os serviços são associados a suas portas padrão consultando o arquivo **/etc/service**. Para quaisquer conexões solicitadas. As regras para o Packet Filter normalmente são simples e podem ser escritas no arquivo de configuração padrão **/etc/pf.conf** ou qualquer outro arquivo, obviamente que o arquivo deve ser indicado para ser carregado na inicialização.

Segue na **Listagem 3.7** o script de configuração do Packet Filter que atende o modelo de rede do **Cenário 1**.

**Listagem 3.7 – Configuração do firewall Packet Filter para o Cenário 1.**

```
tcp_services = "{ ssh, smtp, domain, www, pop3, auth, https, pop3s }"
udp_services = "{ domain }"

block all
pass out proto tcp to any port $tcp_services
pass proto udp to any port $udp_services
```

Arquitetura do segundo cenário, cujo este mantem servidores na mesma rede das maquinas clientes, foi simulado uma rede com três servidores:

- Servidor WEB;
- Servidor de E-mail;
- Servidor de nomes.

A implementação do segundo cenário, as configurações do Packet Filter basicamente libera a passagem do trafego de dados dos serviços para a interface de rede interna (LAN). Bloqueando a entrada de dados da interface externa (WAN).

Verificando o script da **Listagem 3.8**, as 13 primeiras linhas é onde são mapeadas as interfaces de rede, protocolos e endereços dos servidores, em seguida nas linhas de 16 a 32 informam que por padrão todo e qualquer trafego é bloqueado, as próximas regras vão liberando os acessos aos serviços.

No arquivo de regras (**Listagem 3.8**) existe a opção de *synproxy state*, quando o Packet Filter recebe uma conexão que segundo sua configuração é permite trafegar na rede, este é apenas verificado se pode ser aceito ou não. Porem com a opção de *synproxy state* o Packet Filter age como um intermediário entre os hosts que se comunicam prevenindo ataques sobre a negociação das conexões (*tree-way handshake attack*). Ataques como *SYN-Flood* pode causar exaustão do servidor que está recebendo a conexão.

**Listagem 3.8 – Configuração do firewall Packet Filter para o Cenário 2.**

```
ext_if = "em0"
int_if = "em1"

localnet = $int_if:network
webserver = "192.168.10.102"
webports = "{ http, https }"
emailserver = "192.168.10.103"
email = "{ smtp, pop3, imap, imap3, imaps, pop3s }"
nameservers = "{ 192.168.10.100, 192.168.10.101 }"
client_out = "{ ssh, domain, pop3, auth, nntp, http,\
 https, cvspserver, 2628, 5999, 8000, 8080 }"
udp_services = "{ domain, ntp }"
icmp_types = "{ echoreq, unreach }"


block all

pass quick inet proto { tcp, udp } from $localnet to any port $udp_services

pass log inet proto icmp all icmp-type $icmp_types

pass inet proto tcp from $localnet to any port $client_out

pass inet proto { tcp, udp } from any to $nameservers port domain

pass proto tcp from any to $webserver port $webports synproxy state

pass log proto tcp from any to $emailserver port $email synproxy state

pass log proto tcp from $emailserver to any port smtp synproxy state

pass on $ext_if proto tcp from any to any port ssh
```

Outros serviços como serviço de nomes (DNS) esta acessível apenas localmente, isso garante maior segurança sobre vazamento de informações da rede interna (LAN/DMZ), pois o mesmo não pode ser acessado externamente. Vale lembrar que o serviço de nomes pode e é muito utilizado com um endereço fixo na Internet, deve se tomar cuidado para configurar adequadamente este serviço para que informação de servidores da rede internet mantenha acessível apenas internamente.

Seguindo para **Cenário 3** de implementação do gateway. Este possuindo uma terceira interface que separa os serviços e servidores dos clientes. Este modelo de rede é chamado de DMZ (*DeMilitarized Zone* do inglês, rede desmilitarizada) que tem o objetivo isolar aplicações do público com regras de acessos restritos (Kioskea, 2013).

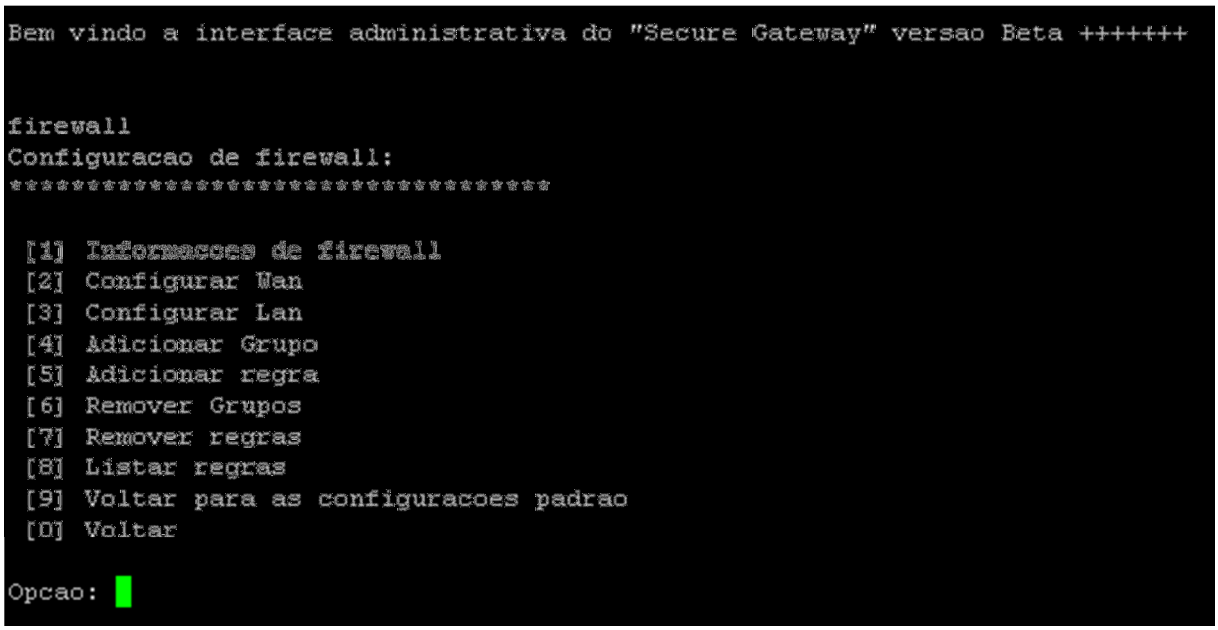
**Listagem 3.9 – Configuração do firewall Packet Filter para o Cenário 3.**

```
ext_if = "em0"
int_if = "em1"
dmz_if = "em2"

localnet = $int_if:network
webserver = "192.168.10.102"
webports = "{ http, https }"
emailserver = "192.168.10.103"
email = "{ smtp, pop3, imap, imap3, imaps, pop3s }"
nameservers = "{ 192.168.10.100, 192.168.10.101 }"
client_out = "{ ssh, domain, pop3, auth, nntp, http,\
 https, cvspserver, 2628, 5999, 8000, 8080 }"
udp_services = "{ domain, ntp }"
icmp_types = "{ echoreq, unreach }"


block all

pass in on $ext_if proto { tcp, udp } from any to $nameservers port domain

pass in on $int_if proto { tcp, udp } from $localnet to $nameservers port domain

pass out on $dmz_if proto { tcp, udp } from any to $nameservers port domain

pass in on $ext_if proto tcp from any to $webserver port $webports

pass in on $int_if proto tcp from $localnet to $webserver port $webports

pass out on $dmz_if proto tcp from any to $webserver port $webports

pass in log on $ext_if proto tcp from any to $emailserver port smtp

pass in log on $int_if proto tcp from $localnet to $emailserver port $email

pass out log on $dmz_if proto tcp from any to $emailserver port smtp

pass in on $dmz_if proto udp from $emailserver to any port smtp

pass out log on $ext_if proto tcp from $emailserver to any port smtp
```

Verificando o arquivo de configuração (**Listagem 3.9**), é possível analisar o fluxo dos dados, sendo estes permitidos entre os serviços da DMZ e da rede de clientes, conexões da interface externa são mais limitados.

A configuração do firewall, o gerenciamento de grupos e regra pode ser realizada utilizando a interface administrativa (**Figura 3.6**).

**Figura 3.6 – Menu de configuração de firewall.**




Mais algumas configurações de segurança deverão ser implementadas para maior eficácia do firewall. Sendo estas as principais para ataques em massa e o abuso da rede:

- *Brute force* – É um ataque aonde o objetivo é de encontrar as credencias de algum usuário valido na forma de tentativa e erros, normalmente aleatoriamente utilizando usuários e senhas mais comuns. Esse tipo de ataque é muito comum em servidores que tenha acesso a ssh. Uma forma de bloquear este tipo de ataque é limitando as tentativas por certo tempo, isto é, se houver 15 tentativas de acesso por menos de 5 segundos o IP do atacante é inserido na *blacklist*, este trabalho pode ser facilmente configurado com o Packet Filter utilizando tabelas (**Listagem 3.10**).

**Listagem 3.10 – Configuração do firewall Packet Filter contra ataques de bruta force .**

```
table <bruteforce> persist
block quick from <bruteforce>
pass inet proto tcp from any to $localnet port $tcp_services \
 keep state (max-src-conn 100, max-src-conn-rate 15/5, \
 overload <bruteforce> flush global)
```

- Spammers – Outro serviço ao qual se deve ter especial cuidado é o serviço de e-mail, sendo este amplamente utilizado por usuário para troca de mensagens.

Com esta popularidade se torna um dos principais alvos de ataques de spammers, que são usuários mal intencionando que enviam uma grande quantidade de e-mail (normalmente e-mails pequenos que contem vírus entre outros malwares) muitas vezes os usuários que são pegos por um ataque de spammer ajudam a propagar mais vírus. Da mesma forma, a melhor maneira de bloquear spammers é a utilização de *blacklist* e *whitelist*, sendo *whitelist* e-mail autorizado a enviar varias mensagens (como um serviço de noticia por e-mail) (**Listagem 3.11**).

**Listagem 3.11 – Configuração do firewall Packet Filter contra spammers.**

```
table <spamd> persist
table <spamd-white> persist
rdr pass on $ext_if inet proto tcp from <spamd> to \
 { $ext_if, $localnet } port smtp -> 127.0.0.1 port 8025
rdr pass on $ext_if inet proto tcp from !<spamd-white> to \
 { $ext_if, $localnet } port smtp -> 127.0.0.1 port 8025
```

Para melhor conhecimento do que acontece com a rede, a configuração de logs de todos os pacotes que trafegam pela rede são essenciais. O OpenBSD trás consigo o pftop, um aplicativo que mostra de uma forma bem simplificada o consumo da rede no exato momento de sua utilização.

Deve-se então, alterar as regras do Packet Filter para que gere logs de todo o trafego como mostrado **Listagem 3.12**.

**Listagem 3.12 – Configuração do firewall Packet Filter para criar logs do trafego de dados da rede.**

```
block log (all) all
pass log (all) quick proto tcp from any to any port ssh keep state
```

A **Listagem 3.13** demonstra os logs gerados segundo a configuração apresentada na **Listagem 3.12**, com o uso do **tcpdump** (Apêndice A).

**Listagem 3.13 – Logs de conexão ssh.**

```
Mar 04 04:51:07.743485 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: . ack 1 win 256 (DF)
Mar 04 04:51:07.758153 rule 29/(match) pass out on em0: 192.168.2.6.22 > 192.168.2.5.16666: P 1:22(21) ack 1 win 2190 (DF)
Mar 04 04:51:07.758568 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: P 1:29(28) ack 22 win 256 (DF)
Mar 04 04:51:07.758645 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: P 29:541(512) ack 22 win 256 (DF)
Mar 04 04:51:07.758666 rule 29/(match) pass out on em0: 192.168.2.6.22 > 192.168.2.5.16666: . ack 541 win 2122 (DF)
Mar 04 04:51:07.758682 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: P 541:669(128) ack 22 win 256 (DF)
Mar 04 04:51:07.831108 rule 29/(match) pass out on em0: 192.168.2.6.22 > 192.168.2.5.16666: P 22:1006(984) ack 669 win 2110 (DF)
Mar 04 04:51:07.831495 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: P 669:685(16) ack 1006 win 252 (DF)
Mar 04 04:51:07.833988 rule 29/(match) pass out on em0: 192.168.2.6.22 > 192.168.2.5.16666: P 1006:1542(536) ack 685 win 2190 (DF)
Mar 04 04:51:07.868418 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: . ack 1542 win 256 (DF)
Mar 04 04:51:07.974574 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: P 685:1197(512) ack 1542 win 256 (DF)
Mar 04 04:51:07.974776 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: P 1197:1213(16) ack 1542 win 256 (DF)
Mar 04 04:51:07.974928 rule 29/(match) pass out on em0: 192.168.2.6.22 > 192.168.2.5.16666: . ack 1213 win 2124 (DF)
Mar 04 04:51:07.987353 rule 29/(match) pass out on em0: 192.168.2.6.22 > 192.168.2.5.16666: P 1542:2646(1104) ack 1213 win 2190 (DF)
Mar 04 04:51:08.042212 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: . ack 2646 win 252 (DF)
Mar 04 04:51:08.143568 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: P 1213:1229(16) ack 2646 win 252 (DF)
Mar 04 04:51:08.144169 rule 29/(match) pass in on em0: 192.168.2.5.16666 > 192.168.2.6.22: P 1229:1281(52) ack 2646 win 252 (DF)
```

**Listagem 3.14 – Logs de spammer entrando na blacklist.**

```
Nov  6 09:50:25 delilah spamd[23576]: 210.214.12.57: connected (1/0)
Nov  6 09:50:32 delilah spamd[23576]: 210.214.12.57: connected (2/0)
Nov  6 09:50:40 delilah spamd[23576]: (GREY) 210.214.12.57: <gilbert@keyholes.net> ->
<wkitp98zpu.fsf@datadok.no>
Nov  6 09:50:40 delilah spamd[23576]: 210.214.12.57: disconnected after 15 seconds.
Nov  6 09:50:42 delilah spamd[23576]: 210.214.12.57: connected (2/0)
Nov  6 09:50:45 delilah spamd[23576]: (GREY) 210.214.12.57: <bounce-3C7E40A4B3@branch15.summer-
bargainz.com> -> <adm@dataped.no>
Nov  6 09:50:45 delilah spamd[23576]: 210.214.12.57: disconnected after 13 seconds.
Nov  6 09:50:50 delilah spamd[23576]: 210.214.12.57: connected (2/0)
Nov  6 09:51:00 delilah spamd[23576]: (GREY) 210.214.12.57: <gilbert@keyholes.net> ->
<wkitp98zpu.fsf@datadok.no>
Nov  6 09:51:00 delilah spamd[23576]: 210.214.12.57: disconnected after 18 seconds.
Nov  6 09:51:02 delilah spamd[23576]: 210.214.12.57: connected (2/0)
Nov  6 09:51:02 delilah spamd[23576]: 210.214.12.57: disconnected after 12 seconds.
Nov  6 09:51:02 delilah spamd[23576]: 210.214.12.57: connected (2/0)
Nov  6 09:51:18 delilah spamd[23576]: (GREY) 210.214.12.57: <gilbert@keyholes.net> ->
<wkitp98zpu.fsf@datadok.no>
Nov  6 09:51:18 delilah spamd[23576]: 210.214.12.57: disconnected after 16 seconds.
Nov  6 09:51:18 delilah spamd[23576]: (GREY) 210.214.12.57: <bounce-3C7E40A4B3@branch15.summer-
bargainz.com> -> <adm@dataped.no>
Nov  6 09:51:18 delilah spamd[23576]: 210.214.12.57: disconnected after 16 seconds.
Nov  6 09:51:20 delilah spamd[23576]: 210.214.12.57: connected (1/1), lists: spamd-greytrap
Nov  6 09:51:23 delilah spamd[23576]: 210.214.12.57: connected (2/2), lists: spamd-greytrap
Nov  6 09:55:33 delilah spamd[23576]: (BLACK) 210.214.12.57: <gilbert@keyholes.net> ->
<wkitp98zpu.fsf@datadok.no>
Nov  6 09:55:34 delilah spamd[23576]: (BLACK) 210.214.12.57: <bounce-
3C7E40A4B3@branch15.summer-bargainz.com> -> <adm@dataped.no>
```

Para melhor desempenho da rede, alguns optimizações de regras e pacotes são necessárias na configuração do gateway (**Listagem 3.15**). Sendo a primeira a opção *block-police*, que determina o que fazer com hosts que tentam criar novas conexões tendo registro de conexões bloqueadas. A correta maneira de se tratar estas conexões é bloqueando. Quando

configurado o *block-police* rejeita novas conexões sem mandar resposta nenhuma para o host solicitante.

**Listagem 3.15 – Configuração do firewall Packet Filter para a opção block-policy.**

```
set block-policy return
```

Outra opção de otimização é a utilização do *skip* (**Listagem 3.16**), utilizado para ignorar interfaces de rede que não necessitam de controle de firewall (a interface *loopback*, por exemplo). Ignorar interfaces que não necessitam de filtros resultam em ganho de desempenho na verificação das regras.

**Listagem 3.16 – Configuração do firewall Packet Filter para a opção skip.**

```
set skip on lo0
```

A opção *ruleset-optimization* (**Listagem 3.17**), remove regras duplicadas automaticamente e reordena tabelas resultando em maior desempenho.

**Listagem 3.17 – Configuração do firewall Packet Filter para a opção ruleset-optimization.**

```
set ruleset-optimization basic
```

Para a normalização do trafego de dados na rede, utiliza-se a opção *scrub* (**Listagem 3.18**). Com a opção *scrub* pacotes fragmentados são remontados eliminando pacotes inválidos ou duplicados, desta forma “limpando” a rede e protegendo contra alguns tipos de ataques baseado em explorar falhas de montagem de pacotes.

**Listagem 3.18 – Configuração do firewall Packet Filter para a opção scrub.**

```
scrub in all
```

A opção *antispoof* (**Listagem 3.19**) foi implementada para um caso especial: proteger contra endereços IP *spoofed*[19], bloqueando pacotes que entram na interface de rede com trajetórias logicamente possível.

**Listagem 3.19 – Configuração do firewall Packet Filter para a opção antispoof.**

```
antispoof for $ext_if
antispoof for $int_if
```

Analisando o processo de pentest, observasse que a principal etapa está em "aquisição de informação" e "mapeamento de rede". Pritak (Pritak, 2011) publicou a configuração do nmap para bloquear aplicativos (Nmap, Apêndice B) que mapeiam a rede **Listagem 3.20**.

**Listagem 3.20 – Configuração do firewall Packet Filter para bloquear aplicativos que mapeiam de rede (Fonte: Pritak, 2011).**

```
block in quick proto tcp flags FUP/WEUAPRSF
block in quick proto tcp flags WEUAPRSF/WEUAPRSF
block in quick proto tcp flags SRAFU/WEUAPRSF
block in quick proto tcp flags /WEUAPRSF
block in quick proto tcp flags SR/SR
block in quick proto tcp flags SF/SF
block in quick proto tcp flags FUP/FUP
```

## 3.4 CONFIGURAÇÕES DO VPN

A implementação do serviço de VPN foi baseando em clientes que utilização IP dinâmicos (providenciado por ISP), desta forma a utilização de certificados digitais garante a autenticidade dos clientes da VPN.

Uma forma pratica de configurar o serviço de VPN no OpenBSD é usando o utilitário **ikectl**.

[19] Pacotes com endereços de origem falsificados.

Com este utilitário a VPN trabalhara no protocolo IKEV2 (denominado VPN IKEV2) que utiliza duas portas para se comunicar, sendo a porta 500 e 4500 utilizando o protocolo UDP. Por padrão todo o trafego dentro da VPN é criptografado utilizando SSL. Para o funcionamento correto é necessária liberação das portas utilizadas (500 e 4500) no firewall (Mazzocchio, Daniele, 2009).

Antes da configuração da VPN propriamente dita, necessita-se de um certificado valido, o OpenBSD fornece um modelo de configuração para gerar certificados, este pode ser encontrado no diretório **/usr/src/usr.sbin/ikectl/ikeca.cnf**. A **Figura 3.7** mostra a geração do certificado e em seguida a configuração do **IKED**.

**Figura 3.7 – Certificado digital VPN.**

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [FR]:BR
State or Province Name (full name) [NA]:Parana
Locality Name (eg, city) [REUNION]:Medianeira
Organization Name (eg, company) [.aise]:utfpr
Organizational Unit Name (eg, section) [VPN]:
Common Name (eg, fully qualified host name) [VPN CA]:secure-gateway
Email Address [wesley@mouedine.net]:baron.rodrigo0@gmail.com
Signature ok
subject=/C=BR/ST=Parana/L=Medianeira/O=utfpr/OU=VPN/CN=secure-gateway/emailAddre
ss=baron.rodrigo0@gmail.com
Getting Private key
Using configuration from /etc/ssl/ikeca.cnf
```

E por fim a liberação das portas utilizadas pela VPS no firewall Packet Filter da mesma forma mostrado na **Figura 3.8**.

**Figura 3.8 – Configuração firewall packet filter pra VPN.**

```
pass in on $ext_if proto udp from any to any port (500,4500)
```

## 3.5 CONFIGURAÇÕES DO LOAD BALANCE

*Load balance* é o redirecionamento do trafego de dados para servidores replicados, a fim de diminuir sobrecargas no servidor principal (Promon, 2006). As configurações *de Load Balance* no OpenBSD normalmente é realizada utilizando o firewall Packet Filter.

A configuração no firewall deve conter quais são os hosts e qual algoritmo utilizado para decidir a qual servidor o cliente vai se conectar.

**Listagem 3.21 – Configuração do firewall Packet Filter para *load balance*.**

```
ext_if = "em0"
int_if = "em1"
table <webpool> persist { 192.168.10.101, 192.168.10.102 }
webserver = "{ 192.168.10.101 }"
webports = "{ http, https }"

block log (all) all
pass out

pass in log (all) on $ext_if proto tcp to port $webports rdr-to <webpool> \
    round-robin sticky-address
```

Observa-se na **Listagem 3.21** que após mapear as interfaces de rede é criada uma tabela que especifica os IPs que serão replicados juntamente com o servidor principal. Todo o trafego de dados entrando na interface externa (WAN) utilizando o protocolo TCP na porta 80 é redirecionado para um IP na tabela *webpool*. Que IP que receberá a conexão é decidida pelo algoritmo *round-robin* (algoritmo cíclico) isto é, o próximo IP da lista recebera a próxima conexão, quando chegar ao fim começa pelo primeiro novamente, a ordem dos IPs é decidida pela distancia (fisicamente) entre o gateway e o servidor (sticky-address).

**Listagem 3.22 – Log de negociação *load balance*.**

```
Mar 06 17:12:30.601004 rule 3/(match) pass in on em0: 192.168.2.2.65038 > 192.168.2.6.80: . ack 1478 win 256 (DF)
Mar 06 17:12:30.601051 rule 3/(match) pass in on em0: 192.168.2.2.65038 > 192.168.2.6.80: . ack 2002 win 254 (DF)
Mar 06 17:12:30.601708 rule 3/(match) pass in on em0: 192.168.2.2.65038 > 192.168.2.6.80: F 420:420(0) ack 2002 win 254 (DF)
Mar 06 17:12:30.602563 rule 3/(match) pass out on em0: 192.168.10.101.80 > 192.168.2.2.65038: . ack 422 win 980 (DF)
```

A **Listagem 3.22** mostra um trecho do arquivo de logs mostrando a negociação (tree-way-handshake) entre o cliente, gateway e o servidor. Em uma das tentativas não houve resposta e em seguida, na segunda tentativa, houve uma resposta (ACK[20]) e por fim a resposta do servidor.

## 3.6 CONFIGURAÇÕES DO IDS/IPS

Quando o objetivo é disponibilizar um IPS ou IDS rodando como um *daemon* o snort se mostra uma boa opção. Snort é uma ferramenta capaz de identificar e deter alguns tipos de ataques em sistemas de nível critico tais como Banco de Dados, Servidor HTTP utilizando filtros que buscam por assinaturas maliciosas.

O OpenBSD por padrão não disponibiliza nenhuma ferramenta especifica para este objetivo, (com o Packet Filter é possível implementar as características de IPS, mas não é efetivo quando comparado ao snort) desta forma à instalação e configuração deve ser realizada. Usando o utilitário de instalação pkg_add (**Listagem 3.23**) para a instalação e em seguida habilitando ele no arquivo **/etc/rc.conf.local** (**Listagem 3.24**).

**Listagem 3.23 – Utilitário pkg_add.**

```
# pkg_add -r snort-2.8.6p1
```

**Listagem 3.24 – Habilitando snor no OpenBSD.**

```
snort=YES
```

E em seguida a configuração de inicialização automática no processo de boot do sistema (**Listagem 3.25**).

[20] ACK – Flag Acknowledgement, flag de confirmação do protocolo TCP.

**Listagem 3.25 – Configuração de inicialização do snort.**

```
if [ X"${snort}" == X"YES" -a -x /usr/local/bin/snort ]; then
echo -n "Iniciando Snort"; /usr/local/bin/snort -D -d -c /etc/snort/snort.conf -u _snort -g _snort -t /var/snort -l /var/snort/log
fi
```

O snort implementa as características de IPS gerando dados de forma organizada e como IDS analisando os dados gerado pelo IPS, buscando assinaturas de ataques na rede. As assinaturas de ataque estão contido junto com as regras e são carregados conforme o arquivo de configuração **/etc/snort/snort.conf** esta definido (**Listagem 3.26**).

**Listagem 3.26 – Configuração de regras do snort.**

```
include $RULE_PATH/local.rules

include $RULE_PATH/exploit.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/dns.rules

include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-coldfusion.rules
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-frontpage.rules
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-client.rules
include $RULE_PATH/web-php.rules

include $RULE_PATH/sql.rules
include $RULE_PATH/x11.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/oracle.rules
include $RULE_PATH/mysql.rules

include $RULE_PATH/smtp.rules
include $RULE_PATH/imap.rules
include $RULE_PATH/pop2.rules
include $RULE_PATH/pop3.rules

include $RULE_PATH/nntp.rules
include $RULE_PATH/backdoor.rules
```

As atividades de intrusão contem assinaturas de dados similares comparando com assinaturas de vírus. Para capturar assinaturas de ataques pode-se utilizar *honeyports* ou então com a posse de *exploits*, analisar os dados gerados.

Com base nas assinaturas são criados a regras para o snort. Apos o filtro de dados (IPS) captura o pacote de dados da interface de rede, a primeira analise é do cabeçalho (IP, Protocolo e Flags) e em seguida os dados propriamente dito. Após a analise são aplicadas as regras (IDS) que podem gerar mensagens de alerta, gerar logs, permitir que o pacote chegue até o destinatário ou descartar. Muito similar comparando com as regras de firewall.

As regras para o snort normalmente são pequenas, a **Listagem 3.27** demonstra uma regra simples que gera alerta toda vez que um pacote com protocolo icmp é detectado na interface de rede que o snort analisa.

**Listagem 3.27 – Configuração de regras do snort.**

```
alert icmp any any -> any any (msg: "Regra Ping para snort";)
```

O exemplo da regra na **Listagem 3.27** indica que:

- A ação a ser aplicada é Alert (Manda uma mensagem junto ao log);
- O protocolo é icmp (outros protocolos serão ignorados);
- O endereço de IP do Host de origem;
- A porta de origem;
- A direção do pacote (entrando ou saindo da interface);
- O endereço IP de destino;
- A porta de destino;
- A mensagem e o parâmetro de comparação (neste caso não houve comparação de assinatura).

A ação é a primeira parte da regra do snort, outros tipos de ações podem ser aplicados as mais utilizadas são:

- **pass** – Está ação diz ao snort para ignorar pacotes que acionam a regra. Importante caso houver um serviço que envie falsos positivos;

- **log** – Esta ação diz ao snort gerar logs dos pacotes que acionam a regra;
- **alert** – Esta ação diz ao snort para gerar log com uma mensagem dos pacotes que acionam a regra e que acionam a condição;
- **Active** – Esta ação aciona uma ação **Alert** e outras regras;

O protocolo é a segunda parte da regra do snort, os protocolos atualmente suportados são:

- IP
- ICMP
- TCP
- UDP

O endereço é a terceira parte da regra do snort, sendo possível especificar um endereço IP comum ou uma rede o calculo de sub-rede é feita automaticamente pelo snort. Isto é, pode se especificar um endereço como 192.168.1.10 ou uma rede seguida de sua mascara como 192.168.1.0/24.

A quarta parte da regra do snort é a porta, é possível especificar uma porta (80, por exemplo) ou varias portas (80:8080 [porta 80 e porta 8080]).

A direção determina a origem e o destino dos pacotes:

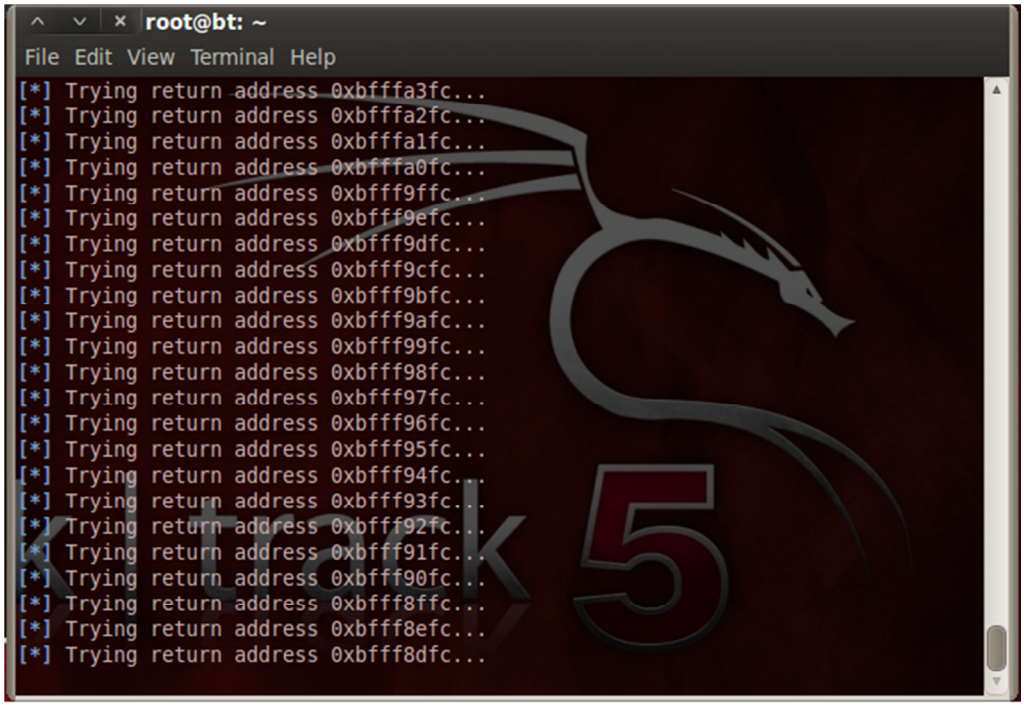
- O símbolo → indica que o endereço de rede e a porta à esquerda é a origem e o endereço e porta da direita é o destino;
- O símbolo ← indica que o endereço de rede e a porta à esquerda é o destino e o endereço e porta da direita é a origem;
- O símbolo <> faz com que a regra seja aplicada em abas direção.
- Opção da regra a ser aplicada. Uma boa pratica é indicar uma mensagem (caso a ação seja *Alert*), opção e em seguida a condição. O objetivo deste projeto não é sobre o snort em questão, desta forma não será abordado todas as opções.

Porem a opção mais utilizada é *content*, capaz de compara o conteúdo do pacote com uma assinatura de ataque.

No gateway de segurança as regras são as regras padrão, que compões o snort. Que pode defender contra mais de 1100 vulnerabilidades (Linuxsecurity, 2010).

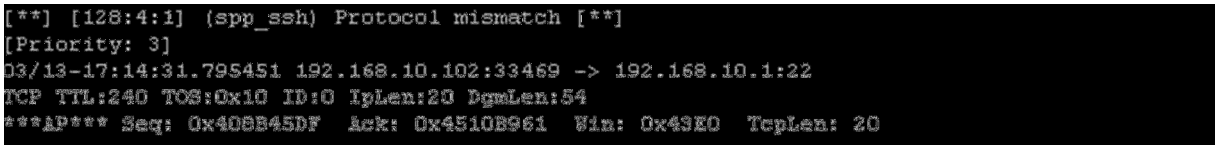
Para exemplificar a eficácia do snort, um teste de estress na porta ssh simulando um endereço arbitrário no protocolo ssh utilizando o Metasploit (Apêndice C), como pode ser visto na **Figura 3.8**.

**Figura 3.8 – Códigos arbitrários no protocolo ssh gerado pelo metasplot.**



Em seguida na **Figura 3.9** observasse os logs gerados a partir desta simulação:

**Figura 3.9 – Logs de alerta no protocolo do ssh.**




## 3.7 CONFIGURAÇÕES DO HONEYPOT

Existem varias formas de implementar uma *honeypot*. Em servidor BSD a mais famosa é a utilização do firewall Packet Filter combinado com IDS/IPS Snort. Porem existe outras maneiras mais simples e mais inteligentes de criar bons **honeypots**, uma delas é a atualização do aplicativo **netcat** (NC). Com uma shellscript simples criasse uma replica de um serviço famoso e com grandes probabilidades de ser alvo de ataques de *Black Hat Hackers.*

**Listagem 3.28 – Honeypot com netcat.**

```
while [ 1 ]; do echo "Honeypot iniciado!"; nc -lvv 443 < apache.txt >> honeypot_apache.log; done
```

O script da **Listagem 3.28** mantem sempre a porta 443 aberta esperando por novas conexões, após receber uma conexão o conteúdo do arquivo “apache.txt” é enviando ao cliente é mostrado o cabeçalho de um servidor HTTP que contem sua versão e respectivamente o sistema operacional como pode ser visto na **Listagem 3.29**. Após uma tentativa de ataque todos os dados enviados serão gravados no arquivo “honeypot_apache.log” que assim permite analisar o método de ataque utilizado.

**Listagem 3.29 – Cabecalho HTTP honeypot.**

```
HTTP/1.1 302 Found
Date: Thu, 07 Mar 2013 10:10:59 GMT
Server: Apache/2.2.3 (Fedora)
X-Powered-By: PHP/5.1.6
Location: server.com
Content-Length: 0
Connection: close
Content-Type: text/html; charset=ISO-8859-1
```

## 3.8 CONFIGURAÇÕES DO FAIL HOVER

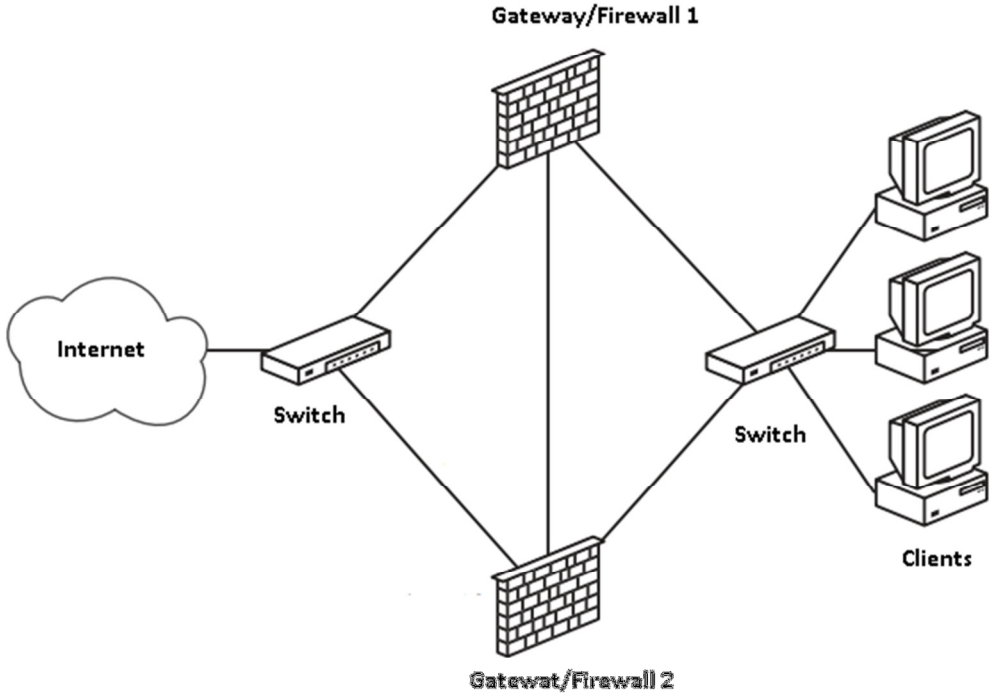
O OpenBSD disponibiliza duas ferramentas para criar redundância de dois ou mais gateways CARP e pfsync:

- CARP – O CARP foi desenvolvido para manter a rede funcional mesmo quando necessita fazer manutenções ou atualização planejadas. As configurações do CARP são baseados em grupos de maquinas, aonde uma delas é a maquina principal e as demais são redundantes.

- PFSYNC – Complementando o CARP, pfsync é um protocolo que tem como objetivo manter syncronizado o estado do Packet Filter com outros Packet Filter redundantes. Isto é necessário porque o Packet Filter pode criar regras dinamicamente. Na realidade pfsync é um tipo de interface virtual designado a trocar informações entre firewalls. Para realizar a sincronização de forma segura, é especificado uma senha de autenticação.

A arquitetura de rede apresentada no projeto é virtualmente similar a **Figura 3.10**, devido ao fato do projeto ser desenvolvido utilizando maquinas virtuais e redes virtuais.

**Figura 3.10 – Arquitetura de rede *failhover* (Fonte: Promon, 2006).**




A principal configuração está em habilitar o modulo CARP, e em seguida o modulo *preempt* que disponibiliza a comunicação entre interfaces de rede CARP. Está configuração devem ser realizados em ambos os gateways que participam da redundância (**/etc/sysctl.conf**) (Bukac, Vít, 2010) (**Listagem 3.30**).

**Listagem 3.30 – Habilitando CARP para failhover.**

```
net.inet.carp.allow=1
net.inet.carp.preempt=1
```

Em seguida deve ser feita a configuração das interfaces CARP. Esta configuração indica o IP para as interfaces CARP, a senha e o grupo. O grupo especificado será o MASTER para está maquina (**Listagem 3.31 e Listagem 3.32**).

**Listagem 3.31 – Configurando interfaces de rede para honeypot.**

```
# ifconfig carp0 pass secure 192.168.10.11 vhid 1
# ifconfig carp1 pass secure 192.168.100.11 vhid 1
```

**Listagem 3.32 – Mostrando configuração de interfaces de rede para honeypot.**

```
carp0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        lladdr 00:00:5e:00:01:01
        priority: 0
        carp: MASTER carpdev em1 vhid 1 advbase 1 advskew 0
        groups: carp
        status: master
        inet6 fe80::200:5eff:fe00:101%carp0 prefixlen 64 scopeid 0x6
        inet 192.168.10.11 netmask 0xffffff00 broadcast 192.168.10.255
```

Na maquina de backup o procedimento é similar, apenas difere marcando a interface no grupo BACKUP (Bukac, Vít, 2010) (**Listagem 3.33**).

**Listagem 3.33 – Configurando interfaces de rede para honeypot na maquina backup.**

```
# ifconfig carp0 pass secure 192.168.10.11 vhid 1 advskew 100
# ifconfig carp1 pass secure 192.168.100.11 vhid 1 advskew 100
```

O próximo passo é configurar a interface pfsync para apenas ser utilizada como um ponto de Sincronização (**Listagem 3.34**).

**Listagem 3.34 – Configurando ´fsync para honeypot.**

```
# ifconfig pfsync0 syncdev ep2
```

E por fim a configuração do firewall Packet Filter (**Listagem 3.35**).

**Listagem 3.35 – Configurando interfaces de rede para honeypot na maquina backup.**

```
pass on $carpdevs proto carp keep state
pass on $syncdev proto pfsync
set skip on $syncdev
```

# 4 APÊNDICE

## 4.1 APÊNDICE A - INTRODUÇÃO AO TCPDUMP

**Tcpdum** (**Listagem 8.1**) é uma ferramenta para capturar pacotes que trafegam pela(s) inteface(s) de rede. Após capturar o pacote ele é capaz salva uma copia do pacote em um arquivo. É comum encontrar esta ferramenta em sistemas UNIX e Linux.

Quando combinado com os logs do Packet Filter e Snort, se torna uma ferramenta capaz de uma analise de dados mais precisas (Rehman, Rafeeq Ur, 2003).

**Listagem 8.1 – Utilização do tcpdump.**

```
# tcpdump
tcpdump: listening on em0, link-type EN10MB
tcpdump: WARNING: compensating for unaligned libpcap packets
14:43:23.640768 192.168.2.6.ssh > 192.168.2.2.6973: P 961775903:961775987(84) ack 3665958019 win 2190 (DF) [tos 0x10]
14:43:23.640930 192.168.2.2.6973 > 192.168.2.6.ssh: . ack 84 win 251 (DF)
14:43:24.638495 192.168.2.6.ssh > 192.168.2.2.6973: P 84:152(68) ack 1 win 2190 (DF) [tos 0x10]
14:43:24.639232 192.168.2.6.ssh > 192.168.2.2.6973: P 152:236(84) ack 1 win 2190 (DF) [tos 0x10]
14:43:24.639768 192.168.2.2.6973 > 192.168.2.6.ssh: . ack 236 win 256 (DF)
14:43:24.641197 192.168.2.6.44232 > 192.168.203.1.domain: 35131+ PTR? 2.2.168.192.in-addr.arpa. (42)
14:43:24.733684 192.168.203.1.domain > 192.168.2.6.44232: 35131 NXDomain 0/0/0 (42) (DF)
14:43:24.735771 192.168.2.6.39111 > 192.168.203.1.domain: 65441+ PTR? 6.2.168.192.in-addr.arpa. (42)
14:43:24.784577 192.168.203.1.domain > 192.168.2.6.39111: 65441 NXDomain 0/0/0 (42) (DF)
14:43:24.786046 192.168.2.6.ssh > 192.168.2.2.6973: P 236:672(436) ack 1 win 2190 (DF) [tos 0x10]
14:43:24.786237 192.168.2.6.ssh > 192.168.2.2.6973: P 672:788(116) ack 1 win 2190 (DF) [tos 0x10]
14:43:24.786701 192.168.2.2.6973 > 192.168.2.6.ssh: . ack 788 win 254 (DF)
14:43:24.787446 192.168.2.6.37633 > 192.168.203.1.domain: 63527+ PTR? 1.203.168.192.in-addr.arpa. (44)
14:43:24.837427 192.168.203.1.domain > 192.168.2.6.37633: 63527 NXDomain 0/0/0 (44) (DF)
14:43:24.839033 192.168.2.6.ssh > 192.168.2.2.6973: P 788:1032(244) ack 1 win 2190 (DF) [tos 0x10]
14:43:24.889615 192.168.2.2.6973 > 192.168.2.6.ssh: . ack 1032 win 253 (DF)
14:43:25.838063 192.168.2.6.ssh > 192.168.2.2.6973: P 1032:1644(612) ack 1 win 2190 (DF) [tos 0x10]
14:43:25.938262 192.168.2.6.ssh > 192.168.2.2.6973: P 1644:1952(308) ack 1 win 2190 (DF) [tos 0x10]
14:43:25.938748 192.168.2.2.6973 > 192.168.2.6.ssh: . ack 1952 win 256 (DF)
14:43:25.939515 192.168.2.2.6973 > 192.168.2.6.ssh: P 1:85(84) ack 1952 win 256 (DF)
14:43:25.939600 192.168.2.2.6973 > 192.168.2.6.ssh: P 85:137(52) ack 1952 win 256 (DF)
14:43:25.939859 192.168.2.6.ssh > 192.168.2.2.6973: . ack 137 win 2179 (DF) [tos 0x10]
14:43:25.941185 192.168.2.6.ssh > 192.168.2.2.6973: P 1952:1988(36) ack 137 win 2190 (DF) [tos 0x10]
14:43:25.990922 192.168.2.2.6973 > 192.168.2.6.ssh: . ack 1988 win 256 (DF)
14:43:26.837791 192.168.2.6.ssh > 192.168.2.2.6973: P 1988:2568(580) ack 137 win 2190 (DF) [tos 0x10]
14:43:26.837989 192.168.2.6.ssh > 192.168.2.2.6973: P 2568:2796(228) ack 137 win 2190 (DF) [tos 0x10]
14:43:26.838311 192.168.2.2.6973 > 192.168.2.6.ssh: . ack 2796 win 253 (DF)
14:43:27.837282 192.168.2.6.ssh > 192.168.2.2.6973: P 2796:3120(324) ack 137 win 2190 (DF) [tos 0x10]
14:43:27.888101 192.168.2.2.6973 > 192.168.2.6.ssh: . ack 3120 win 252 (DF)
```

## 4.2 APÊNDICE B – INTRODUÇÃO AO NMAP

**Nmap** (*Network Mapper* - Mapeador de rede) é uma ferramenta gratuita e sob a licença *open source* (GNU GPL), utilizada para realizar descobertas e auditoria de redes como mostra a **Figura 8.2**. Ele utiliza pacote IP RAW (Pacote de IP Manipulável) para determinar quais hosts disponíveis na rede, quais os serviços que o host disponibiliza (nome da aplicação e versão), qual é o sistema operacional, consegue identificar a existência de firewall e possui muitas outras características (Nmap, 2012).

Foi principalmente desenhado para analisar grandes redes (Rede de computador com vários hosts), disponível nas plataformas Linux, Windows e Mac. Foi desenvolvido por vários programadores e administradores de sistemas com grande experiência em sistemas da informação e redes (Nmap, 2012).

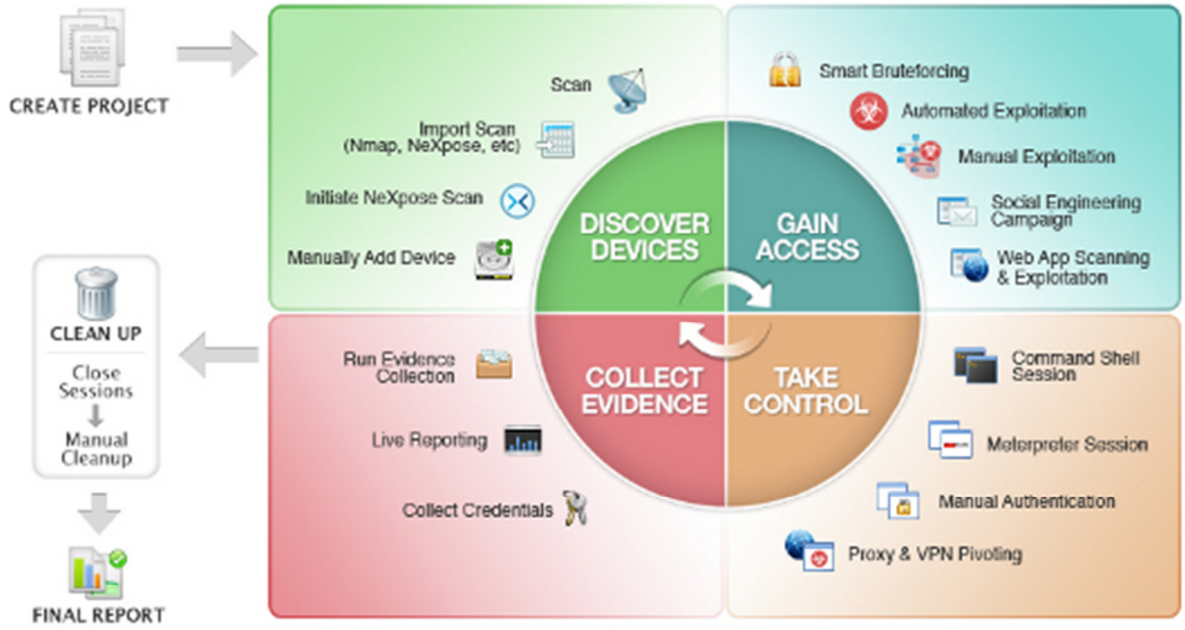
**Figura 8.2 – Utilização do nmap.**


## 4.3 APÊNDICE C – INTRODUÇÃO AO METASPLOIT

**Metasploit** é uma ferramenta modular, *open source* (Licença GNU GPL) para testes de vulnerabilidades. Altamente utilizado por profissionais de segurança da informação, por ser fácil e objetivo (Singh, Abhinav, 2012).

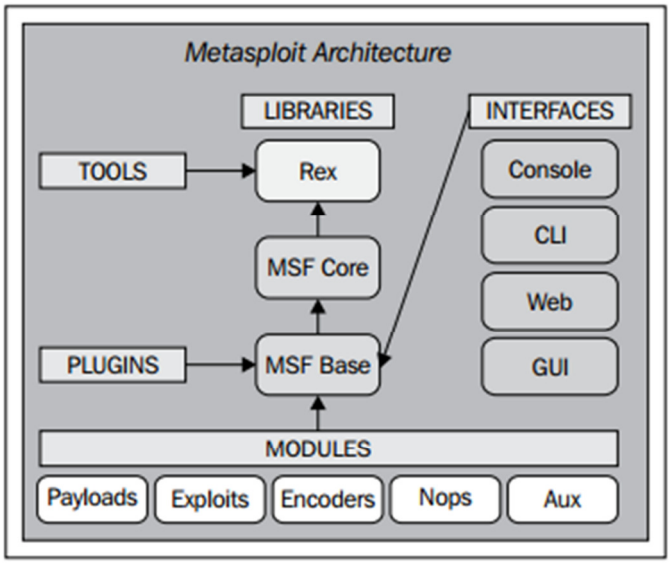
Em perspectiva de segurança da informação, o metasploit atende toda a gama necessária para realizar testes de penetração em sistemas (Aragão, Francisco, 2011) (**Figura 8.3**).

**Figura 8.3 – Metasploit para pentest.**




Abhinav ilustra a arquitetura modular do metasploit para melhor entendimento do seu funcionamento como representado na **Figura 8.4** (Singh, Abhinav, 2012).

**Figura 8.4 – Arquitetura modular metasploit (Fonte: Singh, Abhinav, 2012).**



# 5 CONSIDERAÇÕES FINAIS

## 5.1 CONCLUSÃO

Neste trabalho foram apresentados aspectos relacionados à segurança da informação, tais como processos envolvidos e características de ataques e defesas. Também foram apresentados métodos e técnicas que crackers utilizam para analisar e realizar ataques em sistemas de informação.

No que se refere a matérias, foram apresentados o projeto OpenBSD e a suas demais ferramentas como packet filtere, OpenSSL e o sistema operacional OpenBSD; para *load balance* e *failhover*.

Foram apresentados três cenários de rede de computadores para o firewall sendo este concluído com suas respectivas configurações, foram apresentadas também configurações para VPN, *load balance* e *fail hover* estes configurados utilizando o packet filter. Foram necessárias a instalação e implementação de script de automatização do snort, implementação de script para *honeypot* e o desenvolvimento de interface administrativa simples.

Todo este conjunto de configuração no sistema operacional OpenBSD resulta em um gateway de segurança que atende as necessidades de segurança básicas de uma rede de computadores.

## 5.2 TRABALHOS FUTUROS/CONTINUAÇÃO DO TRABALHO

Para trabalhos futuros, pode ser desenvolvido pesquisa e implementações de assinaturas de ameaças para o Snort, softwares de analise de logs para rápida remediação após

uma identificação de ataques, estudos de caso de balanceamento de carga de rede em ambientes de alta disponibilidade e desenvolvimento de regras para o snort.

# 6 REFERÊNCIAS BIBLIOGRÁFICAS

ARAGÃO, Francisco, (2011). "Metasploit - Sabe o que é" Disponível em: <http://pplware.sapo.pt/internet/metasploit-sabe-o-que-e/>.

BUKAC, Vít, (2010). "IDS System Evasion Techniques". Disponível em <http://is.muni.cz/th/172999/fi_m/MT_Bukac.pdf>.

Computerworld, (2012). "Um milhão de dólares por vulnerabilidades no Chrome". Disponível em: <http://www.computerworld.com.pt/2012/03/01/um-milhao-de-dolares-por-vulnerabilidades-no-chrome/>.

Dicionário do Aurélio, (2013). "hacker" Disponível em: <http://www.dicionariodoaurelio.com/Hacker.html>.

EECKHOUTTE, Peter Van, (2009). "Exploit writing tutorial – Stack Based Overflow". Disponível em: <https://www.corelan.be/index.php/2009/07/19/exploit-writing-tutorial-part-1-stack-based-overflows/>.

HANSTEEN, Peter N.M, (2008). "The book of PF".

HERZOG, Pete. (2010), "OSSTMM3 – The open Source Security Testing Methodology Manual". Disponível em <http://www.isecom.org/mirror/OSSTMM.3.pdf>.

KIOSKEA, (2013). "DMZ (Zona desmilitarizada)". Disponível em: <http://pt.kioskea.net/contents/protect/dmz-cloisonnement.php3>.

LUCAS, Michael W., (2003). "Absolute OpenBSD".

KENT, Anderson (2008), “A business Model for Information Security”. Disponível em: <http://www.isaca.org/Journal/Past-Issues/2008/Volume-3/Pages/A-Business-Model-for-Information-Security1.aspx>. Acesso em: 12 de junho de 2012.

MALERBA, César, (2010). “Vulnerabilidades e exploits: técnicas, detecção e prevenção”. Disponível em: <http://www.lume.ufrgs.br/bitstream/handle/10183/26337/000757768.pdf?sequence=1>.

MARCIANO, João Luiz,. Lima-Marques, Mamede, (2006). “O enfoque social da segurança da informação”. Disponível em: <http://www.scielo.br/pdf/ci/v35n3/v35n3a09.pdf>.

MARQUES, Eduardo, (2010). “Hacker, oque é?”. Disponível em: <http://www.tiespecialistas.com.br/2010/11/hacker-o-que-e/#.UTqKSByZcnJ>.

MAZZOCCHIO, Daniele, 2009. “Building VPNs on OpenBSD”. Disponível em <http://www.kernel-panic.it/openbsd/vpn/>.

MCCARTHY, Linda,. Weldon-Siviy, (2010). “Own your space – Compliments of Microsoft”. Disponível em: <http://download.microsoft.com/download/3/C/B/3CBADB0A-19AA-4959-8ABA-4B43756CF219/Own%20Your%20Space%20Teen%20Book/Own%20Your%20Space%20Teen%20Book%20_All%20Chapters.pdf>.

Nmap, (2012). Disponível em: <http://nmap.org>

Openbsd, (2012). "PF: The OpenBSD Packet Filter". Disponível em: <http://www.openbsd.org/faq/pf/pt/>.

Open Information Systems Security Group, (2005). “Information Systems Security Assessment Framework (ISSAF)”. Disponível em: <http://www.oissg.org/wiki/index.php?title=ISAAF-PENETRATION_TESTING_FRAMEWORK>.

Priberam Informática, (2012). “hacker”. Disponível em: <http://www.priberam.pt/dlpo/default.aspx?pal=hacker>.

Prithak, (2011). “Blocking nmap scans wtih pf and iptables”. Disponível em: <http://prithak.blogspot.com.br/2011/12/blocking-nmap-scans-with-pf-and.html>.

PROMON (2006), “Segurança da informação – Um diferencial determinante na competitividade das organizações”. Disponível em: <http://www.promon.com.br/portugues/noticias/download/Seguranca_4Web.pdf>.

REHMAN, Rafeeq Ur, (2003). "Advanced IDS Techniques Using Snort, Apache, MySQL, PHP, and ACID" Disponível em: <http://ptgmedia.pearsoncmg.com/imprint_downloads/informit/perens/0131407333.pdf>

RFC1928, (1996). “Address Allocation for Private Internets”. Disponível em: <http://tools.ietf.org/html/rfc1918>

ROUSE, Margaret, (2007). “Gray hat (or grey hat)”. Disponível em: <http://searchsecurity.techtarget.com/definition/gray-hat>.

Securimetric (2001), “Understanding Computer Viruses”.

SINGH, Abhinav, (2012). "Metasploit Penetration Testing Cookbook".

WEAVER, Paxson, Staniford, Cunningham (2005), “A Taxonomy of Computer Worms Summary”.