UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO

WEBSERVICE HONEYPOT UTILIZANDO A BIBLIOTECA JHONEY

RICARDO DE LIMA

BLUMENAU
2015

2015/1-26

RICARDO DE LIMA

WEBSERVICE HONEYPOT UTILIZANDO A BIBLIOTECA JHONEY

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Ciência da Computação do Centro de Ciências Exatas e Naturais da Universidade Regional de Blumenau como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Prof. Francisco Adell Péricas, Mestre - Orientador

BLUMENAU
2015

2015/1-26

WEBSERVICE HONEYPOT UTILIZANDO A BIBLIOTECA JHONEY

Por

RICARDO DE LIMA

Trabalho de Conclusão de Curso aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II pela banca examinadora formada por:

Presidente:	Prof. Francisco Adell Péricas, Mestre – Orientador, FURB
Membro:	Prof. Paulo Fernando da Silva, Mestre – FURB
Membro:	Prof. Miguel Alexandre Wisintainer, Mestre – FURB

Blumenau, 09 de julho de 2015

Dedico este trabalho à minha família, por acreditar em mim e investir em mim. Mãe, seu cuidado e dedicação foi que deram, em alguns momentos, a esperança para seguir. Pai, sua presença significou segurança e certeza de que não estou sozinho nessa caminhada.

AGRADECIMENTOS

À minha família que sempre me apoiou e incentivou para que eu pudesse estudar cada vez mais e alcançar meus objetivos.

Ao meu orientador em especial, pela confiança na ideia e auxílio durante os semestres em que este trabalho foi desenvolvido.

Aos meus amigos pelas ajudas prestadas em todos os momentos e que contribuíram para a conclusão deste trabalho.

Ao colegiado do curso de Ciência da Computação da FURB, pelo aprendizado adquirido durante a minha graduação.

Que os vossos esforços desafiem as impossibilidades, lembrai-vos de que as grandes coisas do homem foram conquistadas do que parecia impossível.

Charles Chaplin

RESUMO

O objetivo deste trabalho é disponibilizar uma ferramenta de simulação de serviços de rede para que sejam expostos na internet e recebam ataques. Ao receber um ataque fingindo-se ser um serviço real, respondendo as requisições do atacante e registrando informações como endereço de origem, comandos e *scripts* utilizados para o ataque, é possível estudar o comportamento dos *hackers* a fim de melhorar a segurança de uma rede. Para o desenvolvimento deste trabalho foi utilizada uma biblioteca em Java que simula alguns serviços que podem ser configurados de acordo com o tipo de captura, por exemplo: um servidor FTP ou *Web*. A ferramenta funciona de forma oculta em uma rede, com base nisso, qualquer acesso feito a ela é considerado um ataque. Pode ser aplicada em uma rede local para identificar ataques dentro de uma organização ou pode ser aplicada em uma extremidade com a internet, para receber ataques externos. As informações geradas pela ferramenta ficam armazenadas em um banco de dados para que possam ser consultadas no sistema.

Palavras-chave: Segurança de redes. *Honeypot*. IDS.

ABSTRACT

The goal of this study is to provide a network service simulation tool to be exposed on the Internet and receive attacks. Upon receiving an attack pretends to be a real service by answering the requests of the attacker and recording information such as source address, commands and scripts used for the attack, it is possible to study the behavior of hackers in order to improve the security of a network. To develop this work we used a Java library that simulates some services that can be configured according to the type of capture, eg. An FTP or Web server The tool works in a hidden way in a network, based on this, any access made it is considered an attack. It can be applied in a local network to identify attacks within an organization or can be applied directly to the Internet, to receive external attacks. The information generated by the tool are stored in a database and can be consulted in the system.

Key-words: Network security. Honeypot. IDS.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

CERT.BR - Grupo de Resposta a Incidentes de Segurança para a Internet Brasileira

DMZ - *Demilitarized Zone*

DOS - *Denial of Service*

EA - Enterprise Architect

IDS - *Intrusion Detection System*

IP - *Internet Protocol*

IPS - *Intrusion Prevention System*

UML - *Unified Modeling Language*

TTL - *Time to Live*

SUMÁRIO

1 INTRODUÇÃO

O ativo mais valioso atualmente em uma organização é a informação. Ao se informatizar tudo a nossa volta, tornando a internet um universo sem limite, o cuidado com essas informações passou a ter muita importância (CAMPOS, 2007).

Costa (2013, p. 13) comenta que essa preocupação se deve aos riscos que os usuários e empresas correm ao trafegar dados desprotegidos pela internet. Os chamados *hackers* monitoram a internet em busca de fragilidades em redes e sistemas para poder roubar informações valiosas ou sigilosas.

A segurança das informações é um desafio também para os países, pois ataques às redes e sistemas do governo podem comprometer informações e parar serviços críticos da infraestrutura de um país. Os setores de energia, defesa, transporte, telecomunicações e finanças, se não estiverem funcionando 100% do tempo e com segurança, podem afetar gravemente a economia de um país (MANDARINO JUNIOR; CANONGIA, 2010).

Buscando entender melhor o comportamento desses *hackers* e proteger-se de suas invasões, existem sistemas classificados como *Intrusion Detection System* (IDS), que têm por objetivo simular falhas de segurança em redes ou serviços. Dentre esses sistemas existe o *honeypot:* um sistema que já não é mais considerado como experimental, sendo utilizado em grandes projetos para coleta de informações sobre invasões (ASSUNÇÃO, 2009).

Para o desenvolvimento deste trabalho será adaptada a biblioteca JHoney, desenvolvida por Hagelback (2004), que utiliza a técnica *honeypot* para monitorar o comportamento dos invasores ao invadir um sistema e entender as técnicas usadas nos ataques, armazenar informações sobre os endereços de origem, os possíveis disfarces, comandos ou programas usados na invasão, armazenar essas informações em um banco de dados para análises e comparações posteriores e também a comunicação com *firewalls* para bloqueio dos endereços que fizeram os ataques.

1.1 OBJETIVOS

O objetivo deste trabalho é disponibilizar uma ferramenta para simular serviços de rede, coletar dados de invasores e melhorar a segurança de uma rede utilizando a biblioteca JHoney (HAGELBACK, 2004).

Os objetivos específicos são:

a) permitir que o *honeypot* responda a requisições enviadas a ele;

b) persistir os dados coletados pelo *honeypot*;

c) configurar o sistema para enviar alertas via e-mail;

d) gerar um um arquivo *blacklist* para que seja lido por um *firewall*.

1.2 ESTRUTURA

Este trabalho está subdividido em quatro capítulos, sendo o primeiro a introdução ao tema abordado, os objetivos e a estrutura deste trabalho.

O segundo capítulo aborda a fundamentação teórica, explicando os conceitos gerais sobre segurança de redes para permitir um melhor entendimento do trabalho.

O terceiro capítulo apresenta o desenvolvimento da ferramenta, mostrando seus requisitos, especificação de casos de uso e diagramas.

O quarto refere-se às conclusões do trabalho e sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo é apresentada a fundamentação teórica necessária para o entendimento do assunto, abordando assuntos relativos à segurança de redes, *Intrusion Detection System* - IDS, *Intrusion Prevension System* - IPS, *firewall*, segurança por obscuridade, *honeypot*, *honeynet* e trabalhos correlatos.

2.1 SEGURANÇA DE REDES

Segundo Andrucioli (2005), os avanços tecnológicos trouxeram maior comodidade, facilidade e agilidade para qualquer cidadão conectado à internet. Em contrapartida, esse avanço veio acompanhado de um problema muitas vezes oculto, o crime digital. Este tipo de crime pode envolver diversos cenários, desde o roubo de informações pessoais de uma pessoa qualquer até o roubo de informações confidenciais de uma grande organização.

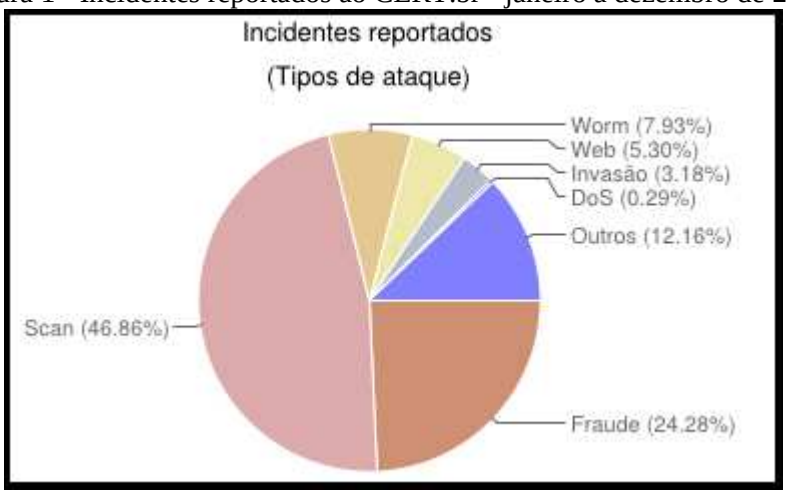
Um usuário desprotegido na internet apresenta os seguintes riscos (BERNSTEIN, 1997, p. 22):

a) senhas poderão ser roubadas quando se conectar a outros sistemas;
b) linhas de comunicação podem ser grampeadas e informações secretas da empresa poderão ser comprometidas;
c) sistemas poderão ser violados e as contas de sistema poderão ser sequestradas;
d) a rede poderá ser inundada com informações e acabar entrando em pane;
e) a propriedade intelectual poderá ser roubada;
f) os direitos autorais e patentes poderão ser infringidos;
g) documentos confidenciais poderão ser divulgados;
h) os funcionários poderão ser pegos acessando pornografia;
i) as finanças também poderão ser alteradas;
j) os fundos poderão ser desviados;
k) alguém poderá se passar por outra pessoa e efetuar transações financeiras em seu nome.

O CERT.BR é o Grupo de Resposta a Incidentes de Segurança para a Internet Brasileira, mantido pelo NIC.BR, do Comitê Gestor da Internet no Brasil. É responsável por tratar incidentes de segurança em computadores que envolvam redes conectadas à Internet brasileira. O CERT.BR divulga estatísticas sobre os incidentes de segurança reportados no Brasil. Na Figura 1 segue gráfico onde é possível ver que a porcentagem de incidentes do tipo *Scan* corresponde a maior parte de todos os outros tipos.

Segundo o CERT.BR (2015), incidentes do tipo *Scan* caracterizam-se por notificações de varreduras em redes de computadores com o intuito de identificar quais computadores estão ativos e quais serviços estão sendo disponibilizados por eles. É amplamente utilizado por atacantes para identificar potenciais alvos, pois permite associar possíveis vulnerabilidades aos serviços habilitados em um computador.

Figura 1 - Incidentes reportados ao CERT.br - janeiro a dezembro de 2013



Fonte: CERT (2015).

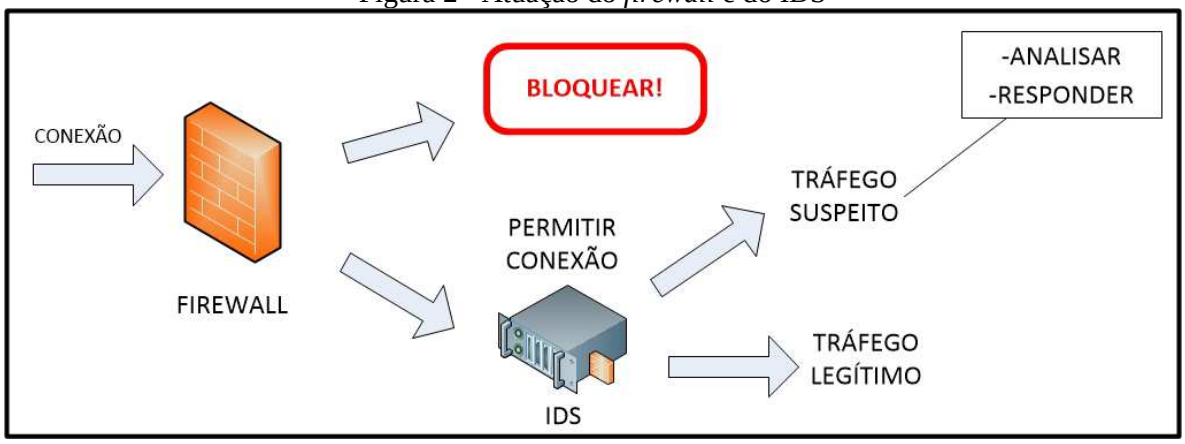
2.2 *INTRUSION DETECTION SYSTEM* - IDS

Conhecer os primeiros passos que um *hacker* executa para contornar *firewalls* ajuda muito a detectar e reagir a um ataque (MCCLURE; SCAMBRAY; KURTZ, 2000, p. 312).

IDS, em português significa Sistema Detector de Intrusão, é uma categoria de software que tem por objetivo detectar possíveis ataques maliciosos à rede. Seus sensores devem ser colocados em vários pontos da rede, principalmente nos pontos críticos que fazem a divisa da rede local com a internet. O IDS se baseia em comportamentos pré-definidos de eventos maliciosos, chamados assinaturas (DUARTE; JABOUR, 2003).

Os sistemas de detecção de intrusão (Figura 2) são mecanismos para monitoramento de ataques ou acessos indevidos a redes, com a capacidade de fornecer resposta. Um IDS tem como funções a coleta e análise de informações, além da geração de alertas e alarmes (NUNES, 2009).

Figura 2 - Atuação do *firewall* e do IDS



Na Figura 2, é exemplificada uma rede protegida por um *firewall* e um IDS, onde uma conexão de entrada passa pelo filtro do *firewall* e é bloqueada caso a porta de conexão não esteja liberada. Se a porta de conexão estiver liberada ela passa pelo IDS, que faz a verificação do tráfego, caso ele seja legítimo chegará ao destino sem outras verificações.

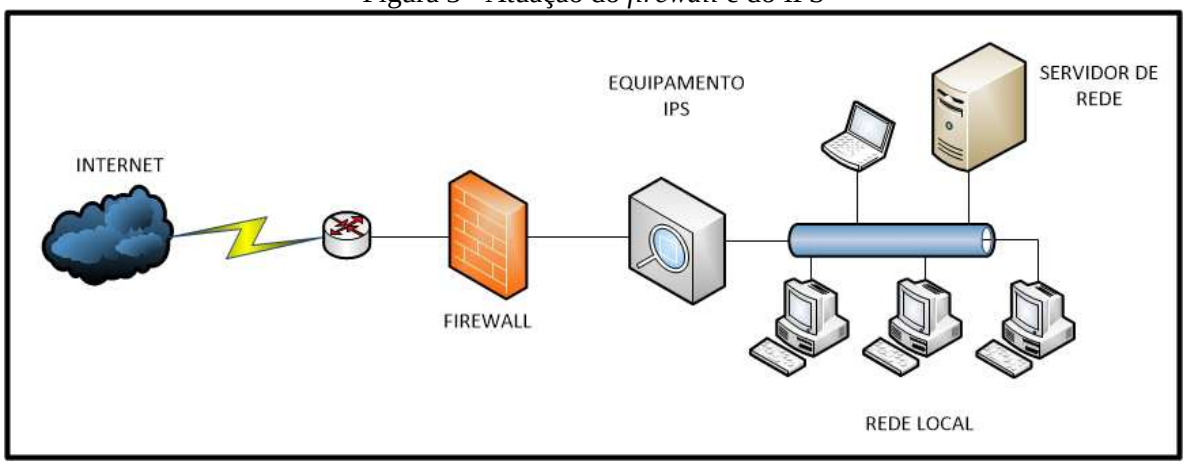
Porém se o tráfego for suspeito, o IDS fará a análise e responderá ao invasor modificando a resposta, afim de não permitir que informações importantes sejam enviadas até o invasor.

2.3 *INTRUSION PREVENSION SYSTEM* - IPS

O IPS, em português significa Sistema de Prevenção de Intrusão, tem por objetivo detectar e bloquear ataques a recursos protegidos da rede. Ele complementa o IDS, que apenas identifica e analisa atividades maliciosas (MAIA; REHEM, 2005).

Trabalhando aliado a um *firewall*, o IPS tem a capacidade de detectar um ataque antes mesmo que ele tenha sucesso, pois ele se posiciona diferente de um IDS para que possa analisar todo o tráfego que entra pela extremidade do *firewall*, conforme a Figura 3 (NUNES, 2009).

Figura 3 - Atuação do *firewall* e do IPS



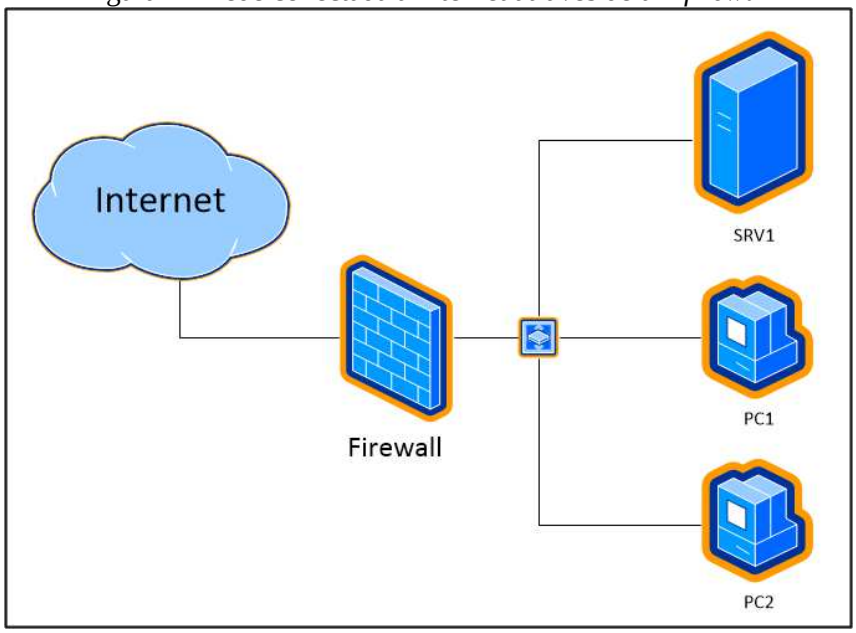
Na Figura 3 é apresentada uma rede protegida por um *firewall* e um IPS, onde uma conexão que passa pelo filtro do *firewall* é verificada e, caso for constatado que é um ataque, ela passa a ser descartada pelo IPS, não chegando ao destino que são os dispositivos conectados a rede interna.

2.4 FIREWALL

Segundo Berstein (1997, p. 176), os *firewalls* têm a finalidade de filtrar acessos e proteger uma rede no caso de uma invasão através da internet. Em outras palavras, o *firewall* faz o intermédio entre duas redes, com a capacidade de verificar o tráfego entre elas.

A melhor maneira de entender como age um *firewall* é imaginá-lo como sendo um guarda, ou sentinela, da rede, que inspeciona a documentação para os pacotes que chegam e depois decide se dará passagem ou não. A Figura 4 apresenta um exemplo de uma rede com um *firewall* em sua extremidade.

Figura 4 - Rede conectada a internet através de um *firewall*



Na Figura 4 é mostrada uma rede protegida por um *firewall* que faz o papel de filtragem de pacotes que entram e saem da rede. A filtragem é feita verificando as portas e endereços de destino das requisições que chegam até o *firewall*: caso existam regras de liberação ou bloqueio configuradas no *firewall*, ele aplicará essas regras de acordo com o destino dos pacotes.

2.5 SEGURANÇA POR OBSCURIDADE

Segurança por obscuridade é utilizada em redes e softwares para aumentar a segurança. Mesmo que nem sempre atinja seu objetivo principal, ela não pode ser deixada de

lado. Segundo Assunção (2009, p. 24), "a segurança por obscuridade depende basicamente de um segredo, esse segredo é o *honeypot*, assim qualquer acesso destinado a ele será considerado um ataque, eliminando falso-positivos".

Para entender melhor o conceito, ao se adicionar um servidor de arquivos em uma rede, ele não receberá acessos se não avisar os usuários e estações da sua existência, permanecendo oculto na rede. Ele poderá ser encontrado somente através de um escaneamento dessa rede, que pode ser feito por algum usuário malicioso ou alguma máquina infectada tentando explorar a rede, a partir desse princípio, conclui-se que qualquer acesso a esse servidor deve ser considerado uma invasão (ASSUNÇÃO, 2009, p. 25).

2.6 HONEYPOT

O primeiro relato do uso da técnica, que na época não tinha ainda recebido o nome de *honeypot*, é do ano de 1986, onde um famoso centro de pesquisas dos Estados Unidos começou a receber acessos não autorizados. Após a identificação de que se tratava de um ataque, os responsáveis pela rede do laboratório não quiseram bloquear o acesso ao atacante, optando por deixar o acesso aberto e monitorar o comportamento do *hacker* para poder se proteger em um futuro ataque (BATISTELA; TRENTIN, 2009, p. 2).

O conceito mais aceito para a técnica *honeypot* é o citado por (SPITZNER, 2002): *honeypot* é um sistema de segurança que disfarça seus valores ao ser testado, atacado ou comprometido.

A técnica permite monitorar com eficiência os ataques a uma rede ou máquina, possuindo diversas estruturas para captura das informações. Pode ser implementada de várias formas a fim de se adaptar ao tipo de rede ou máquina na qual estará monitorando. (ANDRUCIOLI, 2005).

> A principal vantagem de um *honeypot* é que ele não depende de assinaturas, regras ou algoritmos avançados, ele simplesmente captura os ataques que lhe são direcionados, facilitando a análise e correlação da pequena, mas valiosa, coleção de dados. Mas possui algumas desvantagens, sua capacidade limitada de visão, ou seja, ele só captura os dados direcionados a ele mesmo. E ao risco, toda vez que uma nova tecnologia é introduzida existe o risco deste recurso ser quebrado, no caso específico dos *honeypots*, ser usado para atacar outros sistemas. (DUARTE; JABOUR, 2003).

Por esses motivos o *honeypot* não é uma ferramenta de proteção, não podendo substituir de forma alguma as ferramentas específicas de proteção que existem. Seu papel é complementar a segurança e aumentar seu valor. (DUARTE; JABOUR, 2003).

As principais funcionalidades de um sistema *honeypot*, segundo Andrucioli (2005, p. 3), são:

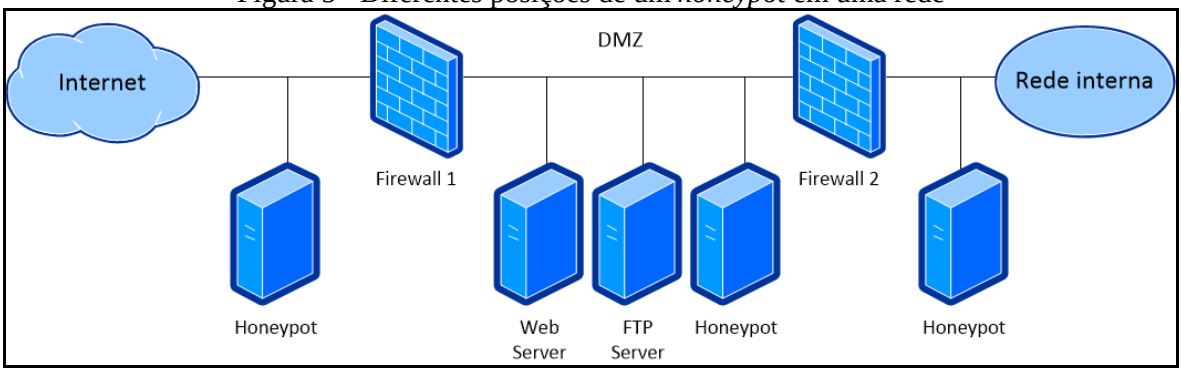
a) tráfego de ataques separados de uma rede real de produção;
b) emulação de serviços e/ou sistemas, quando necessário;
c) utilização de diversas ferramentas com o objetivo de capturar informações relevantes;
d) possibilidade de estudar o atacante e suas técnicas, antes e após uma invasão bem sucedida;
e) captura completa de todos os pacotes envolvidos em um ataque.

Segundo Assunção (2009, p. 28), a Figura 5 apresenta graficamente diferentes posições que o *honeypot* pode assumir, dependendo da estrutura da rede:

a) antes do primeiro *Firewall*: nessa posição o *honeypot* estará mais exposto, pois ficará antes de qualquer proteção (*firewall*) da rede interna. Esse posicionamento é proposital para que receba facilmente mais ataques. É característica de um *honeypot* para pesquisas;
b) na *Demilitarized Zone*[1] (DMZ): nessa posição o *honeypot* não estará completamente exposto, mas estará no mesmo nível dos outros servidores da rede (DMZ), aumentando a efetividade do disfarce e é a principal posição quando usado para auxiliar na proteção de ataques da internet a uma rede local;
c) após o segundo *firewall*: nessa posição o *honeypot* dificilmente receberá um ataque vindo da internet, mas o objetivo dele é identificar ameaças internas na rede.

[1]*Demilitarized Zone*: em português significa zona desmilitarizada, é uma sub-rede que pode ser uma sub-rede física ou lógica que contém e expõe serviços externos de uma organização para acesso a uma rede maior não confiável, por exemplo a internet.

Figura 5 - Diferentes posições de um *honeypot* em uma rede



2.7 HONEYNET

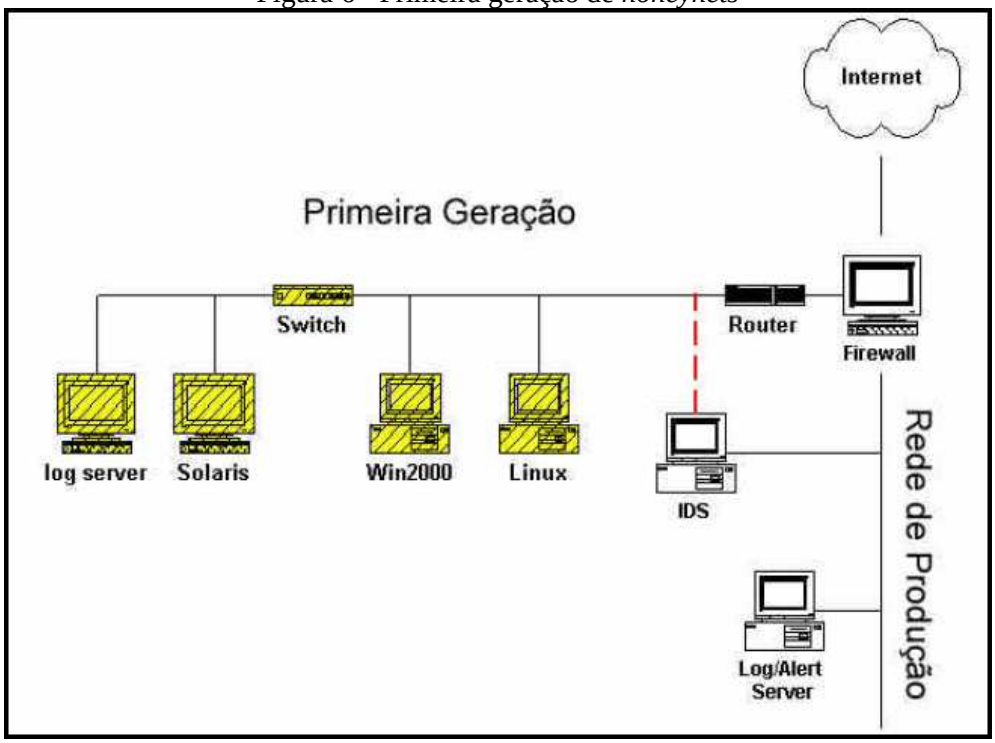
É possível também ampliar a atuação de um *honeypot* tornando a armadilha ainda mais eficaz, criando uma rede inteira com vários *honeypots*, Essa rede tem como objetivo servir inteiramente de armadilha para os ataques, a capacidade de captura de dados será muito maior e será mais difícil para o atacante descobrir ou perceber que não se trata de uma rede de produção (ASSUNÇÃO, 2009, p. 31).

Todos os computadores dentro dessa rede serão *honeypots*, todo o tráfego de entrada é considerado malicioso e todo tráfego de saída é iniciado pelo invasor. Com base nesses fatos, a análise terá apenas dados de verdadeiros ataques, eliminando falso-positivos (ROCHA, 2006).

Segundo Duarte e Jabour (2003), *honeynets* podem ser divididas em duas categorias:

a) *Honeynet* Gen I: a primeira geração possui um controle e captura de dados simples e o nível de informações coletadas é limitado. Tem um *firewall* como ferramenta primária, sendo ele também o *gateway* da rede, permitindo a total entrada dos dados na *honeynet* e limitando a saída, para que o atacante não use a *honeynet* para iniciar ataques a outras redes. Um roteador é colocado entre o *firewall* e a *honeynet* para esconder o *firewall*, pois após a invasão o atacante pensará que apenas o roteador está dividindo a rede interna da internet, tornando a situação mais realista (Figura 6).

Figura 6 - Primeira geração de *honeynets*

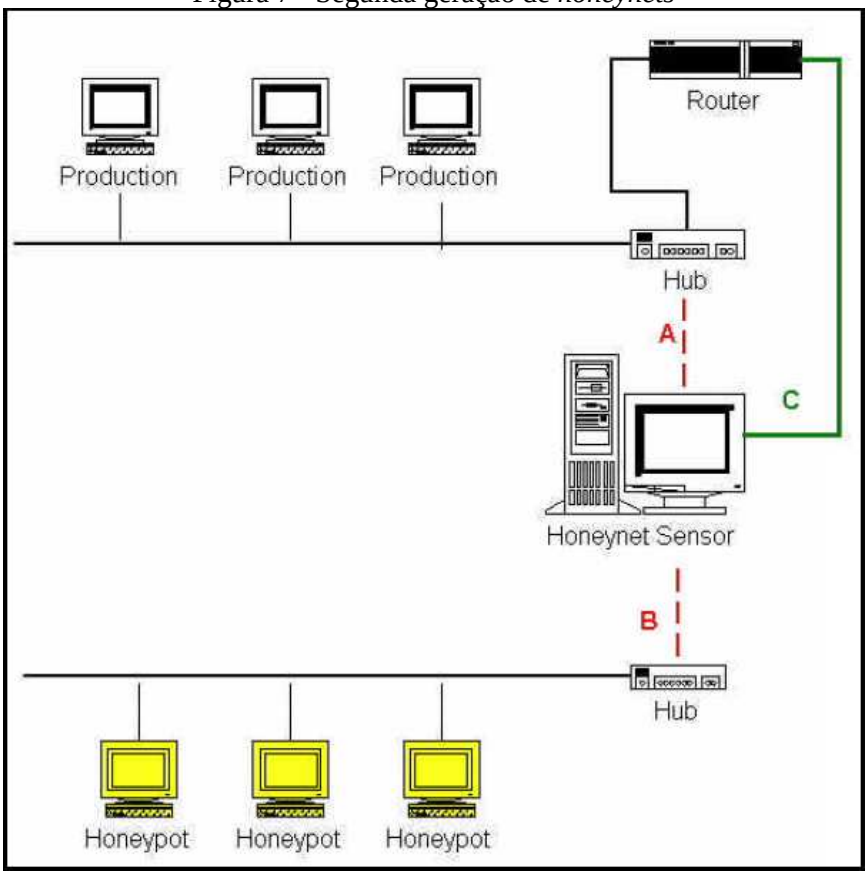


Fonte: Duarte e Jabour (2003).

b) *Honeynet* Gen II: na segunda geração houve uma melhoria nos processos de controle e captura de dados e o foco principal foi criar uma forma de facilitar a implementação da solução e deixá-la mais difícil de ser detectada. Em uma *honeynet* de segunda geração o controle e captura dos dados acontece em apenas um dispositivo, o *gateway* que será responsável por essas duas funções, sendo configurado em modo *bridge*[2] sem rotear ou aumentar o TTL (*Time to Live*) dos pacotes, dessa forma o atacante não consegue perceber que o tráfego está sendo analisado. A resposta das ações não autorizadas melhorou. E os ataques disparados de dentro da *honeynet* não são mais bloqueados, são modificados. Assim esses ataques não têm mais eficácia, os pacotes são modificados quando passam pelo *gateway* (Figura 7).

[2] Modo *bridge*: significa que os pacotes recebidos são diretamente enviados para a outra ponta em que ele está conectado, não é feito nenhum controle de rota, um equipamento em modo *bridge* apenas recebe a informação de um lado e entrega do outro.

Figura 7 - Segunda geração de *honeynets*



Fonte: Duarte e Jabour (2003).

2.8 TRABALHOS CORRELATOS

Existem trabalhos acadêmicos que utilizam a técnica do *honeypot* para detecção de invasões, avaliando modelos de estrutura e empregando o *honeypot* para proteção de um sistema de comércio eletrônico contra ataques DoS. Destacam-se: Proposta e Avaliação de um Modelo Alternativo Baseado em *Honeynet* para Identificação de Ataques e Classificação de Atacantes na Internet (ANDRUCIOLI, 2005) e Protegendo Plataforma de Comércio Eletrônico Contra Ataques DoS Utilizando *HoneyPot* (COSTA, 2013).

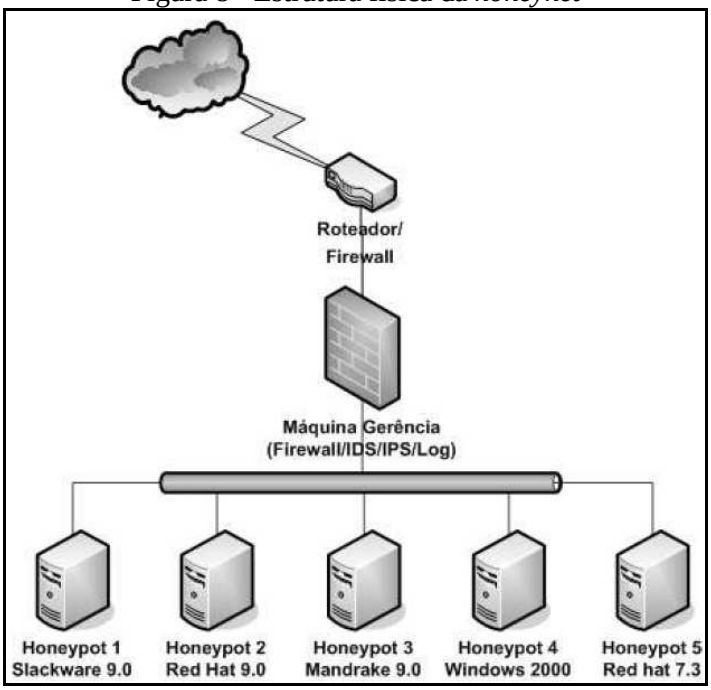
2.8.1 Modelo Alternativo Baseado em *Honeynet*

Andrucioli (2005) apresentou um trabalho com o objetivo de construir um *honeynet* como ferramenta de captura e monitoramento de ataques pela internet. Com os dados coletados sobre os ataques sofridos, o autor faz uma avaliação e classificação dos ataques em relação aos riscos apresentados. O autor implementou um *honeynet* composto por um roteador/*firewall* (utilizando iptables[3]), uma máquina de gerência (com um snort[4] como IDS)

[3] Sistema de filtragem padrão de pacotes da rede do linux.

e cinco *honeypots* (máquinas físicas exclusivas para serem atacadas) responsáveis pela obtenção dos ataques (Figura 8).

Figura 8 - Estrutura física da *honeynet*



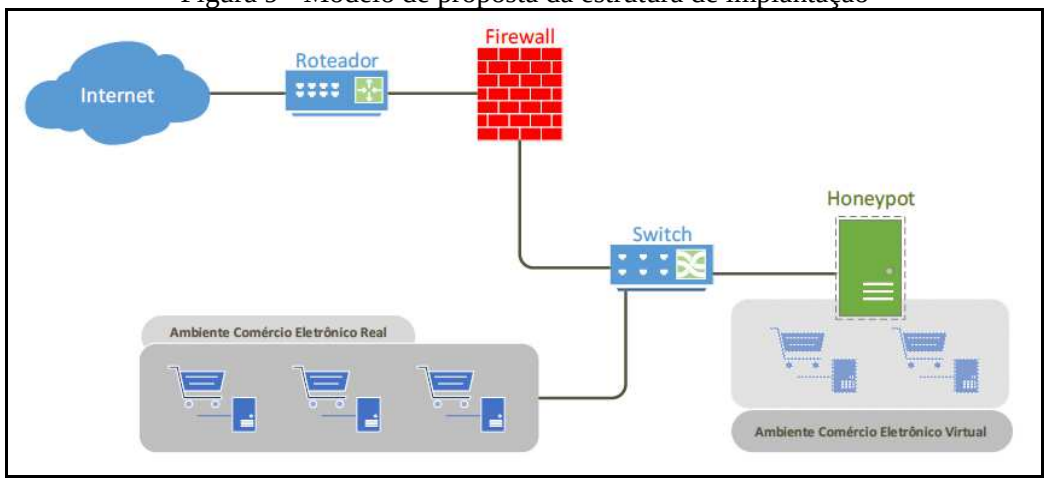
Fonte: Andrucioli (2005).

2.8.2 Plataforma de Comércio Eletrônico Contra Ataques DoS

Costa (2013) apresentou um Trabalho de Conclusão de Curso (TCC) com o objetivo de implantar um *honeypot* para prover disponibilidade em uma plataforma de comércio eletrônico. Um dos ataques mais comuns a serviços na internet é o de negação de serviço ou DoS e, para defender o ambiente de comércio eletrônico, Costa (2013) elaborou o modelo de implantação apresentado na Figura 9.

[4] Snort: é uma ferramenta de código aberto para monitorar o tráfego de pacotes em redes IP, realizando análises em tempo real sobre diversos protocolos (nível de rede e aplicação).

Figura 9 - Modelo de proposta da estrutura de implantação



Fonte: Costa (2013).

Para o desenvolvimento do trabalho, Costa (2013) utilizou o Arpd para fazer o mapeamento da rede e identificar os IPs disponíveis para monitoramento. Como ferramenta *honeypot*, utilizou o Honeyd, que é desenvolvido em C e é largamente utilizado para criação de ambientes com *honeypots* e um *firewall* Mikrotik RouterOS, que vem embarcado em um *hardware* próprio e possui ferramentas específicas para servir como roteador, *firewall*, *proxy* e monitoramento de tráfego.

3 DESENVOLVIMENTO DO *HONEYPOT*

Neste capítulo são descritas as informações sobre a biblioteca utilizada, detalhes da especificação, os diagramas de casos de uso e entidade relacionamento, a operacionalidade do sistema e, ao final, os resultados e discussões.

3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

Nessa subseção são apresentados os principais requisitos funcionais (RF) e requisitos não funcionais (RNF). O Quadro 1 apresenta os requisitos funcionais e sua rastreabilidade com seus respectivos casos de uso.

Quadro 1 - Requisitos funcionais

Requisitos funcionais	**Caso de Uso**
RF01: O sistema deve permitir iniciar, parar e reiniciar o serviço *honeypot*	UC01
RF02: O sistema deve permitir consultar *logs* por data e hora de ataque	UC02
RF03: O sistema deve permitir a configuração de um e-mail para receber os alertas	UC03
RF04: O sistema deve permitir a configuração de uma *blacklist* dinâmica para comunicação com um *firewall*	UC03
RF05: O sistema deve permitir escolher a porta em que o serviço *honeypot* será iniciado	UC03
RF06: O sistema deve registrar um *log* referente à inicialização do serviço *honeypot*	UC04

O Quadro 2 apresenta os requisitos não funcionais do sistema.

Quadro 2 - Requisitos não funcionais

Requisitos não funcionais
RNF01: O sistema deve ter um cadastro de usuários para controle de acesso
RNF02: O sistema deve ser desenvolvido para a plataforma *web*
RNF03: O sistema deve ser desenvolvido utilizando a linguagem Java
RNF04: O sistema deve utilizar banco de dados MySQL

3.2 ESPECIFICAÇÃO

Nesta seção serão apresentados a biblioteca JHoney original desenvolvida por Hagelback (2004) e seus diagramas de classes. Após serão apresentados os diagramas de casos de uso desenvolvidos para esse trabalho com suas respectivas descrições, uma visão geral do *honeypot* aplicado em uma rede, seguido do diagrama de classes desenvolvidas, diagrama de atividades e diagrama de entidade-relacionamento. Para a elaboração dos diagramas foram utilizadas a linguagem de modelagem UML, a ferramenta *case* DBDesigner Fork versão 1.5 para o diagrama de entidade relacionamento e a ferramenta *online* Creately para o diagrama de casos de uso.

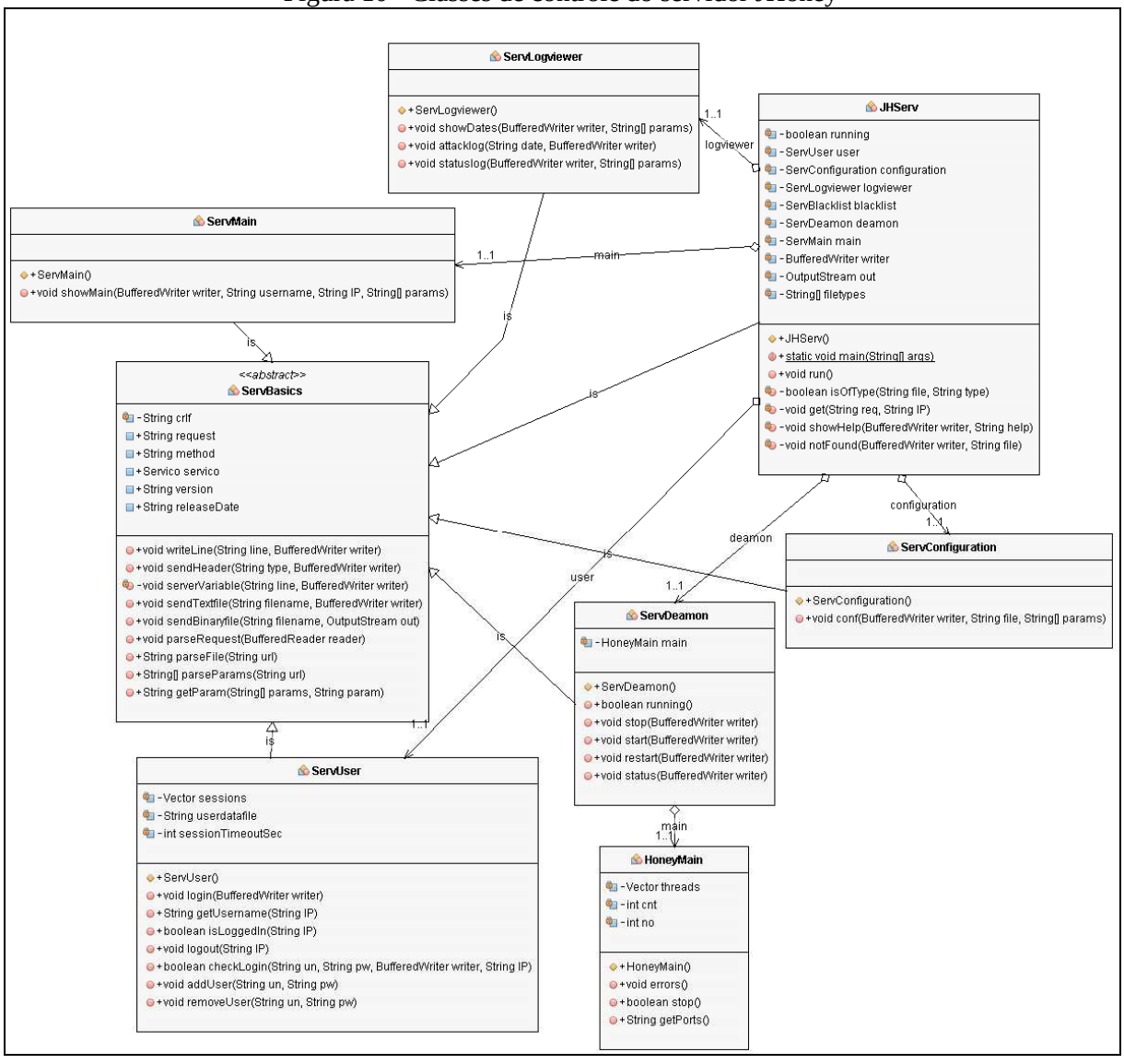
3.2.1 Biblioteca JHoney

Desenvolvida na linguagem Java por Hagelback (2004) e estendida neste trabalho, é uma ferramenta para simular vulnerabilidades em serviços de rede, deixando os serviços expostos sem qualquer segurança ou controle. Possui um servidor *web* para administração e controle do *honeypot*. Na ferramenta é possível configurar uma porta para o serviço do *honeypot*, porém ao iniciar um ataque a essa porta o invasor não receberá retorno, o que limita muito a coleta de dados, pois os invasores não ficarão muito tempo conectados.

3.2.1.1 Estrutura de classes da JHoney

Originalmente a biblioteca possui classes para controle do servidor *web*. Na Figura 10 é possível verificar as classes envolvidas nesse processo.

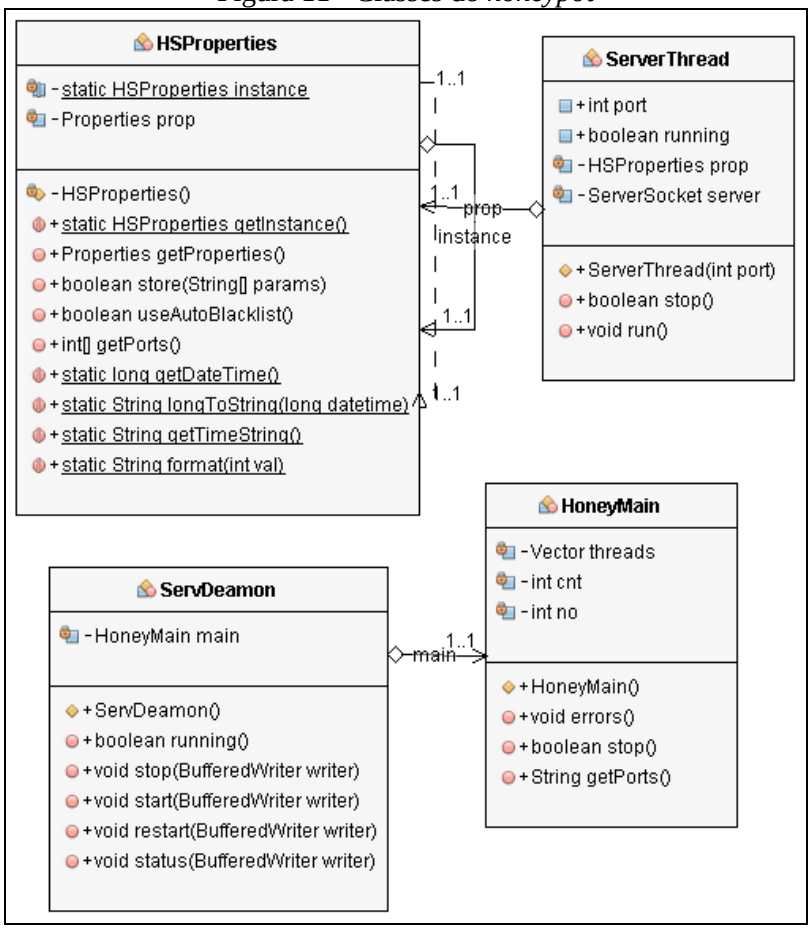
Figura 10 - Classes de controle do servidor JHoney



Das classes mostradas na Figura 10 a principal é a `JHServ`. Essa classe instancia o servidor *web* para controle do *honeypot*, recebe as requisições *web* e as responde através dos métodos descritos na `ServBasics`. A classe `ServUser` faz o controle dos usuários e sessões. E a classe `ServLogviewer` apresenta os *logs* dos ataques.

Na Figura 11 são mostradas as classes que originalmente envolvem o serviço do *honeypot* dentro da biblioteca

Figura 11 - Classes do *honeypot*

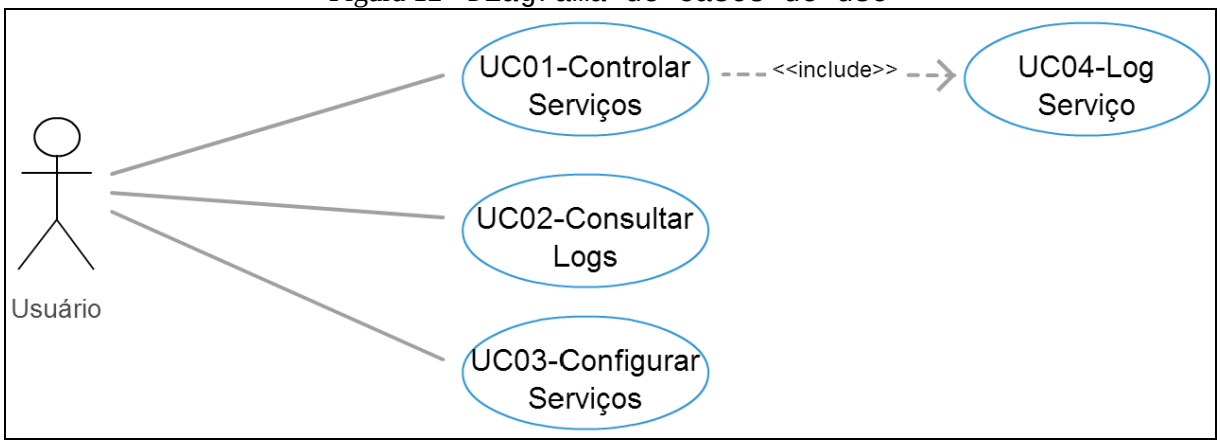


Após fazer a configuração dos parâmetros do *honeypot*, a biblioteca faz as chamas de controle do *honeypot* através dos métodos da classe `ServDeamon`, que inicia a classe `HoneyMain` para que essa controle a inicialização do serviço. A classe `HoneyMain` inicia a classe `ServerThread` passando por parâmetro a porta a ser usada pelo serviço, o serviço é iniciado por um `ServerSocket` que ficará aguardando as requisições. Por fim, a classe `HSProperties` armazena e fornece informações a respeito dos serviços para a classe `ServerThread`.

3.2.2 Diagrama de casos de uso da ferramenta *honeypot*

Nesta subseção serão apresentados o diagrama de casos de uso e a descrição das funcionalidades desenvolvidas neste trabalho (Figura 12).

Figura 12 - Diagrama de casos de uso



O caso de uso UC01 descreve a funcionalidade que permite ao usuário controlar o serviço *honeypot*, podendo iniciar, parar e reiniciar esse serviço. O Quadro 3 apresenta o detalhamento desse caso de uso.

Quadro 3 - UC01 - Controlar Serviços

UC01 - Controlar Serviços	
Descrição	Permite o usuário iniciar, parar e reiniciar o serviço do *honeypot*.
Pré-condições	Ferramenta em execução
Cenário principal	1) O usuário faz *login* no sistema. 2) O usuário clica em uma das opções de controle. 3) Se o comando for de inicialização, o sistema verifica se a porta definida no serviço não está em uso e faz a inicialização, caso o comando for de paralisação, o sistema para o serviço do *honeypot*.
Pós-condições	Aguardar um ataque ao sistema e fazer os registros.
Exceção	Se a porta do serviço já estiver em uso por outro aplicativo, o sistema mostrará uma exceção informando que a porta deve ser liberada ou alterada.

O caso de uso UC02 descreve a visualização das informações coletadas pelo *honeypot*, onde é possível escolher a data e hora dos ataques. O Quadro 4 apresenta o detalhamento desse caso de uso.

Quadro 4 - UC02 - Consultar *Logs*

UC02 - Consultar *Logs*	
Descrição	Permite o usuário visualizar os *logs* de ataques ao *honeypot*.
Pré-condições	Ferramenta em execução
Cenário principal	1) O usuário faz *login* no sistema. 2) O usuário seleciona a data do ataque na primeira tela e depois seleciona o ataque dependendo da hora em que foi recebido.
Pós-condições	Apresentar detalhamento do ataque selecionado.
Exceção	Nenhuma informação a ser apresentada.

O caso de uso UC03 descreve a funcionalidade para configurar os parâmetros de inicialização do serviço *honeypot*. O Quadro 5 apresenta o detalhamento desse caso de uso.

Quadro 5 - UC03 - Configurar Serviço

UC03 - Configurar Serviço	
Descrição	Permite o usuário configurar os parâmetros de inicialização do serviço *honeypot*.
Pré-condições	Ferramenta em execução
Cenário principal	1) O usuário faz *login* no sistema. 2) O usuário especifica a porta do serviço, a criação de uma *blacklist* dinâmica e um e-mail para receber os alertas de ataques.
Pós-condições	Salvar configurações definidas pelo usuário.
Exceção	Não há.

O caso de uso UC04 descreve a funcionalidade para consultar o *log* de inicialização do serviço do *honeypot*. O Quadro 6 apresenta o detalhamento desse caso de uso.

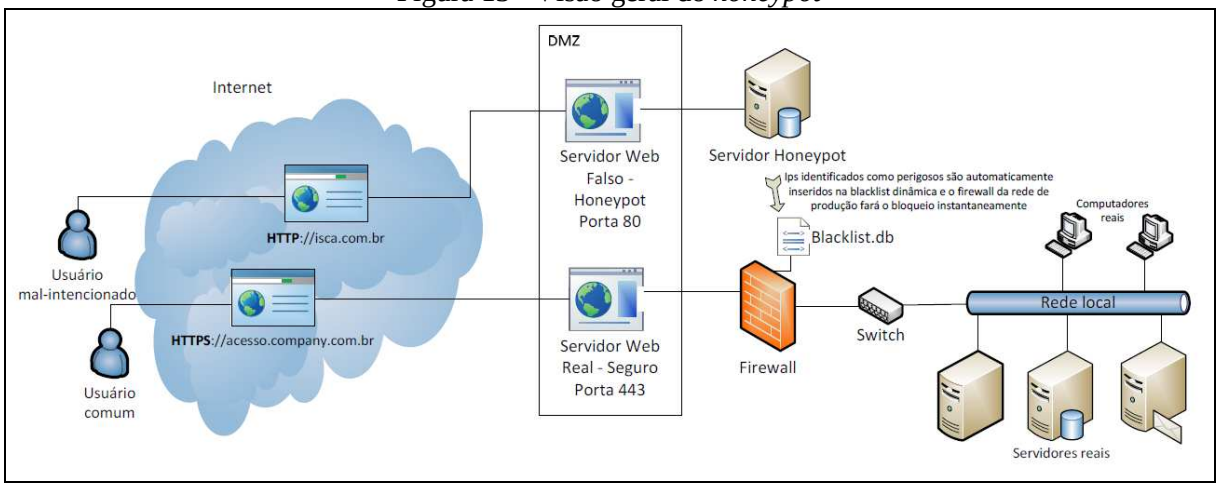
Quadro 6 - UC04 - *Log* Serviço

UC04 - *Log* Serviço	
Descrição	Permite o usuário consultar o *log* de inicialização do serviço *honeypot*.
Pré-condições	Ferramenta em execução
Cenário principal	1) O usuário inicia o serviço do *honeypot*. 2) O usuário verifica o *log* do serviço caso perceber algum erro na inicialização do mesmo.
Pós-condições	Apresenta detalhamento da inicialização do serviço.
Exceção	Não há.

3.2.3 Visão geral

Para um melhor entendimento do trabalho desenvolvido, foi ilustrada na Figura 13 uma visão geral do *honeypot* inserido em uma rede.

Figura 13 - Visão geral do *honeypot*

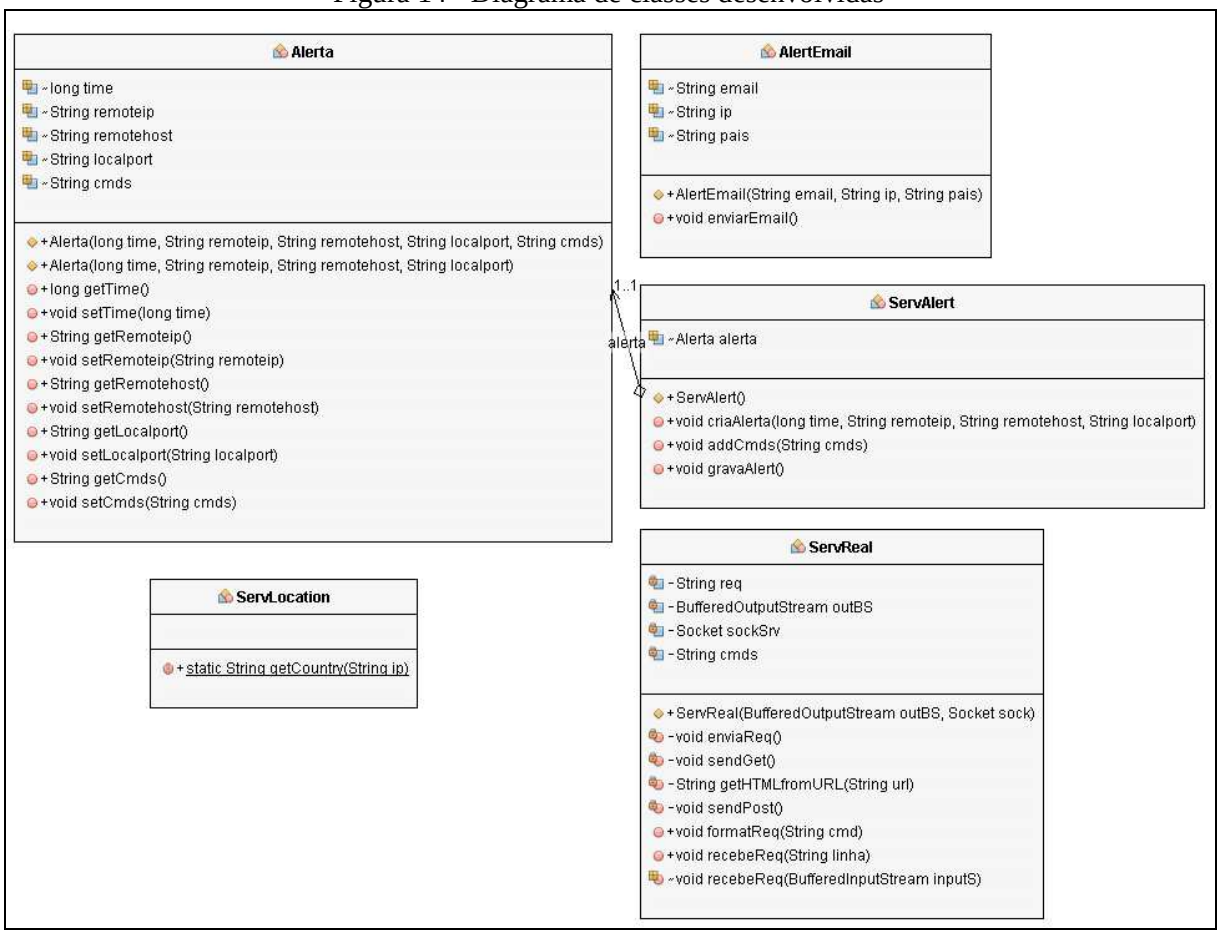


O *honeypot* foi inserido em paralelo a um servidor seguro HTTPS, utilizando o HTTP em sua porta padrão 80. Ele é um alvo fácil para um usuário mal-intencionado que tentará invadir o servidor *web*, enquanto isso o *honeypot* faz a coleta de dados do usuário mal-intencionado e ao mesmo tempo insere o seu endereço em uma lista de bloqueios do *firewall* da rede principal, onde está o servidor seguro HTTPS. Dessa forma pode-se complementar a segurança da rede principal, pois se o usuário mal-intencionado tentar um acesso a ela será impedido pelo *firewall* que já possui o endereço do atacante em sua lista de bloqueios.

3.2.4 Diagrama de classes

Na Figura 14 é apresentado o diagrama de classes que define as novas estruturas desenvolvidas na ferramenta *honeypot*.

Figura 14 - Diagrama de classes desenvolvidas



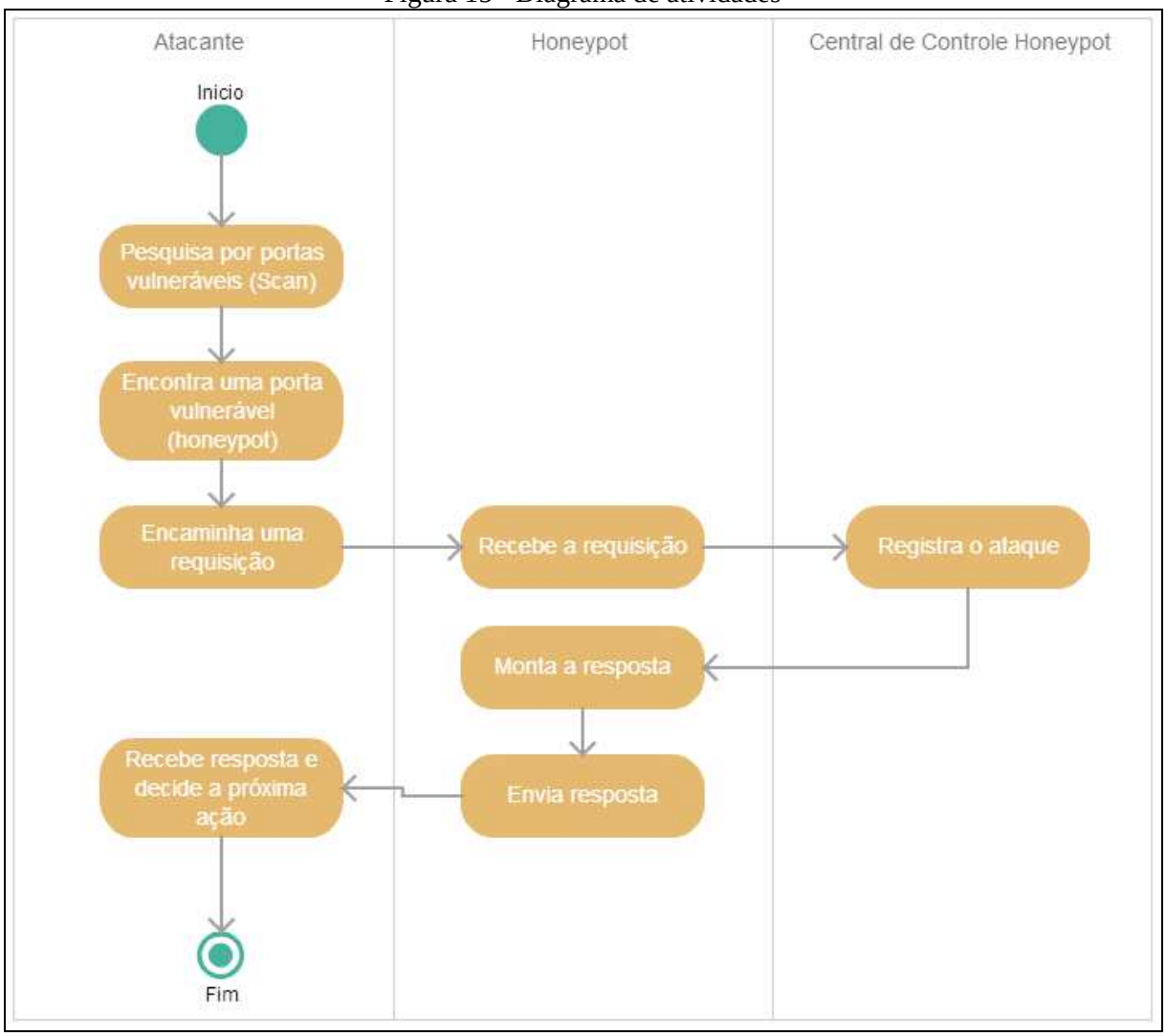
Às classes da Figura 14 foram adicionadas a ferramenta, onde a classe `ServReal` faz o recebimento das requisições encaminhadas ao *honeypot* e monta as respostas que serão enviadas ao invasor. Essa classe é importante pois implementa um dos objetivos principais ao *honeypot* que é a resposta aos ataques. A classe `ServAlert` controla a criação e gravação de alertas, ao receber um ataque o *honeypot* cria o alerta através do método `criaAlerta`, durante o ataque o honeypot adiciona os comandos recebidos através do método `addCmds` e ao final do ataque o alerta é gravado pelo método `gravaAlert`.

Em paralelo, ao receber um ataque, o *honeypot* envia um e-mail (cadastrado nos parâmetros do serviço) de alerta, avisando que o sistema foi invadido. A classe `AlertEmail` é responsável por enviar o e-mail, esse aviso é enviado apenas uma vez. Somente é enviado novamente se um novo endereço atacar o *honeypot*. A classe `ServLocation` faz uma consulta em um *webservice online* para verificar o país de origem do invasor.

3.2.5 Diagrama de atividades

Na Figura 15 é apresentado o diagrama de atividades que irá exibir o fluxo de dados quando o *honeypot* recebe um ataque até o seu retorno para o atacante.

Figura 15 - Diagrama de atividades

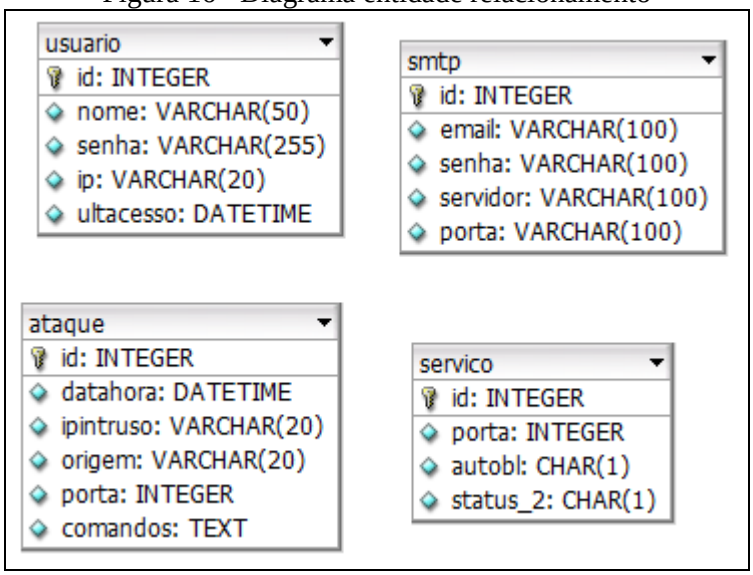


O processo inicia com o atacante fazendo uma pesquisa na rede por portas vulneráveis através de um *Scan*. Ao encontrar uma porta ele envia uma requisição, o *honeypot* recebe a requisição, registra o ataque (data, hora, IPintruso e comandos), monta a resposta (é necessário a alteração da resposta para que o atacante não perceba que é um serviço falso) e envia essa resposta para o atacante. O atacante recebe a resposta e decide se vai continuar a conexão ou encerrar.

3.2.6 Diagrama entidade relacionamento

O diagrama entidade relacionamento (DER) descreve o modelo de dados de um sistema com alto nível de abstração. A Figura 16 apresenta o DER com as entidades que serão persistidas no banco de dados do sistema.

Figura 16 - Diagrama entidade relacionamento



Para o desenvolvimento da ferramenta, foram criadas 4 tabelas, sendo:

a) tabela `usuario`: armazena os dados dos usuários para acesso ao sistema, campo `ip` para gravar o último IP de onde o usuário acessou, campo `ultacesso` para gravar a data e hora do último acesso, que é utilizado no controle de *timeout* de sessão da página de administração *web*;
b) tabela `smtp`: armazena os dados de e-mail para que o sistema possa enviar os alertas;
c) tabela `servico`: armazena a configuração do serviço *honeypot*;
d) tabela `ataque`: grava todos os ataques individualmente, coluna `datahora` armazena a data e hora do ataque, coluna `ipintruso` armazena o IP do invasor, coluna `origem` armazena o país de origem do ataque, coluna `porta` armazena a porta atacada, coluna `comandos` armazena os comandos coletados pelo *honeypot* durante o ataque.

3.3 IMPLEMENTAÇÃO

A seguir são mostradas as técnicas e ferramentas utilizadas e a operacionalidade da implementação.

3.3.1 Técnicas e ferramentas utilizadas

As técnicas e ferramentas utilizadas para o desenvolvimento da ferramenta proposta foram:

a) a linguagem de programação Java versão 7 para codificação da ferramenta;
b) a IDE NetBeans versão 8.0.2, como ambiente de desenvolvimento;

c) a biblioteca JHoney como base para o desenvolvimento da ferramenta;

d) a biblioteca JDBC MySQL Connector/J para comunicação com o banco de dados;

e) o banco de dados MySQL na versão 5.6.

A biblioteca JHoney permite fazer a configuração e controle do serviço *honeypot*, faz a leitura das requisições recebidas e fornece uma interface para acesso ao histórico.

3.3.2 Operacionalidade da implementação

Esta subseção apresenta as principais telas do sistema desenvolvido com uma breve apresentação de suas funcionalidades.

O sistema possui apenas um nível de acesso, portanto os usuários têm a mesma visão dentro do sistema, podendo consultar históricos de ataques e também configurar e iniciar o serviço do *honeypot*.

3.3.2.1 Acessando o sistema

Na tela apresentada na Figura 17, o usuário deve informar seu nome de usuário e senha para acessar o sistema.

Figura 17 - Tela de *login* do sistema

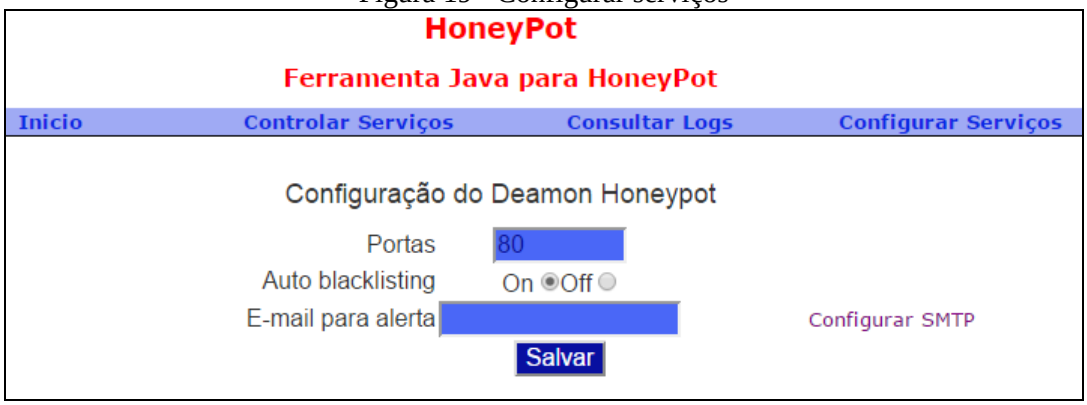
Ao acessar o sistema o usuário visualiza a tela de boas vindas e as opções do *honeypot* (Figura 18).

Figura 18 - Tela de boas vindas

3.3.2.2 Configurar Serviços

Na Figura 19 mostra a tela de configuração do serviço, onde é possível alterar a porta do *honeypot*, definir se o *honeypot* vai criar automaticamente uma *blacklist* e alimentá-la com os IPs que atacarem. Essa lista será lida por um *firewall* para que possa impedir o acesso desses IPs a rede principal. O campo de e-mail para alerta permite cadastrar um e-mail para receber alertas caso o *honeypot* seja atacado, o destinatário receberá apenas um aviso que o sistema está sendo atacado.

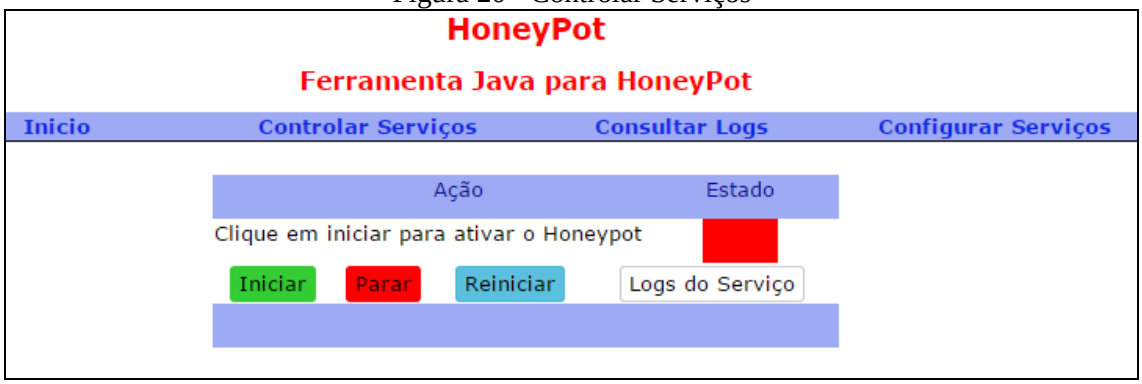
Figura 19 - Configurar serviços



3.3.2.3 Controlar Serviços

A tela de controle dos serviços permite iniciar, parar e reiniciar o *honeypot*, caso ocorra algum problema na inicialização é possível consultar os *logs* do serviço para verificar se houve erros no processo (Figura 20).

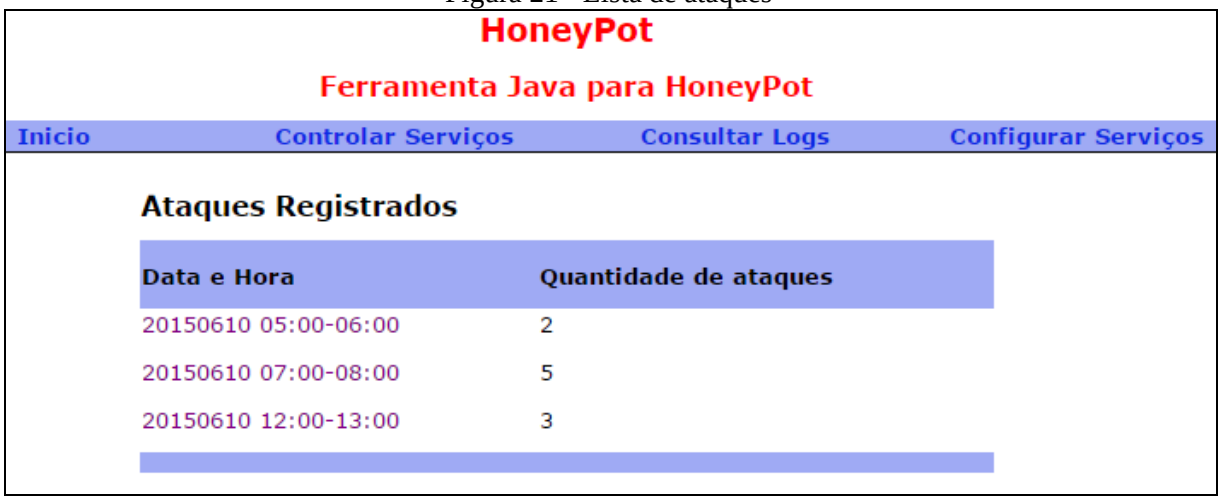
Figura 20 - Controlar Serviços



3.3.2.4 Consultar *Logs*

Na Figura 21 é possível visualizar o número de ataques recebidos em intervalos de uma hora.

Figura 21 - Lista de ataques



Ao clicar em um período serão mostrados os comandos recebidos durante a hora selecionada, conforme Figura 22.

Figura 22 - Detalhes do ataque

3.4 RESULTADOS E DISCUSSÕES

O objetivo deste trabalho foi aumentar a usabilidade da biblioteca JHoney através da implementação de mais funções para que o mesmo possa, simular serviços de redes e deixá-los vulneráveis em uma rede. O desenvolvimento deste trabalho permitiu à biblioteca JHoney ampliar o seu uso além do nível experimental, podendo responder de acordo com o serviço simulado e armazenar o *log* de dados recebidos.

A possibilidade de consultar os ataques e visualizar os comandos recebidos permite ao analista da rede entender como o ataque foi feito e os padrões utilizados pelos atacantes. Com essas informações detalhadas e armazenadas, poderão ser feitos estudos mais aprofundados sobre ataques e aprimoramento dos métodos de segurança atuais.

Observando a biblioteca JHoney original, pode-se fazer o seguinte comparativo das principais funcionalidades do trabalho desenvolvido (Quadro 7).

Quadro 7 - Comparativo entre JHoney original e trabalho desenvolvido

Funcionalidade	**JHoney**	**Trabalho desenvolvido**
Persistência em banco de dados	Não	Sim
Registrar comandos recebidos	Não	Sim
Responder comandos recebidos	Não	Sim
Alertar via e-mail ataques recebidos	Não	Sim
Alterar porta de inicialização do *honeypot*	Sim	Sim
Alimentar uma *blacklist* dinâmica	Sim	Sim

Levando em consideração que o JHoney original não possui persistência dos dados e não responde aos comandos recebidos, dificilmente ele coletaria dados importantes, visto que o atacante ao perceber que o serviço que ele está atacando não responde, ele imediatamente abandonará a conexão. Essas duas funcionalidades são cruciais para a atuação do *honeypot* em seu papel principal, que é enganar os atacantes e coletar dados sobre os mesmos.

O alerta via e-mail é um recurso importante, pois ao iniciar o *honeypot* não se sabe em qual data e hora ele será atacado. Após a inicialização o analista da rede não precisará ficar consultando o *honeypot* para ver se houve algum ataque, basta ele aguardar e um e-mail será enviado quando houver alguma ocorrência.

4 CONCLUSÕES

Atualmente empresas e governo necessitam manter um alto nível de segurança de suas informações, onde suas redes têm sido protegidas por *firewalls*, antivírus e inclusive proteções físicas. Mas nem sempre isso é suficiente, aumentar a segurança é sempre um desafio e deve ser sempre prioridade para um profissional de segurança de redes.

Entender um ataque torna-se uma ferramenta muito importante para proteger uma rede. Neste trabalho foi proposto a implementação de novas funcionalidades na biblioteca JHoney, possibilitando a mesma atuar de forma mais completa como um *honeypot*, mantendo os princípios de invisibilidade e simulação de serviços reais. A utilização da linguagem Java é pioneira para esse tipo de projeto e por ser difundida possibilitará uma grande extensão da técnica para estudos e ampliação de funcionalidades.

Os trabalhos correlatos dão uma visão de como pode-se usar o *honeypot* para o monitoramento de ataques a redes, pois focam situações específicas e utilizam várias estruturas (físicas e virtuais) que trabalham juntas para coletar dados. Através deste trabalho, é possível configurar um *honeypot* de forma mais independente, que facilitará a prática e testes de suas funcionalidades por pessoas que se interessarem pelo estudo de ataques na internet.

O objetivo deste trabalho foi aumentar a usabilidade da biblioteca JHoney através da implementação de mais funções para que o mesmo possa de forma completa, simular serviços de redes e deixá-los vulneráveis. Ampliou o uso da biblioteca, podendo responder de acordo com o serviço simulado e armazenar o *log* de dados recebidos. Sem essas duas funcionalidades o trabalho não teria total característica e nem poderia atuar de forma segura como um *honeypot*. A geração de uma *blacklist* permite que o *honeypot* possa complementar a proteção de um *firewall*.

4.1 EXTENSÕES

O presente trabalho oferece as funcionalidades básicas para o funcionamento de um *honeypot*. Segue algumas sugestões para sua continuidade:

a) extrair relatórios personalizados a partir dos dados coletados;
b) permitir simular sistemas operacionais;
c) permitir simular mais do que um serviço por vez.

REFERÊNCIAS

ANDRUCIOLI, Alexandre Pinaffi. **Proposta e avaliação de um modelo alternativo baseado em honeynet para identificação de ataques e classificação de atacantes na internet**. 2005. 146 f. Dissertação (Mestrado) - Curso de Ciências em Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2005.

ASSUNÇÃO, Marcos Flávio Araújo. **Honeypots e honeynets**: aprenda a detectar e enganar invasores. Florianópolis: Visual Books, 2009. 128 p.

BATISTELA, Vinícius; TRENTIN, Marco Antônio Sandini**. Identificação e análise de tráfego malicioso através do uso de honeypots**. Revista Brasileira de Computação Aplicada, Passo Fundo, v. 1, n. 1, p.2-14, set. 2009. Disponível em: <http://www.upf.br/seer/index.php/rbca/index>. Acesso em: 08 set. 2014.

BERNSTEIN, Terry. **Segurança na internet**. Rio de Janeiro : Campus, 1997. 461 p, il.

CAMPOS, André. **Sistema de segurança da informação – controlando riscos**. Florianópolis: Visual Books, 2007. 218 p.

CERT BR - Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (Org.). **Cartilha de segurança para internet**. São Paulo: Comitê Gestor da Internet no Brasil, 2012. 140 p. Disponível em: <http://cartilha.cert.br/livro/cartilha-seguranca-internet.pdf>. Acesso em: 28 maio 2015.

CERT.BR - Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (Org.). **Estatísticas dos Incidentes Reportados ao CERT.BR**. São Paulo: Comitê Gestor da Internet no Brasil, 2012. Disponível em: <http://www.cert.br/stats/incidentes/2013-jan-dec/tipos-ataque.png>. Acesso em 20 abr. 2015.

COSTA, Luis Henrique Sousa. **Protegendo plataforma de comércio eletrônico contra ataques DoS utilizando honeypot**. 2013. 80 f. Trabalho de conclusão de curso (Graduação) - Curso de Sistemas de Informação, Centro Universitário Eurípides de Marília – Univem, Marília, 2013. Disponível em: <http://aberto.univem.edu.br/bitstream/handle/11077/989/protegendo plataforma de comércio eletrônico contra ataques dos utilizando honeypot.pdf?sequence=1>. Acesso em: 09 set. 2014.

DUARTE, Otto Carlos Muniz Bandeira; JABOUR, Eugênia Cristina Müller Giancoli. **Honeynets**: invasores, ferramentas, técnicas e táticas. Rio de Janeiro: UFRJ, 2003. Disponível em: <http://www.gta.ufrj.br/seminarios/cpe825/tutoriais/eugenia/honeynets.pdf>. Acesso em: 11 set. 2014.

HAGELBACK, Johan. **JHoney**. [S. l.], 2004. Disponível em: <http://linux.softwareweb.com/jhoney-1.00-rw21n.html>. Acesso em: 15 set. 2014.

MCCLURE, Stuart; SCAMBRAY, Joel; KURTZ, George. **Hackers expostos**: [segredos e soluções para a segurança de redes]. São Paulo : Makron Books, 2000. xxviii, 469p, il

MAIA, Igor da Silva Neiva; REHEM, Sandro Herman Pereira. **Sistemas de prevenção de intrusão baseado em software livre**: DEBIAN e SNORT. 2005. 89 f. Monografia (Especialização) - Curso de Redes de Computadores, Universidade Católica de Brasília, Brasília, 2005. Disponível em: <http://pt.scribd.com/doc/13224983/sistemas-de-prevencao-de-intrusao-baseado-em-software-livre-igor-neiva-e-sandro-herman-ucb>. Acesso em: 30 nov. 2014.

MANDARINO JUNIOR, Raphael; CANONGIA, Claudia. **Livro verde**: segurança cibernética no brasil. Brasília: Gsipr, 2010. 63 p. Disponível em: <http://dsic.planalto.gov.br/documentos/publicacoes/1_livro_verde_seg_ciber.pdf>. Acesso em: 28 out. 2014.

NUNES, Mauricio Slobodicov de Sousa. **Prevenção contra crackers e hackers**: a utilização de detecção e prevenção de intrusão em empresas dependentes da internet - estudo de caso. In: NETCOM, 1., 2009, São Paulo. Anais eletrônicos... São Paulo: Aranda Editora, 2009. Disponível em: <http://xa.yimg.com/kq/groups/21753468/669090003/name/preven%c3%a7%c3%a3o+contra+hackers+e+crackers+usando+ids+e+ips-+um+estudo+de+caso.pdf>. Acesso em: 12 nov. 2014.

ROCHA, Daniel Lyra. **Utilização de um ambiente de honeynet no treinamento de redes neurais artificiais para detecção de intrusão**. 2006. 193 f. Dissertação (Mestrado) - Curso de Engenharia de Redes de Comunicação, Universidade de Brasília, Brasília, 2006. Disponível em: <http://repositorio.unb.br/bitstream/10482/5735/1/2006_Daniel Lyra Rocha.pdf>. Acesso em: 28 mar. 2015.

SPITZNER, Lance. **Honeypots: tracking hackers**. Boston: Addison Wesley, 2002.