**UNIVERSIDADE REGIONAL DE BLUMENAU**

**CENTRO DE CIÊNCIAS EXATAS E NATURAIS**

**CURSO DE CIÊNCIA DA COMPUTAÇÃO – BACHARELADO**

# FERRAMENTA PARA MONITORAÇÃO E GERENCIAMENTO DE TRÁFEGO EM UMA REDE LOCAL

**EDERSON BENNERTZ**

**BLUMENAU**
**2014**

**2014/2-06**

**EDERSON BENNERTZ**

# FERRAMENTA PARA MONITORAÇÃO E GERENCIAMENTO DE TRÁFEGO EM UMA REDE LOCAL

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 - Orientador

**BLUMENAU**
**2014**

**2014/2-06**

# FERRAMENTA PARA MONITORAÇÃO E GERENCIAMENTO DE TRÁFEGO EM UMA REDE LOCAL

Por

**EDERSON BENNERTZ**

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. Mauro Marcelo Mattos, Doutor – FURB |
| | _____ |
| Membro: | Prof. Paulo Fernando da Silva, Mestre – FURB |

Blumenau, 09 de Dezembro de 2014

## AGRADECIMENTOS

À minha família, pela confiança e apoio durante a realização do trabalho.

Ao meu orientador, Francisco Adell Péricas pela confiança na conclusão desse trabalho.

Com a força da sua mente, seu instinto e também com sua experiência, você pode voar alto.

Ayrton Senna

# RESUMO

Este trabalho apresenta a especificação e implementação de uma ferramenta para monitoração e gerenciamento de redes, através da análise do cabeçalho das mensagens que trafegam pela rede local. Para o desenvolvimento da ferramenta foi realizado um estudo sobre a arquitetura TCP/IP e sobre a segurança de redes de computadores. O trabalho discorre sobre o uso da biblioteca JPCAP para captura dos pacotes TCP/IP e da biblioteca Winpcap para o acesso a interface de rede. Com o auxílio da biblioteca que foi utilizada nesse trabalho é possível visualizar, salvar, excluir, consultar e aplicar filtros a partir dos pacotes capturados. A ferramenta além de capturar os pacotes, permite definir a geração de alertas quando determinado conteúdo especificado trafegar pela rede, permitindo filtrar informações dos protocolos da camada de rede, transporte e aplicação. Os resultados obtidos mostram que a ferramenta desenvolvida nesse trabalho fornece informações para que o administrador da rede tome decisões de forma rápida e eficaz conforme a necessidade.

Palavras-chave: JPCAP. Gerenciamento de redes. Segurança de redes. TCP/IP.

# ABSTRACT

This work presents the specification and implementation of a tool for monitoring and network management, by analyzing the header of the messages that travel over the local network. For the development of the tool was a study on the TCP/IP architecture and the security of computer networks. The paper discusses the use of JPCAP library for capturing TCP/IP packets and the Winpcap library to access the network interface. With the aid of the library that was used in this work you can view, save, delete, query and apply filters from the captured packets. The tool while capturing packets, allows to define the generation of alerts when certain specified content move across the network, allowing to filter information from the network layer, transport and application protocols. The results show that the tool developed in this work provides information for the network administrator to take quick and effective decisions as needed.



Key-words: JPCAP. Network management. Network security. TCP/IP.

# LISTA DE FIGURAS

# LISTA DE QUADROS

# LISTA DE SIGLAS

FTP – *File Transfer Protocol*

HTTP – *Hypertext Transfer Protocol*

HTTPS - *Hypertext Transfer Protocol Secure*

IP – *Internet Protocol*

ICMP – *Internet Control Message Protocol*

ISO – *International  Organization for Standardization*

MAC – *Media Access Control*

SGBD – Sistema de Gerenciamento de Banco de dados

SMTP – *Simple Mail Transfer Protocol*

TCP – *Transmission Control Protocol*

UDP – *User Datagram Protocol*

UML – *Unified Modeling Language*

# SUMÁRIO

# 1 INTRODUÇÃO

Os avanços tecnológicos exercem hoje um grande impacto na sociedade. A informação tem se tornado cada vez mais uma vantagem competitiva para as empresas e organizações. O fato é que, a cada dia, as empresas e organizações, para se tornarem competitivas e sobreviverem no mercado, têm investido em tecnologia da informação. É diante deste quadro que as redes de computadores se proliferam, encurtando as distâncias e diminuindo o tempo de resposta entre as organizações de todo o mundo (BLACK, 2008).

O contínuo crescimento em número e diversidade dos componentes das redes de computadores tem levado a necessidade de gerenciamento de redes cada vez mais complexa. O gerenciamento permite controle sobre recursos da rede assim como a identificação e prevenção de problemas, sendo tal investimento justificado quando se quer controle dos recursos, de sua complexidade, serviços melhores e controle de custo (BLACK, 2008).

Os dados que trafegam em uma rede passam por um processo de codificação, passando de informações alfanuméricas para bits. Os bits ficam agrupados em forma de pacotes e então são transmitidos pela rede. A partir daí não se tem mais o controle de quais pacotes estão trafegando, qual o destino dos pacotes, qual sua origem e que dado está contido neles (HILGENSTIELER, 2003).

A função de um monitor de pacotes consiste basicamente em interceptar pacotes que trafegam entre o meio interno e externo de um determinado host e armazenar dados sobre estes pacotes para análise. Além de armazenar os dados, o monitor de pacotes pode ser utilizado para a exibição do conteúdo dos pacotes, com vários níveis de abstração para que possam ser melhor interpretados (SILVA, 2001).

Diante do exposto, este trabalho propõe o desenvolvimento de um monitor de pacotes para o gerenciamento e captura dos pacotes, possibilitando, através da especificação de filtros, obter o tráfego de diferentes dispositivos conectados na rede, assim como identificar situações suspeitas.

Já foram desenvolvidos TCCs nesta área, tais como Silva (2001), o qual faz a monitoração de pacotes TCP/IP, com o objetivo de verificar o tráfego da rede em relação aos endereços da camada de rede e transporte, ou como Hilgenstieler (2003), que visa à camada de aplicação, analisando o conteúdo da navegação da internet, com o intuito de filtrar os pacotes do host especificado junto a filtros.

## 1.1 OBJETIVOS

O objetivo deste trabalho é disponibilizar uma ferramenta para monitoração e gerenciamento de redes, através da análise do cabeçalho das mensagens que trafegam em uma rede local.

Os objetivos específicos do trabalho são:

a) interceptar e interpretar pacotes TCP/IP;
b) monitorar dados contidos nas mensagens dos pacotes que trafegam em uma rede local;
c) implementar filtros para identificação do tráfego de diferentes dispositivos da rede;
d) armazenar as informações monitoradas em um banco de dados central.

## 1.2 ESTRUTURA

Este trabalho está estruturado em quatro capítulos. O capítulo um apresenta uma introdução ao tema abordado, os objetivos e a estrutura deste trabalho.

O capítulo dois contém a fundamentação teórica necessária para permitir um melhor entendimento sobre o assunto.

O capítulo três apresenta o desenvolvimento da ferramenta, contemplando também os seus requisitos e a especificação por meio dos diagramas de caso de uso, classe, atividades e o modelo entidade relacionamento. Em seguida, prossegue com o detalhamento da implementação, resultados e problemas encontrados.

Ao final, no capítulo quatro são apresentadas as conclusões finais sobre o trabalho e sugestões para trabalhos futuros.

# 2 FUNDAMENTAÇÃO TEÓRICA

As seções foram distribuídas entre três itens principais, sendo que na seção 2.1 será apresentada uma descrição da arquitetura TCP/IP e seus protocolos. A seção 2.2 procura esclarecer os conceitos relacionados ao gerenciamento de redes. Na seção 2.3 são abordados conceitos e técnicas de segurança em redes de computadores e por fim a seção 2.4 traz os trabalhos correlatos.

## 2.1 ARQUITETURA TCP/IP

O TCP/IP é um conjunto de protocolos usado em redes de computadores, que envolve praticamente toda a internet. Os recursos e possibilidades que a internet oferece aos usuários, através da conectividade de redes de tecnologias distintas, somente são possíveis devido a implementação do TCP/IP.

Os diversos tipos de protocolos disponíveis no TCP/IP trabalham em conjunto para que a comunicação possa ser efetuada (HILGENSTIELER, 2003).

Segundo Tanenbaum (2003), o TCP/IP executa essa conectividade em nível de rede, o que permite a comunicação entre aplicações em computadores de redes distintas sem a necessidade de conhecimento da topologia envolvida nesse processo. Outra característica importante do TCP/IP é a adaptação às tecnologias de redes existentes e futuras, que é possível porque o TCP/IP foi construído de forma independente das tecnologias de rede.

O TCP/IP é o resultado da pesquisa e desenvolvimento de protocolos realizados na rede experimental de comutação de pacotes *Advanced Research Projects Agency Network* (ARPANET), patrocinada pelo Departamento Norte-americano de Defesa, no final da década de 60. O objetivo era criar um software de comunicação entre computadores de forma que fosse possível a comunicação remota ou local entre sistemas operacionais iguais ou distintos, utilizando ou não o mesmo tipo de hardware. O protocolo permitiu que, no começo da década de 70, os computadores pudessem comunicar-se, sendo de suma importância em navios de guerra ou entre porta- aviões.

Segundo Péricas (2003), o modelo de referência TCP/IP, especificado pelo *Internet Engineering Task Force* (IETF) é baseado em um conjunto de cinco camadas.

a) camada física: trata da transmissão de bits através de um canal de comunicação. Esta camada é referenciada pelo IETF conjuntamente á camada de enlace, formando a camada de acesso ao meio;
a) camada de enlace: vária de estação para estação e de rede para rede. Seu objetivo é permitir que quadros enviados pela camada de rede sejam transportados entre

dois nós adjacentes;

b) camada de rede: seu objetivo é entregar pacotes emitidos a qualquer destino independentemente das tecnologias de transmissão utilizadas. Define um formato de pacote e um protocolo denominado *Internet Protocol* (IP). É responsável pelo roteamento e pelo controle de congestionamento;
c) camada de transporte: permite que as entidades pares de origem e de destino mantenham um canal de comunicação. Dois protocolos fim a fim foram definidos *Transport Control Protocol* (TCP), orientado a conexão e confiável e o *User Datagram Protocol* (UDP), sem conexão e não confiável;
d) camada de aplicação: contém os protocolos de suporte as aplicações de alto nível. Por exemplo, os protocolos *File Transfer Protocol* (FTP), *Simple Mail Transfer Protocol* (SMTP), *Simple Network Management Protocol* (SNMP), *Hypertext Transfer Protocol* (HTTP).

Quando as informações são enviadas, elas passam por todas essas camadas. Por exemplo, quando se transfere um arquivo para um servidor através do FTP, os dados e mais um cabeçalho contendo informações de controle do FTP serão enviados ao TCP (SILVA, 2001).

### 2.1.1 O Protocolo TCP

O TCP é um protocolo da camada de transporte que oferece um serviço orientado a conexão, isto é, quando um segmento é recebido, é identificado a que conexão está associada. (ALBUQUERQUE, 2001, p.32).

Em uma conexão TCP, cada estação é identificada por seu endereço IP na rede e as conexões são estabelecidas através de uma porta, que se trata de um número composto por 16 bits e variam de acordo com o serviço a ser utilizado. Este conjunto de endereços, número IP e porta, são conhecidos como soquete. As portas utilizadas para identificação do soquete são usadas por serviços diferentes. O número de portas possíveis que podem ser utilizadas é de 65535, porém as portas abaixo de 1024 são reservadas para serviços padrão (LIBERATO, 2006). O quadro 1 mostra alguns dos principais serviços utilizados em uma rede de computadores e as portas que são utilizadas.

Quadro 1– Serviços e portas atribuídas

| Porta | Serviço |
|---|---|
| 20 | FTP |
| 21 | FTP-DATA |
| 22 | SSH |
| 23 | TELNET |
| 25 | SMTP |
| 80 | HTTP |
| 110 | POP3 |

Todas as conexões TCP são full-duplex e ponto a ponto, isto é, o tráfego pode ser feito em ambas as direções ao mesmo tempo (TANENBAUM, 2003, p.568).

Segundo Albuquerque (2001), os dados são transferidos através da conexão em segmentos. Cada segmento é composto por cabeçalho, dados e seu tamanho é variável. O TCP responsabiliza-se por identificar e corrigir perda de segmentos. Para garantir a entrega dos dados o protocolo TCP espera que cada segmento recebido seja confirmado pela máquina destino. Se a recepção não for confirmada dentro de um intervalo de tempo, a máquina origem transmite novamente o segmento.

Segundo Kurose e Ross (2006), um cabeçalho de segmento TCP é formado pelos seguintes campos principais (Figura 1):

a) porta de origem e destino: são usadas para multiplexação e desmultiplexação de dados para aplicações de camadas superiores;
b) número de sequência e número de confirmação: são usados pelo TCP remetente e destinatário na implementação de um serviço confiável de transferência de dados;
c) tamanho da janela: usado para o controle de fluxo;
d) tamanho do cabeçalho: especifica o comprimento do cabeçalho TCP. O cabeçalho TCP pode ter comprimento variável devido ao campo de opções TCP;
e) opções: opcional e de comprimento variável, é usado quando um remetente e um destinatário negociam o *Maximum Segment Size* (MSS), ou como um fator de aumento de escala da janela para utilização em redes de alta velocidade;
f) ack: usado para indicar se o valor carregado no campo de reconhecimento é válido, isto é, se o segmento contem um reconhecimento para um segmento que foi recebido com sucesso;
g) urg: envia uma mensagem ao destino de que os dados urgentes estão esperando para serem enviados a ele;
h) psh: indica que o destinatário deve passar os dados para a camada superior imediatamente;

i) rst: reinicia a comunicação entre os hosts;

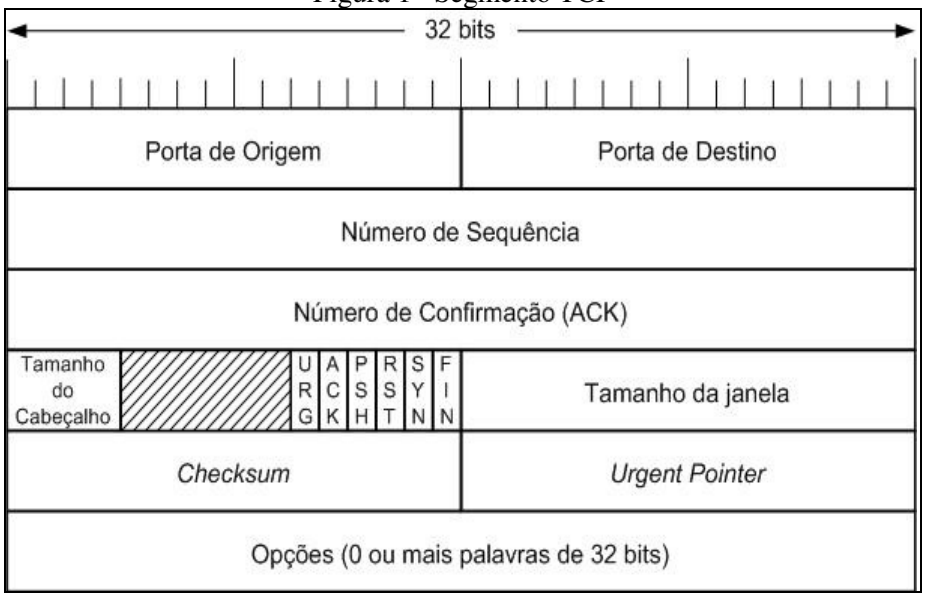
j) syn: usado na inicialização e para estabelecer um número de sequência;

k) fin: mais nenhum dado está vindo da estação de origem;

l) ponteiro urgente: a localização do último byte dos dados urgentes é indicada no campo de ponteiro de urgência;

m) checksum: método de detecção de erros.

Figura 1– Segmento TCP




Fonte: Wordpress (2010)

Para encerrar uma conexão, qualquer dos lados pode enviar um segmento com o bit FIN ativado, o que significa que não há mais dados a serem transmitidos. No entanto, os dados podem continuar a fluir indefinidamente no outro sentido. Quando os dois sentidos da conexão estiverem desativados a conexão será encerrada (TANENBAUM, 2003, p.575).

### 2.1.2 O Protocolo IP

O IP é um protocolo para a entrega universal de dados através de qualquer tipo de rede. Os dados são empacotados em datagramas que compreendem algumas informações de controle de dados a serem entregues (FARREL, 2005, p.315).

A tarefa do IP é fornecer a melhor forma possível de transportar datagramas da origem para o destino, independente de essas máquinas estarem na mesma rede ou de haver outras redes entre elas (TANENBAUM, 2003, p.460).

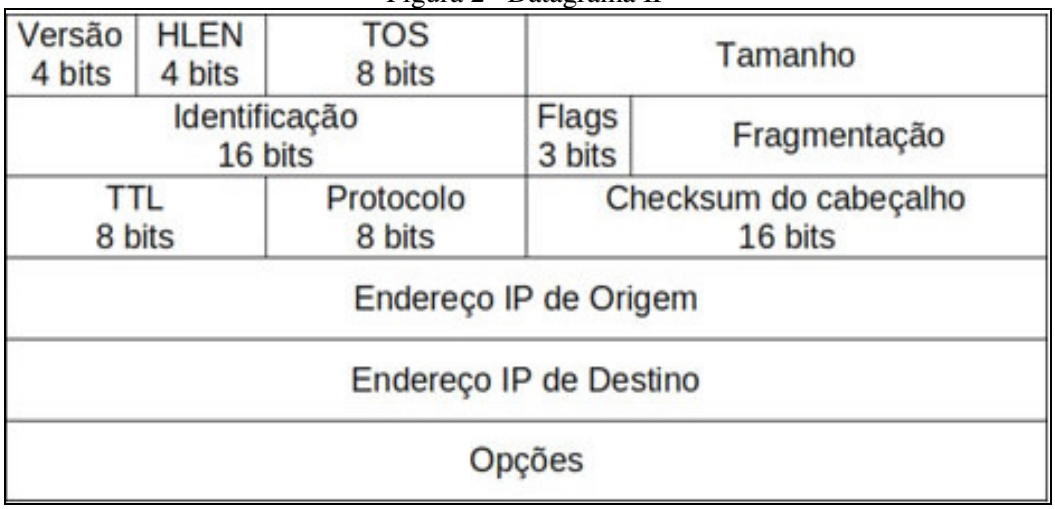
O IP não estabelece uma sessão entre o computador de origem e o destino. Ele é responsável por pegar os dados da camada de interface de rede e apresentá-los ao protocolo da camada acima que os solicitou. Quando os dados chegam a eles vindos da camada superior,

adiciona suas informações de controle e um cabeçalho aos dados. A partir daí, aquele bloco de dados é chamado de datagrama (SILVA, 2001).

Segundo Kurose e Ross (2006), um cabeçalho do datagrama IP é formado pelos seguintes campos principais (Figura 2):

a) número da versão: especifica a versão do protocolo IP;
b) comprimento do cabeçalho: determina onde no datagrama IP, os dados realmente começam;
c) tipo de serviço: foram incluídos no cabeçalho do *IPv4* para poder diferenciar os diferentes tipos de datagramas IP;
d) comprimento do datagrama: é o comprimento total do datagrama IP;
e) identificador: identifica quais fragmentos pertencem a qual datagrama, para que o destino não se confunda na hora de remontá-lo;
f) flags: indicam em uma fragmentação de datagramas, se mais fragmentos chegarão ou se não serão enviados mais dados pertencentes ao datagrama. Também identifica se um datagrama pode ou não ser fragmentado;
g) deslocamento de fragmentação: indica que parte do datagrama a estação de destino deve continuar a remontá-lo para determinado fragmento;
h) tempo de vida: é incluído para garantir que datagramas não fiquem circulando para sempre na rede;
i) protocolo: usado somente quando um datagrama IP chega a seu destino final. O valor desse campo indica o protocolo de camada de transporte específico ao qual a porção de dados desse datagrama IP deverá ser passada;
j) soma de verificação: auxilia um roteador na detecção de erros de bits em um datagrama IP recebido;
k) endereço IP de fonte e de destino: quando uma fonte cria um datagrama, insere seu endereço IP no campo de endereço de fonte IP e insere o endereço do destino final no campo de endereço de destinatário IP;
l) opções: permite que um cabeçalho IP seja ampliado. A intenção é que as opções de cabeçalho sejam usadas raramente, por isso à decisão de poupar sobrecarga não incluindo a informação em campos de opções em todos os cabeçalhos do datagrama.

Figura 2– Datagrama IP




O endereçamento lógico no TCP/IP é definido pelo protocolo IP, que determina que cada computador na rede deve ter um endereço IP único. Esse endereço define a rede á qual o computador pertence e o identifica nessa rede. Essas informações permitem ao protocolo IP implementar o mecanismo de roteamento das redes TCP/IP (HILGENSTIELER, 2003).

O protocolo IP utilizado atualmente na maioria das redes TCP/IP é o IPv4. Cada endereço IPv4 é composto por um numero de 32 bits, portanto existem mais de 4 bilhões de endereços. Porém os endereços IPv4 estão se esgotando e logo não haverá mais endereços para a criação de novas redes (SILVEIRA, 2012).

Para resolver o problema de esgotamento de endereços IPv4 foi criado o IPv6. Nesta versão do protocolo IP, os endereços são compostos por 128 bits, gerando assim uma quantidade virtualmente ilimitada de endereços IP. O IPv6 também é provido de novos recurso, tais como o suporte a novas tecnologias de rede, novo formato do cabeçalho, infraestrutura hierárquica e eficiência de roteamento e endereçamento, configuração de endereçamento com ou sem estado, segurança embutida, melhor suporte para a qualidade dos serviços e novo protocolo para iteração entre nós vizinhos (BRAGA, 2011).

A transição do protocolo IPv4 para o IPv6 deverá ser de forma gradual para garantir a funcionalidade das redes. Foram criados dispositivos para que se possa fazer uma transição de modo que os dois protocolos de rede possam coexistir (BRAGA, 2011).

### 2.1.3 Protocolos da camada de aplicação

Na camada de aplicação funcionam os protocolos usados pelas aplicações de usuários, sendo a camada de mais alto nível em uma pilha de protocolos. Os protocolos de aplicação utilizam os serviços prestados pelos protocolos de transporte para enviar e receber dados através da rede (ALBUQUERQUE, 2001, p.147).

Existem vários protocolos desenvolvidos para esta camada, dentre os protocolos mais conhecidos e utilizados da camada de aplicação pode-se destacar:

a) SMTP: protocolo utilizado em aplicações de transferência de mensagens de correio eletrônico;
b) FTP: protocolo desenvolvido para transferência de arquivos entre um *host* local e um *host* remoto;
c) HTTP: protocolo desenvolvido para oferecer serviços de transmissão e navegação através de hipertexto, suportando a transmissão de textos, gráficos e qualquer tipo de arquivo;
d) TELNET: este protocolo oferece serviço de emulação de um terminal de uma rede remota.

## 2.2 GERENCIAMENTO DE REDES

As redes de computadores são de importância vital e crescente em todos os tipos de empresas. A tendência é em direção a redes maiores e mais complexas, que aceitam mais aplicações e mais usuários. À medida que as redes locais crescem e se interligam com redes de outras organizações, torna-se necessária a utilização de sistemas que facilitem sua gerência (STALLINGS, 2005, p.409).

O crescimento das redes de computadores e a necessidade de disponibilizar informações são diretamente proporcionais aos riscos que as empresas correm ao não tomar providências de monitoração, podendo sofrer invasões, ter informações roubadas, alteradas ou danificadas. Por menor que seja uma rede de computadores, precisa ser gerenciada, a fim de garantir, aos seus usuários a disponibilidade de serviços com um nível de desempenho e segurança aceitável.

Gerenciar uma rede é uma atividade complexa. Nos últimos anos o tráfego de informações dentro das redes aumentou significativamente devido ao surgimento de novas aplicações. Novas tecnologias e padrões proporcionaram uma grande proliferação de dispositivos heterogêneos conectados à rede (BLACK, 2008).

Uma grande rede não pode ser organizada e gerenciada unicamente pelo esforço humano. A complexidade desse tipo de sistema obriga ao uso de ferramentas automatizadas de gerenciamento de redes (STALLINGS, 2005, p.409).

Segundo Lopes, Nicolleti e Sauvé (2003), a arquitetura geral dos sistemas de gerência de redes apresenta quatro componentes básicos:

a) elementos gerenciados: possuem um software especial chamado agente, que

permite que o equipamento seja monitorado e controlado de uma ou mais estações de gerência;

b) estação de gerência: em um sistema de gerência de redes deve haver pelo menos uma estação de gerência. É denominado gerente o software da estação de gerência que conversa diretamente com os agentes nos elementos gerenciados, seja com o objetivo de monitorá-los, seja com o objetivo de controlá-los. A estação de gerência oferece uma interface através da qual usuários autorizados podem gerenciar a rede;

c) protocolo de gerência: para que a troca de informações entre gerentes e agentes seja possível é necessário um protocolo de gerência. Este protocolo permite operações de monitoramento e controle;

d) informações de gerência: definem os dados que podem ser referenciados em operações do protocolo de gerência, isto é, dados sobre os quais gerente e agentes conversam.

Ferramentas de monitoração são importantes também para determinar se está havendo uma invasão à rede interna, auxiliando o administrador de rede a tomar providências para que os danos causados sejam contidos o mais rápido possível (SILVA, 2001).

A chave para um perfeito gerenciamento de redes encontra-se nas informações que possam ser obtidas, podendo-se assim, ter a possibilidade de visão e manipulação das informações, bem como, compartilhá-las com outros profissionais que trabalham nas mesmas funções ou em colaboração, para desta forma obter-se conhecimento e evolução nos trabalhos realizados (MELLO, 2005).

### 2.2.1 Modelo de Gerenciamento

Para facilitar a identificação do gerenciamento de redes, a *International Organization for Standardization* (ISO) definiu cinco áreas funcionais para a gerência dentro de uma rede. Vale ressaltar que segundo Albuquerque (2001), a maioria dos sistemas de gerência não abrange todas as áreas.

A não abrangência de todas as áreas é justificável pela necessidade de objetividade na prática do gerenciamento. Um sistema de gerência deve ser capaz de passar de forma rápida e clara, qualquer falha que possa ocorrer na rede. A apresentação de vários dados da rede poderia promover uma falta de concentração e confusão do gerente de rede (STANGE, 2008).

Segundo Albuquerque (2001), a função de cada uma das áreas funcionais é:

a) gerência de falhas: envolve a identificação e a correção de falhas;

b) gerência de configuração: envolve a análise e a alteração das configuração das entidades gerenciadas;
c) gerência de desempenho: envolve a coleta de dados sobre o desempenho das entidades gerenciadas e a execução de ações visando otimizá-las;
d) gerência de contabilidade: envolve a imposição de cotas aos usuários, a cobrança pelo uso dos recursos e a sua monitoração;
e) gerência de segurança: envolve o controle de acesso aos recursos, o sigilo, a autenticação e a identificação de acessos não autorizados.

Os sistemas para gerência de redes podem auxiliar nas diversas fases citadas, coletando dados e informações sobre eventos ocorridos, apresentando-os em um formato que facilite a análise, sugerindo procedimentos a serem seguidos, seguindo automaticamente procedimentos previamente definidos, possibilitando a inspeção e a alteração das configurações das entidades gerenciadas através da comparação com configurações armazenadas em bases de dados (ALBUQUERQUE, 2001, p.279).

Um sistema para gerência de redes pode ter uma arquitetura centralizada ou distribuída. Na arquitetura centralizada é usada apenas uma estação de gerência, que é adequada para redes de pequeno porte. Na arquitetura distribuída são usadas várias máquinas, organizadas em uma hierarquia, para gerenciar a rede. As máquinas no topo da hierarquia operam como estações de gerência, apresentando uma visão integrada da rede e interagem com as outras máquinas dentro da hierarquia (ALBUQUERQUE, 2001, p.277).

Segundo Albuquerque (2001), independente das necessidades do responsável pela gerência da rede, o sistema de gerenciamento segue um modelo básico. Este pode optar por realizar apenas as gerências desejadas. No cotidiano a gerência mais utilizada é a de falhas, devido ao impacto que esta tem sobre o funcionamento de uma rede de computadores. Uma empresa pode, por exemplo, manter seu funcionamento com uma rede operando com certa lentidão, porém de maneira alguma conseguira realizar suas tarefas se a rede sofrer alguma falha e for indisponibilizada.

## 2.3 SEGURANCA EM REDES DE COMPUTADORES

As redes de computadores são ferramentas indispensáveis no mundo dos negócios, pois elas trazem maior facilidade, velocidade, maior competitividade e em consequência disso um grande aumento na produtividade (QUEIROZ, 2007).

Diversos pontos devem ser considerados quando uma rede passa a constituir parte importante da organização. Aspectos como o mal uso de senhas, perigos da falta de controle

de acesso ou demais riscos precisam ser entendidos para que uma melhor segurança possa ser implementada e gerenciada (QUEIROZ, 2007).

Segundo Queiroz (2007), alguns dos assuntos mais importantes a serem entendidos para o gerenciamento da segurança do sistema são os que estão relacionados com o tipo de informação que deve entrar e percorrer a rede, com a definição do controle de acesso, com a dificuldade de controle e de informação do administrador sobre os sistemas, com a hostilidade do ambiente da internet, a sujeição das informações que trafegam pela rede, com a conexão entre redes internas e pontos externos como meio de invasão à rede e a complexidade de implantar meios totalmente seguros.

Segundo Péricas (2003), os problemas de segurança das redes podem ser divididos nas seguintes áreas:

a) sigilo: está relacionado ao fato de manter as informações longe de usuários não autorizados, ou seja somente usuários autorizados podem ser capazes de entender o conteúdo de mensagens ou de arquivos;
b) autenticação: cuida do processo de determinar com quem se está falando antes de revelar informações sigilosas ou entrar em transação comercial, ou seja, é preciso se assegurar da identidade da outra parte com quem se quer trocar informações;
c) integridade: certifica que uma determinada mensagem recebida é realmente legitima e não algo modificado ou impropriamente criado, tratando também de assinaturas, ou seja, garantia de que uma determinada transação foi realmente requisitada naqueles termos.

A segurança na rede não envolve apenas proteção, mas também detecção de falhas em comunicações seguras e ataques a infraestrutura e reação a esses ataques. Em muitos casos um administrador pode implementar mecanismos especiais de proteção para reagir a esses ataques. Nesse sentido, a segurança da rede é conseguida por meio de um ciclo contínuo de proteção, detecção e reação (KUROSE; ROSS, 2006, p.514).

Deve-se ter em mente que a segurança se enquadra em um processo constante e evolutivo, uma luta do administrador de segurança que mantém os sistemas atualizados, suas políticas de segurança coerentes e seus planos de contenção abrangente em um ambiente hostil.

### 2.3.1 Políticas de Segurança

A necessidade de proteção deve ser definida em termos das possíveis ameaças e riscos dos objetivos de uma organização, formalizados nos termos de uma política de segurança (KOBUSZEWSKI, 2004).

A política de segurança é um mecanismo preventivo de proteção dos dados e processos importantes de uma organização que define um padrão de segurança a ser seguido pelo corpo técnico, gerencial e pelos usuários internos e externos. Pode ser usada para definir as interfaces entre usuários, fornecedores, parceiros, para medir a qualidade e a segurança dos sistemas atuais. As políticas de segurança devem ter uma implementação realista e definir claramente as áreas de responsabilidade dos usuários, do pessoal de gestão de sistemas e redes e da direção (PUPOLIN, 2007).

O principal objetivo da implantação de uma política de segurança é preservar a informação quanto a sua integridade, disponibilidade e confidencialidade (PUPOLIN, 2007).

Segundo Pupolin (2007), a política de segurança deve ir além dos aspectos de sistemas de informação e recursos computacionais, integrando as políticas institucionais relativas a segurança em geral, as metas de negócios da organização e ao plano estratégico de informática.

Para desenvolver uma política de segurança, é necessário traçar os objetivos de segurança da organização, pois o ambiente computacional varia de uma empresa para outra. Para localizar os objetivos mais prioritários é necessário fazer uma análise da natureza da aplicação, dos riscos e dos prováveis impactos. A implementação de uma política de segurança pode ser feita através de diversos mecanismos, desde a criação de cópias de segurança, até ferramentas de detecção de invasão (KOBUSZEWSKI, 2004).

### 2.3.2 Formas de Defesa

As formas de defesa mais eficientes dizem respeito à implementação e adoção de políticas de segurança, utilização de ferramentas ou outro esquema que sirva par proteger ao máximo o sistema. Mas as dificuldades não param por ai. Além de implementar as técnicas de segurança, faz-se necessário identificar onde está a vulnerabilidade para que seja possível corrigí-la, fazer o levantamento dos possíveis prejuízos e identificar o mais rápido possível os invasores (HILGENTIELER, 2003).

Segundo Kurose e Ross (2006), entre as formas de defesa, algumas técnicas e medidas se destacam.

a) *firewall*: combinação de hardware e software que isola a rede interna de uma

organização da internet em geral, permitindo que alguns pacotes passem e bloqueando outros;

b) criptografia: permitem que um remetente disfarce os dados de modo que um intruso não consiga obter nenhuma informação dos dados interceptados. O destinatário deve estar habilitado a recuperar os dados originais a partir dos dados disfarçados;

c) autenticação: técnica através da qual um processo confirma que seu parceiro na comunicação é quem deve ser. A autenticação é efetuada através da troca de mensagens constituintes de um protocolo de autenticação;

d) análise de pacotes: funciona em um dispositivo acoplado a rede e que recebe passivamente todos os quadros da camada de enlace que passam por sua interface de rede. Esses quadros podem então, ser passados aos programas de aplicação que extraem dados da camada de aplicação. A análise de pacotes é uma faca de dois gumes, pois pode ser de extrema importância para o administrador de rede realizar a monitoração e a administração da rede, mas também pode ser utilizada para eventuais ataques a rede.

### 2.3.3 Medidas de Segurança

Por maior que seja o aparato de equipamentos e de softwares de segurança de uma organização, algumas medidas consideradas simples e básicas podem ser empregadas para dificultar o acesso não autorizado às informações e ao sistema como um todo. Segundo Hilgentieler (2003), algumas medidas básicas seriam:

a) senhas seguras: fazer com que os usuários passem a criar e utilizar senhas mais seguras e promover a troca periódica das mesmas;

b) arquivos anexados: deixar claro o perigo contido em arquivos anexados em e-mails de procedência desconhecida e a instalação de qualquer tipo de software desconhecido;

c) serviços: eliminar serviços que não sejam realmente necessários para a empresa;

d) contas de usuários: fazer um levantamento periódico de contas de usuários inativos e eliminá-los;

e) atualização: estar sempre atualizado em questões de segurança.

## 2.4 TRABALHOS CORRELATOS

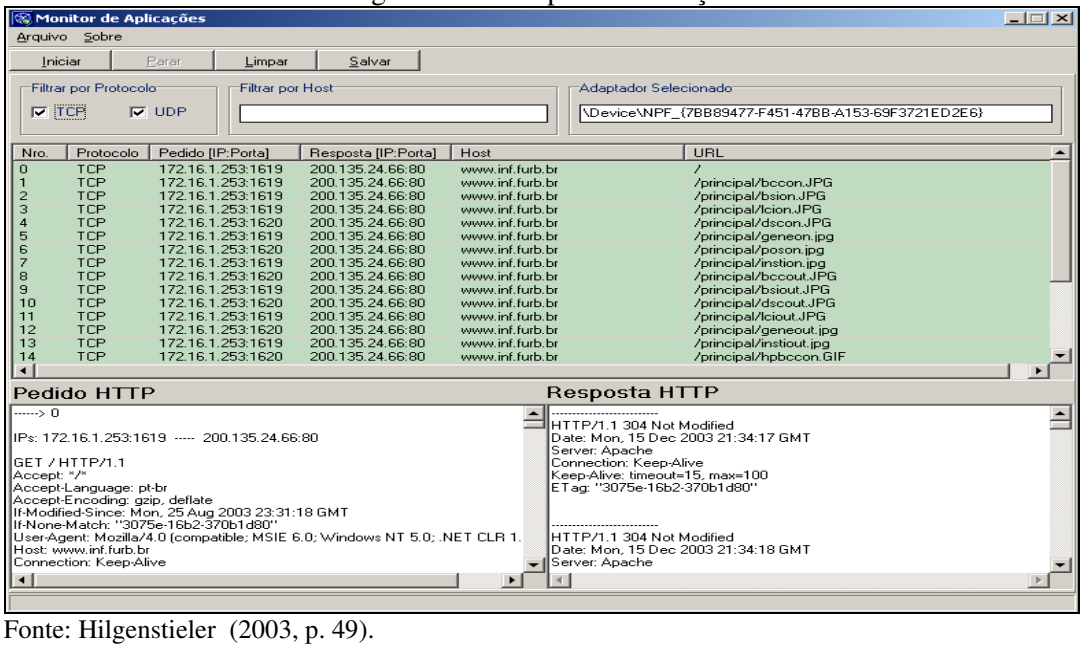
Existem disponíveis ferramentas comerciais e acadêmicas que abordam a temática de monitoração e gerência de redes, baseadas na captura de pacotes da camada de aplicação. Desta maneira, é possível encontrar desde trabalhos de conclusão de curso até dissertações de mestrado. Dentre estes trabalhos, foram selecionadas as ferramentas criadas por Hilgenstieler (2003), para monitoração do cabeçalho do protocolo *Hypertext Transfer Protocol* (HTTP) da camada de aplicação e a ferramenta criada por Silva (2001), para monitoração dos pacotes que trafegam entre um *host* e a internet.

### 2.4.1 Protótipo de software para monitoração do cabeçalho do protocolo HTTP em uma rede TCP/IP

Hilgenstieler (2003) desenvolveu uma ferramenta para a monitoração do cabeçalho do protocolo HTTP da camada de aplicação em um computador conectado a uma rede Ethernet, utilizando o sistema operacional Windows. O objetivo da ferramenta é interceptar pacotes TCP/IP, interpretando e monitorando dados contidos nas mensagens de camada de rede, transporte e aplicação, possibilitando armazenar as informações e identificar situações suspeitas.

Para o desenvolvimento da ferramenta foi utilizada a linguagem de programação Borland C++ baseando-se na biblioteca Winpcap. A figura 3 demonstra um exemplo de monitoração do cabeçalho HTTP.

Figura 3– Protótipo em execução




Fonte: Hilgenstieler (2003, p. 49).

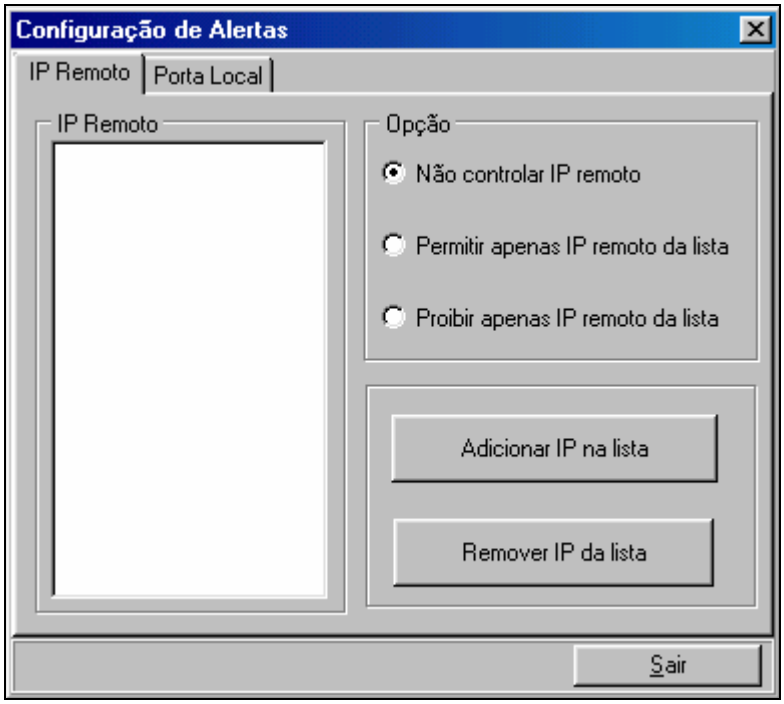
### 2.4.2 Protótipo de software de segurança em redes para a monitoração de pacotes em uma conexão TCP/IP

Silva (2001) desenvolveu uma ferramenta para a monitoração de pacotes TCP/IP, interceptando pacotes TCP/IP que trafegam entre um host e a internet, alertando o usuário quando ocorrem situações suspeitas. A ferramenta monitora pacotes de um host conectado a internet através de um acesso discado, utilizando para tanto o protocolo *Point to Point Protocol* (PPP). O host disponibiliza serviços para a internet os quais estarão sendo monitorados pelo monitor de pacotes. Quando algo suspeito ocorrer, o monitor de pacotes gera um alerta.

A implementação foi realizada na linguagem Object Pascal utilizando para tal o ambiente de desenvolvimento Borland Delphi 5.0. Para a implementação do modulo driver, que corresponde às funções de controle de conexão, desconexão e recebimento de pacotes foi utilizado funções da API do sistema operacional Windows 98. Para desenvolver o driver também foi utilizado o ambiente de desenvolvimento Microsoft Visual C++ 6.0.

A figura 4 apresenta a página "IP Remoto" da tela de configurações de alertas. Nesta página o usuário monta uma lista de endereços IP remotos e informa ao monitor como devem ser tratados os endereços da lista.

Figura 4– Tela de configuração de alertas




Fonte: Silva (2001, p. 75).

# 3 DESENVOLVIMENTO

Neste capítulo são abordadas as etapas de desenvolvimento da ferramenta. A primeira seção apresenta os principais requisitos funcionais e não funcionais da ferramenta desenvolvida. A segunda seção descreve a especificação do aplicativo através de diagramas da *Unified Modeling Language* (UML). A terceira seção apresenta os detalhes da implementação da ferramenta, incluindo os trechos de códigos mais relevantes ao seu funcionamento. Por fim, a quarta seção aborda os resultados deste trabalho.

## 3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

Os Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF) para a criação da ferramenta são:

a) monitorar os protocolos da arquitetura TCP/IP (RF);
b) capturar e analisar os pacotes que trafegam pela camada de aplicação (RF);
c) possibilitar mudanças dos filtros durante a execução (RF);
d) mostrar todos os pacotes capturados ao administrador (RF);
e) filtrar os dados dos pacotes de um determinado host (RF);
f) permitir que os dados capturados sejam salvos (RF);
g) utilizar a biblioteca JPCAP para captura dos pacotes TCP/IP (RNF);
h) ser implementado utilizando a linguagem de programação Java (RNF);
i) utilizar o ambiente de desenvolvimento Eclipse para o desenvolvimento(RNF);
j) permitir ao administrador cadastrar endereços IP que desejar monitorar (RF);
k) gerar mensagens de alerta quando o IP cadastrado trafegar pela rede (RF).

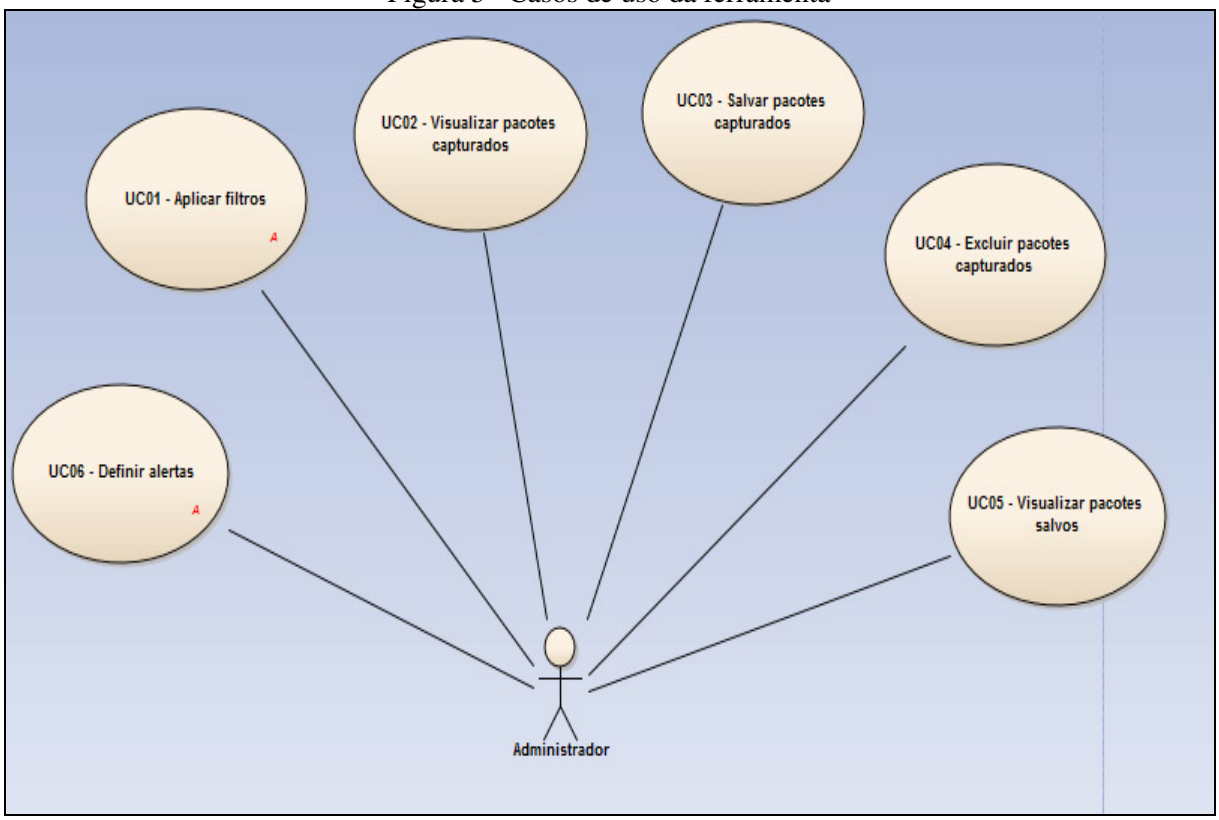
## 3.2 ESPECIFICAÇÃO

A especificação deste trabalho foi desenvolvida utilizando os diagramas da UML. Os casos de uso são apresentados através de um diagrama de casos de uso, seguidos das descrições dos casos de uso. Em seguida, são apresentadas as classes criadas para a ferramenta, através do diagrama de classe. Para maior entendimento de alguns fluxos básicos, também serão apresentados diagramas de atividade que representarão casos de uso da ferramenta. Os diagramas foram desenvolvidos na ferramenta Enterprise Architect 10.

### 3.2.1 Diagrama de casos de uso

Nesta seção são descritos os casos de uso com as funcionalidades da ferramenta. A figura 5 ilustra o diagrama de casos da ferramenta desenvolvida. Nele o ator administrador fará uso de todas as funcionalidades da ferramenta.

Figura 5– Casos de uso da ferramenta




O caso de uso UC01 descreve a funcionalidade que permite ao administrador aplicar filtros para a apresentação das informações capturadas. O quadro 2 apresenta o detalhamento desse caso de uso.

Quadro 2– UC01 - Aplicar filtros

| **UC01 – Aplicar filtros** | |
|---|---|
| **Descrição** | Permite ao administrador especificar filtros para apresentação das informações. |
| **Pré-condições** | Ferramenta em execução. |
| **Cenário principal** | 1) O administrador inicia a captura dos pacotes.<br>2) O administrador especifica filtros para a visualização das informações.<br>3) A ferramenta verifica se os pacotes capturados correspondem aos filtros definidos. |
| **Pós-condições** | Apresentar as informações dos pacotes. |
| **Exceção** | Nenhuma informação apresentada. |

O caso de uso UC02 descreve a visualização das informações dos pacotes capturados. A ferramenta permite definir filtros para a visualização das informações, caso nenhum filtro seja definido, todos os pacotes capturados são apresentados. O quadro 3 apresenta o caso de uso.

Quadro 3– UC02 - Visualizar pacotes capturados

| **UC02 – Visualizar pacotes capturados** | |
|---|---|
| **Descrição** | Permite ao administrador exibir as informações dos pacotes capturados. |
| **Pré-condições** | Ferramenta em execução. |
| **Cenário principal** | 1) O administrados inicia a captura dos pacotes.<br>2) O administrador tem a possibilidade de filtrar as informações.<br>3) A ferramenta verifica os filtros, caso algum filtro seja definido.<br>4) A ferramenta apresenta as informações dos pacotes capturados. |
| **Pós-condições** | Apresentar as informações dos pacotes. |
| **Exceção** | Nenhuma informação apresentada. |

O caso de uso UC03 descreve a funcionalidade de salvar na base de dados às informações dos pacotes capturados. As informações são salvas junto à base Oracle versão 10g. O quadro 4 apresenta o detalhamento do caso de uso.

Quadro 4– UC03 - Salvar pacotes capturados

| **UC03 – Salvar pacotes capturados** | |
|---|---|
| **Descrição** | Permite ao administrador salvar as informações na base de dados. |
| **Pré-condições** | Pacotes capturados. |
| **Cenário principal** | 1) O administrador inicia a captura dos pacotes.<br>2) A ferramenta apresenta as informações dos pacotes capturados.<br>3) O administrador define os pacotes a serem salvos.<br>4) A ferramenta salva na base de dados as informações. |
| **Pós-condições** | Salvar na base de dados Oracle 10g as informações. |
| **Exceção** | Não há. |

O caso de uso UC04 descreve a funcionalidade de excluir na base de dados às informações dos pacotes salvos. O quadro 5 apresenta o detalhamento do caso de uso.

Quadro 5– UC04 - Excluir pacotes capturados

| **UC04 – Excluir pacotes capturados** | |
|---|---|
| **Descrição** | Excluir os pacotes salvos na base de dados. |
| **Pré-condições** | Informações salvas na base de dados. |

| | |
|---|---|
| **Cenário principal** | 1) O administrador define as informações que serão excluídas na base.<br>2) A ferramenta exclui na base de dados as informações. |
| **Pós-condições** | Informações excluídas na base de dados. |
| **Exceção** | Nenhuma informação salva na base de dados. |

O caso de uso UC05 descreve a funcionalidade para consultar na base de dados às informações dos pacotes salvos. A ferramenta permite definir parâmetros para consultar as informações salvas. O quadro 6 apresenta o detalhamento do caso de uso.

Quadro 6– UC05 - Visualizar pacotes salvos

| **UC05 – Visualizar pacotes salvos** | |
|---|---|
| **Descrição** | Permite ao administrador consultar as informações dos pacotes salvos na base de dados. |
| **Pré-condições** | Informações salvas na base de dados. |
| **Cenário principal** | 1) O administrador define as informações que serão consultadas na base de dados.<br>2) A ferramenta seleciona na base de dados as informações.<br>3) A ferramenta apresenta as informações salvas na base de dados. |
| **Pós-condições** | Apresentar informações salvas. |
| **Exceção** | Nenhuma informação na base de dados. |

O caso de uso UC06 descreve a definições de alertas. A ferramenta permite definir gerações de alerta, ao capturar determinadas informações. O quadro 7 apresenta o detalhamento do caso de uso.

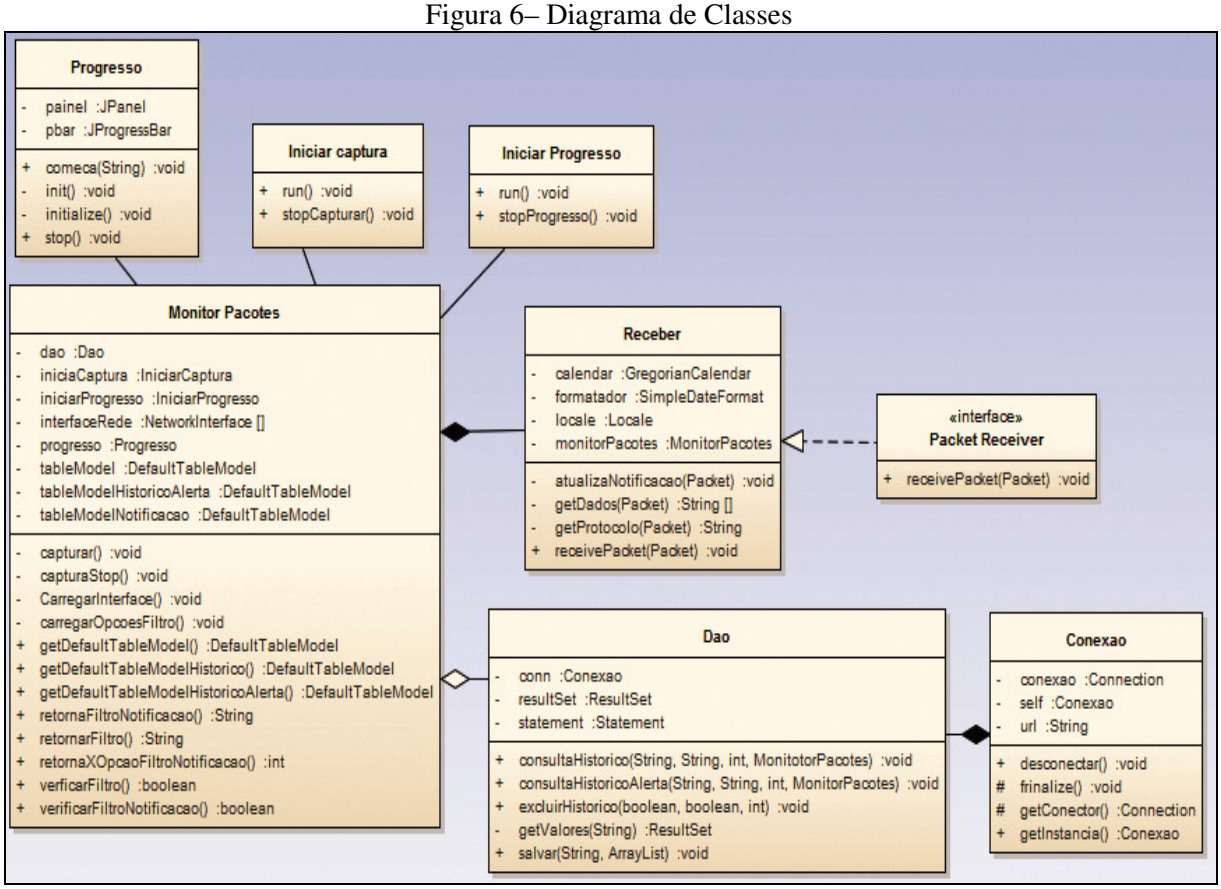
Quadro 7– UC06 - Definir alertas

| **UC06 – Visualizar pacotes salvos** | |
|---|---|
| **Descrição** | Permite ao administrador definir alertas quando determinada informação for capturada. |
| **Pré-condições** | Ferramenta em execução. |
| **Cenário principal** | 1) O administrador inicia a captura dos pacotes.<br>2) A ferramenta apresenta as informações dos pacotes capturados.<br>3) O administrador define as informações que irão gerar alertas.<br>3) A ferramenta verifica as definições de alerta.<br>4) A ferramenta apresenta as informações dos pacotes definidos como alerta. |
| **Pós-condições** | Gerar definições de alerta. |
| **Exceção** | Nenhuma informação de alerta capturada. |

### 3.2.2 Diagrama de Classes

Nesta seção são demonstradas as classes desenvolvidas para o protótipo. A figura 6 apresenta o diagrama de classes.

Figura 6– Diagrama de Classes




A interface `PacketReceiver` disponibilizada pela biblioteca JPCAP, permite que através do método `receivePacket` todos os pacotes possam ser capturados. A classe `Receber` implementa a interface `PacketReceiver` e é registrada para receber os pacotes capturados. O método `receivePacket` da classe `Receber` é responsável por filtrar e analisar todo o conteúdo que é capturado. Além disso, a classe `Receber` é responsável por analisar a geração de alertas, conforme a análise do conteúdo filtrado.

A classe `Dao` é responsável por realizar a seleção, exclusão e salvar as informações capturadas. Ao realizar a seleção das informações na base de dados, o conteúdo é atualizado junto às *grids* da classe `Monitor Pacotes`. A conexão com a base de dados é realizada através da classe `Conexao`. Essa é realizada utilizando o padrão de projeto `Singleton`, de forma que seja estabelecida apenas uma conexão com a base de dados.

Através da classe `Monitor Pacotes` as informações capturadas podem ser visualizadas. Na classe `Monitor Pacotes` é possível iniciar e finalizar a captura dos pacotes,

definir os filtros para seleção, consulta e exclusão de todo o conteúdo, além de permitir definir a interface de rede na qual os pacotes serão capturados.

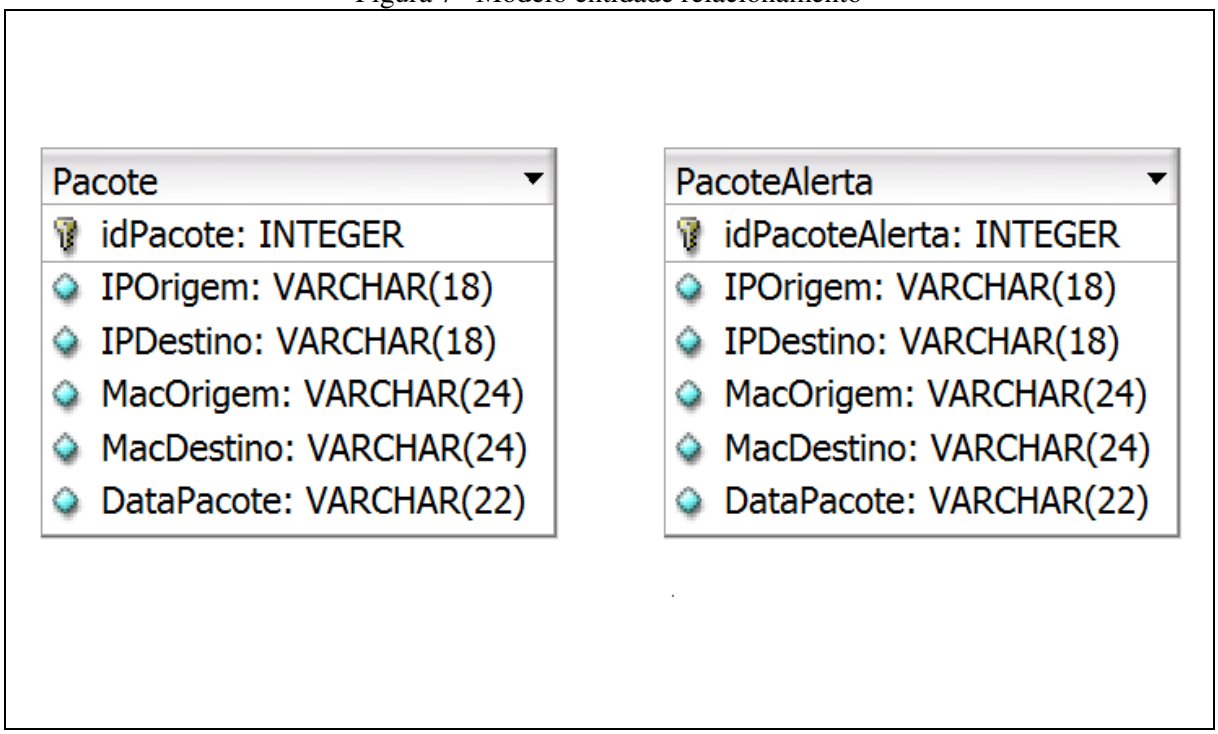
A classe `Iniciar Captura` é responsável por implementar uma *thread* responsável pela captura dos pacotes, de acordo, com a interface de rede definida na classe `Monitor Pacotes`. Através da classe `Iniciar Captura` a *thread* responsável por capturar os pacotes pode ser inicializada ou finalizada.

A classe `Progresso` representa apenas uma tela de espera, que pode ser visualizada ao salvar uma grande quantidade de informações ao mesmo tempo na base de dados, causando certa lentidão durante o processo. A classe `Iniciar Progresso` define uma *thread* responsável por apresentar e ocultar a tela de progresso.

### 3.2.3 Diagrama de entidade relacionamento

Na figura 7 é apresentada a base de dados criada utilizando-se como base as camadas de entidades da ferramenta. No diagrama foram especificadas as tabelas, colunas e chaves primárias. A tabela Pacote representa as informações capturadas e salvas, enquanto que, a tabela PacoteAlerta, representa as informações definidas como alertas.

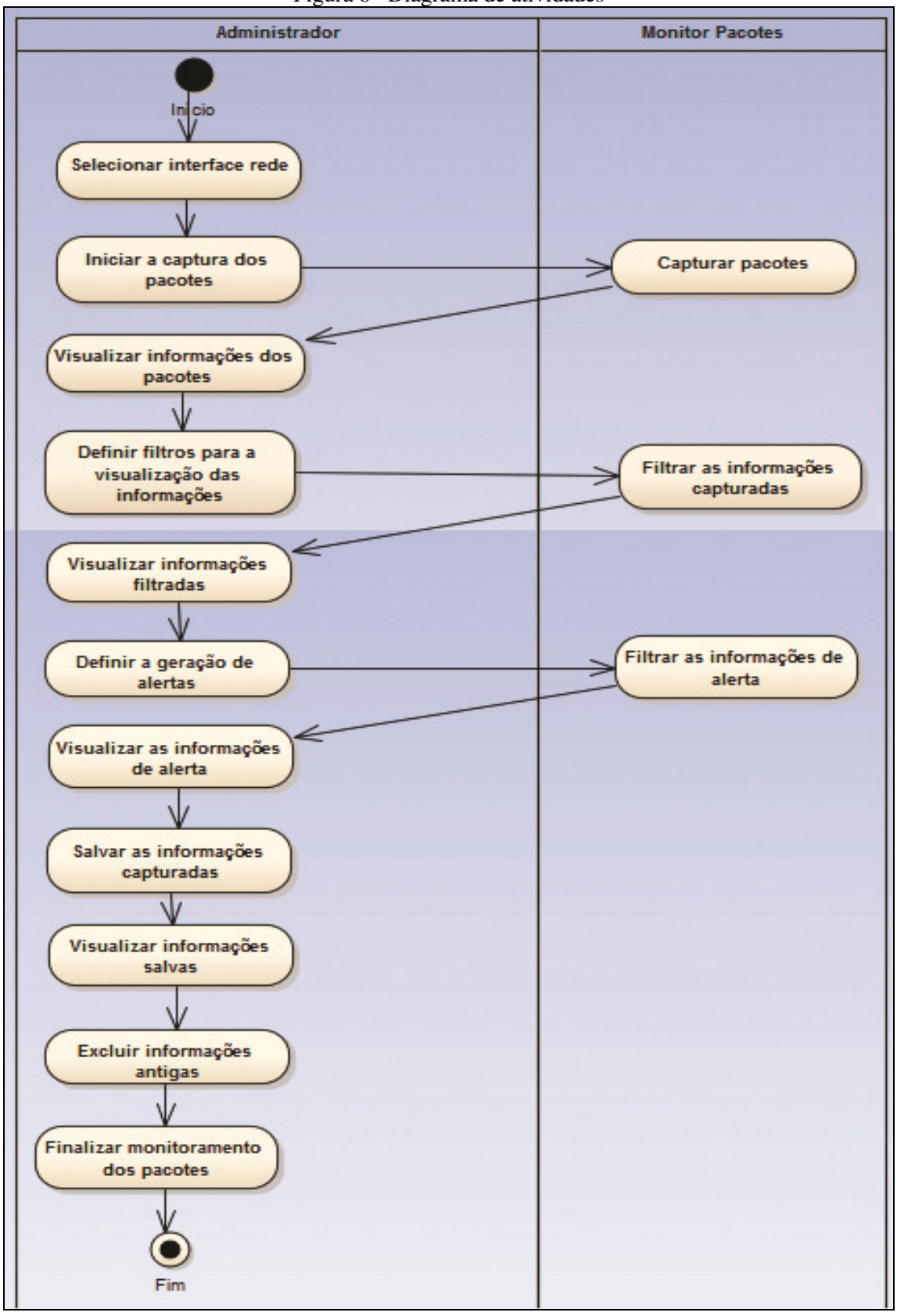
Figura 7– Modelo entidade relacionamento




### 3.2.4 Diagrama de atividades

Na figura 8 é apresentado o diagrama de atividades que irá exibir o fluxo de captura dos pacotes.

Figura 8– Diagrama de atividades



O processo se inicia com o administrador selecionando a interface de rede que possibilitará que os pacotes sejam capturados. Após selecionar a interface, o administrador deverá iniciar a captura dos pacotes.

Ao iniciar a captura dos pacotes, o administrador poderá visualizar as informações capturadas, podendo então definir filtros para a visualização de forma individual das informações, que serão filtradas e disponibilizadas pelo monitor de pacotes. Ao mesmo tempo em que visualiza as informações, o administrador pode definir a geração de alertas, que serão filtrados e apresentados de forma separada e em destaque.

Após a visualização das informações, o administrador pode salva-las, assim como, consulta-las na base de dados. Além de consultar as informações, o administrador pode exclui-las. Por fim, o processo é encerrado ao finalizar a captura.

## 3.3 IMPLEMENTAÇÃO

A seguir estão elencadas as técnicas e ferramentas utilizadas e a operacionalidade da ferramenta.

### 3.3.1 Técnicas e ferramentas utilizadas

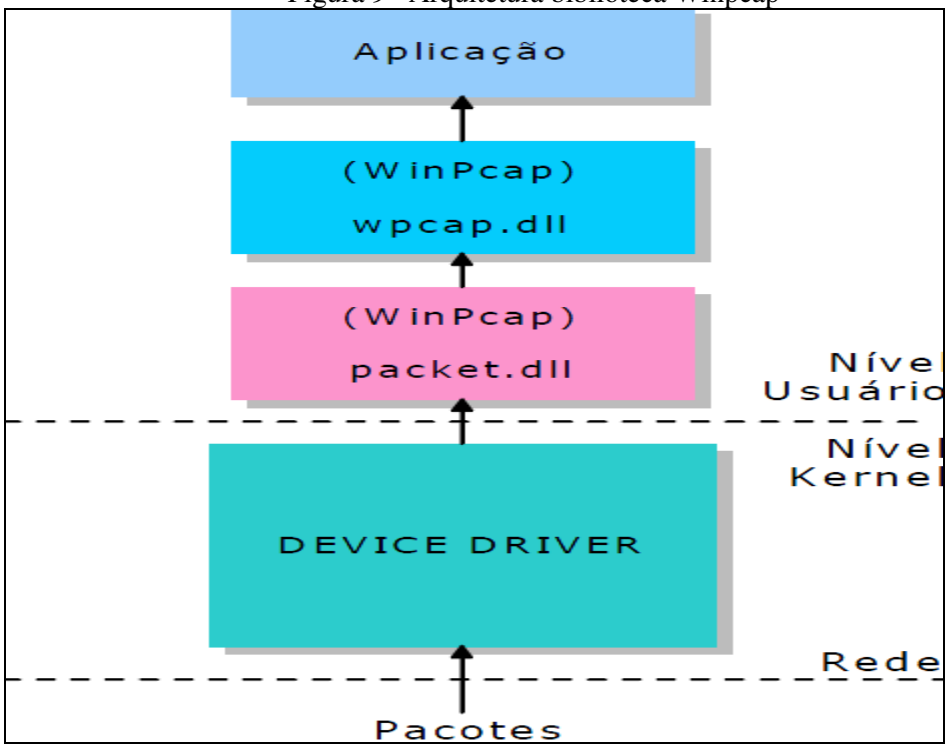
As técnicas e ferramentas utilizadas para o desenvolvimento da ferramenta proposta foram:

a) a linguagem de programação Java, na versão 7.0, para codificar a ferramenta;
b) a IDE Eclipse, na versão Juno, como ambiente de desenvolvimento;
c) a biblioteca JPCAP para a captura de pacotes para a linguagem Java;
d) a biblioteca Winpcap para acesso à interface de rede;
e) o sistema de gerenciador de banco de dados relacional Oracle na versão 10g.

A biblioteca Winpcap é uma biblioteca para acesso a interface de rede baseadas na plataforma Windows. Ela implementa o acesso direto ao tráfego da rede, sem a intermediação das entidades relacionadas.

São duas as bibliotecas responsáveis pelo filtro de pacotes, sendo a de nível mais baixo (packet.dll) responsável pela ligação dinâmica, ou seja, pode ser usada para acessar diretamente as funções do *driver* com uma programação de interface independente do sistema operacional (wpcal.dll), conforme apresentado na figura 9.

Figura 9– Arquitetura biblioteca Winpcap




Fonte: Hilgenstieler (2003, p. 39).

Para que a biblioteca Winpcap seja instanciada para ser utilizada junto à ferramenta é necessário que o administrador se logue na primeira vez no sistema operacional. Ao ser iniciado, o Winpcap habilita o modo promíscuo da interface de rede, permitindo que todos os pacotes que passem pela rede sejam capturados.

Para a captura dos pacotes foi utilizada a biblioteca JPCAP. A biblioteca JPCAP é uma biblioteca Java *open source* para a captura e envio de pacotes. JPCAP é baseado em duas bibliotecas, LipCap para sistemas Unix e WinpCap para sistemas Windows (JPCAP, [2014]).

JPCAP suporta os seguintes tipos de protocolos: Ethernet, *Internet Protocol* (IP), *Address Resolution Protocol* (ARP), *Transmission Control Protocol* (TCP), *User Datagram Protocol* (UDP) e *Internet Control Message Protocol* (ICMP).

Para que as informações capturadas sejam salvas para uma futura análise, foi utilizado o sistema gerenciador de banco de dados (SGBD) Oracle na versão 10g.

### 3.3.2 Captura e apresentação dos pacotes

Para realizar a captura do tráfego de rede foram utilizadas funções da biblioteca JPCAP. O quadro 8 apresenta o método `iniciarCaptura` executado pela *thread* da classe `Captura`, onde é obtida a interface de rede que será monitorada.

Quadro 8– Método `iniciarCaptura`

```
01 public void iniciarCaptura() throws Exception {
02   jpcap.NetworkInterface[] devices = JpcapCaptor.getDeviceList();
03   jpcap=JpcapCaptor.openDevice(devices[dispositivo.getSelectedIndex()],
04                                        2000,false,20);
05   jpcap.loopPacket(-1,new Receiver(this));
06 }
```

Para que ocorra a captura dos pacotes é necessário chamar o método `loopPacket` da interface de rede obtida, passando como parâmetro uma classe que implementa a interface `PacketReceiver`. Essa interface possui um único método chamado `receivePacket`, onde são entregues os pacotes capturados.

A classe `Receiver` implementa a interface `PacketReceiver` conforme podemos observar no quadro 9. O método `receivePacket` da classe Receiver no qual recebe os pacotes capturados, verifica se os pacotes capturados correspondem aos filtros informados junto a tela principal da ferramenta da linha (04 até a 24). Caso não seja informado nenhum filtro todos os pacotes são apresentados.

Quadro 9 – Método `receivePacket`

```
01 public void receivePacket(Packet packet) {
02    EthernetPacket ethernet_packet = (EthernetPacket) packet.datalink;
03    String xDados[] = null;
04    if(!getProtocolo(packet).equalsIgnoreCase("")){
05     if (monitorPacotes.verificarFiltro()) {
06      if ((monitorPacotes.retornaOpcaoFiltro() == 0) &&
07        monitorPacotes.retornaFiltro().equalsIgnoreCase
08        (getProtocolo(packet))){
09           xDados = getDados(packet);}
10    else if(monitorPacotes.retornaOpcaoFiltro() == 1) {
11      if (String.valueOf(((IPPacket)packet).src_ip).equalsIgnoreCase
12         ("/" + monitorPacotes.retornaFiltro())) {
13          xDados = getDados(packet);} }
14    else if (monitorPacotes.retornaOpcaoFiltro() == 2) {
15      if (String.valueOf(((IPPacket)packet).dst_ip).
16            equalsIgnoreCase("/" + monitorPacotes.retornaFiltro())) {
17          xDados = getDados(packet); } }
18    else if (monitorPacotes.retornaOpcaoFiltro() == 3) {
19      if (String.valueOf(ethernet_packet.getSourceAddress()).
20          equalsIgnoreCase(monitorPacotes.retornaFiltro())){
21          xDados = getDados(packet);} }
22    else if (monitorPacotes.retornaOpcaoFiltro() == 4) {
23      if (String.valueOf(ethernet_packet.getDestinationAddress()).
24          equalsIgnoreCase(monitorPacotes.retornaFiltro())) {
25          xDados = getDados(packet); } } }
26     else {
27       xDados = getDados(packet); } }
28
29     if (xDados != null) {
30      monitorPacotes.getDefaultTableModel().addRow(xDados); }
31
32    if (monitorPacotes.verificarFiltroNotificacao()) {
33                atualizaNotificacao(packet); } }
```

Após os pacotes capturados serem filtrados, é verificada a existência de notificações de alerta (linha 32), para serem carregados junto à *grid* de alertas. O método `addRow` (linha 30) atualiza as informações na grid da aba `monitor`.

O método `getProtocolo` da classe `Receiver` utilizado para filtrar os pacotes e montar as informações que serão apresentadas tem o papel de verificar a que protocolo o pacote pertence, consultando a porta de origem e destino do pacote.

O quadro 10 descreve a implementação do método `getProtocolo`. A classe `TCPPacket` da biblioteca JPCAP representa um pacote TCP. É verificado se o pacote capturado é uma instância do tipo TCP, a partir daí são verificadas as portas de origem e destino. É utilizado o número das portas padrões dos protocolos HTTP, HTTPS e FTP para verificar a que protocolo o pacote capturado pertence. Caso o pacote não seja uma instância da classe `TCPPacket`, na (linha 16), é verificado se o pacote capturado é uma instância da classe `ICMPPacket` da biblioteca JPCAP.

Quadro 10 – Método `getProtocolo`

```
01 public String getProtocolo(Packet packet) {
02   if ((packet instanceof TCPPacket)
03     && ((((TCPPacket) packet).src_port == 80)
04     || (((TCPPacket) packet).dst_port == 80))) {
05       return "http";}
06    else if (((packet instanceof TCPPacket))
07      && ((((TCPPacket) packet).src_port == 443)
08      || (((TCPPacket) packet).dst_port == 443))) {
09        return "https";}
10    else if (((packet instanceof TCPPacket))
11      && ((((TCPPacket) packet).src_port == 20)
12      || (((TCPPacket) packet).dst_port == 20)
13      || (((TCPPacket) packet).src_port == 21)
14      || (((TCPPacket) packet).dst_port == 21))) {
15        return "ftp";}
16    else if (packet instanceof ICMPPacket) {
17        return "icmp";}
18
19    return "";
20
21  }
```

### 3.3.3 Salvar pacotes capturados

A captura dos pacotes é feita de forma simples e rápida. O conteúdo monitorado e filtrado é apresentado em tempo real, possibilitando uma ação rápida em caso de suspeita de invasão do sistema. As informações capturadas podem ser salvas para uma futura análise do tráfego da rede. Conforme pode ser observado no quadro 11, o método salvar da classe `Dao` é responsável por salvar as informações junto à base de dados.

Quadro 11– Método `salvar`

```
01 public void salvar(ArrayList valores, String tabela) throws SQLException
02   String valor = "";
03
04   if (valores != null) {
05     for (int i = 0; i < valores.size(); i++) {
06       if (i == valores.size() - 1) {
07         valor = valor + "'"+valores.get(i)+"'";}
08       else {
09         valor = valor +"'"+ valores.get(i)+"'" + ",";
10       }
11     }
12   }
13
14   try {
15    String sql = "INSERT INTO " + tabela
16      +"(IPORIGEM,IPDESTINO,PROTOCOLO,MACORIGEM,MACDESTINO
17    ,DATAPACOTE)VALUES"+
18    "(" + valor + ")";
19
20     stm.addBatch(sql);
21     stm.executeBatch();}
22
23   catch (SQLException e) {
24     e.printStackTrace();}
25  }
```

Na (linha 15) é montada a instrução sql que será passada para o objeto `Statement` (linha 20), utilizado para a inserção das informações na base de dados.

### 3.3.4 Consultar pacotes salvos

No quadro 12 é apresentado o método `consultaHistorico`. Este método tem o papel de, a partir dos filtros passados, consultar as informações dos pacotes salvos junto à base de dados e a partir das informações selecionadas carregar as informações junto a aba de `histórico`.

Quadro 12– Método `consultaHistorico`

```
01 public void consultaHistorico(MonitorPacotes monitorPacotes,int filtro,
02 String val, String val2) throws SQLException, ClassNotFoundException {
03
03   String xDados[] = null;
04   String sql ="SELECT IDPACOTE,IPORIGEM,IPDESTINO,PROTOCOLO,MACORIGEM,"+
05                 MACDESTINO,DATAPACOTE FROM PACOTE ";
06   String param = " WHERE 1=1 ";
07   String order = " ORDER BY 1 ";
08   if((!val.equalsIgnoreCase(""))&&(val2.equalsIgnoreCase(""))){
09     switch (filtro) {
10       case 1:{
11         param = param + " AND PROTOCOLO = " + "'" + val + "'";
12         break;}
13       case 2:{
14         param = param + " AND IPORIGEM = " + "'/" + val + "'";
15         break;}
```

```
16        case 3:{
17          param = param + " AND IPDESTINO = " + "'/" + val + "'";
18
19        break;}
20        case 4:{
21          param = param + " AND MACORIGEM = " + "'" + val + "'";
22          break;}
23        case 5:{
24          param = param + " AND MACDESTINO = " + "'" + val + "'";
25           break;}
26        default:
27          break; }}
28    else if((!val.equalsIgnoreCase(""))&&(!val2.equalsIgnoreCase(""))){
29        if (filtro == 6 ){
30         param = param + " AND DATAPACOTE BETWEEN  to_date("+"'"+val+
31         "'"+","+"'"+ dd/mm/yy "+ "'"+")" +"  AND  "+
32         " to_date("+"'"+val2+"'"+","+"'"+"dd/mm/yy "+ "'"+")" ;}
33    }
34    sql = sql + param + order;
35    ResultSet res = getValores(sql);
36    try{
37      if(monitorPacotes.getDefaultTableModelHistorico().getRowCount()>0){
38        monitorPacotes.tableModelHistorico.setRowCount(0); }
39
40      if(!res.next()){
41        monitorPacotes.getDefaultTableModelHistorico().setRowCount(0); }
42      else{
43        while (res.next()) {
44          xDados = new String[] {
45          String.valueOf(res.getString("IPORIGEM")),
46          String.valueOf(res.getString("IPDESTINO")),
47          String.valueOf(res.getString("PROTOCOLO")),
48          String.valueOf(res.getString("MACORIGEM")),
49          String.valueOf(String.valueOf(res.getString("MACDESTINO"))),
50          String.valueOf(String.valueOf(res.getString("DATAPACOTE")))};
51
52       monitorPacotes.getDefaultTableModelHistorico().addRow(xDados);}}}
53
54    catch (Exception e) {
55      e.printStackTrace(); }
56    finally{
57      res.close(); }
58
59     }
```

No objeto `ResultSet` (linha 35) é carregado as informações referentes a consulta na base de dados, esses dados são extraídos e atualizados junto a *grid* na aba histórico, através do método `addRow` (linha 52).

### 3.3.5 Excluir histórico dos pacotes

No quadro 13 é apresentado o método `excluiHistorico`. Este método tem o papel de, a partir dos filtros passados, excluir as informações dos pacotes salvos junto à base de dados.

Quadro 13– Método `excluirHistorico`

```
01 public void excluirRegistro(int valor, boolean historico,
02 boolean historicoAlerta) throws SQLException {
03   String sqlHis = " DELETE FROM PACOTE ";
04   String sqlHisAlerta = " DELETE FROM PACOTEALERTA ";
05   String param = "  WHERE 1=1 ";
06   PreparedStatement sm = null;
07   switch (valor) {
08   case 0: {
09     param = param + " AND DATAPACOTE = TO_DATE(sysdate,'dd/mm/yy')  ";
10     break;}
11   case 1: {
12     param = param + " AND DATAPACOTE = TO_DATE(sysdate+7,'dd/mm/yy')  ";
13     break;}
14   case 2: {
15     param = param+ " AND DATAPACOTE = TO_DATE(sysdate+30,'dd/mm/yy')  ";
16     break;}
17   default:
18     break;}
19   try {
20     if ((historico) && (historicoAlerta)) {
21       sqlHis = sqlHis + param;
22     sqlHisAlerta = sqlHisAlerta + param;
23       sm = (PreparedStatement) Conexao.getInstacia().getConector().
24     prepareStatement(sqlHis);
25     sm.executeQuery();
26       sm = (PreparedStatement) Conexao.getInstacia().getConector().
27     prepareStatement(sqlHisAlerta);
28     sm.executeQuery();}
29     else if ((historico) && (!historicoAlerta)) {
30       sqlHis = sqlHis + param;
31     sm = (PreparedStatement) Conexao.getInstacia().getConector().
32     prepareStatement(sqlHis);
33     sm.executeQuery();}
34     else if ((!historico) && (historicoAlerta)) {
35       sqlHisAlerta = sqlHisAlerta + param;
36     sm = (PreparedStatement) Conexao.getInstacia().getConector().
37     prepareStatement(sqlHisAlerta);
38     sm.executeQuery();}}
39   catch (ClassNotFoundException e) {
40     e.printStackTrace();}
41   catch (SQLException e) {
42     e.printStackTrace();}
43   finally {
44     if (sm != null) {
45      sm.close();}}
46 }
```

Na (linha 37) foi usado o objeto `PreparedStatement` para passar uma instrução sql para excluir os registros junto ao banco de dados.

### 3.3.6 Operacionalidade da implementação

Esta seção tem por objetivo apresentar a operacionalidade da implementação desenvolvida. Para realizar a demonstração da ferramenta foi realizada a captura de diversos

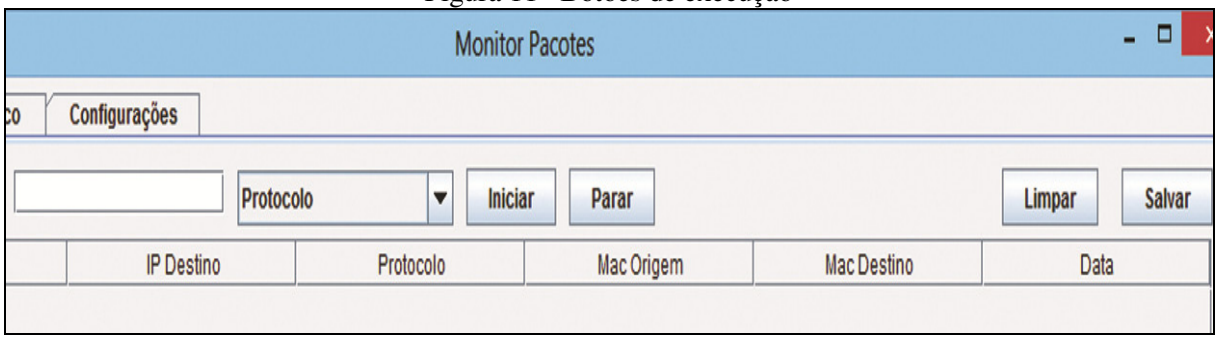
pacotes que trafegavam pela rede, filtrando o conteúdo das informações que se desejava monitorar.

Para iniciar a utilização da ferramenta é necessário selecionar a interface de rede que será responsável pela captura dos pacotes. A figura 10 apresenta a janela para a seleção da interface de rede, disponível na aba `configurações`.

Figura 10– Seleção interface de rede

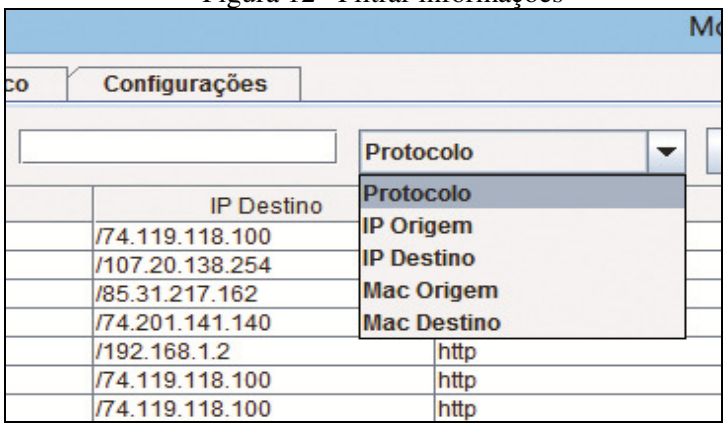

Depois que a interface de rede estiver selecionada é possível iniciar a captura dos pacotes. O botão iniciar inicializa a captura de todos os pacotes que estão trafegando pela rede. O botão parar permite que a ferramenta interrompa o processo de captura dos pacotes. O botão limpar limpa o conteúdo das visualizações apresentadas. O botão salvar armazena o conteúdo filtrado para uma futura consulta. Os botões limpar e salvar presentes na *grid* de alertas apresentam a mesma funcionalidade dos botões descritos anteriormente, conforme a figura 11.

Figura 11– Botões de execução




Logo ao lado dos botões de execução é possível definir filtros para a apresentação das informações. Os filtros podem ser ativados ou desativados durante a execução da ferramenta. Se a caixa de texto estiver preenchida e a opção de filtro definida, o monitor passa a apresentar apenas o conteúdo referente aos filtros selecionados.

Figura 12– Filtrar informações

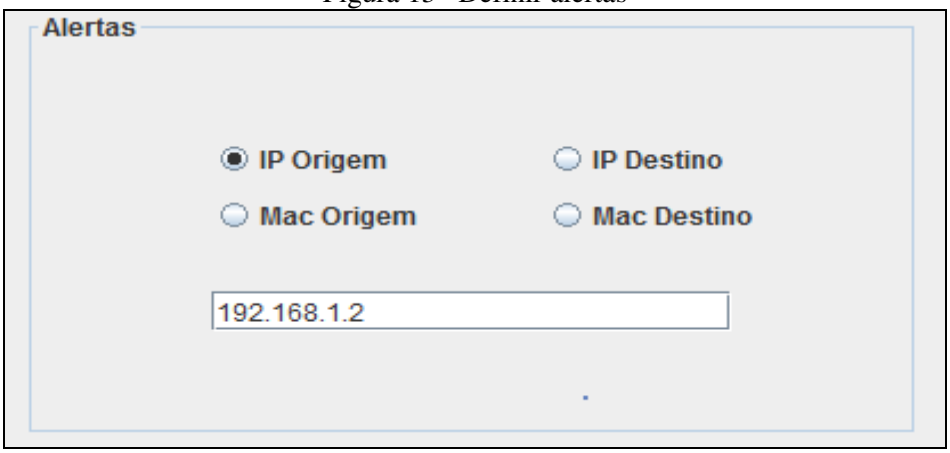



Na figura 12 é possível visualizar as opções de filtro. É possível filtrar os pacotes capturados informando as informações de IP de origem e destino, endereço MAC de origem e destino ou através de determinado protocolo. A ferramenta apresenta e permite filtrar os seguintes protocolos: TCP, IPv4, HTTP, HTTPS, FTP e ICMP. O endereço MAC corresponde a um endereço físico associado à interface de comunicação, que conecta um dispositivo a rede: é composto de 48 bits, os primeiros 24 bits do endereço MAC identificam o fabricante do dispositivo de rede e os últimos 24 bits são designados pelo fabricante do dispositivo.

#### 3.3.6.1 Definir alertas

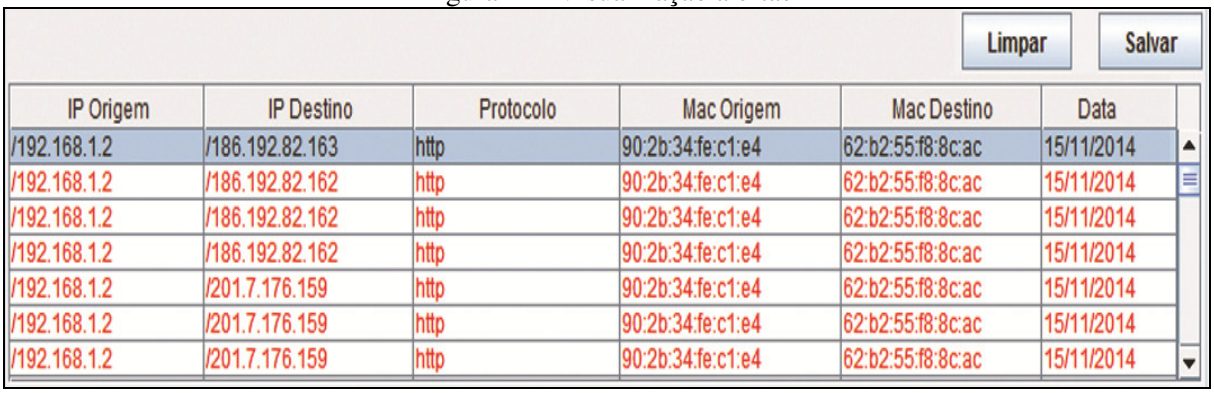
Na aba `configurações` é possível definir a geração de alertas. Definir alertas permite que, quando determinado endereço IP de origem, destino ou endereço MAC de origem ou destino, for capturado, as informações sejam apresentadas na *grid* de alertas da aba `monitor`. Na figura 13 é possível visualizar a definição de determinado endereço IP como situação de alerta.

Figura 13– Definir alertas



Na aba `monitor` na *grid* de alertas, é possível visualizar as informações definidas como situações de alerta. Figura 14.

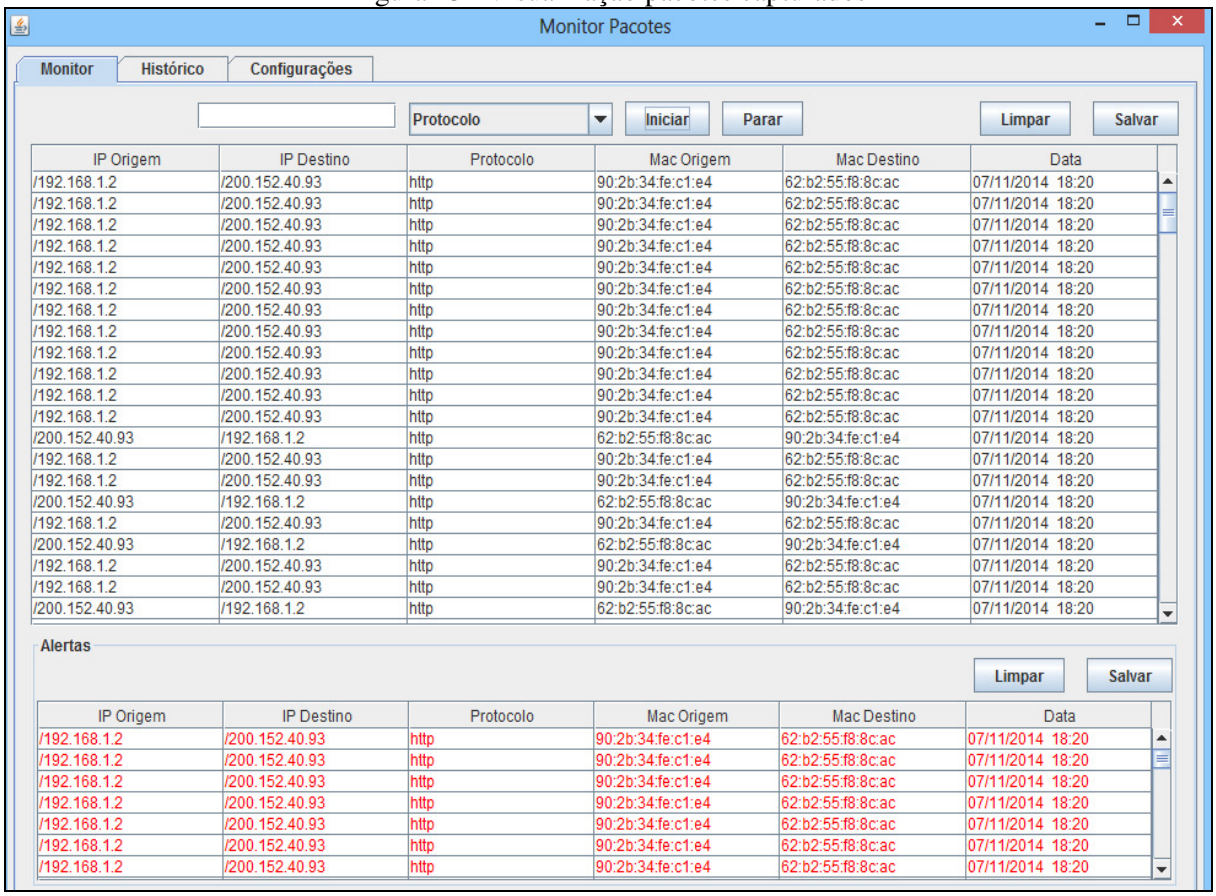
Figura 14– Visualização alertas




As colunas e campos listados na *grid* de alertas correspondem às informações de IP de origem e destino, protocolo, endereço MAC de origem e destino e a data que os pacotes foram capturados pela ferramenta.

Na figura 15 é possível visualizar os pacotes capturados e os alertas gerados.

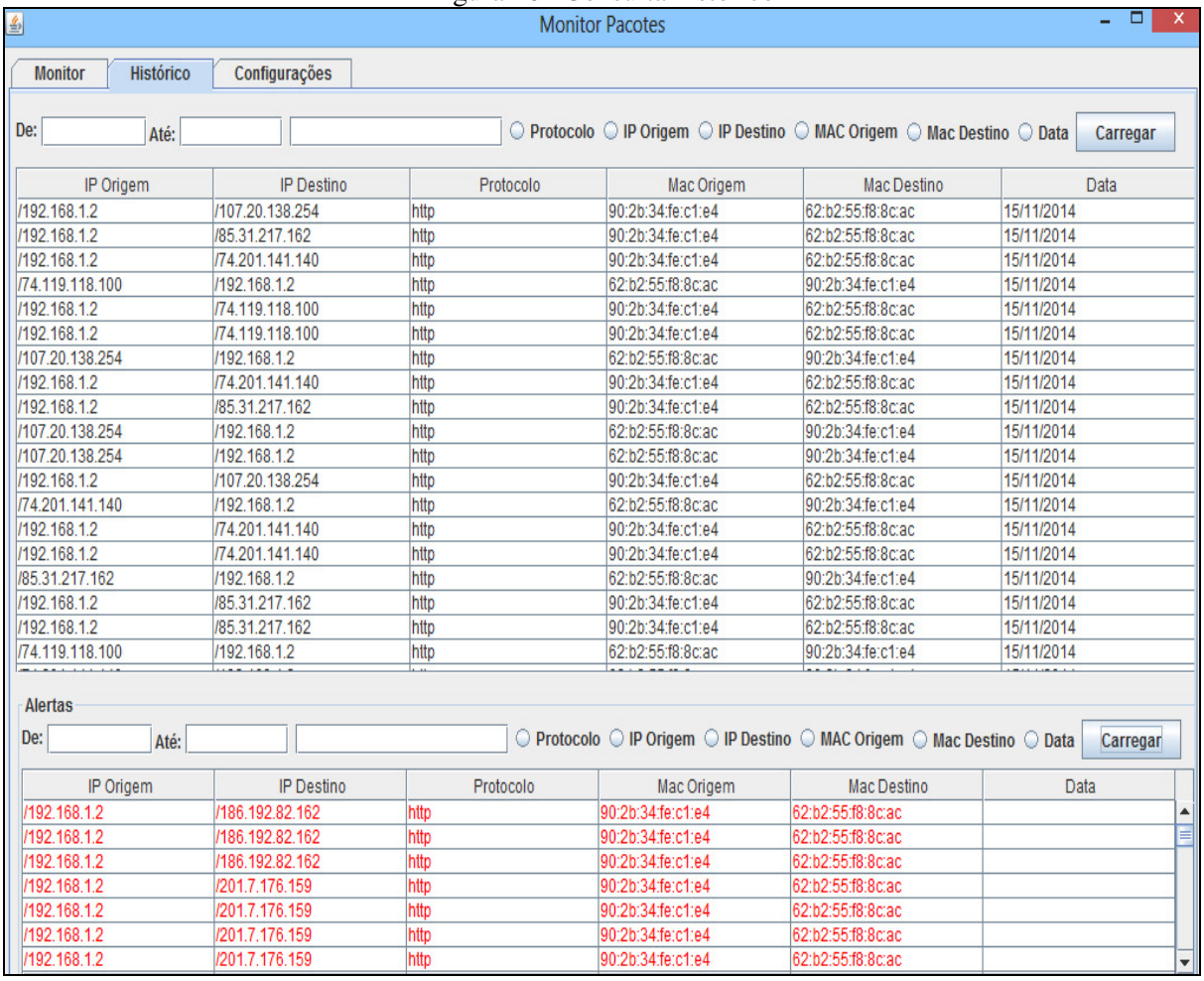
Figura 15– Visualização pacotes capturados



### 3.3.6.2 Consultar histórico

Para consultar as informações salvas na base de dados deve-se acessar a aba `histórico`. Na aba `histórico` é possível visualizar as informações dos pacotes salvos e das definições de alerta, conforme a imagem 16.

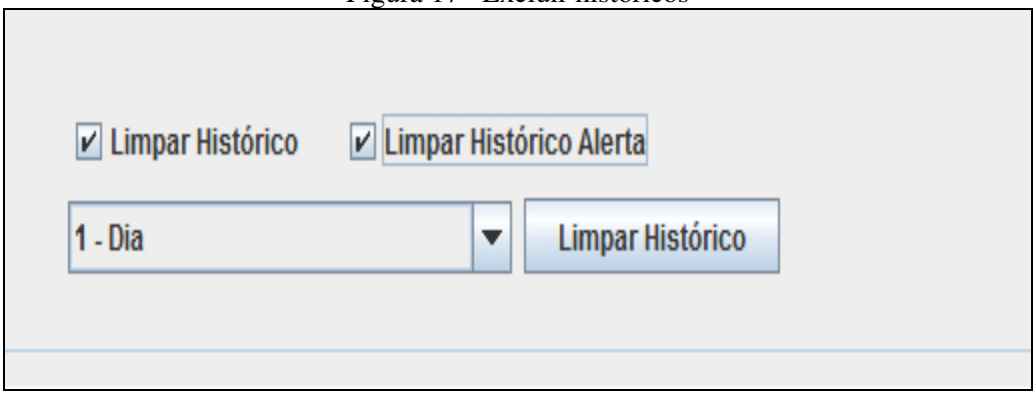
Figura 16– Consulta histórico




### 3.3.6.3 Excluir histórico

Para excluir as informações salvas na base de dados deve-se acessar a aba `configurações`. Na aba `configurações` é possível excluir as informações dos pacotes salvos e das definições de alerta (Figura 17).

Figura 17– Excluir históricos




## 3.4 RESULTADOS E DISCUSSÃO

O presente trabalho teve como objetivo pesquisar e desenvolver uma ferramenta para monitoração e gerenciamento de redes, através da análise do cabeçalho das mensagens que trafegam em uma rede local.

O protótipo desenvolvido neste trabalho demonstrou ser uma ferramenta de monitoração eficaz para redes Ethernet, utilizando a estrutura da arquitetura TCP/IP, sendo capaz de monitorar, extrair e armazenar informações dos pacotes que trafegam em uma rede local.

A captura e visualização das informações são realizadas de forma simples, flexível e rápida. O conteúdo monitorado é visualizado em tempo real. Através das informações capturadas o administrador pode aplicar filtros que permitem analisar de forma individual o tráfego da rede.

Outro detalhe a ser observado em relação ao monitoramento e visualização das informações é a possibilidade de gerar alertas. A geração de alertas possibilita que a informação monitorada possa ser visualizada separadamente no mesmo instante em que foi capturada da rede.

Ao mesmo tempo em que visualiza as informações, o administrador da rede pode salvar as informações na base de dados. O administrador da rede poderá então possuir um histórico das informações e alertas gerados pela ferramenta, permitindo realizar consultas das informações capturadas. Vale resaltar que ao salvar uma grande quantidade de informação ao mesmo junto à base de dados, poderá resultar em uma pequena lentidão no processo.

Através da visualização das informações monitoradas ou salvas, o administrador da rede pode identificar e tomar decisões de forma rápida e eficaz, auxiliando na administração e controle do tráfego da rede.

Em relação aos trabalhos correlatos, o protótipo desenvolvido possui todas as características destacadas dos demais trabalhos. No entanto, existem diferenças em relação aos trabalhos relacionados. O trabalho de Silva (2001) realiza o monitoramento de pacotes TCP/IP com o objetivo de verificar o tráfego da rede em relação aos endereços da camada de rede e transporte, este trabalho, além de monitorar pacotes, extrai do cabeçalho informações que são relevantes para a segurança de uma rede de computadores. Através da ferramenta desenvolvida é possível visualizar as informações correspondentes ao IP de origem e destino, protocolo, endereço MAC de origem e destino e a data em que os pacotes trafegaram pela rede e foram capturados. Já Hilgenstieler (2003) limita-se ao monitoramento e análise apenas do protocolo HTTP, além de possibilitar poucas opções para filtrar os dados dos pacotes capturados.

No quadro 14 podem ser verificadas algumas comparações feitas com os trabalhos correlatos.

Quadro 14 – Características do trabalho desenvolvido e dos correlatos

| Características | Trabalho desenvolvido | Hilgenstieler (2003) | Silva (2001) |
|---|---|---|---|
| Biblioteca Winpcap | X | X | |
| Extrair informações da camada de aplicação | | X | |
| Extrair informações da camada de rede e transporte | X | X | X |
| Filtrar informações capturadas | X | X | X |
| Salvar informações capturadas | X | X | X |
| Consultar informações capturadas | X | X | X |
| Gerar alertas das informações capturadas | X | X | X |
| Biblioteca JPCAP | X | | |

Por fim a ferramenta desenvolvida diferencia-se dos trabalhos correlatos em função da linguagem de programação utilizada e pela utilização da biblioteca JPCAP, que possibilita o desenvolvimento de uma ferramenta de monitoração de redes para diferentes plataformas.

# 4 CONCLUSÕES

O crescimento das redes de computadores e a disponibilidade cada vez maior das informações são fatores que podem trazer grandes riscos para as empresas. Por menor que seja a rede precisa ser gerenciada e monitorada, a fim de garantir, aos seus usuários, segurança, disponibilidade e um aceitável nível de desempenho.

Este trabalho apresentou a criação de uma ferramenta para filtrar os pacotes dos protocolos IPv4, TCP, HTTP, HTTPS e FTP das camadas de rede, transporte e aplicação de todos os hosts dentro da rede, além da possibilidade de determinar endereços IP específicos.

A biblioteca JPCAP que foi utilizada para programar as funções relacionadas à captura dos pacotes se mostrou eficaz e conseguiu abstrair diversas tarefas de análise dos pacotes capturados, assim como a biblioteca Winpcap que foi utilizada para acesso às funções da interface de rede.

Os pacotes capturados e filtrados foram armazenados com sucesso na base de dados e através da especificação de filtros, as funções de consulta e exclusão das informações foram feitas de forma clara e específica, conforme a necessidade. Com isso, a referida ferramenta, através da monitoração de pacotes TCP/IP, pode extrair informações úteis para análise do administrador da rede, possibilitando assim tomar decisões rápidas e eficazes na medida em que sejam necessárias.

Por fim, pode-se dizer que o resultado obtido da implementação atendeu aos requisitos previamente formulados e as bibliotecas utilizadas se mostraram eficazes em relação ao que delas se esperava.

Em relação aos trabalhos correlatos, pode-se afirmar que este trabalho utilizou-se de uma abordagem inspirada nos mesmos. No entanto existem diferenças em relação aos trabalhos relacionados. Uma das principais diferenças é em relação à linguagem de programação utilizada e pela utilização da biblioteca JPCAP, que possibilita o desenvolvimento para diferentes plataformas.

## 4.1 EXTENSÕES

Como sugestões para a continuidade do presente trabalho têm-se:

a) permitir especificar uma quantidade maior de filtros;
b) permitir especificar mais endereços IP para monitoramento;
c) disponibilizar um sistema de bloqueio ao acesso de determinados endereços IP definidos;
d) permitir visualizar um número maior de informações dos pacotes capturados;

e) permitir enviar e-mail para o administrador ao capturar determinado pacote;
f) permitir capturar pacotes de redes distintas.

# REFERÊNCIAS

ALBUQUERQUE, Fernando. **TCP/IP Internet:** protocolos e tecnologias. 3. ed. Rio de Janeiro: Axcel Books, 2001.

BLACK, Tomas L. **Comparação de ferramentas de gerenciamento de redes**. 2008. 64 f. Trabalho de Conclusão de Curso (Especialização em Tecnologias, Gerência e Segurança de Rede de Computadores) - Centro de Ciências Exatas e Naturais, Universidade Federal do Rio Grande do Sul, Porto Alegre. Disponível em: <http://www.lume.ufrgs.br/bitstream/handle/10183/15986/000695315.pdf?sequence=1>. Acesso em: 30 mar. 2014.

BRAGA, Fagner. G. **A continuidade da internet passa pelo IPv6.** 2011. 53 f. Trabalho de Conclusão de Curso (Tecnólogo em Processamento de Dados) - Centro de Ciências Exatas e Naturais, Faculdade de Tecnologia do Estado de São Paulo, São Paulo. Disponível em: < http://www.fatecsp.br/dti/tcc/tcc0010.pdf>. Acesso em: 06 dez. 2014.

FARREL, Adrian. **A internet e seus protocolos:** uma análise comparativa. Rio de Janeiro: Campus, 2005.

HILGENSTIELER, Fernando. **Protótipo de software para monitoração do cabeçalho do protocolo HTTP em uma rede TCP/IP**. 2003. 51 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau. Disponível em: < http://www.bc.furb.br/docs/MO/2003/278729_1_1.pdf>. Acesso em: 30 mar. 2014.

**JPCAP.** [2004]. Disponível em http://jpcap.sourceforge.net/javadoc/index.html. Acesso em: 30 mar. 2014.

KOBUSZEWSKI, André. **Protótipo de software para ocultar textos compactados em arquivos de aúdio utilizando esteganografia**. 2004. 51 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau. Disponível em: <http://www.bc.furb.br/docs/MO/2004/305290_1_1.pdf >. Acesso em: 22 nov. 2014.

KUROSE, James. F; ROSS, Keith. W. **Redes de computadores e a internet**: uma abordagem top-down. 3.ed. São Paulo: Pearson, 2006.

LIBERATO, Rafael. **Comparação de algoritmos de controle de congestionamento do protocolo TCP visando um melhor desempenho em redes sem fio**. 2006. 64 f. Trabalho de Conclusão de Curso (Bacharelado em Sistemas da Informação) - Centro de Ciências Exatas e Naturais, Instituto Superior Tupy - IST, Joinville. Disponível em: <http://www.cpgmne.ufpr.br/redeticbr/index.php?option=com_phocadownload&view=category&download=19:2006-2-alteraes-no-controle-de-congestionamento-do-protocolo-tcp-visando-um-melhor-desempenho-em-redes-sem-fio-rafael-liberato&id=13:redes-de-computadores&Itemid=89>. Acesso em: 22 nov. 2014.

LOPES, Raquel. V; NICOLLETI, Pedro. S; SOUVE, Jacques. P. **Melhores práticas para gerencia de rede de computadores**. Rio de Janeiro: Campus, 2003.

MELLO, Valmes. D. **Sistema para acompanhamento de correção de falhas na rede de computadores da empresa Koerich Telecom baseado em TTS (Trouble Ticket Systems)**. 2005. 60 f. Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau. Disponível em:<http://dsc.inf.furb.br/arquivos/tccs/monografias/TCC2005_2_22_VF_VALMESDMELLO.pdf >. Acesso em: 22 nov. 2014.

PÉRICAS, Francisco, A. **Redes de computadores:** conceitos e arquitetura internet. 2. ed. Blumenau: Furb, 2010.

PUPOLIN, Mirella. N. **Alternativas de segurança em redes de computadores Linux em arquitetura TCP/IP**. 2007. 67 f. Trabalho de Conclusão de Curso (Pós-graduação em segurança de redes Linux) - Centro de Ciências Exatas e Naturais, Faculdade Salesiana de Vitória, Vitória. Disponível em:< http://www.multicast.com.br/sergio/arquivos/monografia-pos-seguranca-alternativas-seguranca-no-linux-com-tcp-ip.pdf >. Acesso em: 22 nov. 2014.

QUEIROZ, Claudemir. C. **Segurança digital:** um estudo de caso. 2007. 59 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Faculdade Lourenço Filho, Fortaleza. Disponível em: <http://www.flf.edu.br/revista-flf/monografias-computacao/seguranca_digital.pdf>. Acesso em: 22 nov. 2014.

SILVA, Paulo. F. **Protótipo de software de segurança em redes para monitoração de pacotes em uma rede TCP/IP**. 2001. 112 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau. Disponível em: < http://campeche.inf.furb.br/tccs/2001-II/2001-2paulofernandosilvavf.pdf >. Acesso em: 30 mar. 2014.

SILVEIRA, André. M. **Rede IPv6 com integração IPv4.** 2012. 49 f. Trabalho de Conclusão de Curso (Sistemas de Telecomunicações) - Centro de Ciências Exatas e Naturais, Centro Federal de Educação Tecnológica de Santa Catarina, São José. Disponível em: < http://wiki.sj.ifsc.edu.br/wiki/images/e/e3/TCC_AndreManoeldaSilveira.pdf>. Acesso em: 06 dez. 2014.

STALLINGS, Wiliam. **Redes e sistemas de comunicação de dados**. 5. ed. Rio de Janeiro: Campus, 2005.

TANENBAUM, Andrew, S. **Redes de computadores**. 4. ed. Rio de Janeiro: Campus, 2003.

WORDPRESS**. Camada de transporte TCP/IP.** [2014]. Disponível em http://rapazes.wordpress.com/2010/05/27/camada-de-transporte-tcpip. Acesso em: 22 nov. 2014.