Faculdade de Tecnologia de Americana
Curso Superior de Tecnologia em Segurança da Informação

Vulnerabilidades de Segurança em Serviços de Rede (FTP, SSH, E-MAIL e WEB)

RODOLFO DA SILVA SANTOS

Americana, SP
2012

Faculdade de Tecnologia de Americana
Curso Superior de Tecnologia em Segurança da Informação

Vulnerabilidades de Segurança em Serviços de Rede (FTP, SSH, E-MAIL e WEB)

RODOLFO DA SILVA SANTOS

rodolfo.santos1@fatec.sp.gov.br

Trabalho de conclusão de curso apresentado a Faculdade de Tecnologia de Americana como parte das exigências do Curso de Tecnologia em Segurança da Informação para obtenção de título de Tecnólogo em Segurança da Informação.

Orientador: Prof. Dr. José Luís Zem.

Americana, SP
2012

BANCA EXAMINADORA

Orientador:_____
Prof. Dr. José Luiz Zem – FATEC

Presidente:_____
Prof. Ms. Carlos Henrique Rodrigues Sarro – FATEC

Convidado: _____
Prof. Edson Roberto Gaseta – FATEC

AGRADECIMENTOS

Agradeço a Deus por tudo ter dado certo.

Agradeço aos meus pais e a meus irmãos que sempre me apoiaram.

Agradeço a Faculdade de Tecnologia de Americana pela oportunidade de estudar e me desenvolver como profissional e ser humano.

Agradeço ao meu orientador Prof. Dr. José Luiz Zem, por toda ajuda e dedicação.

Agradeço ao Prof. Ms. Carlos Henrique Rodrigues Sarro pela atenção e todas as orientações que ajudaram a elaboração deste trabalho.

DEDICATÓRIA

Dedico esse trabalho a meu pai Jorge a minha mãe Teresinha e a meus irmãos sem os quais eu não seria nada. E dedico a todas as pessoas que me acompanharam por esses anos de curso e o tornaram mais motivador.

RESUMO

Este trabalho tem como objetivo estudar serviços que operam em redes de computadores e vulnerabilidades de segurança que eles possam apresentar. Para isso serão discutidas as características que permitem o funcionamento desses sistemas e vulnerabilidades comuns a eles. Apresentaram-se também conceitos sobre softwares específicos para auditória de segurança em redes de computadores. E o resultado e discussão de testes realizados em ambiente computacional previamente apresentado.

Com a crescente utilização das redes de computadores a necessidade de segurança nos diversos serviços disponibilizados através delas aumenta rapidamente. Apesar de existirem inúmeros sistemas destinados a melhorar a segurança nesses serviços. Também existem muitos softwares destinados a explorar (com fins legais ou ilegais) vulnerabilidades de segurança em redes de computadores. Logo a segurança em uma rede de computadores e, portanto nos serviços providos por ela deve ser objeto de constante estudo dos profissionais de Segurança da Informação.

Palavras chave: Serviços de rede. Vulnerabilidades. Segurança da Informação.

ABSTRACT

This paper aims to provide services that operate on computer networks and security vulnerabilities they may have. For this we discuss the features that allow the functioning of these systems and vulnerabilities common to them. Also presented ware concepts on specific software to audit security in computer networks. And the result and discussion of tests of previously presented computational environment.

With the increasing use of computer networks to security needs in the various services available through them increases rapidly. Although there ware numerous systems to improve security in these services. There is also much software designed to explore (legal or illegal purpose) security vulnerabilities in computer networks. Soon the security in a network of computers and therefore the services provided by it should be the object of constant study of Information Security professionals.

Keywords: Network Services. Vulnerabilities. Information Security.

SUMÁRIO

Lista de Figuras

Lista de Tabelas

1 INTRODUÇÃO

A expansão de aplicações de rede nos mais diversos ambientes sociais vem causando profundas mudanças na forma de relacionamento entre os elementos que compõem a sociedade. Durante as primeiras décadas de existência, o uso das redes de computadores estava restrito a pesquisadores, estudantes e funcionários de empresas. Sob essas condições, a segurança nunca precisou de maiores cuidados. Porém, atualmente a Internet possui milhões de usuários que usam a rede para fazer operações bancárias, compras, declarações de impostos e muitas outras ações onde a segurança é crítica. Com isso a segurança em redes de computadores tornou-se um problema em potencial (TANENBAUM, 2003).

Vulnerabilidades de segurança são os pontos de partida para violações de sistemas computacionais. Através delas o invasor consegue acessar o sistema alvo ou pode prejudicar seu funcionamento. Muitas vezes uma vulnerabilidade de segurança pode ser causada por erros de configuração ou escolha de tecnologias inapropriadas para a aplicação implementada. Há também a possibilidade (muito frequente) de o próprio *software* sair da fabrica com erros em seu projeto ou código. Entretanto neste trabalho não se propõe estudar todos os assuntos relacionados à segurança da informação. Ele limitara-se ao estudo de vulnerabilidades de segurança em serviços específicos de rede.

Vários protocolos de rede foram criados nos primórdios da Internet. E por isso não foram projetados para serem seguros no atual panorama da Internet, formada por milhões de computadores e milhares de aplicações diferentes. Muitos órgãos internacionais e empresas de tecnologia realizaram e realizam um grande esforço visando à criação de novas tecnologias que atendam mais apropriadamente a atual realidade da internet. Mas a substituição de tecnologias amplamente difundidas entre empresas e usuários domésticos é um processo caro e às vezes possui muitos empecilhos jurídicos e sociais. Isso resulta em uma rede cheia de pontos suscetíveis a vulnerabilidades de segurança (TANENBAUM, 2003).

O objetivo principal desse trabalho é fazer um estudo de vulnerabilidades de segurança em serviços de rede importantes para a sociedade. Levando em consideração apenas os fatores tecnológicos que influenciam o funcionamento dos

serviços estudados e sua maior ou menor segurança. Para isso será feito um levantamento bibliográfico com a finalidade de obter-se uma base teórica que permitirá a realização de testes práticos. Os testes possibilitarão um maior entendimento dos fatores existente na implementação da segurança em redes de computadores.

A complexidade que envolve a segurança em redes de computadores não se limita a equipamentos eletrônicos e a programas de computadores. Ela envolve todas as pessoas que dela façam uso, seja fornecendo serviços ou usufruindo deles. Porém neste trabalho se discutirá apenas fatores técnicos que possibilitam o funcionamento e segurança de serviços de rede, mais especificamente: a WEB, o E-MAIL, o SSH e o FTP.

O levantamento teórico de todos esses serviços será feito no segundo capítulo. Ele terá por base obras de autores consagrados na área de redes de computadores e segurança da informação. Sites técnicos relacionados ao assunto e trabalhos, reconhecidos de outros autores. Esse levantamento teórico abrangerá a história de tecnologias que possibilitam, em conjunto, que os serviços de rede estudados, operem. Será também abordado o funcionamento desses serviços. E por fim, serão discutidas as falhas de segurança mais comuns nos serviço de rede estudado.

No terceiro capítulo serão apresentados os *softwares* envolvidos nos testes, e o ambiente de teste. No quarto capítulo apresentaram-se os testes que consistirão em um *scanning* de vulnerabilidades utilizando-se o *scanner* de vulnerabilidades Nessus. Que é uma poderosa ferramenta de auditoria de segurança em sistemas computacionais. Esses testes serão feito em um computador executando os servidores que proveem os serviços discutidos nesse trabalho. Em seguida as falhas encontradas serão discutidas e suas correções apresentadas. Esse capítulo corresponderá à parte prática desse trabalho. E por fim no ultimo capitulo serão apresentadas considerações finais baseadas no que foi estudado nos capítulos anteriores.

2 VULNERABILIDADES EM SERVIÇOS DE REDE

O estudo de vulnerabilidades em serviços de rede é de grande importância para garantir a continuidade do serviço e a confiança dos usuários. Esse estudo faz parte da rotina de muitos desenvolvedores e projetistas de sistemas. Neste capítulo será feito um levantamento bibliográfico dos principais aspectos que permitem o funcionamento de cada serviço de rede estudado neste trabalho. Cada um desses serviços será abordado em um subcapitulo diferente, na seguinte sequencia WEB, E-MAIL, SSH e FTP. As vulnerabilidades de segurança mais comuns a cada um desses serviços serão apresentadas após o estudo do serviço de rede.

2.1 Serviços ou Servidores

A Internet nas duas últimas décadas tornou-se uma imensa infraestrutura de engenharia capaz de fornecer diversos serviços a milhares de usuários de forma eficiente e barata. Serviços que abrangem desde o envio de correspondência eletrônica, o amplamente difundido e-mail, até aplicações mais restritas a profissionais de tecnologia da informação como é o caso do serviço de terminal remoto. A maioria dos usuários desconhece completamente o funcionamento de uma rede de computadores. O fato de a Internet ter sido projetada para ser completamente transparentes ao usuário final o livra de se preocupar com detalhes técnicos como o canal de comunicação, interfaces e protocolos. Devido às características intuitivas de serviços de rede, praticamente qualquer pessoa consegue usufruir facilmente de seus recursos (MAIA, 2009).

Muitos autores consagrados definirão serviço de rede em suas obras. A seguir são exemplificadas algumas. Segundo Tanenbaum (2003, 42p) a primitiva de serviço se define como:

> *"Um serviço é especificado formalmente por um conjunto de primitivas (operações) disponíveis para que um processo do usuário acesse o serviço. Essas primitivas informam ao serviço que ele devê executar alguma ação ou relatar uma ação executada por uma entidade par. [...] com frequência, as primitivas serão normalmente chamadas do sistema."*

Segundo Maia (2009, 4p):

> *"Um serviço é uma funcionalidade da rede disponível de forma transparente para seus usuários e aplicações."*

Logo pode-se definir serviço como um conjunto de operações que permitem a rede disponibilizar várias funcionalidades de forma transparente para processos de usuários e aplicações. As redes de computadores fornecem vários serviços, mas os mais difundidos são o correio eletrônico, a transferência de arquivos e a WEB. Dentre estes serviços a WEB tem se destacado por seu estrondoso.

Uma rede de computadores é formada por diversos equipamentos e programas que juntos permitem seu perfeito funcionamento. Porém nesse trabalho serão estudados exclusivamente sistemas de camada de aplicação[1] que possibilitam que tais serviços de rede funcionem, ou seja, *softwares* que compõem computadores usados diretamente por usuários finais (que podem ser especialistas em redes de computadores ou usuários comuns). Dois desses componentes são o servidor e o cliente, cuja definição pode ser observada a seguir, a partir das visões de diversos autores. A Figura 01 apresenta uma típica rede de computadores.

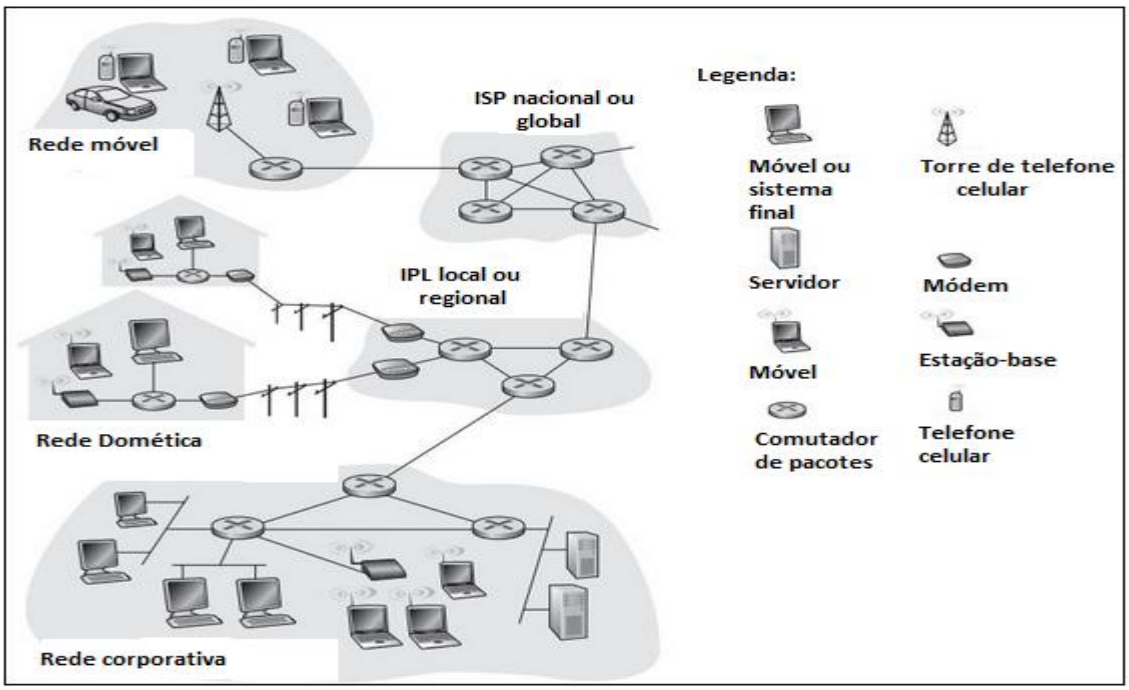

Figura 01 - Rede de Computadores (KUROSE; ROSS, 2003, 3p)

[1] A camada de aplicação é uma das camadas da pilha de protocolos de comunicação da Internet. No modelo TCP/IP ela fica acima da camada de transporte. Ela contém todos os protocolos de nível mais alto como o FTP e SMTP (TANENBAUM, 2003).

Conforme Kurose e Ross (2010, 62p):

> *"Em uma arquitetura cliente – servidor há um hospedeiro sempre em funcionamento, denominado servidor, que atende a requisições de muitos outros hospedeiros, denominados clientes. Estes podem estar em funcionamento às vezes ou sempre."*

Para Stallings (2005, 136p):

> *"Conforme o termo sugere, um ambiente cliente servidor é preenchido por clientes e servidores. As máquinas cliente geralmente são PCs ou estações de trabalho de único usuário, que apresentam uma interface altamente amigável ao usuário final. A estação baseada no cliente geralmente apresenta o tipo de interface gráfica que é mais apropriada aos usuários. [...] Cada servidor no ambiente cliente/servidor fornece um conjunto de serviços compartilhados do usuário para os clientes."*

De acordo com Maia (2009 12p):

> *"No modelo cliente-servidor existem as figuras do cliente e do servidor. O cliente é o dispositivo que solicita um servidor um, enquanto o servidor recebe, processa e responde às solicitações do cliente. Um servidor pode ser responsável por um ou mais serviços, ...Como os servidores concentram todas as solicitações, esses dispositivos devem ter características e hardware e software que permitam oferecer requisitos mínimos de disponibilidade e desempenho, o que influencia no parâmetro custo."*

No modelo cliente-servidor existe as figuras do cliente e do servidor. O cliente é o dispositivo que solicita um serviço, enquanto o servidor recebe, processa e responde ás solicitações ao cliente. Um servidor pode ser responsável por um ou mais serviços.

Como os servidores concentram todas as solicitações, esses dispositivos devem ter características de *hardware* e *software* que permitam oferecer requisitos mínimos de disponibilidade e desempenho, o que influencia no parâmetro custo (MAIA, 2009).

2.2 *World Wide Web* (WWW)

A WEB originou-se no CERN (Centro Europeu para Pesquisas Nucleares), em março de 1989, a partir da proposta de interligar documento em teia, do físico Tim Berners-Lee; esse pretendia solucionar as dificuldades na trocar de informação entre os vários pesquisadores do CERN (TANENBAUM, 2003).

Um ano e meio depois o primeiro protótipo, em modo texto, já era operacional. Em dezembro de 1991 foi realizada uma demonstração pública na conferência Hypertext 91, em San Antonio, no Texas.

Toda a publicidade gerada neste evento chamou a atenção de outros pesquisadores como Marc Andreessen que em fevereiro de 1993 lançava o primeiro navegar web com interface gráfica, o Mosaic.

Já em 1994 o CERN e o MIT (Instituto de Tecnologia de Massachusetts), assinaram um acordo onde criaram a W3C (*World Wide Web Consortium*), organização que tem por fim desenvolver a Web, padronizar protocolos e demais tecnologias envolvidas em seu funcionamento.

Em pouco mais de uma década a Web se tornou um das mais bem sucedidas aplicações de rede sendo confundida por muitos usuários como a própria Internet (TANENBAUM, 2003).

A WWW (*World Wide Web*) ou WEB é uma estrutura que permite o acesso a documentos distribuídos por máquinas espalhadas pela Internet. Esses documentos ou páginas WEB que podem conter textos, imagens, áudio, e vídeo são interconectadas por *hiperlinks*[2].

As páginas são visualizadas com auxílio de um programa (cliente) denominado *browser* ou navegador que busca a página solicitada, interpreta o conteúdo dela e a exibe na tela do computador.

A navegação entre diversas páginas pode ser feita facilmente através do esquema de *hiperlinks* que apontam de uma página para outra e forma o que é

[2] *Strings* (sequencia de caracteres) de texto ou imagens que são *links* para outras páginas (TANENBAUM, 2003).

conhecido como *hipertexto*. Um exemplo de como isso é feito pode ser visto na Figura 02, onde um usuário pode clicar em um *link* e ir à outra página, que por sua vez possui *links* que o levará a outras páginas com conteúdos diversos (TANENBAUM, 2003).

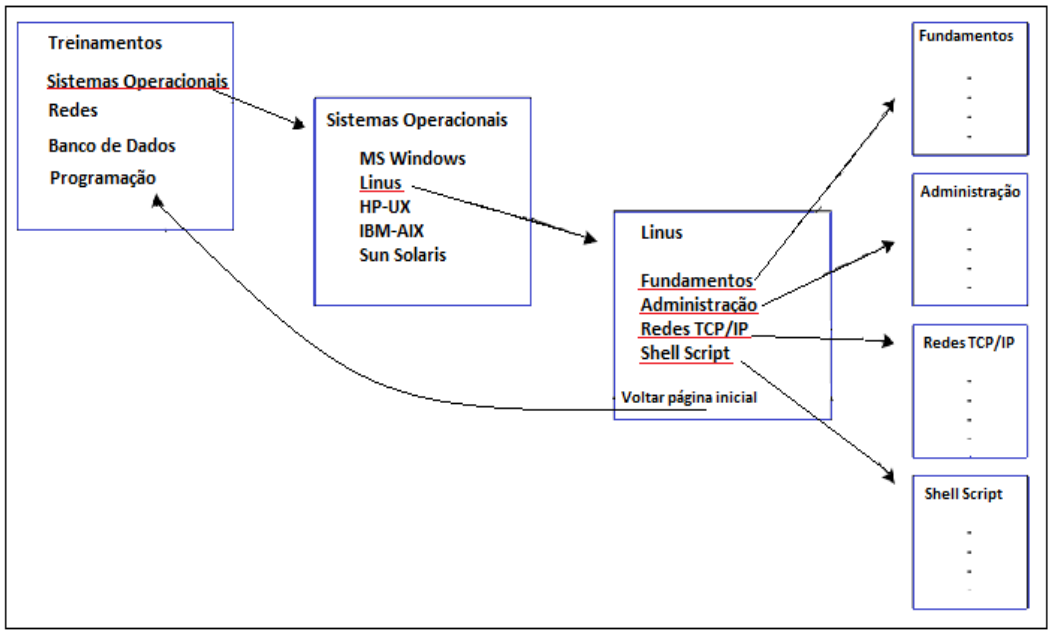

Figura 02 – Navegação através de hipertextos (MAIA, 2009, 210p)

A arquitetura da WEB é formada por três elementos básicos: o servidor WEB, o cliente WEB e o protocolo HTTP (Figura 03). O servidor armazena fisicamente os arquivos das páginas, que podem estar armazenados em diversos servidores, e o esquema de *hiperlinks* torna a localização física delas completamente transparente para o usuário (MAIA, 2009).

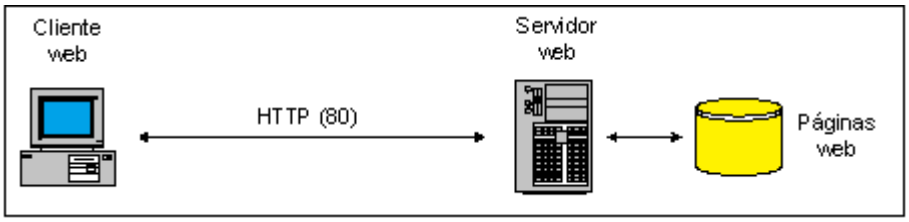

Figura 03 – Serviço WEB (MAIA, 2009, 211p)

2.2.1 Cliente WEB

O cliente WEB, também chamado de *browser* ou navegador, é responsável basicamente por receber as páginas WEB, processar e exibi-las ao usuário.

Quando o usuário clica em um *hiperlink* ou digita uma URL[3] na barra de endereços, o *browser* executa uma série de etapas em ordem que resultaram na exibição da página requisitada no monitor do computador.

Suponha-se que um usuário esteja pesquisando sobre tecnologias empregadas na edição de páginas Web e encontre o seguinte link: http://www.exemplo.br/Home/WebHome. Quando o usuário clicar nesse link o navegador irá determinar a URL, verificando o link selecionado; em seguida perguntará ao DNS[4] o endereço IP de http://www.exemplo.br/Home/WebHome (1); o DNS responderá com o endereço IP(2); logo que receber o IP o browser estabelecerá uma conexão TCP com a porta 80 do servidor WEB www.exemplo.br (3); com a conexão estabelecida o navegador envia um comando solicitando o arquivo /Home/WebHome(4); o servidor www.exemplo.br envia o arquivo /Home/WebHome para o navegador(5); a conexão TCP será encerrada; o navegador exibirá todo o texto de /Home/WebHome; e por fim busca e exibirá todas as imagens que o arquivo contem, através das referências que ele possui dos caminhos para essas imagens (TANENBAUM, 2003).

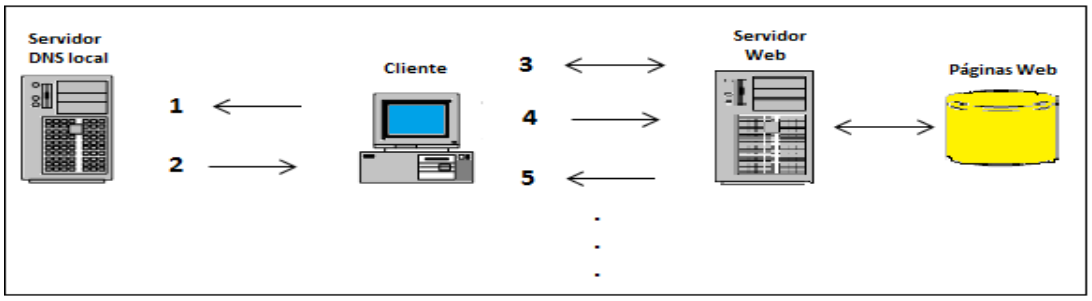

Figura 04 - Comunicação entre navegador e servidor WEB (Autoria própria, 2012)

[3] O URL (*Uniform Resource Locator*) funciona como o nome universal de uma página. Os URLs têm três partes: o protocolo (também chamado esquema), o nome DNS da máquina em que a página está e o caminho do local onde a página está armazenada (TANENBAUM, 2003).

[4] O DNS é um esquema hierárquico de atribuição de nomes a hosts baseado no domínio e em um sistema de bancos de dados distribuídos para implementar esse esquema de nomenclatura. Ele é usado principalmente para mapear nomes de hosts e destinos de mensagens de correio eletrônico em endereços IP, mas também pode ser usado para outros objetivos (TANENBAUM, 2003).

Os navegadores modernos possuem muitos recursos que facilitam seu uso. Botões que permitem ao usuário “navegar” entre as páginas e recursos de pesquisa. Praticamente todo o histórico de navegação fica armazenado, seja de páginas visitadas a minutos ou a semanas, e é exibido com apenas alguns clicks. O usuário também pode salvar o endereço de páginas nos “favoritos” para visita-las posteriormente.

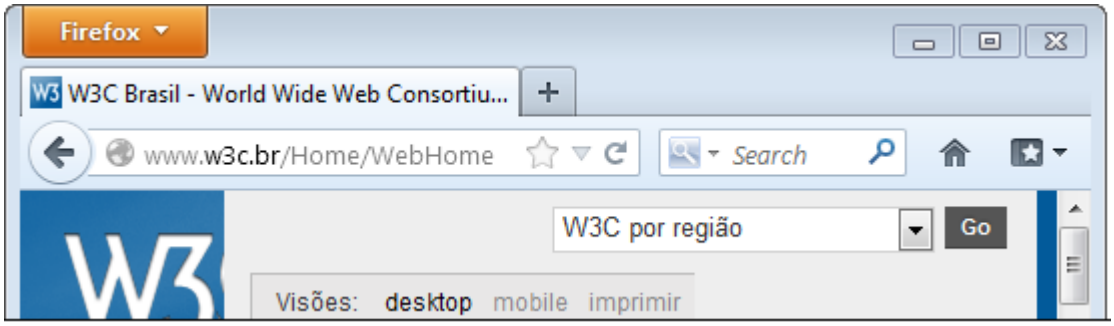

Figura 05 – Navegador Mozilla Firefox 15.0.1(Autoria própria, 2012)

A maioria dos navegadores possui uma linha de status no rodapé, que indica qual etapa do processo de solicitação-recebimento da página está sendo executado no momento. Dessa forma se algum erro ocorrer o usuário poderá saber o motivo do problema.

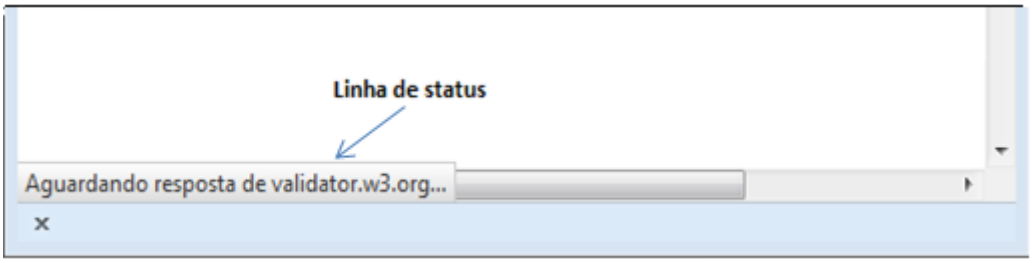

Figura 06 – Linha de status do *browser* Mozilla Firefox 15.0.1 (Autoria própria, 2012)

Apesar dos navegadores serem basicamente interpretadores de HTML[5], as páginas web podem conter vários outros objetos como: desenhos gráficos, ícones, mapas e imagens. Porém nem todas as páginas web contém HTML. Algumas podem conter desde arquivos PDF até vídeos, ou qualquer outro tipo de arquivo dentre centenas de possibilidades.

[5] A HTML é uma linguagem de marcação, ou seja, uma linguagem para descrever como os documentos devem ser formatados. É usada para se escrever páginas Web (TANENBAUM, 2003).

2.2.2 Servidor WEB

O servidor WEB é geralmente um computador com grande capacidade de memória e processamento responsável por armazenar páginas web. Paginas web que poderão ser enviadas para os clientes assim que solicitadas. Os servidores web mais usados no mercado são o Apache e o Microsoft Internet Information Services (IIS).

De acordo com Tanenbaum (2003, 659p), apesar de os servidores mais modernos terem várias características, em essência as etapas que o servidor executa em seu *loop* principal são:

1. Aceitar uma conexão TCP de um cliente (um navegador).

2. Obter o nome do arquivo solicitado.

3. Obter o arquivo (do disco).

4. Retornar o arquivo ao cliente.

5. Encerrar a conexão TCP.

Um dos grandes problemas de servidores WEB é o desempenho. Como a maioria das solicitações de páginas exige acesso ao disco, à etapa três no exemplo acima, o número de requisições que podem ser atendidas pelo servidor fica limitado. Uma solução para essa limitação é armazenar os arquivos mais requisitados em *cache*[6]. O que exige um grande volume de memória principal e justifica o alto custo desses equipamentos (TANENBAUM, 2003).

Em função dos avanços tecnológicos, o servidor WEB deixou de ser apenas um repositório de páginas estáticas para tornar-se um servidor ativo, integrado a outros servidores de aplicação e banco de dados. Nesse caso, as páginas residentes no servidor podem ser criadas dinamicamente conforme, por exemplo, uma solicitação de um usuário. Uma das primeiras iniciativas nesse sentido foi a utilização do padrão CGI (*Common Gateway Interface*). O CGI permite a passagem

[6] Um modo bastante simples de melhorar o desempenho é gravar páginas que foram solicitadas, caso elas tenham de ser usadas novamente. Essa técnica é especialmente efetiva com páginas muito visitadas. Guardar páginas para uso subsequente é uma técnica chamada armazenamento em cache (TANENBAUM, 2003).

de parâmetros de forma padronizada para a execução de um programa no servidor WEB. Na maioria dos casos, os programas executados no servidor são desenvolvidos em linguagens de *script* como Perl ou Python. Atualmente, os servidores WEB utilizam produtos específicos para a geração de páginas dinâmicas, como PHP do PHP *Group*, *Active Server Pages* (ASP) da Microsoft e JavaServer Pages (JSP) da Sun Microsystems (MAIA, 2009).

2.2.3 Protocolo HTTP

O HTTP (*Hyper Texte Transfer Protocol*) é um protocolo de camada de aplicação destinado à distribuição de informação de hipermídia pela Internet. Ele tem sido usado pela iniciativa *World Wide Web* desde 1990. Sua primeira versão foi o HTTP/0.9 que era um protocolo simples de transferência de informação. O HTTP/1.0, que foi a segunda versão melhorada desse protocolo, permitia a transferência de informação no formato MIME[7] além de transferir meta-informações de controle de transferência de dados. O formato da sintaxe de solicitação e resposta também foi modificado. Porém o HTTP/1.0 não englobou em sua base considerações sobre hierarquia de *proxies*, *caching*, necessidade de conexões persistentes ou mesmo virtual *host*. A última versão do HTTP o HTTP/1.1 inclui requisitos mais rigorosos para assegurar a execução confiável de suas características (RFC, 2616).

O protocolo HTTP especificado nos [RFC[8] 1945] e [RFC 2616], como todos os outros protocolos estudados neste trabalho, se comunica de acordo como o modelo cliente-servidor e pode ser usado em qualquer aplicação que envolva hipertexto. Apesar de seu nome (*Hyper Texte Transfer Protocol*), o HTTP não é um protocolo de transferência de *hipertexto*. Em vez disso ele é um protocolo para transferência de informações com uma eficácia necessária para fazer saltos de *hipertexto*. Ele permite que uma página WEB armazenada em um determinado servidor seja copiado para o cliente e exibido por ele, sendo que os dados dessa página poderão

[7] O formato MIME define estruturas e regras para a transmissão de mensagens que não utilizam o código ASCII (TANENBAUM, 2003).

[8] As RFCs (*Request For Comments*) são documentos simples de práticas e padrões da Internet. São elaborados por técnicos especialistas em sua área e cada RFC recebe um número sequencial (ROBBINS, A; REEBE, F, 2005).

ser texto puro, áudio, imagens ou qualquer outra informação disponível pela Internet. A versão do protocolo mais recente e utilizada atualmente é o HTTP/1.1. (STALLING, 2005).

O uso mais típico do HTTP é entre um navegador e um servidor WEB. Para garantir a confiabilidade na transferência de páginas WEB esse protocolo usa o protocolo TCP. Uma página WEB é constituída de objetos que são simplesmente arquivos (arquivos HTML, imagens, arquivos de áudio e vídeo). Se uma página possuir 11 imagens e um vídeo ela terá 13 objetos, pois o arquivo HTML também é um objeto.

As mensagens utilizadas no HTTP são no formato texto e podem ser de dois tipos: solicitações do cliente para o servidor (*request*) e respostas do servidor para o cliente (*response*). As solicitações do cliente para o servidor são feitas através de métodos como, por exemplo, GET, que solicita uma página ao servidor, ou PUT, que manda gravar uma página no servidor. Por exemplo, no comando a seguir, o método GET solicita uma página WEB ao servidor. Uma mensagem de solicitação pode ser seguida por cabeçalho para complementar a mensagem. A seguir pode ser visto um exemplo de mensagem de solicitação HTTP (MAIA, 2009).

GET /index.htm HTTP/1.1

Host: localhost

Uma mensagem de resposta de arquivo é formada pela versão do protocolo HTTP que está sendo utilizada pelo servidor, um código de estado e uma pequena frase explicando o código. Existem diversos códigos de estado que pode indicar o sucesso da solicitação ou mesmo uma falha no servidor. No exemplo a seguir, a primeira linha informa que o protocolo utilizado é HTTP/1.1 e a solicitação foi recebida com sucesso pelo servidor. Além disso, o servidor envia uma série de informações para o cliente, antes de enviar a página WEB solicitada, como, por exemplo, a data e hora do envio, e o tipo servidor WEB acessado.

HTTP/1.1 200 OK

Date: Sat, 10 Nov 2012 12:39:44 GMT

Server: Apache/2.0.63 (Unix) mod_ss1/2.0.63 OpenSSL/0.9.8b

Last-Modified: Thu, 24 Dec 2011 12:27:35 GMT

ETag: “ef11e9-2b0-98489bc0”

Accept-Ranges: bytes

Content-Length: 688

Content-Type: text/html

Devido a sua natureza, o protocolo HTTP não identifica o cliente no momento da conexão e não mantém informações sobre o que o cliente está solicitando ao servidor WEB o que é conhecido como conexão sem estado. O que é bem pertinente para a função que ele desempenha: pegar rapidamente diversas páginas WEB de servidores distribuídos e envia-las a diversos usuários. Para manter o controle das conexões, foi introduzido um artifício, chamado *cookie*. Os *cookies* são enviados pelo servidor WEB, juntamente com as páginas solicitadas pelo cliente, e armazenados em um diretório na estação do cliente. Sempre que novas solicitações forem feitas ao servidor, os *cookies* também serão enviados e o servidor poderá acompanhar e atualizar as solicitações do cliente (MAIA, 2009).

2.2.4 Segurança na WEB

A WEB permite que as aplicações se comuniquem sem que o usuário final necessite conhecer qualquer um dos mecanismos que permitem a comunicação. Muitas vezes por trás de uma aplicação há vários sistemas se comunicando. Garantir que as informações fornecidas por um usuário chegue ao seu destino sem a possibilidade de ser interceptada, perdida ou adulterada é um grande desafio. Uma vez que a WEB foi projetada sem nenhuma preocupação com a segurança, em uma época que sua função se limitava a compartilhamento de documentos. O crescente uso dela para realizar operações bancárias e de comercio gera uma grande preocupação com a segurança por trás das funcionalidades fornecidas (MAIA, 2009).

Os primeiros ataques em redes de computadores exploravam vulnerabilidades ligadas à aplicação dos protocolos da sequência TCP/IP. Com o grande investimento na correção destas vulnerabilidades. Os ataques tem se concentrado na camada de aplicação. Além de compartilhar os problemas de segurança comuns a Internet, a Web está sujeita a ataques de negação de serviço, mensagem antiga e estouro de pilha. Além de possui outros possíveis problemas de segurança como: vulnerabilidades no servidor Web, possibilidade de manipulação dos URLs, fraquezas nos identificadores de sessão e mecanismos de autenticação, injeção de código HTML e injeção de comando SQL.

A proteção do servidor Web é um ponto crítico para muitas organizações que tem na WEB sua principal fonte de recursos. O mau gerenciamento do servidor é um dos principais fatores que colocam em risco a segurança, não só do próprio servidor, mas também da rede. Saber o que já vem configurando por padrão no servidor e manter uma documentação detalhada das modificações feitas no decorrer da vida do equipamento é essencial para manter todo o sistema seguro (FIGUEIREDO, 1999).

A hierarquia de diretórios de um servidor Web é composta pelo diretório raiz do servidor que possui informações de controle, como arquivos de configuração. E pelo diretório raiz de documentos que é responsável por todo armazenamento de conteúdo público disponível através do protocolo HTTP. Ao se executar um servidor WEB todas as informações que estiverem abaixo da raiz do diretório de documentos poderão ser disponibilizadas publicamente. O usuário “root” e o usuário “nobody” nunca devem ser usados para executar o servidor Web, pois eles possibilitaram algumas vulnerabilidades no sistema. O ideal é que se crie um usuário (e um grupo) específico para o servidor Web. Isso restringira o acesso ao sistema de arquivos a esse usuário e ataques feitos ao servidor afetará apenas o usuário específico do servidor (FIGUEIREDO, 1999).

Uma aplicação pode ser projetada para trafegar dados pela URL (utilizando o método GET), isso traz agilidade e praticidade. Entretanto esse recurso abre espaço para ataques de manipulação de URL em que um hacker altera determinadas partes de uma URL e a partir dessa manipulação tem acesso a partes da aplicação, onde não deveria ter acessar ou a informações restritas (BARCELOS, 2010).

Suponha-se uma aplicação onde os usuários façam login e se forem do departamento de recursos humanos, tenha acesso à determinada página. E que o sistema gere uma URL no seguinte formato:

www.exemplo.com.br/listagem?rec_humano=false

Um usuário que possua conhecimentos mais profundos dos mecanismos da aplicação poderá manipular essa URL trocando o false por true o que lhe daria acesso a dados sigilosos. A manipulação de URL além de poder permitir acesso a áreas restritas de uma aplicação também pode permitir recuperação de informações, listagem de diretórios, descoberta da estrutura da aplicação e localização de arquivos (BARCELOS, 2010).

Para evitar que informações trafeguem pela URL deve-se desabilitar o método GET e utilizar o método POST, que envia informações através do corpo das mensagens. Mesmo assim ainda existem formas de se manipular tais informações, por isso é aconselhável que se criptografe o corpo das mensagens.

Quando um usuário requisita alguma informação a alguma aplicação que tenha um banco de dados. Ela envia parâmetros através dos métodos GET ou POST que serão validados e usados para se montar consultas SQL que por sua vez serão executadas no banco de dados. Esse processo permitirá que os dados requisitados sejam retornados pela aplicação.

O ataque de SQL *injection* consiste na inserção de códigos SQL nos campos destinados a entrada de dados. Esse código é executado na aplicação e chega até o banco de dados podendo retornar informações sigilosa, causar a parada da aplicação ou permitir acesso a recursos da aplicação. Caso não existam mecanismos de segurança que validem as entradas de dados e ou impeçam ataques por esse meio, a aplicação terá uma grande falha de segurança. Para se evitar esse tipo de ataque é necessário que se use técnicas de validação nos campos de entrada de dados. Outra forma de evitar esse tipo de ataque é utilizar recursos de segurança do próprio banco de dados que implementam a segurança no recebimento de dados (BARCELOS, 2010).

Cross site scripting consiste na inserção de código HTML malicioso em uma aplicação que interaja com informações fornecidas pelo usuário. Esse tipo de ataque permite uma série de tarefas que prejudicam a aplicação, como a execução de scripts maliciosos no navegador da vítima, roubo de sessões, alterar a aparência de sites, reter o controle do navegador do usuário etc. Assim como o SQL injection, esse ataque se aproveita de falhas na validação de dados logo para se evitar esse tipo de ataque o ideal é fazer uma rigoroso validação na entrada de dados evitando que caracteres especiais sejam aceitos. Um gerenciamento de sessão adequado também contribuirá muita para a segurança da aplicação (BARCELOS, 2010).

O protocolo HTTP não oferece nenhuma confidencialidade no envio das informações, inclusive as senhas, contas bancárias e números de cartões de crédito são enviados em claro, ou seja, sem nenhum tipo de criptografia. A versão segura do protocolo HTTP é conhecida como HTTPS (HTTP Secure). O HTTPS é uma combinação do protocolo HTTP com o protocolo SSL (*Secure Socket Layer*), que permite garantir o sigilo das transações bancárias e no comércio eletrônico na Internet. O protocolo HTTPS responde na porta reservada TCP 442 (MAIA, 2009).

2.3 Correio eletrônico

O E-MAIL (*eletronic mail*) ou correio eletrônico é um meio de comunicação assíncrono, ou seja, remetente e destinatário não necessitam estar coordenados um com o horário do outro, logo enviam e leem mensagens de acordo com sua disponibilidade de tempo. Diferentemente do correio normal o E-MAIL é rápido e barato (KUROSE; ROSS, 2010).

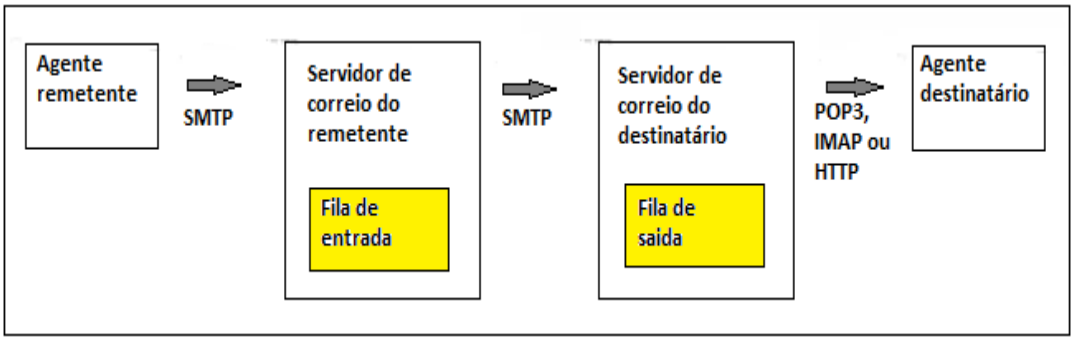

Figura 07 – Agente de usuário e de transferência (Autoria própria, 2012)

O funcionamento do serviço de E-MAIL depende de três componentes principais: agentes de usuário, servidores de correio e o protocolo SMTP. Agentes de usuário são programas que possibilitam o envio, edição, resposta e retransmissão de mensagens entre outras operações. Ao enviar uma mensagem para um destinatário o agente de usuário do remetente transmite essa mensagem para o servidor de correio do remetente. Essa mensagem é colocada na fila de saída desse servidor que a armazena até poder transmiti-la ao servidor de correio do destinatário. Caso exista algum problema impedindo a transmissão da mensagem o servidor de correio do remetente a armazenará até que possa envia-la, caso não consiga enviar a mensagem em um determinado período de tempo o servidor do remetente deverá avisa-lo do ocorrido e excluir a mensagem. Uma vez enviada, a mensagem do servidor de correio do remetente ao servidor de correio do destinatário ela será armazenada na caixa postal desse usuário que poderá acessa-la quando quiser, a Figura 07 demonstra todo esse processo (KUROSE; ROSS, 2010).

2.3.1 O protocolo SMTP

O SMTP, definido no RFC 5321 é o principal protocolo de camada de aplicação envolvido no envio de correspondência pela Internet. Esse protocolo usa o serviço confiável de transmissão de dados do protocolo TCP/IP para transmitir mensagens entre os servidores de correio eletrônico do remetente e do(s) destinatário(s). Do mesmo jeito que os demais protocolos discutidos neste trabalho o SMTP possui um lado cliente e um lado servidor, um lado cliente que funciona no servidor de correio do remetente e um lado servidor que funciona no servidor de correio do destinatário. É importante observar que o SMTP não utiliza servidores intermediários para transmitir correspondência pela Internet (KUROSE; ROSS, 2010).

Embora o SMTP possua muitas qualidades, como evidencia sua ubiquidade na Internet, ele é uma tecnologia antiga com características arcaicas. A mais notável delas é a restrição do envio do corpo de mensagens, e não apenas o cabeçalho, ao formato ASCII de sete bits, o que torna necessária a conversão de qualquer conteúdo que não esteja nesse formato para ser transmitido. Isso era justificável no

passado quando a capacidade de transmissão era reduzida e não se enviava conteúdo de multimídia através desse serviço (KUROSE; ROSS, 2010).

Segundo Stallings (2005) o funcionamento do SMTP consiste na troca de comandos pelos lados emissor e receptor do protocolo sendo que a iniciativa é do emissor. Ao estabelecer uma conexão, o emissor SMTP enviará ao receptor os comandos e receberá para cada um, uma respectiva resposta. Logo o funcionamento básico do SMTP se resume ao estabelecimento de uma conexão, troca de um número de comandos-respostas e encerramento da conexão.

2.3.2 Os protocolos de acesso ao correio eletrônico

Nos parágrafos acima foi demonstrado como é o processo de transmissão de uma mensagem de correio eletrônico do servidor de correio do remetente para o servidor do destinatário, porém o processo de transmissão de arquivos do servidor para o agente de usuário não foi especificado. Até o início da década de 1990 o agente de usuário ficava alocado no hospedeiro servidor, logo o usuário lia suas mensagens nessa máquina. Porém, atualmente o acesso a correio eletrônico ocorre via arquitetura cliente servidor. Logo o usuário acessa sua caixa de correspondência através de um cliente que funciona separado do servidor, por exemplo, um PC no escritório, um *laptop* ou mesmo um PDA. Isso cria a necessidade de um meio de comunicação entre o agente de usuário e servidor de e-mail que possibilite a recuperação de mensagens (KUROSE; ROSS, 2010).

O agente de usuário não pode usar o SMTP para obter as mensagens do servidor de correio. Essa operação é de recuperação (*pull*), e o SMTP é um protocolo de envio (*push*). Para resolver esse problema foi introduzido protocolos de recuperação de mensagens que possibilitam a transmissão de arquivos do servidor de E-MAIL para o agente de usuário. Existem vários protocolos populares que exercem essa tarefa. Os mais usados são: **POP3** (*Post Office Protocol* versão 3), **IMAP**(*Internet Messege Access Protocol*), e **HTTP** a Figura 08 apresenta um resumo dos protocolo usados no correio eletrônico (KUROSE; ROSS, 2010).

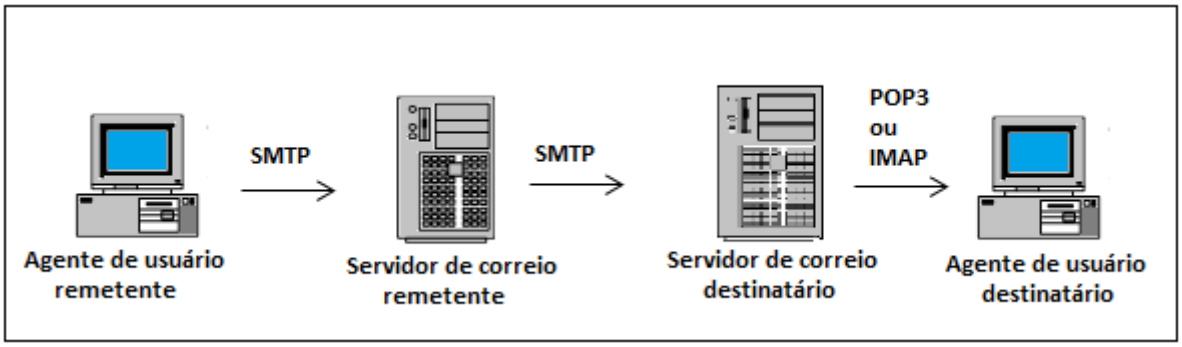

Figura 08 – Protocolos usados no correio eletrônico (KUROSE; ROSS, 2003 p.93)

O *Post Office Protocol* - Versão 3 (POP3) tem por função possibilitar que um usuário acesse dinamicamente sua caixa de correio eletrônico em um servidor. Geralmente, isto significa que o protocolo POP3 é utilizado para permitir que uma estação de trabalho recupere mensagens que estão armazenadas em um servidor. Porém esse protocolo não se destina a manipulação de mensagens, normalmente ele baixa e exclui as mensagens do servidor.

Definido na RFC 1939, o POP3 é um protocolo de acesso ao correio simples e por isso suas funcionalidades são bastante limitadas. Ele inicia a comunicação entre o agente de usuário e o servidor abrindo uma conexão TCP. Com a conexão ativada o protocolo passa por três fases: autorização, transação e atualização. Na fase de autorização, o agente de usuário envia um nome de usuário e uma senha sem usar nenhum recurso de segurança. A segunda fase, transação, é a responsável pela recuperação de mensagens. Nela o agente de usuário também pode marcar mensagens que devem ser apagadas, remover essa marca e obter estatísticas de correio. Por fim, a terceira fase ocorre quando o usuário encerra a sessão POP3. Nela o servidor de correio eletrônico ira apagar as mensagens marcadas.

A maioria dos programas de correio eletrônico que utilizam o POP3, simplesmente baixa todas as mensagens do servidor e esvazia a caixa de correio. Isso significa que a única cópia que o usuário possui esta em seu disco rígido. Portanto se ele perder, por algum problema ou descuido a mensagem, não terá como recuperá-la, além de não poder ver as mensagens que já foram baixadas de outro computador. Para resolver essa limitação um novo protocolo de acesso a correio eletrônico foi desenvolvido, o IMAP (TANENBAUM, 2003).

O IMAP (Internet Message Access Protocol) permite que um cliente acesse e manipule mensagens de correio eletrônico em um servidor. O IMAP tem recursos de manipulação de pastas e mensagens remotas, chamadas "caixas de correio". Ele também inclui operações para criar, excluir e renomear caixas de correio. Verificar novas mensagens, remover mensagens permanentemente, analisar, pesquisar e busca seletiva de atributos de mensagens, textos e suas porções.

Tabela 01 - Comparação entre POP3 e IMAP (TANENBAUM, 2003, p.649)

Característica	POP3	IMAP
Onde o protocolo é definido	RFC 1939	RFC 2060
Porta TCP usada	110	143
Onde o correio eletrônico é armazenado	PC do usuário	Servidor
Onde o correio eletrônico é lido	Off-line	On-line
Tempo de conexão exigido	Pequeno	Grande
Utilização de recursos do servidor	Mínima	Intensa
Várias caixas de correio	Não	Sim
Quem guarda cópias das caixas de correio	Usuário	ISP
Bom para usuários em transito	Não	Sim
Controle do usuário sobre *C*	Pequeno	Grande
Download de mensagens parciais	Não	Sim
Quotas de disco constituem um problema	Não	Possível após algum tempo
Implementação simples	Sim	Não
Suporte difundido	Sim	Crescendo

Criado a partir da necessidade de um protocolo sem as desvantagens do POP3. O IMAP, definido na RFC 2060, diferente do POP3 não limpa a caixa de E-MAIL do usuário quando ele desfaz a conexão com o servidor. Ele simplesmente pressupõe que as mensagens do usuário ficaram indefinidamente em várias caixas de correio. Muitos mecanismos de leitura de mensagens ou parte de mensagens são fornecidos por esse protocolo. Tendo em vista o pressuposto que o usuário não armazenará as mensagens com ele, o IMAP fornece recursos para criar, destruir e manipular várias caixas de correio no servidor. O formato geral do IMAP é semelhante ao POP3. Porém ele possui dezenas de comandos permitindo uma

ampla comunicação entre cliente e servidor. A Tabela 01 apresenta uma comparação entre o POP3 e o IMAP (TANENBAUM, 2003).

Como se pode notar na Tabela 01 o IMAP, apesar de mais complexo, possui muito mais recursos. Com ele o usuário acessa e manipula recursos no servidor de e-mail dele de qualquer computador conectado a Internet. O controle de download também é outra vantagem, pois o POP3 não possui suporte a esse recurso. Entretanto quando se usa o IMAP há a preocupação com quotas de disco, pois o usuário pode acumular muitas mensagens e preencher todo o espaço reservado a ele no disco do servidor.

2.3.3 Segurança nos sistemas de correio eletrônico

O correio eletrônico, assim como os demais serviços providos através da Internet, possui diversas vulnerabilidades que devem ser levadas em consideração. Seja na escolha das tecnologias empregadas para fornecer o serviço ou na configuração de servidores e clientes é necessário uma atenção especial. Mensagens não solicitadas (*SPAM*), vírus (o correio eletrônico é uma das maneiras mais usadas para se distribuir vírus na Internet), proteção de conteúdo de mensagens e falsificação de identidades são algumas das vulnerabilidades que um sistema de correio eletrônico pode apresentar.

Alguns problemas de segurança no correio eletrônico possuem uma grande dificuldade de solução. O *SPAM*, por exemplo, é um problema sem soluções efetivas uma vez que é o próprio usuário que define o grau de importância de uma mensagem. Porém apesar de não se poder eliminar o problema é possível diminuir drasticamente os seus impactos. Medidas como configurar os servidores de e-mail para despachar somente mensagens oriundas de computadores do domínio da própria instituição onde estão localizados elimina na maioria dos casos o problema. Existem sites na Internet que são especializados em mante listas com o nome de servidores que não possuem essa restrição. Logo configurar o servidor para não receber mensagens de servidores que estão nessas listas também será outra forma de proteger os servidores. Apesar de eficiente essas medidas pode ser facilmente contornada se quem envia os *SPAMs,* estiver interessado especificamente no sistema de e-mail da instituição, pois ele pode criar uma conta de e-mail no domínio da empresa e enviar *SPAM* para os servidores. Os *E-mails Proxies*, também são

uma forma de proteger o sistema contra *SPAM*. Além de proverem outras funcionalidades como filtrar mensagens através de diversos critérios (remetente, destinatário, tamanho, hora, etc.) e tomar medidas especificadas para cada tipo de mensagem filtrada (SOUSA, 2006).

Sem dúvida a propagação de vírus por mensagens eletrônicas é o maior problema apresentado pelo serviço de correio eletrônico. Toda uma rede pode ser comprometida a partir de uma simples mensagem aberta por um usuário desavisado ou mesmo prejudicar a comunicação de um sistema caso esse note a presença do vírus. Para se proteger contra vírus deve se interceptar as mensagens e analisa-las antes de integrar elas a caixa de correio do usuário. Para fazer isso deve se usar um *E-mails Proxies* ou configurar essa opção no próprio servidor de *e-mail* caso ele tenha suporte a esse recurso (SOUSA, 2006).

Os sistemas de correio eletrônico tradicionais não proveem nenhuma forma de confidencialidade para o envio de mensagens. Proteger o conteúdo de mensagens com criptografia é outro recurso interessante caso se necessite de privacidade. O SMTP, POP3 e IMAP não possuem mecanismos de confidencialidade (TCP/IP também). Todas as mensagens enviadas circulam pela rede sem nenhuma proteção e podem ser facilmente interceptadas. Logo para prover recursos de criptografia a um sistema de correio eletrônico é necessário o uso de tecnologias específicas para esse fim. Nesse trabalho não se pretende fazer o estudo de nenhuma tecnologia específica para resolver os problemas de segurança estudados. Entretanto uma solução bastante empregada para se criptografar mensagens de e-mail é o PGP (*Pretty Good Privacy*). A criptografia além de impedir que terceiros tenham acesso a mensagens também permite a autenticidade das mensagens que é outro problema dos sistemas de E-MAIL (SOUSA, 2006).

Outra vulnerabilidade do correio eletrônico é a autenticidade. Para se enviar mensagens não é necessário senha. Nos servidores e clientes de E-MAIL atuais essa vulnerabilidade não existe mais, entretanto em sistemas antigos mensagens não criptografadas ainda podem ter a identidade do remetente falsificada. Esse fato mostra o quanto é importante que os usuários atualizem seus sistemas e os deixem adequadamente configurados.

2.4 Terminal remoto

O serviço de terminal remoto ou login remoto permite que um usuário tenha acesso interativo a um computador remoto. O SSH (Secure Shell) definido na RFC 4251 é um protocolo para login remoto e seguro, ele prove também outros serviços que aumentam a segurança de uma rede como criptografia e tunelamento [RFC 4251].

Embora existam pessoas que denominem implementações como o OpenSSH, Tectia e o SunSSH de SSH. Segundo Morimoto (2006, 313p.):

> *"O SSH é, na verdade, um protocolo aberto e não o nome de uma solução específica."*

O SSH é um protocolo para comunicação segura em rede. Criado em 1995 pelo finlandês Tatu Ylönen. Ele foi projetado para ser simples e de fácil implementação. Sua principal função é permite que máquinas remotas sejam administradas, executando comandos em modo texto ou usando aplicativos gráficos. Ele também permite transferir arquivos, compactar dados e encapsula outros protocolos aumentando consideravelmente à segurança deles. As chaves de criptografia são trocadas varias vezes durante a conexão o que torno a descriptografia dos dados por terceiros quase impossível. A grande vantagem do SSH sobre outras ferramentas de acesso remoto é a grande ênfase na segurança.

O SSH funciona utilizando um servidor e um cliente Figura 09. Um servidor que fica residente na máquina que será acessada e um utilitário que é usado para acessá-la de outro computador. Para um usuário poder usufruir dos serviços providos pelo SSH o cliente e o servidor devem trocar entre si suas respectivas chaves públicas o que permite o envio de informação por um canal seguro entre eles. Por esse canal seguro é feita a autenticação. Toda criptografia, do envio inicial das chaves públicas até a autenticação é feito utilizando chaves de tamanho grandes e um algoritmo com um grau maior de complexidade. O que consome muito processamento e é inviável para o envio de todo o fluxo de informações, pois isso deixaria o SSH muito lento. Portanto logo após a autenticação o SSH utiliza uma chave menor e um algoritmo de criptografia que consome menos processamento (Morimoto, 2006).

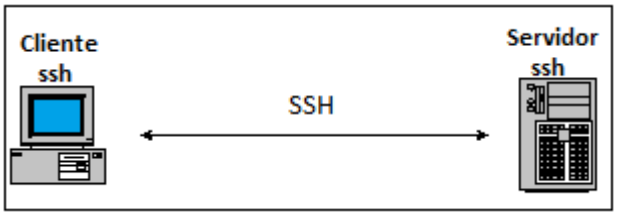

Figura 09 – Terminal remoto com SSH (Autoria própria, 2012)

As duas versões mais difundidas do protocolo SSH são o SSH1 e o SSH2. O SSH1 possui várias falhas de segurança e seu uso tem sido evitado em novos aplicativos. O SSH2 surgiu justamente para corrigir essas falhas além de dar suporte a mais opções de algoritmos de criptografia (o usuário pode escolher os algoritmos que serão usados na troca de informações) ele é mais simples e rápido. O SSH é organizado por meio de três protocolos rodando sobre TCP/IP que são:

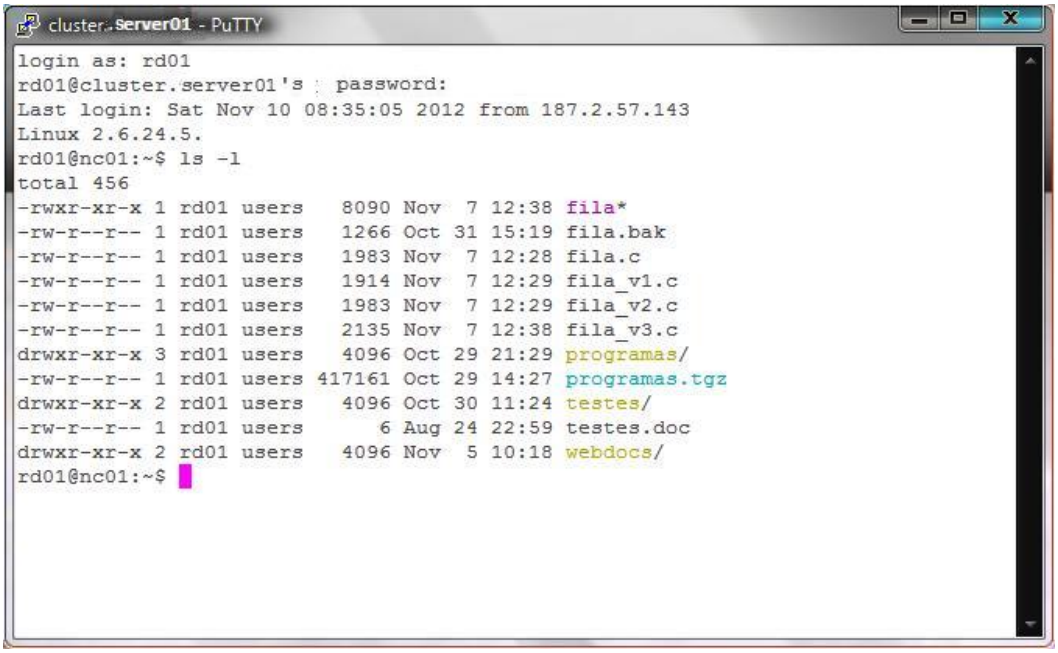

Figura 10- Cliente SSH (Autoria própria, 2012)

- Protocolo da camada de transporte: provê autenticação no servidor, confidencialidade e integridade dos dados. Opcionalmente, esta camada pode prover compressão dos dados;
- Protocolo de autenticação de usuário: autentica o usuário ao servidor;
- Protocolo de conexão: possibilita a utilização de vários canais lógicos de comunicação sobre uma única conexão SSH.

2.4.1 Segurança no SSH

Um servidor SSH corretamente configurado é virtualmente impenetrável, mesmo que a rede esteja comprometida. Com um conjunto de técnicas de criptografia, esse protocolo é capaz de assegurar que apenas pessoas autorizadas tenham acesso aos recursos do servidor. Ele garante também que todos os dados enviados sejam impossíveis de serem decifrados por terceiros que eventualmente possam interceptar as mensagens.

Devido a grande ênfase em segurança que o SSH recebeu em seu projeto. Vulnerabilidades de segurança em sua maioria são causadas por erros de configuração, inexperiência do usuário ao usar os recursos da aplicação de terminal remoto ou ainda por falhas no projeto e implementação de aplicações que utilizem o protocolo SSH para prover serviços de terminal remoto.

A primeira versão do SSH o SSH1 possui vários problemas de segurança e está, inclusive, sujeito a ataques de inserção de dados e interceptação das informação trocadas entre o cliente e o servidor por terceiros. Por isso em redes que necessitam de segurança mais rigorosa devem-se usar versões mais recentes desse protocolo. A maioria das implementações que usam o SSH para prover serviços de terminal remoto suportam tanto a versão um como a dois desse protocolo. Configurar de forma correta a versão do protocolo nesse caso é quesito chave para evitar que um atacante tenha acesso ao serviço. O ideal é configurar o servidor para aceitar apenas canecões com o SSH2 ou outra versão mais recente (MORIMOTO, 2006).

O SSH ao ser instalado, por padrão, permite que qualquer conta de usuário cadastrada no sistema possa acessar o terminal remoto usando seu login e senha. Entretanto isso não é adequado. Apenas usuários que necessitem de usar o serviço de terminal remoto devem ter acesso a ele. Essa precaução diminui a possibilidade de erros e descuidos de usuários inexperientes. As implementações do SSH costumam ter recursos que possibilitam a criação de listas com os usuários que possuem permissão de login no terminal além de ter a opção de impedir o login direto do usuário *root* (MORIMOTO, 2006).

O login direto do usuário root no terminal remoto é um risco de segurança desnecessário. Caso um atacante consiga acessar a rede como root ele terá acesso a muito mais recursos o consequentemente poderá causar mais danos. Para dificultar possíveis invasões o ideal é que apena usuários sem privilégios possam fazer login no sistema inicialmente. E apenas após esse login possa se transformar em usuário root. Essa estratégia também é útil para identificar o usuário que está usando o serviço, caso aja mais de uma pessoa que use o usuário root em suas tarefas (MORIMOTO, 2006).

Por padrão o SSH atende conexões na porta 22. Configurar o servidor para ouvir uma porta maior como 32569, por exemplo, pode dificultar o trabalho de identificação da aplicação por *hackers* e *cracker* que eventualmente possam estar fazendo “varreduras” na rede a procura de servidores vulneráveis.

2.5 Transferência de arquivos

O FTP (*File Transfer Protocol*) teve uma longa evolução até chegar a seu formato atual. A primeira proposto de um mecanismo de transferência de arquivos (RFC 114) ocorreu em 1971 e foi implementado para maquinas no MIT. O RFC 114 acrescentou comentários à discussão em relação ao protocolo de transferência de arquivos. Já o RFC 172 forneceu um protocolo orientado para a transferência de arquivos entre computadores. Uma revisão adicional do FTP foi feita posteriormente no RFC 265. Mais alterações foram propostas no RFC 281. Em julho de 1973, mudanças consideráveis das últimas versões de FTP foram feitas, mas a estrutura geral permaneceu a mesma. Muitas outras mudanças foram feitas em outros RFCs, porém demasiadamente numerosas para serem citadas aqui. Porém desde então muitas mudança foram feitas no protocolo, seja para permitir sua transição do NCP para o TCP, para adicionar comandos adicionais ou corrigir erros na documentação (RFC 959).

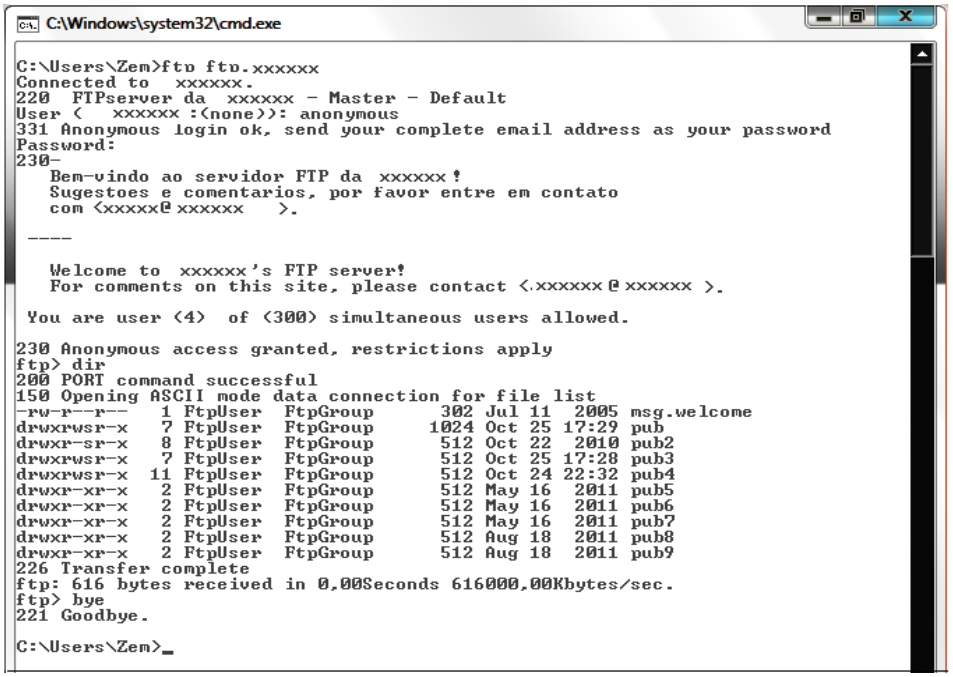

Figura 11- Cliente FTP (Autoria própria, 2012)

Maia (2009, 213p.), afirma que:

> *"A transferência de arquivos é um dos serviços mais básicos em uma rede de computadores. O serviço de transferência de arquivos permite ao usuário copiar arquivos utilizando a rede, além de visualizar o conteúdo de arquivos e diretórios, definir o formato dos arquivos (texto e executável) transferidos e criar e eliminar arquivos e diretórios."*

O protocolo FTP, definido na RFC-959, utiliza o protocolo TCP na camada de transporte. O FTP utiliza o modelo cliente-servidor, e são necessários um cliente FTP e um servidor FTP. Em uma típica sessão FTP o usuário, na frente de um hospedeiro local, fornece o nome de um hospedeiro remoto. Uma vez com esse nome o processo cliente FTP do hospedeiro local estabelece uma conexão TCP com o processo servidor FTP do hospedeiro remoto. Em seguida o usuário precisa se autenticar no servidor FTP através da conexão TCP para poder transferir arquivos do computador local para outro e vice-versa. O protocolo FTP assim como o HTTP é um protocolo de transferência de arquivos e possui muitas semelhanças com ele. Entretanto há algumas diferenças, como o fato de o servidor FTP utilizar duas portas reservadas: a porta 20 para a transferência de dados (conexão de dados), e a porta 21 para a troca de comandos (conexão de controle) (Figura 11) (KUROSE; ROSS, 2010).

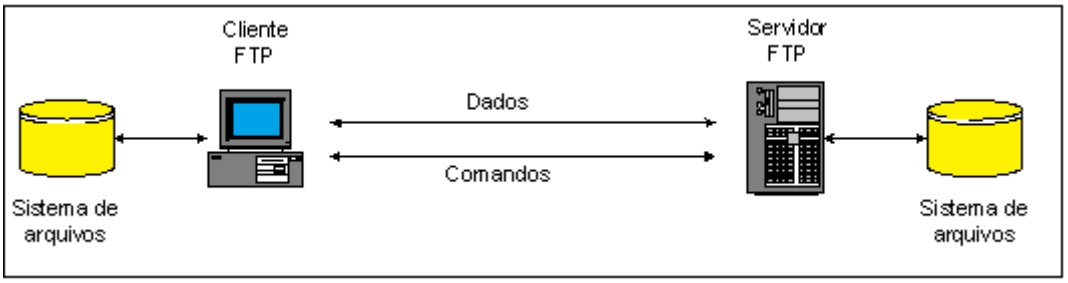

Figura 12 – Transferência de arquivos FTP (MAIA, 2009, 214p)

O acesso aos serviços de transferência de arquivos é feito a partir do utilitário FTP, que, geralmente, acompanha a maioria dos sistemas operacionais. Este utilitário é orientado a caractere na maioria dos sistemas operacionais, porém existe utilitários FTP com interface gráfica. A Tabela 02 descreve os principais comandos disponíveis nos utilitários FTP.

Tabela 02 – Principais comandos do FTP (MAIA, 2009, 214p)

Comando	Descrição
Open	Abre uma conexão com o servidor FTP.
Close	Encerra uma conexão.
Get	Transfere um arquivo do host remoto para o host local.
Put	Transfere do host local para o host remoto.
dir ou ls	Exibe o conteúdo do diretório remoto.
Ascii	Especifica que a transferência é de um arquivo texto.
Binary	Especifica que a transferência é de um arquivo texto binário.

O protocolo FTP não oferece nenhuma confidencialidade no envio das informações, inclusive a senha é enviada em claro. A versão segura do protocolo FTP é conhecida como SFTP (Secure FTP). O SFTP utiliza o protocolo SSH (*Secure Shell*) para garantir o sigilo nas transferências de arquivos e responde na porta TCP 22 (MAIA, 2009).

2.5.1 Segurança no FTP

O principal problema de segurança do FTP é o trafego das informações de autenticação e dados, que são feito sem nenhuma proteção. Isso permite que praticamente qualquer usuário da rede possa interceptar as mensagens. Obtendo acesso total ao sistema (MORIMOTO, 2006).

Outro recurso que pode causar problemas é o FTP anônimo. Através dele qualquer pessoa pode, sem precisar ter conta na máquina, pode depositar e copiar arquivo dela. Esse recurso é muito utilizado para distribuir os mais diversos conteúdos pela rede. Porém deve se tomar o cuidado de, na configuração do FTP anônimo de se restringir o acesso apenas aos documentos que não tenham restrições de acesso. Permitir o deposito de arquivos pode ser muito perigoso, pois arquivos indesejados (como vírus, arquivos com conteúdo ilegais e etc.) podem ser depositados. Recomenda-se que um diretório separado e com espaço limitado sege criado para esse fim. A existência de diversos programas destinados a realizar ataques a servidores FTP é outro fator que torna crítica à segurança neste sistema.

3 DETECÇÃO DE VULNERABILIDADES EM REDES DE COMPUTADORES

Uma rede de computadores é formada por vários equipamentos e *softwares* que possuem funções especificas. A administração da segurança nesses sistemas é uma tarefa complexa e requer profissionais experientes. Entretanto mesmos profissionais com grande experiência podem cometer erros ou desconhecer algum fator que torne a rede insegura. Uma das fases do processo de auditoria ou mesmo invasão de um sistema é a procura por vulnerabilidades. Esse processo apesar de poder ser feito manualmente é na maioria das vezes automatizado por um *scanner* de vulnerabilidades.

Existem muitos s*oftwares* que auxiliam na detecção de vulnerabilidades em redes de computadores. Muitos deles são específicos para uma determinada vulnerabilidade e outros são capazes de analisar toda uma rede e um enorme número de vulnerabilidades. Ao mesmo tempo em que estes programas de computador auxiliam os profissionais de tecnologia da informação a aumentar a segurança em suas redes, eles facilitam o trabalho de quem pretende encontrar vulnerabilidades para invadi-las. E o que é pior, esses programas em geral são de fácil manuseio permitindo que pessoas sem conhecimentos relevantes em segurança da informação possam efetuar invasões em sistemas computacionais e causar grandes estragos. Esse fato reafirma a importância dos investimentos em segurança da informação.

Este capítulo destina-se a apresentação dos *softwares* envolvidos nos testes e o ambiente onde eles irão ser realizados. Os testes consistiram em varreduras de rede com o propósito de detectar vulnerabilidades. Essas varreduras serão feitas com auxilio de um scanner que segundo Ulbrich e Valle (2006, 141p) pode ser definido como:

> *"Scanners são programas usados para varrer os computadores em uma rede à procura de vulnerabilidades. Há portas abertas a ataques tanto em máquinas pessoais como em servidores de todos os tamanhos. Os scanners buscam sistemas que estejam desprotegidos e prontos para receber uma análise minuciosa sobre sua segurança."*

Existem centenas de scanners espalhados pela Internet. O primeiro quesito a ser levado em consideração ao escolher qual usar, deve ser o tipo de sistema a ser explorado no caso Unix/Linux ou Windows. Em geral o usuário pode fornecer um grupo de IPs a ser analisado, evitando que o scanner analise toda a rede. Entretanto fazer varreduras em redes internas de instituições não é tão simples. Segundo Ulbrich e Valle (2006, 144p) os endereços IPs dessas redes não podem ser analisados diretamente da Internet. Pois seria necessário passar pelo *Firewall*[9], pelo *Gateway*[10] e conseguir instalar um programa em uma máquina que possua interface de rede interna e externa (na Internet) para através delas atacar a rede interna.

Scanners de vulnerabilidades são scanners que basicamente, sabendo dos serviços que estão rodando na máquina, testa esses serviços checando se possuem problemas. Para fazer esses testes ele possui um grande repertório de vulnerabilidades conhecidas. Desde problema de configurações até vulnerabilidades nos próprios *softwares* são analisados pelos scanners de vulnerabilidades.

3.1 Produtos usados nos testes

Nos testes serão usados o Apache 2.2.17, o Postfix 2.4.3-1, o OpenSSH 5.1 e o Proftpd 1.3.1. A escolha destes programas justifica-se por serem de uso difundido. De fácil obtenção e por serem *softwares* livres. Entretanto essas escolhas também foram influenciadas pelo sistema operacional que irá abrigar esses programas e, portanto será alvo dos testes.

Todos esses programas serão instalados em um sistema operacional Linux (distribuição Slax). A instalação de *softwares* nesse sistema é feita através de módulos de programas que são disponibilizados no mesmo site onde se pode obter essa distribuição Linux (http://www.slax.org/), bastando ao usuário apenas escolher quais programas ele quer que venham instalados. Porém muitos desses módulos não são checados ou atualizados. Logo esses *softwares* podem estar

[9] *Firewall* são dispositivos utilizados para prover segurança em redes de computadores. Estes aparelhos analisam todos os dados que transitam por um ponto da rede aplicando uma politica de segurança e dessa forma impedindo a livre circulação de dados perigosos para a instituição (TANENBAUM, 2003).

[10] *Gateway* são dispositivos de rede destinados a ligar duas redes de computadores diferentes, que podem possuir protocolos de comunicação diferentes ou mesmo separar domínios (TANENBAUM, 2003).

desatualizados e conter alguma vulnerabilidade não corrigida. Isso aumenta a chance de alguma vulnerabilidade ser encontrada. O que é ideal para a proposta desse trabalho uma vez que possibilitará a analise, com um *scanner* de vulnerabilidades, de *softwares* que possivelmente tenham vulnerabilidades.

3.1.1 Apache

O Apache é o mais bem sucedido servidor WEB livre. Estando presente em uma parcela grande de servidores dedicados a disponibilizar aplicações na WEB. Segundo Marcelo (2005, p3):

> *"O Apache é, sem sombra de dúvida, um dos mais robustos e seguros programas desenvolvidos para ambientes TCP/IP e que mantem em operação mais de 60% homepages/sites disponíveis no mundo."*

O Apache pode ser dividido em duas grandes famílias representadas pelas versões 2.x e 1.3. Apesar de a versão 2.x do Apache possuir muito mais recursos, a versão 1.3 ainda é muito usada pelo fato de muitos módulos da nova versão ser incompatíveis com a versão antiga. O que impede que administradores migrem suas plataformas. Esse servidor WEB possui um módulo chamado mod_ssl que garante a segurança nas transações HTTP, permitindo que ele atenda requisições por meio do protocolo HTTPS. O protocolo HTTPS utiliza uma camada SSL que possibilita a transmissão de dados criptografados entre o cliente e servidor (MORIMOTO, 2004).

3.1.2 Postfix

Segundo Ricci (2004), o Postfix é um agente de correio eletrônico criado por Wiestse Venema como uma alternativa para o sendmail. Esse agente de correio eletrônico é mais rápido que em comparação a outros. Mais é a sua segurança aprimorada que faz dele uma ótima opção para fornecer o serviço de correio eletrônico.

Além de ser de fácil configuração e já vir com configuração padrão suficientemente aceitável do ponto de vista da segurança. O Postfix possui fácil integração com anti-spam e anti-vírus. Possui documentação abundante e um ótimo histórico de segurança. O que faz dele um servidor de E-MAIL muito popular (RICCI, 2004).

3.1.3 OpenSSH

O OpenSSH é baseado na ultima versão da implementação de Tatu Ylonen (criador do protocolo SSH) excluído todo o conteúdo patenteado. Dentre suas várias características destaca-se a conexão cliente/servidor criptografada, transferência de arquivos com conexão criptografada e suporte a compactação de dados. O OpenSSH foi criado por Aaron Campbell, Bob Beck, Markus Friedl, Niels Provos, Theo de Raadt e Dug Song (RICCI, 2004).

A segurança no OpenSSH é possibilitada com o uso de algoritmos de criptografia comprovadamente eficientes e livres de qualquer pagamento de direitos autorais. Esses algoritmos são usados pelo *software* de acordo com a necessidade de velocidade ou de maior segurança. O recurso de compressão de dados também está presente neste programa, o que melhora muito a velocidade de transferência de informações entre cliente e servidor.

3.1.4 Proftpd

O ProFTPD é um dos servidores de FTP mais usados em sistemas Linux. De fácil configuração e gerenciamento ele possui vários recursos como arquivo único de configuração, fácil configuração de servidores virtuais e anônimos, o código fonte está sempre disponível para quem possa interessar e permissões de acesso a diretórios baseadas em esquemas dos sistemas Unix. Além de estar disponível para diversas plataformas.

O desenvolvimento do ProFTPD iniciou-se a partir da necessidade de um servidor FTP com mais recursos e com segurança melhorada. Esse programa foi desenvolvimento sem ser baseado em qualquer outro *software*.

3.1.5 Nessus

Nessus é um *software* destinado a fazer auditorias de segurança em redes de computadores. Ele é uma excelente ferramenta de detecção de vulnerabilidades. Com mais de 10.000 *plugins* e atualizações diárias ele é capaz de encontrar um grande número de vulnerabilidades conhecidas. Constituído por duas partes, um cliente e um servidor que faz o *scan* propriamente dito. Ele basicamente faz uma varredura de portas detectando as que estão ativas e em seguida simula invasões

para detectar vulnerabilidades. O Nessus é atualmente considerado por muitos como o melhor *software* da sua categoria.

Além de ser usado para encontrar vulnerabilidades em redes de computadores, o Nessus pode ser usado para outras tarefas como o gerenciamento de atualizações de *softwares* ou até mesmo auxiliar na administração da rede. Eficiente e veloz esse *scanner* de vulnerabilidades pode fazer até mesmo auditoria em dispositivos móveis. E os relatórios gerados podem ser personalizados facilitando a vida do auditor.

3.2 O ambiente de teste

O ambiente de testes será composto por duas máquinas virtuais formando uma rede. Na montagem de uma das máquinas virtuais será usada a distribuição Linux destinada a auditorias em segurança da informação, Brack Track versão 5. Esse sistema operacional já vem com o Nessus pré-instalado além de uma infinidade de outras ferramentas de auditoria. A outra máquina virtual será composta, como já mencionado, pela distribuição Linux Slax. Ele é uma pequena distribuição Linux baseado no Slackware que pode ser inicializada diretamente de um CD ou *pen-drive* e, portanto não necessita de ser instalada. Na Figura 13 temos uma ilustração exemplificando o ambiente de teste.

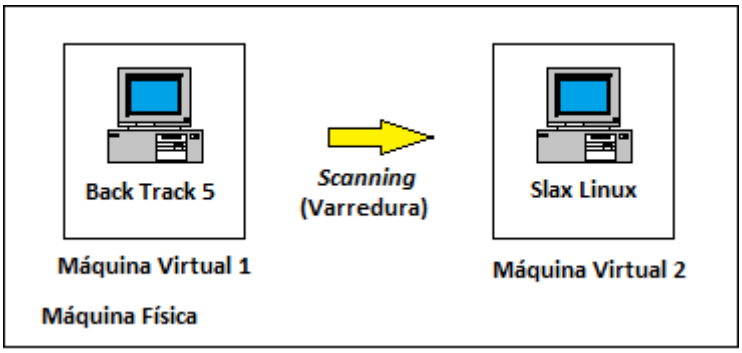

Figura 13 - Ambiente de testes (Autoria própria, 2012)

Com a intenção de evitarem-se longas considerações a cerca da configuração desses servidores. Adotaram-se em todos eles as configurações padrão que os acompanha. Isso será adequado aos testes, pois na falta de alguma modificação importante o Nessus poderá acusar o erro.

Como já discutido anteriormente a varredura em uma rede interna só pode ser feita por meio de programas que estejam nessa rede. Para se conseguir analisar uma máquina em uma rede interna a partir da Internet é necessário que algum *script* ou programa localizado na rede interna, abra uma conexão com uma máquina externa à rede. Dessa forma essa conexão pode ser explorada para realizar varreduras na rede. Entretanto isso além de complexo está fora dos objetivos desse trabalho. Portanto a varredura de uma máquina por outra localizada na mesma rede corresponde na maioria dos casos com a realidade desse tipo de testes.

O Nessus faz varreduras em uma rede baseando-se em politicas de escaneamento. Essas políticas podem ser criadas pelo usuário. Que para isso escolhe, dentre outros parâmetros de configuração, as portas que serão analisadas e os *plugins* que testarão o alvo. Algumas políticas já veem configuradas. Essas políticas, segundo o manual do Nessus, procuram por todas as vulnerabilidades conhecidas pelo programa. Logo já são adequadas para auditorias. Entretanto caso a rede for composta por muitas máquinas o processo de escaneamento tornara-se muito lento o que justificaria a criação de uma politica mais focada em um determinado elemento da rede. Como o ambiente de teste usado neste trabalho possui tamanho limitado se justiça o uso das politicas de escaneamento fornecidas juntamente com o Nessus.

Os testes serão realizados com todos os servidores executando ao mesmo tempo. Isso não diminuirá a eficiência do Nessus e muito menos afetará os resultados uma vez que esse *software*, em um primeiro momento executa *plugins* destinado a reconhecer as portas ativas. E em seguida executa vários outros *plugins* específicos para cada vulnerabilidade, analisando cada porta ativa. Logo todos os testes são feitos independentemente, em cada porta com atividade detectada.

4 RESULTADOS E DISCUSSÃO DOS TESTES

Neste capítulo serão feitas a apresentação e discussão dos testes realizados de acordo com o que foi apresentado acima. Apesar de ser uma excelente ferramenta o Nessus está sujeito como qualquer outro *software* de análise e auditoria em redes de computadores a falsos positivos. Por isso em uma auditoria o ideal é que o auditor cheque as vulnerabilidades encontradas.

O objetivo deste trabalho é o estudo de vulnerabilidades em serviços de rede. Entretanto os testes práticos se concentraram nos servidores que fornecem esses serviços. Por serem esses equipamentos de responsabilidade dos profissionais de tecnologia da informação. E principalmente por ser neles onde a segurança é crítica.

O Nessus classifica as vulnerabilidades encontradas pelo comprometimento que elas podem causar ao sistema. Sendo que vulnerabilidades de alto grau de comprometimento (*high*) são capazes de permitir que prejuízos relevantes sejam causados por terceiros ao sistema. Vulnerabilidades de médio grau de comprometimento (*medium*) não chegam a ser alarmantes, porém merecem atenção por também esporem o sistema a ataques. Por outro lado, vulnerabilidades classificadas como de baixo grau de comprometimento (*low*), podem não oferecer nenhum risco ao sistema, ser um aviso de detecção de serviços ou mesmo um problema com pouca relevância.

4.1 Vulnerabilidades no Apache

O Apache 2.2.17 foi o único *software* testado a apresentar vulnerabilidades classificadas, pelo Nessus, como de alto e médio grau de comprometimento. Também foram encontradas vulnerabilidades de baixo grau de comprometimento. Entretanto, por ter o Nessus acusado muitas vulnerabilidades de baixo grau de comprometimento, apenas as mais relevantes dessa classificação serão discutidas. Por razões práticas as vulnerabilidades encontradas nessa versão do apache serão divididas em três tabelas. Uma tabela para cada nível de comprometimento. Logo abaixo são apresentadas as vulnerabilidades encontradas.

Tabela 03: Vulnerabilidade de alto comprometimento (Autoria própria, 2012)

Vulnerabilidade	Descrição
O servidor Web em execução no host remoto é afetado por uma vulnerabilidade de negação de serviço.	Fazendo-se uma série de solicitações HTTP com sobreposição de faixas no intervalo ou cabeçalhos de solicitação pode-se obter a exaustão da memória e CPU. Um atacante remoto não autenticado pode explorar isso e tirar o servidor de operação

A única vulnerabilidade de alto grau de comprometimento do Servidor Web Apache é descrita pelo Nessus como uma vulnerabilidade no próprio sistema. Essa vulnerabilidade permite ataques de negação de serviço que são realizados explorando-se a forma como os múltiplos intervalos sobrepostos são manipulados pelo servidor HTTP. Segundo o relatório do Nessus a correção dessa vulnerabilidade consiste em atualizar o Apache para uma versão onde essa vulnerabilidade já esteja corrigida. Essa vulnerabilidade permite que um atacante, usando um simples computador, consiga tirar o servidor do ar. Logo ela afeta claramente a segurança do sistema.

Tabela 04: Vulnerabilidade de médio comprometimento (Autoria própria, 2012)

Vulnerabilidade	Descrição
O servidor Web em execução no host tem uma vulnerabilidade de divulgação de informações.	O envio de um pedido com os cabeçalhos HTTP tem tempo suficiente para ultrapassar o limite do servidor fazendo com que o servidor web responda com um HTTP 400. Por padrão, o cabeçalho HTTP e valores são exibidos na página de erro 400.
O servidor web hospeda um script PHP que está propenso a um ataque.	A versão do phpMyAdmin não valida tags de entrada do usuário antes de usá-las para gerar HTML dinâmico. Um invasor pode ser capaz de injetar HTML arbitrária ou código de script no navegador para ser executado dentro do contexto de segurança do local afetado.
A configuração do PHP no host remoto permite a divulgação de informações confidenciais.	A instalação do PHP no servidor remoto esta configurada de uma maneira que permite a divulgação de informações potencialmente sensíveis para um atacante através de uma URL especial.
Funções de depuração estão ativas no servidor WEB.	O servidor web remoto suporta métodos TRACE e / ou métodos TRACK. TRACE e TRACK são métodos HTTP que são utilizados para depurar as conexões do servidor web.

Outra vulnerabilidade descoberta no Apache 2.2.17 é a divulgação de informações através de mensagens de erros. Segundo o relatório do Nessus um pedido com cabeçalhos HTTP podem ultrapassar o limite de tempo do servidor fazendo com que ele gere uma mensagem de erro onde outros valores podem ser exibidos. A correção para esse problema é a atualização do Apache para uma versão onde à vulnerabilidade esteja corrigida.

Na segunda linha da Tabela 04 é apresentada uma vulnerabilidade causada pela presença, no sistema do Servidor, de um *script* (phpMyAdmin) com problemas de segurança. Esse *script* pode permitir a alguém injetar código HTML arbitrário ou código de *script* no navegador. A solução para esse problema, segundo o Nessus é a atualização do phpMyAdmin para a versão 3.4.0-beta1 ou outra versão disponibilizada ao público posteriormente a essa.

As duas últimas vulnerabilidades de médio grau de comprometimento do sistema, linhas três e quatro da Tabela 04, correspondem a problemas de configurações. O arquivo php.ini está incorretamente configurado e permite a divulgação de informações confidenciais. E métodos de depuração do Servidor estão ativados e disponíveis a qualquer usuário. A solução apresentada, evidentemente é que se alterem as configurações desse arquivo e desabilitem-se os métodos de depuração do Apache.

Tabela 05: Vulnerabilidade de baixo comprometimento (Autoria própria, 2012)

Vulnerabilidade	Descrição
Alguns cookies foram criados pelo servidor web.	*Cookies* HTTP são informações que são apresentados por servidores web e são enviados de volta pelo navegador. Como o HTTP é um protocolo sem estado, os cookies são um mecanismo possível para acompanhar as sessões. Este *plugin* exibe a lista dos cookies HTTP que foram definidas pelo servidor web quando foi rastreado.
É possível enumerar os diretórios no servidor web.	Este *plugin* tenta determinar a presença de vários diretórios comuns no servidor web. Ao enviar um pedido de um diretório, o código de resposta do servidor web indica se é um diretório válido ou não.

As duas vulnerabilidades de baixo grau de comprometimento apresentadas na tabela acima correspondem a problemas com *plugins*. O primeiro exibe a lista de

cookies caso esses sejam rastreados. E o segundo permite que se cheque a presença de diretórios no servidor. A solução para ambas as vulnerabilidade é a atualização desses *plugins* para versões corrigidas.

4.2 Vulnerabilidades no Postfix

O Nessus não encontrou nenhuma vulnerabilidade de segurança no Postfix. Na tabela abaixo é apresentado o único resultado retornado pelo *scanner* de vulnerabilidades.

Tabela 06: Resultados dos testes no Postfix 2.4.3-1(Autoria própria, 2012)

Vulnerabilidade	Descrição	Grau de comprometimento
Um servidor SMTP está escutando na porta 25.	O host está executando um servidor de e-mail (SMTP) nesta porta. Este servidor SMTP pode ser alvo de ataques de *SPAM*. É recomendado desativá-lo caso não esteja sendo usando-o.	Baixo

Neste caso apenas foi emitido um aviso de que há um servidor de SMTP rodando na máquina alvo. O Nessus também aconselha a desativação do serviço caso ele não esteja sendo usado. Além de alertar a possiblidade de ele ser alvo de ataques de SPAM.

4.3 Vulnerabilidades no Openssh

Os testes realizados pelo Nessus no OpenSSH não acusaram nenhuma vulnerabilidades de segurança. Na tabela abaixo é apresentado o único resultado com informações relevantes, apresentado pelo *software*.

Tabela 07: Resultados dos testes no OpenSSH 5.1 (Autoria própria, 2012)

Vulnerabilidade	Descrição	Grau de comprometimento
Correções/atualizações de segurança estão feitas.	Atualizações de segurança podem ter sido ‘feitas’ no servidor remoto SSH sem alterar seu número de versão. Alertas de vulnerabilidades foram desativados para evitar falsos positivos. Note-se que este relatórios é apenas informativo .	Baixo

Como pode-se ver na descrição acima o Nessus foi capaz de detectar atualizações no OpenSSH que foram feitas sem que o número de versão do *software* fosse alterado. Isso mostra claramente que esse *scanner* de vulnerabilidades analisado profundamente o sistema alvo, não se “contentando” apenas com informações superficiais para dar seu diagnostico.

Ele também deixou claro que o aviso retornado é apenas informativo, não representando uma vulnerabilidade. Esse resultado reafirma a grande ênfase em segurança do OpenSSH. Porém, a partir dele, se percebe que a falta de atualizações pode comprometer o serviço fornecido. Uma vez que, como ocorrem com quaisquer outros *softwares*, vulnerabilidades podem ser descobertas. E cabe ao administrador estar atento a elas através da comunidade ou empresa que mantem esses programas de computador.

4.4 Vulnerabilidades no Proftpd

Nos testes realizados no Proftpd não foram encontrada nenhuma vulnerabilidade pelo Nessus. Apenas algumas características do serviço FTP foram mencionadas. Como pode ser visto na tabela a seguir.

Tabela 08: Resultados dos testes no Proftpd 1.3.1(Autoria própria, 2012)

Vulnerabilidade	Descrição	Grau de comprometimento
As credenciais de autenticação podem ser interceptadas.	O servidor de FTP remoto transmite o nome do usuário e senha em texto puro, o que permite a interceptação destas informações.	Baixo

O Nessus, neste caso, apenas alertou para as limitações de segurança do *software*: todas as informações transmitidas pelo cliente ao servidor e vise versa, não são protegidas por nenhum esquema de segurança e, portanto podem ser facilmente capturadas por terceiros. Essas limitações de segurança (que já foram discutidas em capítulos anteriores) não são tratadas pelo scanner como vulnerabilidades de segurança em si, pois são de conhecimento dos administradores de rede. Entretanto caso se deseje eliminar essas limitações, o Nessus aconselha o

uso da versão segura do protocolo FTP, o SFTP. Ou qualquer outro esquema que possa aumentar a segurança desse protocolo.

5 CONCLUSÃO

A grande expansão e evolução da rede mundial de computadores, que ocorre continuamente nos dias atuais é uma das responsáveis pela criação de novas aplicações e também por tornar a sociedade cada vez mais dependente dos recursos providos por ela. A segurança na transmissão, armazenamento e recepção de dados nos diversos serviços providos pela Internet é fator chave para o sucesso de muitas instituições que tem seus negócios dependentes dela.

Muito se investe para se criar mecanismo que sejam capazes de assegurar a segurança nas redes de corporativas, entretanto a capacitação dos profissionais responsáveis pela manutenção de tais redes é de extrema importância. Apesar do grande número de equipamentos e programas de computadores que compõem as redes de computadores é possível conseguir um bom nível de segurança utilizando-se tecnologias disponíveis no mercado.

Vulnerabilidades em sistemas computacionais surgem frequentemente, porém as empresas ou grupos responsáveis pelo desenvolvimento desses sistemas investem muitos recursos (tempo, dinheiro e pessoal) na solução dos problemas que surgem. Em geral a solução de uma vulnerabilidade recém-descoberta é publicada em questões de horas ou dias, logo cabe ao profissional de Segurança da Informação estar sempre atento. Para auxiliar no processo monitoramento de vulnerabilidades em redes computadores existem diversos *software* que simplificam a tarefa e permitem rapidez e eficiência.

Referência bibliográfica

BARCELOS, T. M. Segurança em aplicações WEB. 2010. Monografia – Faculdade de Ciências Empresariais, Universidade FUMEC, Belo Horizonte. 2010.

FIELDING, R.;GETTYS, J,; MOGUL, H,; FRYSTYK, L.; MASINTER, P.; LEACH, T.; BERNERS-LEE, T “RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1”, junho 1999

FIGUEREIDO. A; Centro de Computação da Unicamp. Revista Infotec. Administração de Sistemas e Segurança. Disponível em: http://www.ccuec.unicamp.br/revista/infotec/admsis/admsis6-1.html Acesso em: 20 de out. 2012.

KUROSE, J. F.; KEITH, W. R. . Redes de computadores e a Internet: uma abordagem top-down. - 3. ed. - São Paulo: Pearson Addison Wesley, 2006.

MAIA, L. P. Arquitetura de Redes de Computadores. 1. ed. Rio de Janeiro: LTC, 2009.

MARCELO, A. Configurando o Servidor WEB para Linux. 3. ed. Rio de Janeiro: Brasport, 2005.

MORIMOTO, Redes e Servidores Linux, Guia Prático. São Paulo: GDH Press e Sul Editores, 2006.

POSTEL, J.: REYNOLDS, J.”RFC 959: File Transfer Protocol”. Outubro 1985

RICCI, B. Sladkware – Guia Prático. 1. ed. Rio de Janeiro: Editora Ciência Moderna, 2004.

ROBBINS, A.; BEEBE, N. H. F. Classic Shell Scripting. 1. ed. United States: O’Reilly, 2005

SOUSA, M. F. Segurança em Servidores de Correio Eletrônico. Faculdade de Ciências Aplicadas de Minas – União Educacional de Minas Gerais S/C LTDA, Uberlândia. 2006.

STALLING, W. Redes e Sistemas de Comunicação de Dados. 5. ed. Rio de Janeiro: Elsevier, 2005.

TANENBAUM, A. S. Redes de Computadores. 4. ed. Rio de Janeiro: Elsevier,2003.

ULBRICH, H. C; VALLE, J. D. Universidade Hacker. 5. ed. São Paulo: Digerati Books, 2006.

YLONEN, T.; LONVICK, C.”RFC 4051: The Secure Shell (SSH) Protocol Architecture”. Janeiro 2006