UNIPAC
UNIVERSIDADE PRESIDENTE ANTÔNIO CARLOS
FACULDADE DE CIÊNCIA DA COMPUTAÇÃO E
COMUNICAÇÃO SOCIAL DE
BARBACENA

CURSO DE CIÊNCIA DA COMPUTAÇÃO

Felipe Luís da Silva

UMA ABORDAGEM SOBRE TÉCNICAS DE INVASÃO DE COMPUTADORES

BARBACENA
DEZEMBRO DE 2004

FELIPE LUÍS DA SILVA

TÉCNICAS DE INVASÃO DE COMPUTADORES

Trabalho de conclusão de curso apresentado à Universidade Presidente Antônio Carlos, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

ORIENTADOR: Prof. Elio Lovisi Filho

BARBACENA
DEZEMBRO DE 2004

FELIPE LUÍS DA SILVA

TÉCNICAS DE INVASÃO DE COMPUTADORES

Este trabalho de conclusão de curso foi julgado adequado à obtenção do grau de Bacharel em Ciência da Computação e aprovado em sua forma final pelo Curso de Ciência da Computação da Universidade Presidente Antônio Carlos.

Barbacena – MG, 7 de dezembro de 2004.

_____

Prof. Elio Lovisi Filho - Orientador do Trabalho

_____

Prof. Eduardo Macedo Bhering - Membro da Banca Examinadora

_____

Prof. Reinaldo Silva Fortes - Membro da Banca Examinadora

SIGLAS

ACK - Acknowledgment

ACL - Access Control List

ARP - Address Resolution Protocol

BBS - Bulletin Board System

CA - Computer Associates

CIS - Cerberus Internet Scanner

DDoS - Distributed Denial of Service

DLL - Dynamic Link Library

DNS - Domain Name Server

DoS - Denial of Servise

EUA - Estados Unidos da América

EXE - Executável

FTP - File Transfer Protocol

HTTP - Hyper Text Tranfer Protocol

ICMP - Internet Control Message Protocol

IDS - Intrusion Detection System

IP - Internet Protocol

KDE - K Desktop Environment

MAC - Media Access Control

MIME - Multiporpuse Internet Mail Extension

NetBios - NETwork Basic Input Output System

POP3 - Post Office Protocol 3

PWL - Passwords (WASP)

RFC - Request For Comment

RPC - Remote Procedure Call

SMTP - Simple Mail Transfer Protocol

SNMP - Simple Network Management Protocol

SO - Sistemas Operacionais

SYN - SequenceNum

TCP - Trasmission Control Protocol

TELNET - Protocolo de Terminal Virtual

TFN - Tribe Flood Network

TI - Tecnologia de Informação

UDP - User Datagram Protocol

UTP - Unshielded Twisted Pair

SUMÁRIO

LISTAS

TABELAS

1 INTRODUÇÃO

A Internet é uma fonte de informação muito grande e de fácil acesso. Ela foi um enorme avanço para a humanidade, possibilitando a quem a utiliza fazer compras, trabalhos, acessar contas de bancos dentre outras infinidades de coisas. Porém, não se imaginava que poderiam ser cometidos crimes como os que estão acontecendo em grande escala hoje em dia. Pessoas estão com medo de utilizar cartões de crédito e também passar informações pessoais pela Internet, temendo um roubo dessas informações e o prejuízo que possam causar.

Mas, segundo (Hynds, 2004) policial chefe da unidade de combate ao crime cibernético na Inglaterra, pelo seu ponto de vista devemos separar o exagero da realidade. Ele diz "Uma pessoa numa rua respeitável, com lojas conhecidas, em plena luz do dia, provavelmente não terá problemas se usar cartão de crédito. Agora se estiver em uma lojinha desconhecida, numa rua lateral, no escuro, com pessoas esquisitas em volta, saberá que o risco de usar o cartão é maior. Na Internet é a mesma coisa, é preciso usar o bom senso. É um exagero retratar a rede como um lugar escuro e sinistro. Ela é como tudo na sociedade: há lugares seguros e lugares inseguros."

Existem criminosos na Internet sim, e estes utilizam técnicas para cometer crimes que vão evoluindo com o passar do tempo. Antes os criminosos conhecidos eram pessoas como Kevin Mitnik e Raphael Gray. O primeiro, foi incluído na lista de *hackers* mais procurados pelo FBI. Ele era muito ativo mas nunca ganhou dinheiro com isso. Gray obteve vinte e cinco mil números de cartões de créditos na Internet, coisa que, na mão de criminosos, renderia uma fortuna. Mas ele se contentou em pegar o número de cartão de crédito do Bill Gates para mandar uma cobrança de Viagra.

Hoje existem crimes organizados: no leste Europeu, por exemplo, há criminosos agindo na rede exatamente como agiriam no mundo real: impõe sua vontade com violência, tiram competidores do caminho e dirigem seus negócios espúrios como sempre fizeram.

1.1 A PROPOSTA

A proposta é realizar um estudo sobre as técnicas de invasão e as ferramentas utilizadas.

Existem várias formas diferentes de invasão, dediquei a pesquisá-las e passar todo o conteúdo de maior interesse para este projeto. Falaremos das técnicas mais utilizadas e também das ferramentas que são de papel fundamental para poder realizá-las.

Os objetivos pelos quais esta pesquisa foi desenvolvida foram, aprofundar no estudo da invasão dos computadores, organizar algumas técnicas dentro da bibliografia, coleta de material, pois não se encontra um material específico onde mistura as técnicas, ferramentas e conceitos necessários para entender as técnicas.

1.2 ORGANIZAÇÃO

Este trabalho foi dividido em cinco capítulos da seguinte maneira: o primeiro é a introdução, que contem o objetivo, a proposta e organização. No segundo foram abordados conceitos de redes de computadores, protocolos, *firewall*, portas, sistemas operacionais. Conceitos que serão necessários para poder entender as técnicas, caso o leitor já tenha conhecimento de redes de computadores, não será necessário ler o segundo capítulo. As técnicas são descritas no terceiro capítulo, onde são diferenciadas de acordo com o objetivo da invasão. O quarto capítulo, contém algumas ferramentas utilizadas pelos *hackers*, onde são mostrados exemplos, na grande maioria, de *softwares* para a plataforma Windows. E o último capítulo é a conclusão que será feita uma revisão, conclusões chegadas e abordadas algumas propostas para continuação deste estudo.

2 CONCEITOS BÁSICOS

Nesse capítulo, procurou-se descrever conceitos básicos para entender o funcionamento de uma rede e alguns de seus protocolos, origem do ataques, Sistemas Operacionais. Estas definições são importantes, pois serão citadas no decorrer do trabalho.

2.1 SISTEMAS OPERACIONAIS

Sistemas Operacionais (S.O.) são a "camada" de *software* entre os aplicativos e o hardware, conforme demonstrado na figura 1(Tanenbaum, 1994).

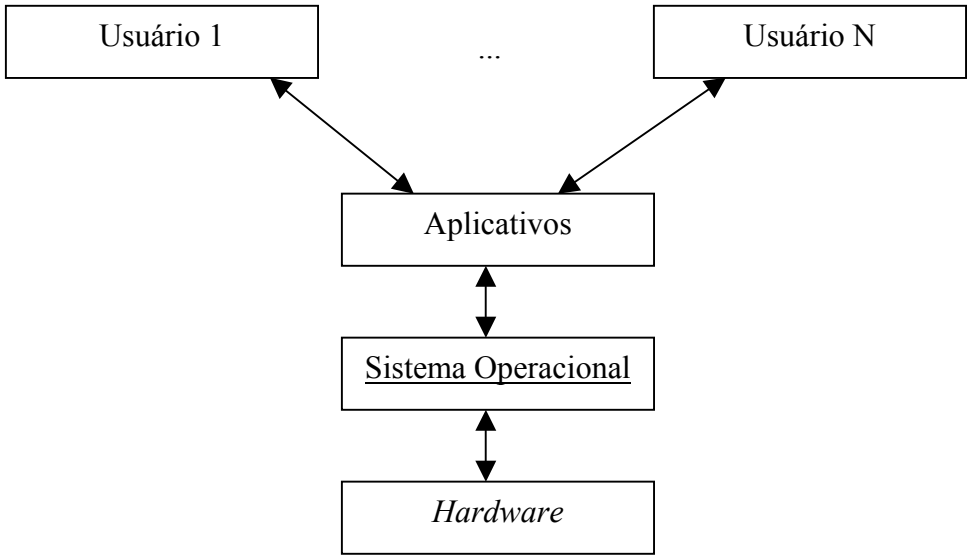

Figura 1- Esquema da localização do Sistema Operacional (Tanenbaum, 1994)

Exemplo de Sistemas Operacionais: Windows, Linux, Unix, DOS, Mac-OS, OS/2, Netware.

Não há um sistema realmente que seja melhor que o outro. Existem vantagens e desvantagens de cada um, por exemplo, o Linux é muito mais seguro que o Windows, mas o Windows é mais fácil de usar. Alguns possuem mais erros que outros, mas podem ser corrigidos. A aplicação do sistema também não importa. Os recursos de segurança dos sistemas windows98 e ME são muito escassos, pois foram feitos para o usuário comum e não para ambiente empresarial. O sistema também vai depender do tipo de rede que você terá. Se você terá um servidor *Web* ou algum tipo acesso externo, seria melhor utilizar o Linux ou o Windows NT. Se for uma rede interna, somente utilize *Novell Netware*, que ainda não fez a sua história quanto à Internet, mas ainda é insuperável nas redes locais.

2.1.1 A Influência do Sistema Operacional

Como citado anteriormente o sistema operacional não influi tanto na segurança quanto algumas pessoas pensavam. Citamos anteriormente que se alguém precisasse de um servidor externo seria melhor que utilizasse o Linux ou o Windows NT se fosse apenas uma rede local. E o *Netware*? A *Novell* passou a apostar na Internet recentemente, suas antigas versões não possuem o suporte devido à rede. E os seus servidores web ainda não são tão utilizados em larga escala quanto o Apache e o IIS. Por isso não nos aprofundaremos muito nele, pois nossos principais focos são os ataques remotos.

2.1.2 Unix Versus Windows

Por serem os dois sistemas mais utilizados em servidores. O Windows possui algumas vantagens sobre o Unix. Mais simples de se usar, é fácil de se instalar programas e drivers, e possui mais programas no mercado. O Linux em comparação, possui inúmeras vantagens sobre o Windows. (Assunção, 2004)

Vamos listar algumas:

- Têm distribuições gratuitas;
- Criptografia inquebrável de senhas. Só se descobre no método de tentativa e erro;
- Melhores ferramentas de rede;
- Melhor gerenciamento de permissões;
- Código fonte aberto.

2.1.3 Vantages do Open Source

As vantagens do código-fonte aberto (ou open-source) são muito grandes. Esse termo significa que os programas criados (ou o próximo sistema operacional) vêm junto com o seu código fonte, ou seja, você pode ver exatamente o que está executando, se for usuário conhecedor. Para começar, qualquer um pode fazer sua própria versão de Unix ou Linux. É só pegar o código fonte de algum sistema já existente e alterá-lo. Como o sistema operacional foi feito de programadores para programadores, ainda possuem alguns recursos que o usuário comum não consegue entender. Mas até isso o *open-source* está mudando. Novas ferramentas gráficas foram criadas para facilitar o uso do Unix. Podem torna-lo tão fácil de usar quanto o Windows. E o melhor, são melhoradas rapidamente pelos seus próprios usuários e distribuídas gratuitamente. Alguns bons exemplos são o *GNOME* e o *KDE*, os ambientes gráficos mais usados na atualidade.

2.2 REDES DE COMPUTADORES

A RFC 2828, define rede de computadores como uma coleção de *hosts* interligados para troca de dados. *Host* é definido pela mesma RFC 2828 como sendo um computador ligado a uma rede de comunicação que possa usar os serviços providos pela rede para trocar dados com outros sistemas interligados.

A Figura 2, demonstra um esquema de redes de computadores. A definição de rede de computadores é utilizada para sistemas de todos os tamanhos e tipos, indo desde a enorme Internet até a um simples computador pessoal interligado remotamente como um terminal de outro computador.

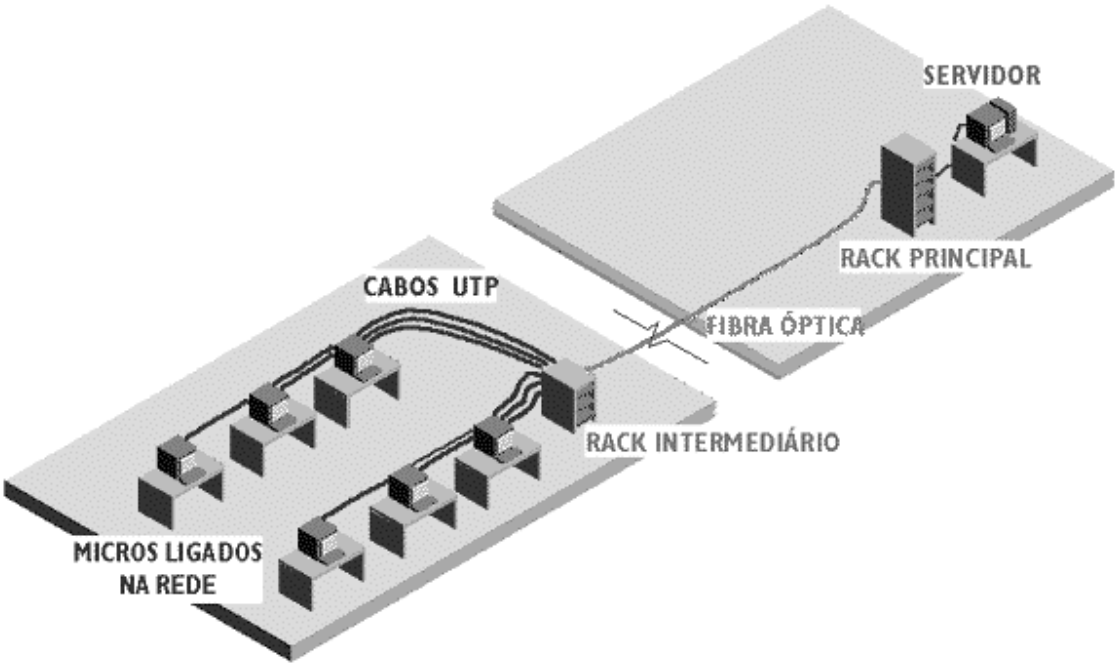

Figura 2 – Esquema de Redes de Computadores.(**Tanenbaum, 1994**)

2.2.1 Objetivos das Redes de Computadores

Segundo (Tanenbaum, 1994), define-se os objetivos da interconexão de computadores como sendo:

Compartilhamento de recursos: fazer com que todos os programas, dados e equipamentos da rede estejam disponíveis a todos os usuários independentemente de sua localização física. Como exemplo, pode-se citar o compartilhamento de uma impressora por vários usuários;

Economia: a substituição gradativa dos antigos *mainframes* para as redes de computadores de pequeno porte, significou uma redução muito grande nos custos de manutenção

dos sistemas de informação. Essas redes de computadores de pequeno porte possibilitam um aumento da capacidade de processamento a medida que a demanda cresce, ao contrário dos grandes *mainframes*, onde a sobrecarga só poderia ser solucionada com a substituição do mesmo por um *mainframe* de maior capacidade, a um custo geralmente muito elevado;

Prover um meio de comunicação: As redes de computadores também são um poderoso meio de comunicação entre pessoas, possibilitando inclusive o trabalho em conjunto mesmo estando a quilômetros de distância.

2.3 PROTOCOLOS

2.3.1 TCP/IP – (*Transmission Control Protocol* / *Internet Protocol* ou Protocolo de Controle da Transmissão / Protocolo de Internet) é uma pilha de protocolos que vem sendo modelada há décadas, desde a criação de uma rede chamada *ARPANET*, em meados dos anos 60, nos EUA. Ao contrário do que muitos acham, não é apenas um protocolo de comunicação, mas uma pilha deles. Estas pilhas de linguagens de comunicação permitem que todas as camadas de comunicação em rede sejam atendidas e a comunicação seja possível. Todas elas, de forma ou de outra, tem de atender a todas as camadas, para que os computadores consigam trocar informações.

Podemos fazer uma analogia dessas, com a comunicação verbal. Se alguém fala com outra pessoa, esta o atende, é porque todas as camadas para que a "fala" seja entendida foram atendidas. Imagine que, para que duas pessoas se comunicarem verbalmente, será necessário:

1. que ambas saibam o mesmo idioma;

2. que ambas tenham toda a estrutura fisiológica para que emitam som (voz – cordas vocais, língua, garganta, pulmões);

3. que ambas possuam toda a estrutura fisiológica que ouçam o som (orelha, ouvido interno, tímpanos).

Nesta pilha de protocolos, temos como mais importantes as mostradas a seguir:

2.3.2 IP - (*Internet Protocol* ou Protocolo de Internet) é o responsável pelo endereçamento lógico de pacotes TCP/IP. Além disso, é responsável pelo roteamento destes pacotes, e sua fragmentação, caso a rede seguinte não possa interpretar pacotes do mesmo tamanho.

Um endereço IP padrão é composto pelas seguintes partes:

- Endereço de Rede;
- Endereço de Sub-rede;
- Número (único) do *host*.

Um endereço IP é composto de 32 bits divididos em 04 octetos de 08 *bits* cada. Este, composto de 32 *bits*, é capaz de endereçar nada mais nada menos que 2^{32} = 4.294.967.296 estações distintas. Cada octeto pode conter valores que variam de 0 à 255 (um *byte*) e, são devidamente separados por um delimitador de campos, neste caso o ponto ("."), por exemplo 192.168.0.1 .

2.3.3 TCP – (*Trasmission Control Protocol* ou Protocolo de Controle da Transmissão) é um protocolo de transporte, responsável pela entrega correta dos pacotes. Sua principal característica é a confiabilidade. Para cada pacote ou conjunto de pacotes que envia, espera do destinatário uma confirmação da chegada dos mesmos. Caso isso não ocorra, ou o pacote chegue corrompido, ele tratará de efetuar a retransmissão. Ele também coloca nos pacotes um número de seqüência, para que o destino possa remontar o dado original, caso os pacotes sigam por caminhos diferentes ou cheguem atrasados (fora de ordem). Este número de seqüência também é usado como recurso de segurança.

2.3.4 UDP – (*User Datagram Protocol* ou Protocolo do Datagrama do Usuário) assim como o TCP, também é um protocolo de transporte. Contudo, não possui nenhuma checagem de erros, confirmação de entrega ou sequênciamento. Ele é muito utilizado em aplicações que necessitam de tráfego urgente, e não sejam tão sensíveis a algumas perdas de pacotes. Exemplos de aplicações que usam UDP como transporte: transmissão de áudio e vídeo pela rede (*RealPlayer*, *Realvideo* ou *Media Player*), jogos online (como *Quake*, *Half-Life*). Pela falta do número de

seqüência ou confirmação de conexão, tráfego UDP é muito mais vulnerável em termos de segurança.

2.3.5 TELNET – (Protocolo de Terminal Virtual) é o protocolo Internet para estabelecer a conexão entre computadores. Através dessa conexão remota, pode-se executar programas e comandos em outra máquina, como se tivéssemos o teclado do computador ligado diretamente a ela. O visual de uma conexão via Telnet é semelhante ao que se tem em BBS's de interface Dos, e a operação do computador remoto se dá da mesma forma, ou seja, através de uma linha de comandos Unix ou a partir de um menu de comandos disponíveis que sempre se apresenta em algum lugar da tela (esta última forma é a mais comum em servidores que permitem acesso público).

O Telnet pode ser usado para a pesquisa de informações e transferência de arquivos, tudo depende do que o computador ao qual estamos conectados permitir que façamos. Ele também é muito usado por operadores de sistemas (*Sysop's*) a fim de fazer algum tipo de manutenção (muitas pessoas pensam que o *Sysop* de nosso provedor sai de casa toda vez que tem algum problema nos servidores, estão muito enganadas, muitas vezes ele faz a manutenção de casa mesmo, via Telnet.). (Anônimo)

2.3.6 FTP – (*File Transfer Protocol* ou Protocolo de Transferência de Arquivos) é o protocolo de transferência de arquivos, serve única e exclusivamente para ser um banco de software. Não se podem executar programas remotamente como no caso do telnet, apenas pegar e colocar arquivos. Desde a criação da internet, o FTP é largamente usado. Uma de suas vantagens é, como ele é usado somente para transferências de arquivos, sua velocidade pode chegar a ser muito maior do que pegar arquivos em HTTP.

2.3.7 DNS – (*Domain Name Server* ou Servidor de Nome do Domínio) A função do DNS é extremamente útil. Já imaginou se tivéssemos que decorar o endereço IP de todas as páginas que visitamos na internet? No máximo 10 decoraremos, mas e o resto? Para acabar com esse problema surgiu o DNS. A sua função é procurar em um banco de dados um nome que

corresponda a um IP. O DNS do nosso provedor de acesso vai checar esse nome em nosso banco de dados e se encarregar de nos direcionar ao IP encontrado.

Nós mesmos podemos configurar e ligar alguns nomes a endereços IP. O método mais fácil de se fazê-lo é utilizar o arquivo *HOSTS*. O processo é o mesmo do *LMHOSTS* do NetBios, e o arquivo é encontrado no mesmo local. O interessante do *HOSTS* é que podemos pregar peças nos nossos amigos, direcionando endereços como *www.fbi.gov* para o IP de alguma homepage *hackeada* ou até seu endereço IP local e contar vantagem de que invadimos o FBI. Muitos *"hackers"* hoje em dia usam isso para aparecerem na televisão e *"hackear"* ao vivo.

2.3.8 SMTP - (*Simple Mail Transfer Protocol* ou Protocolo de Transferência de E-Mail Simples) é o protocolo responsável por entregar mensagens de e-mail a um destinatário. Toda vez que nossos e-mails são enviados, um servidor SMTP se encarrega de leva-los ao nosso destino. Esse servidor geralmente se aloja na porta 25. O interessante do SMTP é que ao contrário do POP3 (visto a seguir), não é necessário senha para enviar um e-mail. Podemos abrir o Microsoft Outlook e mandar e-mails como se fossemos George Bush ou Tom Cruise. A falta de segurança no envio de mensagens é o ponto de partida para facilidade de se enviar *e-mails* anônimos. O SMTP ainda permite anexar a uma mensagem de texto conteúdos binários (programas, por exemplo), utilizando o *MIME*.

2.3.9 POP3 – (*Post Office Protocol*) é outro protocolo de mensagem, só que agora é o responsável por o recebimento dessas mensagens. O POP3 já necessita de senhas para poder habilitar o acesso dos usuários às suas caixas postais, além de saber "re-montar" os arquivos enviados em formato *MIME* com o SMTP. O POP3 geralmente se localiza na porta 113. Por este protocolo é possível fazer um ataque de *bruteforce* para tentar descobrir as senhas, já que a maioria dos serviços possui falhas que possibilitam softwares maliciosos de serem rodados.

2.3.10 HTTP – (*Hyper Text Tranfer Protocol* ou Protocolo de Transferência de Super Textos) Esse sem dúvida é conhecido por muitos. Afinal, quem nunca viu na frente do endereço de uma homepage esse nome? *http://www.altavista.com/*. O HTTP é o protocolo responsável de

transmitir textos, imagens e multimídia na Internet. Sempre que abrimos uma homepage (mesmo que só contenha textos), usamos esse protocolo. Achei interessante comentar sobre ele para que se entenda melhor como a Internet não funciona isolada com um só protocolo. HTTP, FTP, TELNET e os outros muitas vezes trabalham em conjunto e nem percebemos. Quando baixamos um arquivo, preste atenção no *link*. É muito provável que de uma página navegada por HTTP, se envie a um servidor FTP.

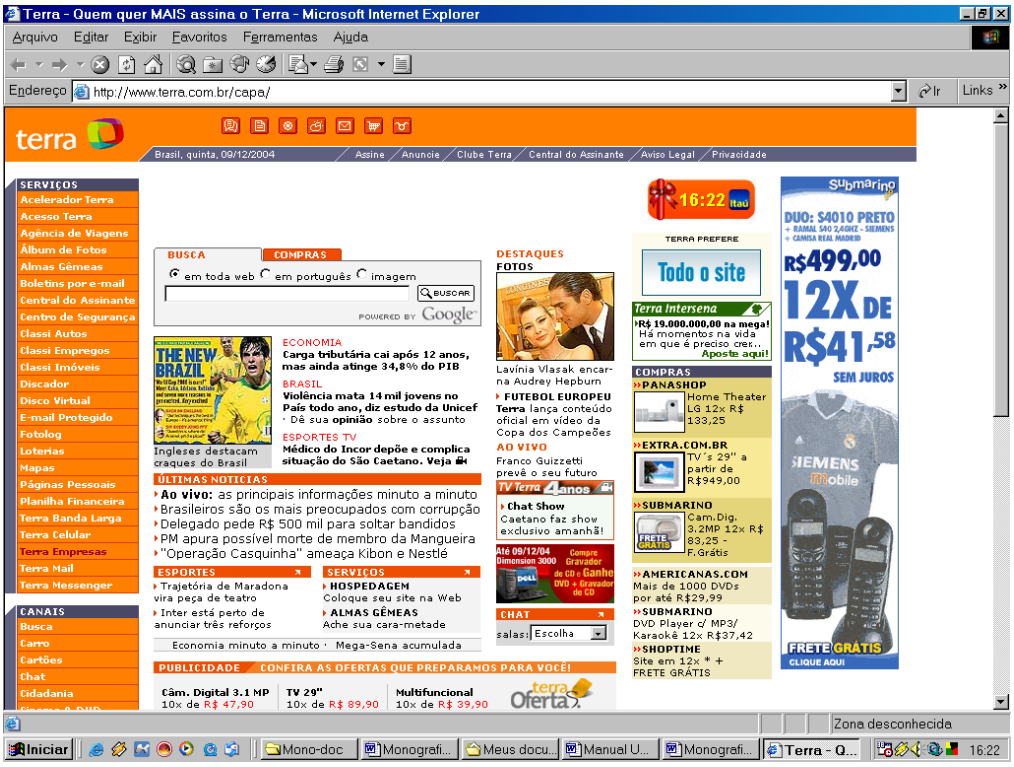

Figura 3 – Exemplo de Protocolo HTTP

A figura 3 mostra um protocolo HTTP, onde é digitado o endereço da página no campo endereço e a página é mostrada.

2.3.11 SNMP – (*Simple Network Management Protocol* ou Protocolo Simples para Manejar a Rede). Usando o SNMP podemos obter informações detalhadas sobre contas de usuário, equipamentos de rede, portas e serviços abertos e muito mais. Existe uma ótima ferramenta, IP *Network Browser* da *SolarWinds* (*www.solarwinds.net*). Ela mostra tudo do administrador.

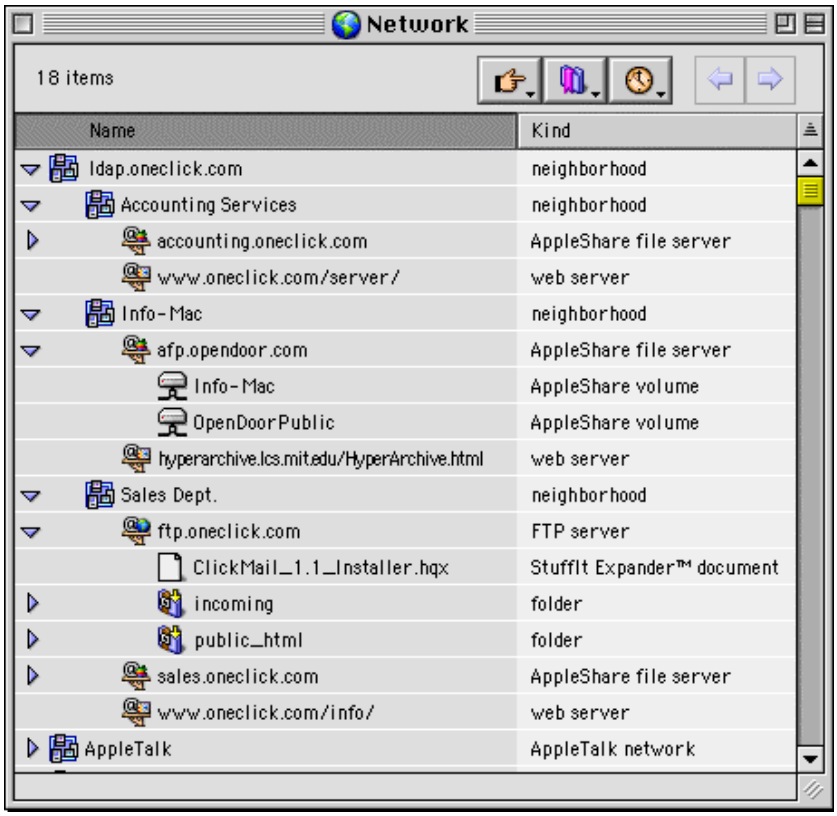

Figura 4 – Screenshot do IP Network Browser (Assunção, 2004)

Esta figura mostra as informações detalhadas da conta de um usuário.

2.3.12 NetBIOS – (*NETwork Basic Input Output System* ou Sistema de entrada e saída básica da rede) A interface NetBios (*NetBEUI*) foi um dos primeiros protocolos disponíveis para uso em redes compostas de computadores pessoais. Como o seu próprio nome diz, foi designado para um protocolo eficiente e pequeno para uso em redes caseiras não roteadas de cerca de no máximo 200 computadores.

Atualmente o NetBIOS é usado mais exclusivamente em pequenas redes não-roteadas podendo ou não estar rodando em vários sistemas operacionais. A implementação NetBIOS do Windows é chamada de *NetBEAUI*. As suas vantagens incluem:

- Grande velocidade de transferência;
- Nenhuma necessidade de configuração;

- Compatibilidade com praticamente todos os sistemas operacionais, inclusive o Linux (usando o Samba).

A única desvantagem é que o NetBIOS não suporta roteamento: o máximo que vamos conseguir invadir usando esse protocolo é o computador de nosso primo ou de nossa namorada que usam o mesmo provedor que nós. Se for um provedor diferente, esqueça. Outro problema: a estrutura de segurança do NetBIOS é extremamente pobre. Facilmente podemos quebrar as senhas utilizadas (usando *bruteforce*). (Assunção, 2004; Tanenbaum,1997)

2.4 PORTAS

Cada protocolo se comunica com a camada de transporte através de portas de comunicação. Existem 65536 portas, as portas de 1 a 1023 são conhecidas como "*Well Know Port Numbers*", portas privilegiadas ou portas baixas, que possuem serviços mais comuns previamente associados, como é o caso do Telnet (porta23) e do FTP (porta21).

Cada protocolo precisa de uma porta, TCP ou UDP, para funcionar. Os mais antigos possuem suas portas padrão já determinadas.

Exemplo:

Protocolo/ Aplicação	Porta Padrão	Transporte
FTP	21	TCP
TELNET	23	TCP
SMTP	25	TCP
WINS *NameServer*	42	UDP
HTTP	80	TCP
POP3	110	TCP
SNMP	161	UDP
SNMP trap	162	UDP

Tabela 1 – Relação porta protocolo (Fuctura, 2004)

As portas acima de 1023 são denominadas portas altas, e são usadas como *end points*, ou pontos de "devolução" de uma conexão. Imagine uma conexão como um cano de água conectando duas casas. A diferença é que neste cano, a água pode ir a qualquer sentido. Portanto, ao tentar lermos nosso correio eletrônico, provavelmente usaremos o protocolo POP3, que funciona na porta 110. Nosso computador estabelecerá uma conexão com o servidor de correio, na porta 110 remota, e 1026 (por exemplo) localmente. A porta local é na maioria dos protocolos, uma porta acima de 1023, desde que não seja sendo usada.

2.5 TIPO DE TRASMISSÃO *BROADCAST*

Esse tipo de transmissão, os dados são enviados apenas uma vez, mas para toda a rede. Esse processo não é muito eficiente, pois faz a velocidade cair bastante já que todos os computadores irão receber os dados. Mesmo os hosts que não fizeram o pedido receberão os dados. Somente não irão processá-los. Esse método é utilizado no ataque de *smurf*, em que é enviado um broadcast para diversos endereços IP e o endereço de origem (que deveria ser o IP de quem enviou) é modificado para o da vítima. Resultado: centenas de máquinas mandarão milhares de unicasts para um único *host*.

2.6 *FIREWALL* OU PAREDE DE FOGO

Como o nome sugere (do inglês, "parede ou porta de fogo"), os *firewalls* são esquemas de hardware, software, ou os dois juntos, capazes de, baseados em características do tráfego, permitir ou não a passagem deste tráfego. Basicamente, o *firewall* analisa informações como o endereço de origem, endereço de destino, transporte, protocolo, e serviço ou porta. Para cada pacote que passar pelo *firewall*, ele consistirá uma ACL (*Access Control List*), (ou lista de controle de acessos), que é uma espécie de tabela de regras, que contém informações sobre que tipo de pacote pode ou não passar. Baseado nesta informação, rejeita ou repassa o dado. Contudo, ao contrário do que muitos pensam, um *firewall* não é apenas um produto, seja hardware ou software (Fuctura, 2004).

Para que um esquema de *firewall* seja eficiente, algumas regras devem ser observadas:

1. todo tráfego entre as redes precisa passar pelo *firewall* ou filtragem;

2. deve existir alguma forma de *reporting* ou *log*, para se ter uma idéia de que tipo de tráfego está sendo rejeitado;

3. o *firewall* em si deve ser imune à penetração / invasão (deve rodar o código mais simples possível, e a menor quantidade de código possível).

A seguir, vemos um esquema simples de *firewall*:

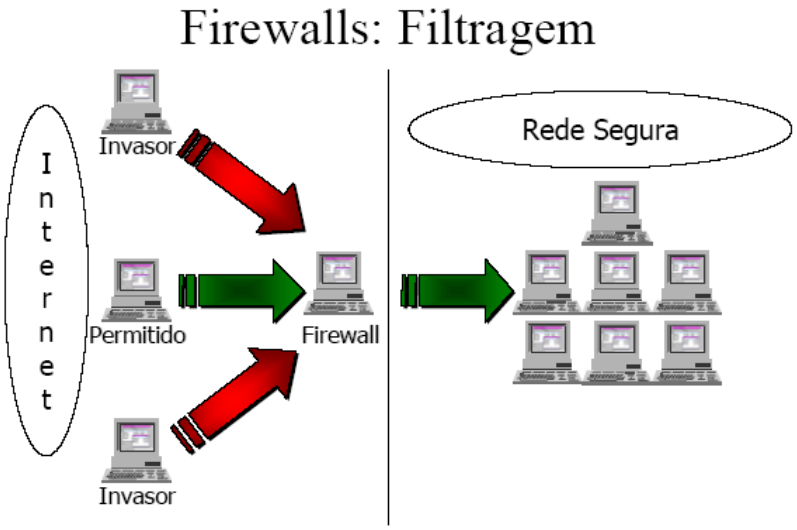

Figura 5 – Esquema do funcionamento do firewall (Fuctura, 2004)

A figura 5 demonstra a tentativa de dois invasores tentando passar pelo *firewall*. O *firewall* faz a filtragem de tudo que está tentando entrar na rede segura, possibilitando somente a passagem dos pacotes permitidos.

Existem outros modelos para uso com *firewall*. O modelo mais eficiente é o de "zona desmilitarizada". Nele, servidores e computadores críticos são protegidos tanto de rede interna quanto da rede externa. Veja:

Existem alguns produtos, ou soluções de software bastante interessantes, que tentam implementar o mesmo princípio de um *firewall* em seu computador. Estes programas são chamados de *Personal Firewalls*, e são bem baratos, ou de graça.

2.7 ORIGEM DOS ATAQUES

2.7.1 *Hacker*

O termo *hacker* é definido pela RFC 2828 (*Request for Comments* nº 2828), como sendo alguma pessoa com um grande interesse e conhecimento em tecnologia, não utilizando eventuais falhas de segurança descobertas, em benefício próprio, pois a política dos Hackers é que todas informações devem ser disponíveis a todos. O termo é usado erroneamente, especialmente pelos jornalistas, como alguém que efetue crimes cibernéticos.

2.7.2 *Cracker*

O termo *cracker ou intruder*, esse sim, é definido pela RFC 2828 como sendo alguém que tente quebrar a segurança ou ganhar acesso a sistemas de outras pessoas sem ser convidado. Não é obrigatoriamente uma pessoa com grande conhecimento de tecnologia como um *hacker*.

3 TÉCNICAS DE INVASÃO

Nesse capítulo, procurou-se descrever as técnicas mais utilizadas para invasão, estas técnicas são utilizadas para se obter lucros financeiros, status entre os *hackers*, apenas para atrapalhar um serviço oferecido pela Internet ou até mesmo por vingança de um ex-funcionário. Estas técnicas foram retiradas de livros e artigos publicados na Internet. Estas podem ser utilizadas por qualquer plataforma, diferenciando somente as ferramentas que serão abordadas no próximo capítulo.

3.1 ENGENHARIA SOCIAL

A engenharia social é uma tática muito usada pelos *crackers*. É o que a lei chama de estelionato. Consiste em enganar uma pessoa para se conseguir vantagem. Por exemplo: Uma pessoa liga para um provedor e diz que perdeu sua senha e pede ao telefonista que a recupere. Alguns provedores pedem documentação para comprovar que é o dono da conta, outros passam a senha sem nenhum problema. Esse método não é utilizado somente para descobrir a senha, mas uma série de informações valiosas como informações pessoais sobre os administradores da rede, sobre os fornecedores de suprimentos e manutenção, quem tem acesso privilegiado a qualquer servidor ou estação, avaliar o grau de conhecimento desta pessoa (quanto menor, melhor, se possuir acesso privilegiado), números de telefones importantes, lista de endereços de correio eletrônicos.

A partir daí, o próximo passo será tentar relacionar as informações coletadas. Baseando nessas informações, o *hacker* irá pesquisar sobre vulnerabilidades existentes nas versões dos programas, serviços e sistemas operacionais usados pela rede.

Independentemente da abordagem adotada, o *hacker* terá duas coisas em mente: objetividade e máxima dissimulação. Tudo será feito sem pressa, para não levantar suspeitas. Ele poderá até fazer amizade com alguém que trabalhe na empresa. Tudo isso dependerá da informação que se deseja obter, o *hacker* avaliará se todo o esforço vale a pena. Alguns fazem isso pelo desafio e superação, dificuldades somente para provar a si mesmos que são capazes. (Assunção, 2004; Anônimo)

3.2 *FOOTPRINTING*

O *Footprinting* nada mais é do que a busca detalhada da maior quantidade de informações possíveis do alvo da invasão, sem cair e sem ser pego por Firewalls, ou seja, ninguém que queira invadir sua rede vai ficar tentando uma invasão sem ter uma estratégia definida, a partir do resultado obtido pelo *Footprinting* é que é traçado o plano ou estratégia de invasão, há casos em que essa busca de informações, chega a durar meses.

Talvez esta seja a parte mais importante para se obter êxito em uma invasão e não é à toa que é uma das partes mais demoradas e a que é dada maior atenção pelos *hacker's*, é a partir dela que o invasor consegue informações essenciais para obter êxito em uma invasão, como a topologia da sua rede, nomes de domínio, sistema operacional usado, enfim o objetivo principal é criar um perfil do seu sistema de segurança, para tentar descobrir falhas e possíveis brechas que possam ser exploradas. (Assunção, 2004; Fuctura, 2004)

3.3 *TROJAN*

O nome *Trojan* é uma alusão à história do antigo cavalo de tróia, em que o governante da cidade de Tróia na antiga Grécia foi presenteado com um cavalo de madeira no qual havia escondido soldados inimigos. Segundo (Assunção, 2004) o *trojan* possui características similares aos vírus, tais como: perda de arquivos, falhas de memória, erros em periféricos. A grande diferença é que o *trojan* pode ser considerado um vírus inteligente, pois é controlado a distância pela pessoa que o instalou. Esse indivíduo então consegue "enxergar" o seu computador, podendo

realizar desde as mais simples tarefas como mexer o mouse à utilização do seu IP como ponte para outros ataques. Conseguem ficar escondidos em arquivos de inicialização do sistema operacional e se iniciam toda a vez que a máquina é ligada. Existem variadas formas de *trojans*, cada qual para uma finalidade, as mais importantes são citadas nos próximos itens:

3.3.1 *Trojans* de Informação

Consiste em instalar um *trojan* que detecta todos os dados vitais do sistema, como todas as senhas digitadas no servidor junto ao endereço IP das máquinas e envia informação para uma conta de e-mail configurada pelo invasor.

3.3.2 *Trojans* de Portas

Esse é o *trojan* mais comum existente na internet hoje. Possuem na sua maioria dois arquivos: um servidor para ser instalado no computador da vítima e um cliente com interface gráfica para manipular o servidor remotamente. As portas de um sistema variam entre 0 e 65535 e servem para identificar serviços rodando no sistema. O servidor se torna mais um serviço ao escolher alguma porta para "escutar" as chamadas do cliente. O *trojan* que utiliza portas TCP estabelece uma conexão com o servidor, atuando diretamente de dentro do sistema, já o que utiliza portas UDP, comunicam-se via pacote de dados enviados no *host* alvo. Não tão confiável como o TCP, não garante a entrega dos pacotes e o recebimento da resposta. Quase todos os trojans atuais são para a arquitetura Windows. Os poucos existentes em outros sistemas, tais como: Unix, Linux, Novell e Macintosh são chamados de *backdoors*. A diferença entre um *trojan* comum e o *backdoor* é que o segundo é mais difícil de se instalar. Em um sistema Unix, por exemplo, para conseguir instalar um *backdoor* é preciso possuir privilégios de super usuário (root).

3.3.3 *Trojans* de Ponte

Largamente usado por *hackers* e *crakers* do mundo inteiro. Consiste em instalar um servidor no computador que possibilite que através dele (e do endereço IP) o invasor possa realizar ataques de invasão e recusa de serviço, que consiste em colocar a culpa em outro *host*.

3.3.4 *Rootkits*

Esse tipo especial de *backdoor* é utilizado no Unix e Linux. Ao ser executado pelo operador do sistema ele substitui arquivos executáveis importantes por versões "infectadas". Essas versões podem ser tanto *trojans* de portas quanto de informação. Fornecendo acesso irrestrito ao invasor como acesso de super-usuário.

3.3.5 Escondendo o *Trojan*

Existem muitos programas que escondem os servidores em arquivos executáveis. A técnica corresponde em juntar o *trojan* com algum outro executável e criar um terceiro contendo os dois. Um método muito utilizado pelos *hackers* e *crakers*, é renomear algum executável para foto e deixar um largo espaço. Por exemplo: supondo que o nosso servidor é o arquivo server.exe. Então iríamos renomeá-lo para loira.jpg .exe.

Estes métodos possuem uma falha, qualquer programa anti-vírus logo detectará o arquivo. Para que o anti-vírus não o detecte, Faça com que o programa quando executado renomeie o servidor e o execute. Assim, como se o servidor estiver como vodoo.dll passe-o para sysconf.exe e execute. Este método passa despercebido pela maioria dos programas de detecção.

3.4 *SNIFFERS* OU FAREJADORES

Após o intruso ter invadido e ter acesso a um determinado computador da rede, o próximo passo será instalar *backdoors*, *trojans* em seguida *sniffers* (farejadores). *Sniffers* são programas que trabalham, geralmente em rede *Ethernet*, utilizando uma interface de rede do computador em modo promíscuo. Esta interface "escuta" tudo o que passa pelo barramento e envia para o arquivo de *log* do *sniffer*.

Através dos *sniffers*, serão capturados todo e qualquer dado que trafegar pelo barramento *Ethernet* onde estiver conectado o *sniffer*. Dados como nomes de usuários, senhas, e informações que identificam os computadores podem ser coletadas da rede. O arquivo de *log*, que é armazenado no computador local que foi instalado pelo invasor, é regularmente consultado pelo mesmo.

Os *trojans* e *backdoors* costumam estar camuflados em arquivos binários do sistema para não serem descobertos. Os *sniffers* também utilizam este procedimento, instalando um *backdoor* no binário "ps" do sistema, por exemplo, para não ser percebido. Eles executam em *background* no sistema gerando como resultados os *logs* (Fuctura,2003; Anônimo).

3.5 *SPOOFING*

O *spoofing* ocorre quando invasores autenticam uma máquina em nome de outra forjando pacotes de um *host* confiável (*trusted*). Nos últimos anos, essa definição foi expandida para abranger qualquer método de subverter a relação de confiança ou autenticação baseada em endereço ou em nome de *host*. (Anônimo).

Os principais e mais largamente utilizados tipos de *spoofing* são: IP *Spooffing*, ARP *Spoofing* e o DNS *Spoofing*.

3.5.1 IP *Spoofing*

Muitos serviços em servidores que utilizam o protocolo TCP/IP se utilizam de um tipo de autenticação baseada em *hosts*, são as *address-based authentication* (autenticação baseada em endereços). Sendo assim a autenticação para um determinado serviço é feita pela simples verificação do *hostname* e/ou IP do solicitante contra uma lista de "*hosts* confiáveis", e se houver relação, o solicitante estará autorizado a utilizar o serviço. Todavia o *spoofing* não é uma coisa tão simples assim. O fato de se alterar o IP da origem não torna o *spoofing* possível, existem outros fatores entre os quais o mais importante é como são gerenciadas as conexões e transferências TCP. Para que haja uma conexão entre dois *hosts* existem uma série de checagens passadas lado a lado até que a conexão seja definitivamente estabelecida. Por esta razão o TCP se utiliza de uma seqüência de números, associando-os aos pacotes como identificadores, que são utilizados por ambos os *hosts* para checagens de erros e informe destes, de maneira que ao ser iniciada a conexão, o *host* solicitante envia um pacote TCP com uma seqüência de números inicial. O servidor responde então com sua seqüência de números e um ACK. Esta seqüência é igual a do cliente + 1. Quando o solicitante ou cliente recebe o ACK do servidor ele envia então o seu, que trata-se da seqüência enviada inicialmente pelo servidor + 1.

A pessoa que irá realizar o *spoofing* tem então dois problemas na verdade: o de alterar o IP origem, mais simples de resolver, e o de manter a seqüência de números, que por serem geradas arbitrariamente, complica em muito esta tarefa.

A vulnerabilidade a este ataque varia de sistema pra sistema sendo mais ou menos efetiva de acordo com o algoritmo utilizado para a geração das seqüências numéricas. E o número de serviços rodando que se utilizam de autenticação *address-based* condenada a muito, pelo menos desde 1985, de acordo com o artigo publicado de Robert Morris então da Bell *Labs*, ou seja, bem antes da Internet virar a febre que é hoje. Ainda assim, hoje em dia, ainda temos uma infinidade de servidores na rede mundial contendo inúmeros serviços que se utilizam de RPC (*Remote Procedure Call*), sabidamente vulneráveis a ataques do tipo IP *spoofing*, como todos os sabores dos todos poderosos Unix, incluindo o Linux, Windows NT, 2000 e outros (Assunção, 2004).

3.5.2 ARP *Spoofing*

Esta técnica é uma variação do IP *spoofing*, que se aproveita do mesmo tipo de vulnerabilidade, diferenciando apenas que na autenticação ARP, apesar de também ser *address-based* utiliza o endereço MAC (*Media Access Control*) ou físico.

O *host* que ataca envia um mapa informações erradas ao ARP *cache* remoto de maneira que os pacotes que saem do alvo para o seu suposto destino são roteados para o *host* que atacou. Porém este ataque tem uma série de limitações sendo uma delas o tempo em que uma entrada dinâmica permanece no ARP *cache*, que é muito curto, desfazendo então a informação errada inicialmente implantada. Uma maneira de se proteger seria criar entradas estáticas no ARP *cache*, no entanto isto se torna inviável devido ao tempo e a administração dispensados.

3.5.3 DNS *Spoofing*

Esta técnica é muito simples e não requer grandes conhecimentos do TCP/IP. Consiste em se alterar as tabelas de mapeamento de *hostname-ipaddress* dos servidores DNS, ou seja, seus registros do tipo *host* (A), de maneira que os servidores, ao serem perguntados pelos seus clientes sobre um *hostname* qualquer, informam o IP errado, ou seja, o do *host* que está aplicando o DNS *spoofing*.

3.6 DoS

Denial of Servise ou em português Recusa de serviço, ataques desse tipo geralmente não comprometem a privacidade dos dados. A finalidade de um ataque DoS é tirar um serviço, servidor, computador ou até mesmo uma rede do ar. Os ataques do tipo DoS são usados muitas vezes em conjunto com invasões, ou porque alguns tipos de invasões exigem que determinados computadores não estejam funcionando (no caso do *spoofing*) ou para despistar ou desativar a

atenção da invasão em si. Ataques DoS também são usados simplesmente para “atrapalhar” ou desacreditar um serviço.

Os ataques DoS na sua grande maioria usam *buffer overflows* para conseguir sucesso. Contudo, qualquer forma de tirar um computador, serviço ou rede do ar é considerando um ataque DoS. Por exemplo, as maiorias dos servidores que possuem alguma segurança possuem também *logs* de acesso. Imagine que o administrador coloque os *logs* no mesmo espaço em disco do sistema. Assim, se gerarmos milhares (talvez milhões) de entradas no *log* , o arquivo irá crescer até ocupar todo o disco. Outro tipo de ataque DoS comum: várias redes possuem programada uma ação, caso um *login* tente por diversas vezes efetuar *logon* e erre suas credenciais. Esta ação geralmente é um bloqueio indeterminado da conta (*login*), que apenas pode ser restaurado com a intervenção do administrador. Forçar o travamento de uma conta dessas é considerado um ataque DoS, principalmente quando essa conta é usada por algum serviço (se a conta for bloqueada, o serviço sairá do ar).

Já ataques que visam tirar do ar uma rede, ou um servidor através de tráfego excessivo, ou enviando pacotes de rede inválidos também são possíveis. A cerca de 3 anos atrás, foi lançado na Internet uma vulnerabilidade em pilhas TCP/IP de computadores Windows. Consistia em enviar para um determinado serviço, pacotes TCP com uma sinalização de “urgência”. Contudo, o conteúdo do pacote era composto de caracteres inválidos. Este ataque DoS ficou conhecido como OOB (Out Of Band data). Hoje em dia, a grande maioria das pilhas TCP/IP são protegidas conta esse tipo de ataque, e variações. Porém, se a quantidade de informação inválida for realmente muito grande, ainda existe a possibilidade de tirar do ar o computador. Para se obter a quantidade suficiente de pacotes, o ataque do tipo DoS foi estendido, para o que conhecemos hoje como DDoS (Anônimo).

3.6.1 DDoS

Distributed Denial of Service consistem geralmente em enviar para uma única máquina ou rede, milhões de pacotes de rede ou requisições de serviço, em um dado momento. Não existe maneira de gerar este tráfego todo de um dado momento. Daí surgiu a idéia do DDoS:

várias máquinas espalhadas por toda a Internet, enviando tráfego simultaneamente, para um mesmo servidor, estação ou rede.

Os ataques do tipo DDoS ficaram conhecidos a partir dos ataques recentes realizados contra *sites* na Internet populares, como *yahoo.com*, *amazon.com*, entre outros. Contudo, utilitários que exploram ou criam ataques DDoS, apesar de difíceis de obter, já existem desde meados de 1999.

A lógica de um ataque DDoS é bem simples. Imagine um servidor de paginas *web*, que normalmente recebe cem mil acessos por dia. Agora, imagine que duzentos ou trezentos computadores espalhados pela Internet, ao mesmo tempo, e continuamente, enviem requisições de acesso à página. Dependendo do número de requisições, o servidor poderá deixar de responder simplesmente porque chegou ao seu limite de conexões.

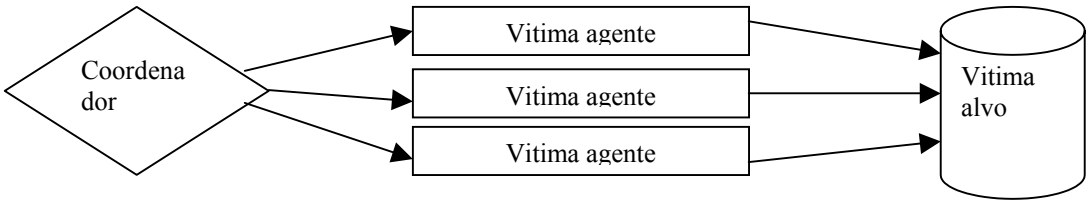

Figura 6 – Exemplo de ataque DDoS (Carvalho, 2003)

3.6.2 *Buffer Overflows*

O *buffer overflows* é uma ataque usado a muito tempo e que ainda será muito usado. Compreende em lotar os *buffers* (memórias disponíveis para aplicativos) de um servidor e incluir na sua lista de processos algum programa ou um trojan. Todos os sistemas são vulneráveis a *buffer overflows* e a solução é a mesma, procurar se já existem correções.

3.7 *SMURF*

Um dos ataques mais devastadores, funciona da seguinte maneira, envia pacotes ICMP (protocolo que informa condições de erro) *spoofados* para centenas, talvez milhares de sites. Envia-se pacotes com o endereço IP da vítima, assim fazendo com que ela receba muitos pacotes *ping* de resposta ao mesmo tempo, causando um travamento total. Ainda não existe uma proteção eficaz contra esse tipo de ataque.

3.8 FILTRANDO PACOTES NA REDE

Não é verdade que o *sniffer* pode ser usado em seu computador para capturar pacotes do seu provedor, na realidade o programa tem que estar instalado no computador central de uma rede em que se querem capturar pacotes. Utilizando o exemplo do provedor, todos os seus usuários realizam o processo de autenticação em um servidor antes de conectar-se à rede. Assim, primeiro é necessário conseguir invadir o servidor e depois colocar o *sniffer*. Ele irá monitorar absolutamente tudo, às vezes até informações pessoais dos usuários, como endereço e telefone. Como são muitos pacotes em uma rede, o farejador é configurado para obter somente o essencial e importante: as senhas (Assunção, 2004).

3.8.1 Capturando Senhas

A principal preocupação de um operador é, ou pelo menos deveria ser as senhas. Afinal, por mais seguro que o sistema seja, uma senha adquirida maliciosamente é sempre perigosa. O único interesse dos *crakers* é capturar *logins* e senhas. Existem algumas opções que ainda possibilitam filtrar os tipos de pacotes recebidos. Vamos supor que eu quero descobrir todas as senhas que comecem com “C”. Após configurar o *sniffer* e esperar, ele começa a me enviar os pacotes recebidos já “selecionados” com o que desejo.

3.9 *SCANNERS* OU VARREDURAS

Nenhum sistema é perfeito. Falhas em programas e sistemas existem sim e são uma ameaça à segurança. Geralmente ocorre do seguinte modo: um administrador acidentalmente descobre que algum recurso de seu sistema gera um erro em resposta a algum tipo de pedido. Para exemplificar, suponhamos que a rede em que o administrador trabalha só se comunica gerando mensagens de "olá". Um dia ele escreve "alô" sem querer e descobre que ao enviar a mensagem para outra máquina, ela fica confusa e trava. Bem, a resposta deveria dizer "Desculpe, só olá aceito". Foi descoberto um *bug*. Agora imagine que centenas de *bugs* são descobertos a cada dia e que o seu sistema "confiável" de hoje, pode ser destruído amanhã. Existem algumas saídas para fazer uma análise mais garantida. A primeira é que você conheça desde o primeiro ao último *bug* existente, o que é inviável. Uma outra saída é a utilização de *scanners*, ou seja, *scanner* é uma varredura que se faz em um sistema para verificar existências de *bugs* (Fuctura, 2004).

3.9.1 Descobrindo falhas em um *host*

Para entender qual parte do seu sistema é mais vulnerável, você terá que pensar com malícia. Se você usa um *firewall* e desabilita o acesso externo aos servidores de FTP e Telnet, eles não serão sua preocupação. Em alguns hosts, deixa-se habilitada apenas a porta 80 (www) para acesso externo. Muitos se sentem seguros desse modo. Mas enganam-se. Atualmente, a quantidades de falhas que existem em servidores *World Wide Web* é enorme. Algumas delas tão perigosas que possibilitam acesso ao interpretador de comandos do sistema, podendo gerar uma "entrada" para o invasor na rede. Outras podem fazer com que se consuma toda a memória existente, causando um *Buffer Overflow*.

3.10 *BRUTEFORCE* OU FORÇA BRUTA

O método de força bruta é muito demorado. Pode levar horas, e as vezes dias. Mas é um dos mais eficazes. Utiliza-se um programa que tenta conectar-se a um sistema utilizando todas as combinações possíveis de letras e números. Este processo também depende da velocidade da máquina e da conexão. Deve-se utilizar uma conexão dedicada (à cabo, via rádio e outras) e vários computadores. Tendo 20 computadores rápidos trabalhando cada um em um setor (um tentando descobrir senha começadas por a, outro por b, outro por c etc) o tempo para se conseguir diminuirá consideravelmente. Existem alguns casos em que a força bruta é mais rápido, como quando se tenta quebrar um arquivo de senha localmente. Pode ser um *passwd* (senha) do Unix ou do Windows (Assunção, 2004).

3.11 FALHAS EM SISTEMAS

3.11.1 Como surge o *bug* - *bug* ou falha, surge a partir do momento que o programador comete um erro. Ou seja, indiretamente é um erro humano que gera falha nos programas. Por serem pequenos erros, muitas vezes passam despercebidos e só são descobertos por algum *hacker* ou analista de segurança. Os erros do Windows, por exemplo. A grande maioria das falhas descobertas, são os próprios usuários que descobrem. Os criadores mesmo que têm o código-fonte e conhecem o programa como a palma da mão, raríssimas vezes percebem algum erro. Para ser mais seguro, um programa tem que ser testado de todas as maneiras possíveis.

3.11.2 Descobrindo as falhas - para o programador experiente é mais fácil verificar se um sistema tem falhas, utilizando por recursos de *debug* que checam por erros de *buffer overflows* e outros. Para o usuário comum é bem mais difícil descobrir algo, o interessante seria visitar páginas especializadas no assunto, que a cada dia publicam novos tipos de erros descobertos. Algumas muito boas são a *Security-focus* (*www.security-focus.com.br*) e a Hacker brasileira (*www.hacker.com.br*).

3.12 SER ANÔNIMO NA REDE

Anonimidade na rede é algo muito discutido atualmente. Existe maneiras de ser totalmente indetectável na rede e é bem simples. Muitos programas e ferramentas prometem tornar seu usuário invisível, mas são pura enganação. O que você precisa é de conhecimento, não de *softwares*. Um usuário pode conseguir passar em computadores no Japão, Alemanha e Finlândia antes de atacar um site no Brasil. Um *craker* pega o *notebook*, vai a um telefone público, utiliza uma conta roubada de internet, se conecta a cinco computadores pelo mundo e utilizando-os conecta-se a um sistema de anonimidade. Após isso entra em um site do FBI, por exemplo, e apaga alguns arquivos. Nunca será pego. Todos os bons *crackers* não são pegos, justamente pela facilidade de se esconder.

3.12.1 ***Anonymizer*** - é um dos muitos serviços de anonimidade na Internet. Visitando a sua *homepage* (*www.anonymizer.com*) ele possibilita que você digite algum endereço e seja redirecionado para ele. Exemplo: eu digito *www.felainternet.com.br* na página do serviço e ele me redirecionará para o provedor de Internet FELA, só que com o endereço IP do *anonymizer*. Ou seja, se os administradores de página consultarem o *log*, não verão meu real endereço. Não é possível utilizar um *anonymizer* para conectar-se a outro (Fuctura, 2004; Assunção, 2004).

4 FERRAMENTAS

Nesse capítulo serão mostradas ferramentas necessárias para realizar a invasão, apesar de a maioria dos exemplos de programas ser para Windows, este trabalho não foi desenvolvido para nenhum sistema operacional. Existem ferramentas excelentes similares as que serão demonstradas para Linux e Unix (entre outros sistemas como Macintosh). Serão citados *sites* durante a explicação onde se podem encontrar ferramentas para várias plataformas.

4.1 PROGRAMAS USADOS PARA OBTER INFORMAÇÕES

Diversos programas podem ser usados para obter informações, como dito no item 3.2 onde talvez seja parte mais importante para se obter êxito, sobre a rede, computadores ou servidores remotos. (Assunção, 2004) Alguns deles são:

4.1.1 *CA Unicenter TNG Frame Work*

Este é um programa da *Computer Associate*, que tem por objetivo a gerência completa de uma infra-estrutura de TI, desde módulos de anti-vírus até *backup*. O módulo básico, chamado "*Framework*", é gratuito, e pode ser encontrado do *site* da CA para avaliação (*http://www.cai.com*). Porém, apesar de gratuito, o módulo básico possui o mapeamento de rede via SNMP incluído. Com essa aplicação, usando o protocolo SNMP, que a grande maioria dos roteadores, *switches*, e outros equipamentos de conectividade suportam, inclusive servidores, é

possível construir o mapa de uma rede com detalhes impressionantes. Portanto, é primordial numa rede, que o *firewall* filtre tráfego SNMP (portas 161 e 162).

Existem outros programas que usam o SNMP para construir o mapa de uma rede. Como exemplo: *Tivolo*, *Lucent NavisAcess*, e o *SNMPc*. Qualquer ferramenta SNMP pode ser usada.

4.1.2 *Essential Net Tools*

É um programa que explora a má configuração de computadores Windows conectados a Internet. Através dele, é possível, dado um intervalo de endereços IP, visualizar quais destes estão com o compartilhamento de arquivos e impressoras ativado, e com algo compartilhado. Sendo que existem umas quantidades enormes de computadores que possuem a raiz do drive C: compartilhada, permitindo acesso a qualquer arquivo dentro do disco, inclusive o .pwl, arquivo que possui a senha salva dos usuários deste computador. Para evitar que o *EssNetTools* seja efetivo, é necessário filtrar no *firewall* as portas usadas pelo NetBIOS(135, 136, 137, 139 e 445, TCP/UDP).

4.1.3 *CIS (Cerberus Internet Scanner)*

O CIS é um programa de análise de vulnerabilidades. Ele executa em ambiente Windows NT4 ou 2000 e, dado um endereço IP, ele produzirá uma página HTML com todos os testes realizados. O CIS testa por vulnerabilidades conhecidas, como VRFY (SMTP), DNS, *Web* entre outras. Ele consegue acessar informações de contas de usuários de máquinas Windows NT, má configuradas. Para evitar a efetividade do CIS, é aconselhável usar ele próprio, analisar quais vulnerabilidades foram encontradas. E saná-las uma a uma.

4.1.4 *WhatsUP Gold*

O *WhatsUp* é um programa desenvolvido pela empresa *IPSwith*, com a intenção de ser uma ferramenta de monitoração de rede. Porém, ele possui internamente uma função usada para "descobrir", dado um intervalo de endereços, quais estão ou não ativos, bem como outras informações, como o nome das máquinas. Bastante eficiente em redes Microsoft, com ele podemos ter uma idéia de quantas máquinas estão ativas numa determinada classe, por exemplo.

4.1.5 Telnet

O próprio Telnet do windows pode ser usado para descobrir que versão um determinado servidor está rodando, por exemplo, de *sendmail*, servidor **web**, POP3 ou FTP. Para isso, basta disparar um TELNET para a porta do serviço desejado.

Vejamos:

telnet empresa.com.br

Resultado:

220 dominus.elogica.com.br ESMTP Sendmail 8.9.3/8.9.3; Wed, 29 Mar 2004 20:38:40 –0300

Agora sabemos que o servidor é um *sedmail*, versão 8.9.3 Aliado ao nmap, descobrimos qual o sistema operacional.

4.2 *TRIN00, TFN (TRIBE FLOOD NETWORK, SCHAFT)*

Estes são exemplos de ferramentas de ataque DDoS, técnica explicada no item 3.6.1. O *trin00* já foi portado para a plataforma Windows, enquanto o TFN é o mais usado. Já o *Schaft*, apesar de relativamente antigo, é bem mais raro de ser encontrado. Atualmente, existe uma forma do agente do *trin00* que infecta computadores como um cavalo de tróia. Já o TFN possui uma

versão chamada TFN2K, com várias melhorias, incluindo criptografia da conversão entre os clientes e os agentes, de forma a burlar detecção destas ferramentas.

Em ambientes corporativos ligados à Internet, a forma mais comum de detecção é através da quantidade de tráfego. Na maioria das redes que possuem monitoração de tráfego, a característica será uma série de tentativas de conexão, ou tráfego gerado de diversas máquinas da rede interna, para um único endereço na Internet.

4.3 NETBUS

O *NetBus* é um programa que foi desenvolvido para fazer manutenção de micro computadores a longa distância, mas caiu em mão erradas, devido a sua facilidade de uso ele se compreende em uma interface muito simples de boa visão, identificação e utilização, para usarmos o *NetBus* precisamos mandar um *patch* (o cavalo de tróia, explicado no item 3.3) cliente (programa que será instalado na maquina vítima, permitindo o acesso do invasor) para a vítima, e para que ela sem saber do que se trata, instalamos em seu computador para que nos de acesso. Explicaremos os passos da utilização do *NetBus* a seguir:

1º Mandar o *patch* cliente para a vitima.

Para se mandar um *patch* para a vitima, podemos mandar o executável para ela ou criarmos um instalador para que ao instalar um programa ele executa um comando instalando automaticamente o *patch* cliente na máquina da vitima, como exemplo de programa instalador, o *WinZip Self Extractor*, que podemos encontrar na página do *Winzip* (*www.winzip.com*). Mandando assim a pessoa instalará o programa sem saber que é um trojan.

2º Depois de instalado, fazer a conexão.

Com o IP da vítima, colocamos no local do *NetBus* onde está denominado "*Host name*/IP" e clicamos em "*connect*" para fazer a conexão. Agora temos controle total da máquina.

A Figura 7, demonstra a tela de controle do Trojan *NetBus* e com ela o invasor pode realizar várias ações maliciosas, como exemplo pode-se citar a função *Open CD-ROM*, que quando acionada abre ou fecha a sua bandeja:

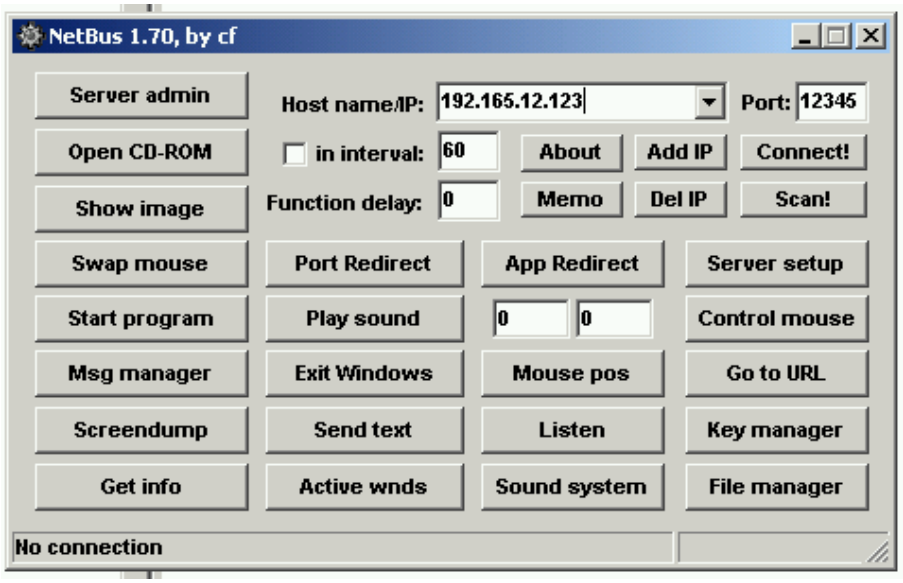

Figura 7 - Trojan NetBus 1.70, tela de controle (Carvalho, 2003)

4.4 PETITE

É um compressor de arquivos executáveis, utilizado para esconder cavalos de tróia. Utilizado no windows, ele diminui cerca de 30% ou mais do arquivo original. Um trojan (ou mesmo um vírus) comprimido é absolutamente indetectável por anti-virus e *scanners*. Isso porquê esses programas se baseiam na estrutura do arquivo para identificá-lo. É como se estivesse fotos na memória e as comparasse. Como não encontrou nenhum igual, não mostra nenhum tipo de aviso. Um operador de sistemas tem que conhecer muito bem os arquivos e conferir sempre novas alterações (como data e hora de outros arquivos) para evitar que um trojan comprimido seja instalado em seu sistema.

Vamos realizar passo a passo o processo de esconder um trojan de um anti-vírus:

1.Passaremos, como exemplo, o *Norton* para que encontre o arquivo infectado:

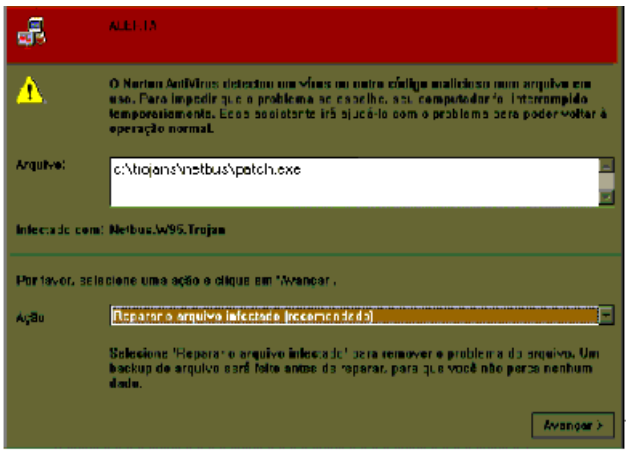

Figura 8 - Tela do Norton Anti-Vírus detectando a existência de vírus (Assunção, 2004)

2. Agora, abriremos o programa PETITE para comprimir o arquivo EXE do servidor do Netbus.

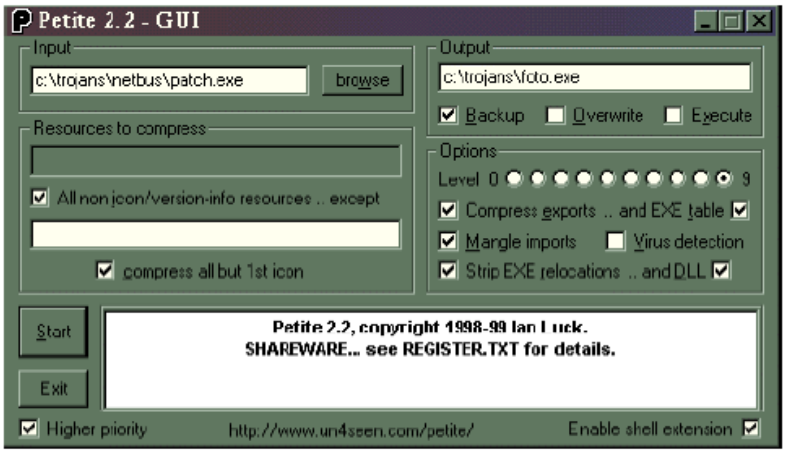

Figura 9 – Tela do Petite compactando um arquivo EXE (Assunção, 2004)

3. Com o arquivo já comprimido, novamente testaremos o anti-vírus:

Figurar 10 – Tela do Norton Anti-Vírus com mensagem da não detecção de vírus

(Assunção, 2004)

4.5 *SOFTWARES ZUMBIS*

Programas que automatizam o processo de causar um DoS, explicado no item 3.6, em algumas máquina. São instalados em computadores estratégicos (como universidades, centros de pesquisas e outros) que possuem conexão rápida à Internet e configurados para atacar ao mesmo tempo. Se instalarmos o programa em vinte máquinas de diferentes endereços e configurá-las para enviar 10.000 pacotes cada uma, com certeza derruba qualquer *host*. Um programa muito utilizado para isso é o *Tribal Flood Network*. *Trojans* também são largamente usados para esse fim. (Anônimo)

4.6 UTILIZANDO O BROADCAST COMO ARMA

Realizar um ataque de DoS é bastante simples. Pode-se utilizar vários tipos de programas e *softwares zumbis* para fazê-lo. Às vezes nem é preciso um programa adicional. *Sites* como Yahoo e Altavista utilizam *Webspiders* (programas utilizados para procurar informações)

para checar o conteúdo de *homepages*. Muitos *webspiders* checando o mesmo servidor ao mesmo tempo pode levá-lo ao colapso. Causar um DoS em algum servidor de *e-mail* também é possível. Utilizando um programa de *e-mail* bomba (*software* que envia milhares de *e-mails* para o mesmo endereço) ou cadastrando o *e-mail* alvo em serviços de *spam* (como mensagens de anjos, piadas, notícias e outros) pode encher uma caixa postal e travar todo o sistema. Ou mandar um *e-mail* para alguém que tenha serviço de resposta automática, utilizando o próprio *e-mail* da pessoa. Exemplo: mande uma mensagem para *fulano@provedor.com.br* usando esse *e-mail* como se fosse o seu, já que não se precisa de senha para enviar. A resposta do Fulano mandará mensagens para ele mesmo, travando sua caixa postal.

4.7 PORTAS ABERTAS COM SERVIÇOS ATIVOS

Como sabemos, existem 65535 portas TCP e UDP, dificultando assim invadir um computador pessoal, que pode estar em qualquer porta acima de 1023. Em servidores, muitas delas possuem serviços rodando, tais como:

21- FTP, 23- TELNET, 25 - SMTP, 80 - WWW

Esses são apenas alguns dos muitos serviços que são rodados em computadores de empresas que precisam estabelecer contato com filiais e clientes. Um sistema que possui os seguintes serviços acima ativos, pode obter sérios problemas com segurança. Mas enquanto, por exemplo, o seu computador cheio de jogos e que você utiliza para mandar uma mensagem pelo ICQ (programa de bate-papo)?. Mas essas são randômicas, ou seja, a cada vez que uma conexão for feita, a porta mudará. Isso impede que algum invasor fique à espreita e tente conectar a portas padrões. Dificulta, mas não impede. Algum cavalo de tróia instalado sem sabermos pode abrir uma porta qualquer e permitir a conexão de qualquer pessoa. Para saber quais portas estão abertas em um sistema remoto, utilizamos o *scan*, técnica explicada no item 3.9, de portas. Existem muito programas desse tipo. Alguns são o *ChaOscan*, *Shadow Scan* e o *HakTek*.

Funcionam da seguinte maneira: vão tentar se conectar a todas as portas de um endereço IP fornecido, mostrando todas as portas encontradas “ativas” e o seu conteúdo. É uma boa tática para encontrar cavalo de tróia sem depender de anti-vírus, já que todos usam portas.

Exemplo: eu quero analisar o meu próprio computador para saber se têm alguma porta aberta.

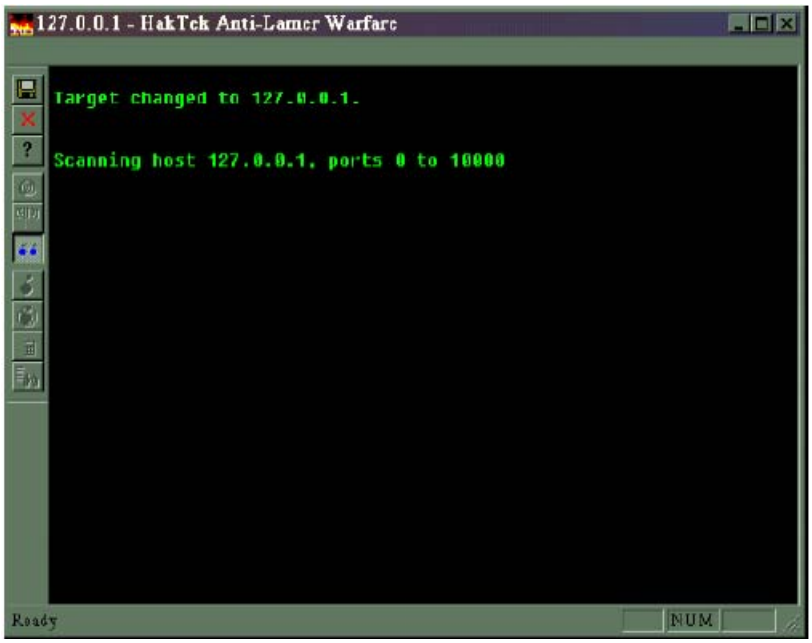

Figura 11 – Tela de demonstração da portas abertas do Software *HakTek* (Anônimo)

Para isso, vamos usar o *HakTek*. Então o programa tenta *scannear* portas no endereço 127.0.0.1 (o chamado endereço de *loopback*. Serve para quando não estamos conectados na Internet e precisamos utilizar algum programa de análise que precise de endereço IP). Foram encontradas as seguintes portas ativas:

80

1256

21554

31337

A primeira porta sabemos que é o servidor de páginas que roda no computador. Mas e as outras três? A porta 1256 era a que o ICQ havia aberto na hora. As outras duas são portas de trojans que usamos como teste. A porta 21554 é do trojan *Girlfriend* e a porta 31337 é do *BackOrifice*.

4.8 *WARDIALERS*

Os *wardialers* ou discadores de guerra, são programas que checam um lista de telefones procurando por telefones conectáveis. Podem ser bem úteis. Por exemplo, supondo que o telefone comercial de uma empresa seja 8829-1122. Mande algum programa (como *Toneloc*) tentar se conectar a telefones do número 8829-1100 a 8829-1300. Bom, pode ficar um pouco caro os impulsos, mas a chance de conseguir algum número de modem externo é grande.

4.9 WORDLISTIS

São listas de palavras criadas especialmente para se descobrir senhas. Quando temos em mão um arquivo de senha do UNIX com o sistema de criptografia do Unix, por exemplo, a criptografia é inquebrável, mas pode-se usar programas como *Craker Jack* e *Shadow Scan*. Eles pegam um arquivo criado por uma pessoa com lista de palavras comuns (geralmente utilizadas como senhas, tal como alien3, tricolor, secreta, 1234) o criptografa utilizando o mesmo sistema de senhas do Unix e compara os arquivos. Se o programa encontrar algum usuário em que a criptografia tenha ficado igual, o nome lhe é informado.

4.10 FERRAMENTAS *MULTI-BRUTEFORCE*

Existem muitos programas de *bruteforce* específicos, como o *WebCrack* (que quebra senhas de páginas *web*). Mas há também excelentes programas que conseguem quebrar senhas de vários tipos diferentes, como senhas de *e-mail*, NetBios, *web*, Unix, enfim, quase tudo. Já citamos o *Shadow Sacan*, mas mostraremos dois outros programas para realizar *multi-bruteforce*:

Brutus: Rápido e com uma configuração muito específica, produz excelentes resultados. Até senhas do NetBus ele quebra. E salva as sessões.

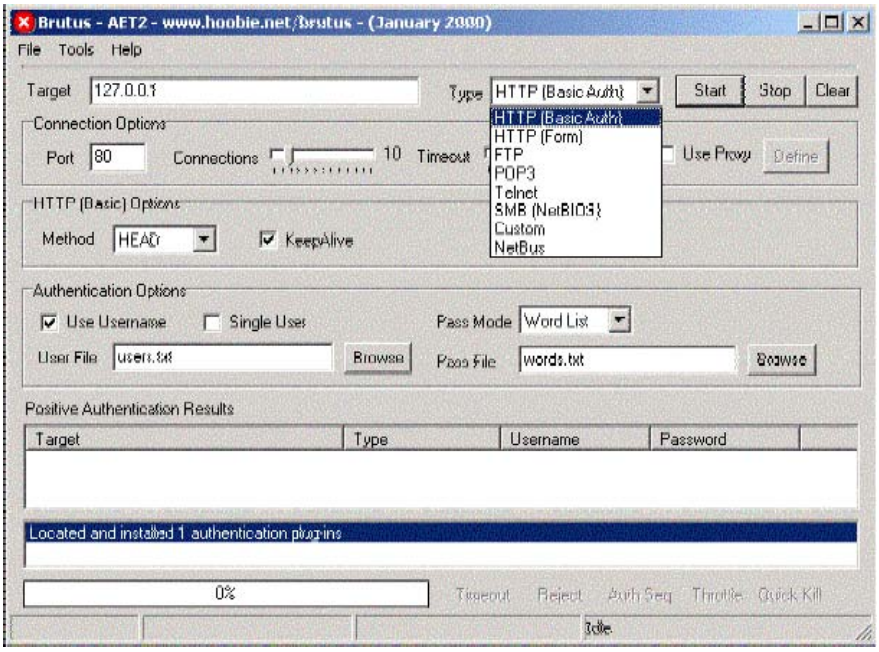

Figura 12 – Menu do programa Brutus (Anônimo)

Unsecure: Mais rápido que o Brutos, esse programa de *bruteforce* é um dos mais usados para o Windows. Sabendo a porta do servidor (TTP, Telnet, etc...), o programa faz todo o serviço.

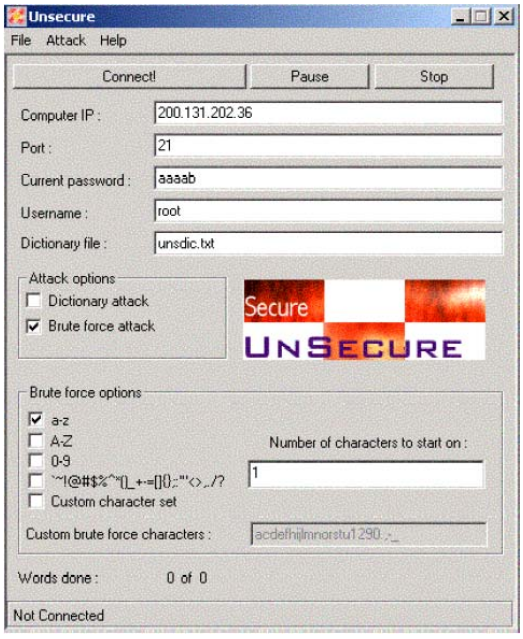

Figura 13 – Menu do programa Unsecure (Anônimo)

4.11 UTILIZANDO *EXPLOITS*

Exploits são programas criados para explorar falhas.O *printe.isapi* do IIS 5.0 possui um *exploit* chamado *iishack2000*. Ele possibilita que apenas digitando o IP de algum computador, consegue ter acesso ao interpretador de comandos (ou *Shell*). Assim podendo fazer o que quiser com o sistema. Existe também o *iishack*, que utiliza um erro de *buffer overflow* para fazer com que se possa mandar o servidor executar qualquer tarefa (como um trojan). Cada *exploit* corresponde a uma falha, por isso geralmente os grande site de segurança publicam os dois juntos. Geralmente vem em código-fonte C. Perl e alguns poucos em executáveis comuns.

Se quisermos encontrar compiladores para rodar os *exploits*, estão disponíveis na página *www.programmersheaven.com*. É uma *homepage* com muitos recursos de várias linguagens de programação. Se não quisermos procurar um compilador, um bom *exploit* que checa mais de 200 vulnerabilidades de *Unicode* (IIS), está disponível em *http://tomktech.n3.net*. (Anônimo)

4.12 CRIANDO UM *TROJAN*

Mostraremos um exemplo da criação de um trojan simples, utilizaremos apenas um componente, o *FtpServer*. É interessante notar que com um simples objeto (que funciona desde o primeiro *Delphi* à sua versão mais atual) podemos criar programas muito sofisticados.

O trojan consistirá em um mini-servidor FTP que ficará ativo na porta 2099. Criaremos um programa em que quando passamos o mouse em cima de um botão, esse some e aparece em outro local do formulário. A nossa intenção é que a pessoa o utilize sem desconfiar que seu computador está aberto para o mundo. Vamos fazê-lo passo a passo:

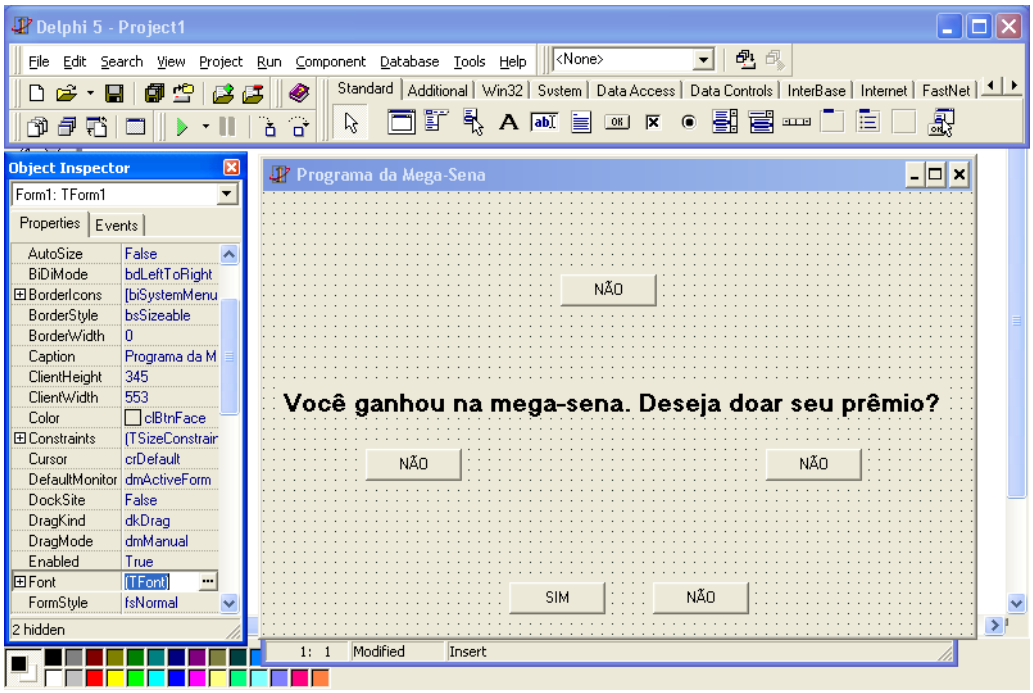

Figura 14 - Trojan feito em *Delphi* (Anônimo)

- Coloque cinco botões no formulário do modo que está demonstrado na figura14.

- Clique em cada um dos botões, vá em suas propriedades (mostradas no *object inspector* à esquerda) e mude o *caption* para "NÃO". Apenas no local que um botão está colado no outro, coloque o *caption* do botão como "sim".

- Coloque um *label* na barra de componentes (ilustrado com a letra A) e coloque-o no formulário.

- Mude o *caption* do *label* para "Você ganhou na Mega-Sena, Deseja doar seu premio?".

- Agora novamente clique em cada um dos botões, com exceção dos dois que estão juntos, vá a propriedades e mude o valor de *visible* de *true* para *false*.

Na verdade o botão que o usuário passar o mouse encima mudara seu *status* de visível para invisível e ao mesmo tempo outro dos botões escondidos ficará visível, dando a impressão de que o botão correu.

Começando pelo botão à esquerda do formulário. Vá a seus eventos (*Events*) e clique duas vezes em *OnMouseMove*. Coloque as seguintes linhas de código:

Button2.visible:=false;

Button3.visible:=true;

Repita este procedimento com todos os botões "não", mudando suas respectivas visibilidades.

Agora com o botão "SIM", vá em seus eventos e selecione *OnClick*. Escreva o seguinte código:

Label1.Caption:= "Você é muito bonzinho. Obrigado.";

Isso foi apenas o que o programa fingirá fazer.

- Na barra de *status*, selecione *FTPServer* e coloque-o no formulário.
- Clique no quadrinho de componentes do componente *FTPServer*, vá em suas propriedades e mude a porta (propriedade *port*) para 2099.
- Clique no formulário para selecioná-lo. Vá em seus eventos e clique duas vezes no evento OnCreate. Este evento é chamado todas vezes que o programa é inicializado.

Escreva o seguinte código:

FTPServer1.Start;

Para você se conectar ao trojan, pelo Windows, vá em Iniciar/Executar : ftp 127.0.0.1 2099.

Para terminar o trojan, resolveremos um ultimo problema. Para que o usuário não feche o programa:

- Selecione o formulário clicando duas vezes em cima dele. Nas propriedades, clique em duas vezes no símbolo "+" que está na frente da propriedade BorderIcons. Logo que a propriedade se expandir, clique na frente de bitSystemMenu e selecione false. Isso fará com que aquele X que clicamos para fechar programas desapareça.

- Coloque um novo botão, e mude o caption para “Fechar”. Clique duas vezes nele para acessar o evento OnClique. Agora coloque o seguinte códico-fonte:

Form1.Visible:=false;

Quando o botão for clicado ele apenas irá esconder o formulário, não o fechará.

4.13 FERRAMENTAS PARA SE TORNAR ANÔNIMO NA REDE

4.13.1 ***Proxys*** - possibilita uma ponte entre um computador e um servidor. Para exemplificar melhor, imagine que você possui uma rele local, mas somente um dos seus computadores tem placa *fax-modem*. Então você se conecta por ele e usa um *proxy* para que outro computador da rede faça uma ponte e acesse a Internet pelo servidor. O endereço IP utilizado será do servidor. Acontece que existem muitos *proxys* gratuitos na Internet. Brasileiros ou internacionais, eles possibilitam que você navegue tranqüilamente e às vezes ficam até mais rápidos do que a conexão comum.

4.13.2 ***Wingates*** - parece muito com o *proxy*, mas sua aplicação é um pouco mais perigosa por dois fatores. Primeiro: o *wingate* é acessado Pelo Telnet, então possibilita a conexão a qualquer tipo de servidores, sejam Telnet, FTP, SMTP, POP, ou até algum *trojan*. Segundo: ao contrário do *anonymizer* e do *proxy* que só pode ser usado uma vez, o *wingate* não tem limites. Você pode conectar-se a um *wingate* chinês, depois utilizá-lo para entrar em um argentino e um italiano. A cada conexão, você terá um endereço IP diferente.

4.13.3 ***Remailers*** - é muito parecido com os outros, mas é somente para enviar *e-mails* anonimamente. Com ele não é necessário utilizar um *wingate* para se conectar a um servidor SMTP, o próprio *remailer* já é um servidor anônimo.

4.13.4 *Shells* - “o bom *craker* não é o que consegue utilizar bem um sistema Unix e invadir uma rede. É o que utiliza Windows e consegue o mesmo resultado”, máquinas utilizando serviços Unix na Internet que possibilitam que você se conecte nelas por Telnet e FTP e as utilize como se fossem locais.

4.13.5 *Outdials* - consiste em se conectar via Telnet em alguns sistemas que possibilite conexão via modem. Exemplo: você quer invadir um sistema nos EUA. Não tem dinheiro para conectar-se diretamente, então procura um *outdial*, se conecta via telnet e indica o telefone do sistema a ser invadido. O computador que roda o *outdial* discará e você conectará no sistema sem pagar absolutamente nada.

5 CONCLUSÃO

Os computadores e a Internet são instrumentos de grande valor para as pessoas, produto da capacidade e inteligência do ser humano e, como todas as coisas existentes, usado segundo a ética de cada um.

Quando estamos na rede, devemos saber que há uma certa exposição de nossa identidade e que a rede é utilizada por milhares de pessoas com vários tipos de intenções. Caso haja necessidade de passar informações pessoais, algumas precauções podem ser tomadas, tais como: procurar saber se é um site confiável, desconfiar de e-mails que não sejam de pessoas conhecidas, pois não há dificuldade adquirir conhecimento sobre invasão e informações sobre isso podem ser encontradas na própria rede, pela falta uma lei rígida para crimes cibernéticos no Brasil; podendo ser encontrados até cursos de hackers. Apesar de invasão ser crime e ser passível de pena de prisão.

Quem comete a invasão, primeiramente tem de estudar todo o local, obtendo o máximo de informações possíveis. E também deve ser muito cauteloso. Uma outra sugestão importante seria, como foram apresentados várias técnicas de invasão, é necessário saber qual o objetivo da invasão para utilizar a técnica e a ferramenta correta.

Podemos concluir que as técnicas são sempre as mesmas, o que muda são as ferramentas, gerando uma guerra. Descobre-se uma falha em algum sistema, criam-se ferramentas para explorar esta falha, então os criadores dos sistemas tentam de alguma maneira consertar esta falha. E então se procuram novas falhas, gerando um *looping*.

5.1 REVISÃO

Este trabalho foi dividido em cinco capítulos da seguinte maneira: o primeiro constando a introdução que contém o objetivo, a proposta e organização. No segundo foram abordados os conceitos de redes de computadores, protocolos, *firewall*, portas, sistemas operacionais. Conceitos que foram necessários para poder entender as técnicas, estas foram descritas no terceiro capítulo, onde foram diferenciadas de acordo com o objetivo da invasão. No quarto capítulo, foram mostrados ferramentas utilizadas pelos criminosos da rede. Onde foram mostrados exemplos, na grande maioria, de *softwares* para a plataforma Windows.

5.2 RECOMENDAÇÕES

Para quem após ter estudado esta monografia, interessado no assunto e que queira dar uma seqüência ou aprofundar em algum item específico, eu recomendo:

- Aprofundar no estudo do Telnet, pois se encontra uma enorme quantidade de material sobre o assunto, onde os ataques mais devastadores ocorridos no Brasil foram utilizando o Telnet;
- Pode-se também implementar uma ferramenta de invasão, para qualquer técnica citada nesta pesquisa;
- Fazer uma pesquisa para uma plataforma específica, por exemplo, Linux;
- Analise dos prejuízos causados pelos ataques.

BIBLIOGRAFIA

CARVALHO, Monografia de Denis Rocha de Carvalho, sobre **Segurança da Informação** (UNIPAC- 2003)

TANENBAUM, Tanenbaum, Andrew S. **Rede de Computadores**, 3ª ed. Rio de Janeiro, RJ: Campus, 1994.

InfoGuerra - Segurança e Privacidade, InfoGuerra. **InfoGuerra - Segurança e Privacidade** <*http://www.infoguerra.com.br/infoguerra.php?newsid=1037215495,16188*,> Acesso em 10 jun. 2004.

ANÔNIMO, Anônimo. **Segurança Máxima**, 3ª ed. Rio de Janeiro, RJ: Campus, 2001.

ASSUNÇÃO; FUCTURA, ebook-pdf-Hacking-Hugo Cornwall-The Hacker'sHandbook, Hacker – Livro, mini_curso_hacker **–** arquivos adquiridos no Kazaa, 2004.

ANÔNIMO, Hackers – **Apresentação quem são? O que querem ? Como fazem?** Disponível em **<***www.niconti.com.br/curioso/hackers.html*>

Acesso em: Mar. 2004

ANÔNIMO, Hackers – **Telnet** Disponível em **<***guiracz.hpg.ig.com.br/hacker.htm*>

Acesso em: Mar. 2004

ANÔNIMO, Ataque ao pop por telnet Disponível em <*http://www.geocities.com/subuns/browser.htm*>

Acesso em: Abr. 2004

ANÔNIMO, Crackeando... Disponível em <http://www.guiracz.hpg.ig.com.br/phacker.htm>

Acesso em: Maio. 2004

ANÔNIMO, Invasão pela porta 666 mas o que é isso? Disponível em <*http://www.dequasetudo.hpg.ig.com.br/Hacker%27s.htm*>

Acesso em Abr. 2004

SLOHA, Sloha, Liliana Esther Velásquez Alegre. Os *logs* como ferramenta de detecção de intrusão. Disponível em <*http://www.rnp.br/*> Acesso em 8 out. 2004.

HYNDS, Len Hynds. **Revista Veja**, 3 de novembro de 2004. Edição 1878, páginas 12 e 13.

GLOSSÁRIO

ACK - indica que a porta esta no estado *listening*(escuta).

ARP - (Analisador / Simulador de Redes de Petri) - é um programa para o auxílio ao projeto com redes de Petri.

Backup - é o processo de copiar um conjunto de arquivos ou pastas para outro local, que pode ser no mesmo disco ou em outro tipo de mídia, como disquete*, Zip drive*, CD-RW, Fita *Dat*, etc. Este processo tem a finalidade de garantir a continuidade do trabalho, mesmo que venha a ocorrer uma pane no disco ou no micro.

Banco de Dados - representa o arquivo físico de dados, armazenado em dispositivos periféricos, onde estão armazenados os dados de diversos sistemas, para consulta e atualização pelo usuário.

Bug - Falha.

Buffer Overflows - nome dado ao travamento do sistema devido às falhas de memória

Criptografia - é definida como sendo a ciência que estuda a Cigragem e decifração de informações.

Drivers - são bibliotecas para que quando um novo drive (por exemplo: Mouse, Impressora, teclado) for inserido no computador, este o reconheça.

E-mail - ou Correio Eletrônico é um serviço disponível na Internet que tem a função de controlar o envio e o recebimento de mensagens entre usuários.

Endereço MAC - é o endereço físico da placa de rede. É como se fosse a "impressão digital" dela. Cada placa de rede possui um MAC único.

Estação - é um computador.

Fragmentação de pacotes - é a divisão de mensagens grandes em pacotes, visto que estes têm tamanhos máximos, para conseguir fazer o transporte pela rede.

Host - é um computador.

Link - "ligar" uma palavra ou um trecho de texto com outras partes de um documento ou de outros documentos.

LMHOSTS - este arquivo é um banco de dados que mapeia o endereço IP.

Log - são simples arquivos de textos onde são armazenadas informações críticas ex: acesso e autenticação.

Mainframe - são computadores de grande poder de processamento que suportam vários usuários ao "mesmo tempo".

Microsoft Outlook - é um software que permite aos usuários organizarem seus *e-mails*, calendários, agendas de contatos, tarefas e documentos em um único ambiente.

MIME - (Multipurpose Internet Mail Extensions) – é um padrão de codificação dos dados. Para que um servidor Web reconheça o conteúdo como definido nas especificações WAP, é importante configurá-lo corretamente com os tipos MIME apropriados. Ex: texto (wap.wml), figura (wap.wbmp).

Novell Netware - é um Servidor (ver definição).

Provedor - é uma empresa que oferece os serviços relacionados à Internet.

Roteador - é um dispositivo que estabelece rotas para pacotes em uma Internet. Ele faz este trabalho baseado em uma tabela de rotas que contém as indicações de caminhos que o pacote poderá seguir.

Rub - é um dispositivo utilizado para conectar os equipamentos que compõe uma rede local (LAN). Com o *Hub*, as conexões da rede são concentradas, ficando cada equipamento num seguimento próprio.

Samba - é um conjunto de programas que funciona de forma integrada para possibilitar o acesso de clientes Windows a recursos remotos em um servidor Unix ou Linux.

Servidor - é um computador equipado com software que disponibiliza serviços a uma rede de computadores, como serviço de correio-eletrônico, armazenamento de arquivos, *sites*, entre outros.

Site - é um conjunto de páginas da Internet, um portal de informações colocado à disposição do mundo inteiro ou de um grupo selecionado de pessoa. Para uma empresa, um site é uma imagem moderna e sólida, uma identidade registrada e a credibilidade para seus clientes.

Swith - é um dispositivo utilizado para conectar os equipamentos que compõe uma rede local (LAN). Com o *Swith*, as conexões da rede são concentradas, ficando cada equipamento num seguimento próprio.