SIRLEI LOURDES BACH

CONTRIBUIÇÃO DO HACKER PARA O DESENVOLVIMENTO TECNOLÓGICO DA INFORMÁTICA

FLORIANÓPOLIS – SC

2001

UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Sirlei Lourdes Bach

CONTRIBUIÇÃO DO HACKER PARA O DESENVOLVIMENTO TECNOLÓGICO DA INFORMÁTICA

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

João Bosco da Mota Alves

Florianópolis, agosto de 2001.

CONTRIBUIÇÃO DO HACKER PARA O DESENVOLVIMENTO TECNOLÓGICO DA INFORMÁTICA

Sirlei Lourdes Bach

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, Área de Concentração Sistemas de Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Fernando A. Ostuni Gauthier, Dr.
Coordenador do CPGCC

João Bosco da Mota Alves, Dr.
Orientador

Banca Examinadora

Murilo Silva de Camrgo, Dr.

Ilson Wilmar Rodrigues Filho, Dr.

"... é justamente o desafio que impulsiona a vida e a torna interessante..."

DEDICATÓRIA

Ao Adair, constante incentivador.
Ao Mateus, inesgotável fonte de inspiração.

AGRADECIMENTOS

Ao meu orientador, Professor Dr. João Bosco da Mota Alves, pela dedicação, incentivo, e principalmente, pelas sábias palavras nos momentos de desalento.

Aos Professores do Curso de Mestrado em Ciência da Computação, por compartilharem seus conhecimentos.

Aos colegas do curso de Mestrado, pelo companheirismo.

À Adriana Elissa Notoya, pela valiosa colaboração no trabalho e por ser exemplo de coragem e dedicação.

A minha irmã Wanderléia, pelo auxílio e pelas críticas que enriqueceram este trabalho.

Aos meus sogros, Mário e Inelde, especialmente pelo empenho diário na educação de meu filho.

Ao meu esposo Adair e ao meu baixinho Mateus, pela compreensão, pelo carinho e por encherem minha alma de júbilo.

Aos amigos ocultos, mas que se fazem sempre presentes...

SUMÁRIO

1 INTRODUÇÃO **1**

2 RESGATEMOS O TERMO HACKER **3**

2.1 CLASSIFICAÇÃO DOS PIRATAS CIBERNÉTICOS 3

2.2 DIFERENÇAS ENTRE *HACKER* E *CRACKER* 5

2.3 O SIGNIFICADO DO TERMO *HACKER* 7

2.4 HISTÓRIA DOS *HACKERS* 9

2.5 A IDEOLOGIA DO *HACKER* 10

2.6 PERFIL DO *HACKER* 12

3 AMEAÇAS E VULNERABILIDADES NAS REDES **16**

3.1 VULNERABILIDADES NAS TECNOLOGIAS 17

3.1.1 Vulnerabilidade na Rede Local 17

3.1.2 Vulnerabilidade na Arquitetura TCP/IP 18

3.1.3 O protocolo IP 20

3.1.3.1 Opções IP 21

3.1.4 Fragmentação IP 21

3.1.5 O protocolo TCP 23

3.1.5.1 Opções do TCP 24

3.1.5.2 Número de seqüência do TCP 26

3.1.6 O protocolo UDP 26

3.1.7 O protocolo ICMP 27

3.2 VULNERABILIDADE NA CONFIGURAÇÃO 28

3.3 SERVIÇOS E SUAS VULNERABILIDADES. 28

3.3.1 Telnet 28

3.3.2 FTP – *File Transfer Protocol* 29

3.3.3 Finger 30

3.3.4 TFTP 30

3.3.5 Gopher 31

3.3.6 Correio Eletrônico 31

4 *MODUS OPERANDI* – ATAQUES E CONTRAMEDIDAS 33
4.1 VÍRUS 35
4.2 CAVALOS DE TRÓIA 37
4.2.1 Net Bus 38
4.2.2 Back Orifice 39
4.3 ENGENHARIA SOCIAL 41
4.4 SNIFFERS 41
4.5 EXPLORAÇÃO DE PROTOCOLOS 42
4.5.1 TCP *Sequence Number Prediction Attack* (SNPA) 42
4.5.2 Spoofing 44
4.5.2.1 IP Spoofing 45
4.5.2.2 DNS *Spoofing* 45
4.5.2.3 RIP – Rounting Information Protocol 47
4.5.2.4 ARP – Address Resolution Protocol 48
4.5.3 ICMP – *Internet Control Message Protocol* 49
4.5.4 Ataque de Fragmentação 50
4.5.5 DoS – *Denial of Service* 51
4.5.5.1 Consumo de Largura de Banda 51
4.5.5.2 Inanição de Recursos 51
4.5.5.3 Ataques de Roteamento e DNS 51
4.5.5.4 Ataques DoS Genéricos 52
4.5.5.5 SMURF 53
4.5.5.6 SYN FLOOD 54
4.5.6 DDoS - *Distributed Denial of Service* 54
4.5.6.1 Ferramentas de DDoS 57
4.5.6.2 Prevenção e detecção de ataques de negação de serviços 62
4.5.6.3 Ferramentas de detecção específicas 63
5 TECNOLOGIAS DE PREVENÇÃO 64
5.1 POLÍTICA DE SEGURANÇA 66
5.2 CUIDADOS COM O SERVIDOR WEB 68
5.2.1 Estrutura de Diretórios 68
5.2.2 CGI Scripts 69
5.2.3 SSI – *Server-Side Include* 70

5.3 MECANISMOS DE PROTEÇÃO 71
5.3.1 Firewall 71
5.3.1.1 Filtros de pacotes 72
5.3.1.2 Filtros Inteligentes 74
5.3.1.3 Servidor Proxy 74
5.4 CRIPTOGRAFIA 75
5.4.1 Cifras de substituição 76
5.4.2 Cifras de transposição 78
5.4.3 Máquinas de Cifragem 78
5.5 O USO DE CHAVES CRIPTOGRÁFICAS 78
5.5.1 Algoritmo de chave simétrica 79
5.5.2 Algoritmo de chave assimétrica 81
5.6 ASSINATURAS DIGITAIS 82
5.7 CERTIFICADOS DIGITAIS 84
5.8 DADOS BIOMÉTRICOS 85
5.9 FERRAMENTAS DE PREVENÇÃO 87
5.9.1 Wrappers 87
5.9.2 PGP – *Pretty Good Privacy* 88
5.9.3 SSL – *Secure Socket Layer* 90
5.10CONSIDERAÇÕES FINAIS 91

6 O *HACKER* INSERIDO NO DESENVOLVIMENTO TECNOLÓGICO 93
6.1 A FILOSOFIA DO *OPEN SOURCE* 93
6.1.1 A Catedral e o Bazer de Raymond 95
6.2 DIAGNÓSTICO *HACKER* DA QUALIDADE DE PRODUTOS E SERVIÇOS 98
6.3 A PERÍCIA *HACKER* NA IDENTIFICAÇÃO DE FALHAS DE SEGURANÇA 100
6.4 IMAGEM PÚBLICA DOS *HACKERS* 103
6.4.1 Confusão de sentimentos: admiração e rejeição presentes nas notícias/reportagens 105
6.4.2 O caso John Draper (*Captain Crunch*) 107
6.5 CONDUZINDO OS *HACKERS* PARA USAREM SUAS HABILIDADES EM PROL DO DESENVOLVIMENTO TECNOLÓGICO 108

7 CONCLUSÃO 110

8 REFERÊNCIAS 113

RESUMO

Dada a enorme importância política, econômica e social da Internet, urge realizar-se um estudo sobre a evolução das formas de invasão dos chamados piratas eletrônicos. Para tal, buscou-se primeiramente classificá-los de acordo com a terminologia empregada no ambiente virtual, resgatando a origem do termo *hacker*, o qual tem sofrido mutações devido ao uso equivocado pela mídia. Na seqüência, as falhas que são exploradas pelos piratas eletrônicos são apresentadas, seus *modus operandis* e as tecnologias de defesa e de prevenção, as quais são divulgadas e passíveis de conhecimento, embora sejam ignoradas ou desconhecidas por muitos administradores e desenvolvedores de sistemas, já que a insegurança nos sistemas é decorrente de falhas amplamente conhecidas com soluções já determinadas. Por fim, buscou-se resgatar a contribuição que tais piratas exercem no desenvolvimento e no aprimoramento tecnológico, evidenciado em suas habilidades, agregadas à dedicação e ao trabalho cooperativo, exigindo assim qualidade nos produtos e serviços oferecidos. Uma maneira de minimizar a insegurança dos sistemas, é manter um conhecimento detalhado e atualizado das tecnologias existentes, característica inerente aos piratas eletrônicos. Assim, reconhecendo suas habilidades e incentivando-os a aprimorarem os sistemas de segurança, os esforços que hoje são dispendidos nestes sistemas, poderão ser tranferidos para outros desafios.

ABSTRACT

Due to the great political, economical and social Internet importance, it urges to take place a study about the evolution way of the electronic pirate´s invasion. For such, it aimed firstly to classify them in agreement with the terminology used in the virtual atmosphere, rescuing the origin of the term 'hacker', which has been suffering mutation due to the mistaken use by the media. Afterwards, the flaws explored by the electronic pirates, their modus operandi, defense and prevention technologies, which are thoroughly divulged and passive of knowledge, although their are unknown or ignored by many systems administrators and developers, since the systems insecurity is thoroughly occured due to well-known defects with estabilished solutions. Finally, it aimed to rescue the contribution that such pirates exercise in technological development and improvement, evidenced in their abilities, joined to the cooperative work and dedication demanding quality in products and services. An answer to minimize the systems insecurity is to maintain a detailed and modernizes knowledge of the existent technologies, inherent characteristic to the electronic pirates. Thus, recognizing their abilities and motivating them to improve the safety systems, the efforts that are spent today in these systems, can be transferred to other challenges.

LISTA DE FIGURAS

Figura 3.1: Formato do datagrama da Internet, a unidade básica de transferência em uma interligação em redes TCP/IP. Fonte: COMER (1998) 20
Figura 3.2: Fragmentação de dados. Fonte: ZWICKY et al. (2001)..... 22
Figura 3.3: Fragmentos superpostos. Fonte: ZWICKY et al. (2001)..... 23
Figura 3.4: Conexão TCP requer um *handshake* de três vias. Fonte: ZWICKY et al. (2001) 25
Figura 4.1: Ameaças de Segurança. Fonte: STALLINGS (1999)..... 33
Figura 4.2: Dados e Modus operandi do worn ILOVEYOU. Fonte: VIANA (2000). 36
Figura 4.3: Obtenção do ISN para o ataque *sequence number*. Fonte: NORTHCUT (1999) 44
Figura 4.4: Exemplo alterações do cache do DNS . Fonte: MCCLURE , SCAMBRAY e KURTZ (2000) 52
Figura 4.5: Ataque de negação de serviço – smurf . Fonte: GARBER (2000) 53
Figura 4.1: O funcionamento de um ataque DDoS..... 57
Figura 4.7: Método de ataque das ferramentas DDoS. Fonte: BERTHOLDO e ANDREOLI (2000) 58
Figura 5.1: Implementação de um Screening Router. Fonte: SWICKY (2001)..... 73
Figura 5.2: Utilização das chaves nos algoritmos criptográficos Fonte: TRINTA e MACEDO (2000)..... 79
Figura 5.3: Funcionamento do algoritmo da chave privada Fonte: TRINTA & MACEDO (2000) 80
Figura 5.4: Funcionamento do algoritmo da chave pública. Fonte: TRINTA e MACEDO (2000) 81
Figura 5.5: Geração de uma assinatura digital 84
Figura 5.6: Implementação do protocolo SSL..... 90
Figura 5.7: Criptografia SSL 91

LISTA DE SIGLAS E ABREVIATURAS

ACK	Acknowledgment
AES	Advanced Encryption Standard
ARP	Address Resolution Protocol
ARPA	Advanced Research Projects Agency
BGP	Border Gateway Protocol
CGI	Common Gateway Interface
CERT	Computer Emergency Response Team
CRC	Cyclic Redundancy Check
DDoS	Distributed Denial of Service
DES	Data Encryption Standard
DMZ	Demilitarized Zone
DNS	Domain Name Service
DoD	Department of Defense
DDoS	Distributed Denial of Service
DoS	Denial of Service
EDI	Eletronic Data Interchange
FDDI	Fiber Distributed Data Interface
FTP	File Transfer Protocol
GID	Group Id
GNU	Gnu is Not UNIX
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
IPSec	IP Security Protocol
IPv6	IP versão 6
ISN	Initial Sequence Number

MIME	Multipurpose Internet Mail Extensions
MIT	Massachussets Institute of Technology
MTU	Maximum Unit Transmission
NIPC	National Infraestructure Protection Center
PERL	Practical Extraction and Report Language
PGP	Pretty Good Privacy
PHP	Hipertext Preprocessor (sigla recursiva)
PING	Packet InterNet Groper
PPP	Point-to-Point Protocol
RFC	Request for Comment
RIP	Routing Information Protocol
SMTP	Simple Mail Transfer Protocol
SNPA	Sequence Number Prediction Attack
SSI	Server Side Includes
SSL	Secure Socket Layer
SYN	Synchronizing Segment
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
TTL	Time to Live
UDP	User Datagram Protocol
UID	User Id
URL	Uniform Resource Locator
VBS	Visual Basic Script
WORM	Write Once, Read Many
WWW	World Wide Web

1 INTRODUÇÃO

Em qualquer sociedade existe uma pequena parcela de pessoas que não gostam de respeitar limites. Outras possuem disposição e criatividade para buscar formas de contornar e burlar os limites estabelecidos. Tais pessoas, tendo conhecimento de computadores, e estando conectados à Internet, caracterizam a pirataria eletrônica.

A Internet criou inúmeras oportunidades de negócios para os usuários e as corporações. Mas, ao mesmo tempo em que está ajudando o desenvolvimento da sociedade em uma escala jamais vista, a Internet, por sua característica "anárquica", tem propiciado algumas oportunidades para indivíduos utilizarem-se de brechas de segurança para invadir sistemas, retirá-los do ar ou mesmo tornar os serviços lentos. (BORTONE, 2000)

Nesse contexto, a cada dia é mais comum o medo de invasões na Internet. Pânicos causados por pseudo-*hackers* têm atormentado centenas de pessoas e muitas vezes simples ameaças destróem momentos de paz e diversão de uma forma irremediável. O temor aos ataques, principalmente nas organizações, pode alcançar um patamar que, em muitos casos, tem sido reconhecido como *hackerfobia*. Nestes casos, devido às restrições impostas, ou ainda pelo constante estado de alerta (ataque iminente), o bom desenvolvimento das atividades é comprometido.

Apesar da disponibilidade de diversos recursos com intuito de fornecer segurança, não se tem conseguido eliminar tal problema, muito pelo contrário, a cada dia esta vem atormentando mais a vida das pessoas. Talvez isto se dê devido à segurança ser um problema muito complexo, além disso, a complexidade dos sistemas evolui muito rapidamente, pois diversos recursos foram sendo agregados, perdendo-se o controle. Outro fator que tem contribuído para a complexidade da segurança é a informação. Hoje, informação é a chave mestra da era da computação, e conectividade para o mundo externo tornou-se crucial para os negócios do dia a dia. Redes locais e redes globais se conectam através de ligações que tornam possíveis a trocas de arquivo, de dados e de outras informações importantes. Em contrapartida, as conexões também podem ser utilizadas para obter acesso não autorizado a estas informações.

Nos deparamos então com um problema: como prover acesso entre redes externas não confiáveis e uma rede interna? A rede privada contém dados proprietários importantes.

A rede externa é desacreditada. Não prover acesso externo a sua rede também não é viável, já que usuários de fora podem ser empregados em lugares remotos, empregados locais que estão viajando, ou clientes que têm razão autêntica para acessar a sua rede privada. Por outro lado, *crackers* irão tentar de tudo para ganhar acesso e comprometer dados, serviços e a confiança da rede interna.

Devido a falhas existentes, ou brechas deixadas na segurança de um modo geral, como nos protocolos ou nas configurações dos servidores, abrem-se portas para que internautas fascinados por desafios, consigam realizar suas invasões mostrando o quanto são bons no que fazem. Neste contexto, este trabalho tem por objetivo analisar o papel do pirata eletrônico na evolução da Internet. Para tanto, buscar-se-á identificar as contribuições que tais piratas têm deixado, bem como identificar as causas das vulnerabilidades, o *modus operandi* dos invasores, e as tecnologias existentes tanto de proteção como de prevenção.

Para possibilitar o desenvolvimento de sistemas de defesa e prevenção de ataques piratas, é imprescindível ter-se o conhecimento das características dos invasores. E é justamente essa a principal justificativa deste trabalho.

A organização deste trabalho será da seguinte forma: primeiramente, os piratas eletrônicos são classificados em diferentes categorias, de acordo com seus conhecimentos e pretensões. Na sequência, identificam-se as principais vulnerabilidades que ocasionam a insegurança, enfocando o protocolo TCP/IP. Em seguida, o *modus operandi* de diversos ataques é relatado. Posteriormente, são analisadas as tecnologias de defesa e de prevenção. Por fim, é constatada a contribuição e a influência dos hackers para o desenvolvimento tecnológico.

2 RESGATEMOS O TERMO HACKER

O termo *hacker* tem sido usado de forma indevida pela mídia em geral. No entanto, buscar-se-á, neste capítulo, resgatá-lo. Para isso, serão analisadas as seguintes questões: as principais terminologias que classificam os piratas cibernéticos; as diferenças entre os *hackers* e os *crackers;* a origem da palavra *hacker*; a história dos *hackers*; a ideologia *hacker*, e finalmente, o perfil dos *hackers*.

2.1 CLASSIFICAÇÃO DOS PIRATAS CIBERNÉTICOS

No ambiente cibernético, especialmente na Internet, diversos termos podem ser encontrados para classificar os piratas cibernéticos. Tais piratas, que em sua maioria têm em comum algumas características, como: são homens jovens, autodidatas, gostam de ultrapassar os limites, são apaixonados por informática, têm conhecimentos em segurança, em auditoria e em ferramentas para quebra de sistema, dentre outras características específicas de cada categoria. No entanto, embora haja semelhança em algumas características, em outras, como por exemplo, o nível de conhecimento e o objetivo que os impulsiona à pirataria, diferem nas diversas terminologias.

Em ANONYMOUS (1998), ANTI-*HACKERS* (2000), GOMES (2000), OLIVEIRA (2000), RAYMOND (2000*b*) e VASCONCELOS (1999), alguns termos encontram-se com maior freqüência, a saber:

- *Warez*: é o indivíduo que utiliza os conhecimentos de informática para copiar programas de forma ilegal, para fins comerciais. Entre suas atividades estão compreendidas as vendas de programas piratas e desbloqueio de chaves de programas para mascarar a pirataria;
- *Lammer*: É o principiante, o qual está em busca de informações de como se tornar um *hacker*;
- *Script Kiddie:* é um usuário com poucos conhecimentos técnicos em informática, que invade sistemas utilizando ferramentas prontas que estão amplamente disponíveis na

Internet.

- *Wannabe*: já sabe combinar algumas técnicas de ataques prontas e consegue invadir sistemas frágeis;
- Larva: considerado como uma fase de transição entre *Wannabe* e *Hacker*, pois já consegue desenvolver suas próprias técnicas de ataque e penetrar em sistemas de nível de segurança considerado médio;
- *Hacker*: possui profundos conhecimentos de programação e de sistemas operacionais, principalmente Unix e Linux. Tem conhecimento das falhas de segurança dos sistemas e está sempre em busca de novos desafios, mantendo-se fiel aos códigos éticos de sua comunidade;
- *Cracker*: apesar de muitos possuírem conhecimentos similares aos *hackers*, os *crackers* têm intenções criminosas e rompem a segurança de um sistema em busca de informações confidenciais, com o objetivo de causar danos ou obter vantagem pessoal;
- *Phreaker*: fanáticos por telefones e tudo que diz respeito a eles, mesclam conhecimentos de telefonia, eletrônica digital e informática. Têm objetivos de burlar sistemas e evitar cobranças de contas telefônicas. Associam o talento de manipular tecnologia telefônica ao computador para promover ataques, tendo sempre o cuidado de não serem identificados.
- Guru: o mestre dos *hackers*, a escala máxima de um superusuário com habilidades técnicas em todos segmentos.

Com a popularização de ferramentas que exploram falhas de segurança na Internet – geralmente falhas de configuração –, um "atacante" não precisa ter os conhecimentos de um *cracker* ou de um *hacker* para causar insegurança nas redes. Tal fato é confirmado ao analisarmos o tipo de "atacante" que tem causado danos com maior freqüência: o *script kiddie*, o qual detém apenas conhecimentos da utilização das ferramentas e, através destas, "varre" a rede em busca de presas fáceis, pois, geralmente, não está atrás de alvos pré-definidos, apenas busca vulnerabilidades que facilitem a invasão.

De fato, muitos casos de insegurança poderiam ser evitados, se os administradores dessas redes estivessem bem preparados, ou seja, com conhecimentos detalhados dos sistemas que rodam em suas redes.

2.2 DIFERENÇAS ENTRE *HACKER* E *CRACKER*

"A diferença básica é esta: hackers constróem coisas, crackers as destróem."

(RAYMOND, 2000*b*)

A palavra *hacker* assusta a maioria das pessoas desinformadas, pois julgam um *hacker* como sendo aquele indivíduo que invade os computadores, apaga os arquivos, toma o controle da máquina e, ainda, envia mensagens à vitima. Este conceito que a maioria das pessoas tem a respeito dos *hackers* foi criado, possivelmente, pela publicidade negativa gerada pelos filmes que tratam do assunto, como por exemplo: "Jogos de Guerra", e com o mais recente: "A rede e os *Hackers*". A partir destes filmes e das notícias que a mídia difunde, cria-se a muitas pessoas uma confusão de sentimentos antagônicos, como a aversão e, ao mesmo tempo, a admiração aos piratas cibernéticos.

Tanto os *hackers* como os *crackers* são detentores de grandes conhecimentos em sistemas operacionais e de redes; ambos fazem uso do conhecimento adquirido para penetrar em sistemas, porém, suas ideologias são bem distantes. A mídia, em geral, utiliza os termos como sinônimos, talvez por desconhecer ou ignorar as diferenças existentes, como se os termos "detetive" e "bandido" também fossem sinônimos.

Os *hackers* direcionam seu potencial para construir, não destróem ou roubam dados de forma intencional, compartilham informações deixando rastros de passagem abertas para que administradores de rede possam fazer correções; eles têm como objetivo aprender mais, pois são autodidatas e gostam de desafios. Já os *Crackers* são os indivíduos que utilizam suas habilidades em benefício próprio, não importando quantos ou quais prejuízos causem; são elementos perigosos e prepotentes, deixam mensagens do tipo: "Eu sou o melhor! Apaguei e escapei!". (ANONYMOUS, 1998).

GOMES (2000), especialista em criminalidade cibernética, afirma que os alvos preferidos dos *crackers* são as redes encontradas em empresas de pequeno/médio porte ou escritórios particulares, pois empresas maiores costumam ter tecnologias mais sofisticadas na área de segurança. Mas também existem *crackers* que se arriscam, mediante recompensa, fazendo espionagem em empresas maiores.

Dentre os *crackers* que têm seu nome lembrado com maior freqüência, estão o Kevin Mitnick, John Drapper e Robert T. Morris. Mitnick ganhou o título de "o criminoso

de computador mais procurado dos EUA" graças à sua extensa folha corrida. Drapper, o Captain Crunch, modificava um apito (que vinha como brinde em uma caixa de cereais) para gerar um tom de 2600 Hz a fim de fazer ligações telefônicas gratuitas. Morris foi responsável pelo Verme da Internet, que paralisou cerca de 6000 máquinas na Internet em 1988.

Assim como outros *crackers* da velha guarda, relata FRANCO (1999), Mitnick (auto-apelidado de Condor) começou sua carreira como um *phreaker*, violando a segurança de centrais telefônicas. Mitnick tirava proveito dessa habilidade, não somente para fazer ligações telefônicas sem pagar, mas, principalmente, para esconder a origem de seus ataques a computadores de diversas instituições nos EUA.

Após envolver-se em uma longa seqüência de eventos, e passar um bom tempo na cadeia, Mitnick cometeu seu último e grande erro: decidiu invadir os computadores de Tsutomo Shimomura no dia de Natal de 1995, aparentemente atrás de um sofware para telefones celulares que Shimomura estava desenvolvendo. Shimomura, um pesquisador de renome do Centro de Computação de San Diego (Califórnia), seguiu rastros de Mitnick que, em menos de 2 meses, foi localizado e preso pelo FBI. (FRANCO, 1999)

Os *hackers* de verdade são autodidatas, conhecem profundamente hardware, diversas linguagens de programação, vários sistemas operacionais, redes e, principalmente, os protocolos.

Para FRANCO (1999), assim como Richard Stallman, Linus Torvalds, Ken Thompson e Larry Wall, são alguns dos *hackers* respeitosamente lembrados por aqueles que desejam tornar-se *expert* em informática. Todos eles possuem, além de um profundo conhecimento e uma extensa experiência na ciência (ou arte?) de escrever programas de computador, um desejo irrefreável de explorar novas fronteiras de conhecimento.

De forma breve, ressaltemos que Stallman é autor do famoso editor GNU Emacs. Torvalds é o pai do Linux. Thompson é co-autor (em parceria com Dennis Ritchie) do Unix. Wall é criador da linguagem Perl. (FRANCO, 1999)

Segundo VASCONCELLOS (1999), os *crackers* não são bem vistos pelos *hackers*, pois sabem que a má fama que carregam está diretamente relacionada com os ataques que os *crackers* realizam; sendo este um dos motivos pelos quais os *hackers* não gostam de se expor.

No item 2.1, pode-se verificar que existem diversos termos para classificar invasores, porém, conforme destaca REZENDE (2000), o argumento da simplicidade para

utilização do termo *hacker* de forma distorcida, banaliza a questão moral e jurídica do dolo. Traduções simplificadas contribuem para dificultar a compreensão coletiva das novas dinâmicas sociais postas em marcha com os novos meios de comunicação virtual: a Internet.

Os *hackers* não podem ser coletivamente classificados como criminosos indesejáveis, pois desempenham um papel semelhante ao da evolução da natureza, descobrindo falhas de segurança nos softwares em uso pelo mundo, e reportando suas descobertas aos desenvolvedores, às vezes sugerindo estratégias de reparo. São também os grandes responsáveis pelo enorme patrimônio intelectual em software livre que circula pelo mundo hoje, e do qual todos, engajados na revolução digital, estão se beneficiando. Chamar a estes de "*hackers* éticos", como fazem alguns, para atenuar o erro de tradução, apenas agrava tal confusão, pois assume a desonestidade como condição preliminar e natural à ação associada. (REZENDE, 2000)

2.3 O SIGNIFICADO DO TERMO *HACKER*

De acordo com RAYMOND (1999), o surgimento do termo *hacker* que conhecemos hoje, pode ser convenientemente datada em 1961, na cultura de computadores do MIT (*Massachussetts Institute of Technology*), quando este adquiriu seu primeiro PDP-1. O comitê de *Signals and Power* do *Tech Model Railroad Club* do MIT adotou a máquina como seu brinquedo tecnológico favorito, produzindo ferramentas de programação, gírias e cultura em torno do assunto, que permanecem até hoje. Naquela época, *hacker* era o termo usado pelos estudantes do MIT para designar aqueles que "fuçavam" nos computadores da Universidade além dos limites de uso, chamados vulgarmente de "ratos de laboratório". Antes desta data, tais programadores eram conhecidos por "programadores verdadeiros" (*real programmers*).

BRUNVAND (1996) afirma que o termo *hacker* era utilizado para denominar qualquer pessoa que detinha um grande conhecimento em um determinado assunto, isto antes mesmo do termo ser inserido na informática, ou seja, era utilizado em qualquer área.

Desde a época em que o termo *hacker* foi inserido no mundo digital, diversos

autores têm buscado traduzi-lo. Nas obras JARGON FILE[1] (1995), RAYMOND (2000*a*), CUNHA (2000), NETO (2000), além de outras obras especializadas, como tradução mais referenciada encontra-se a de "fuçador", do verbo *to hack* (fuçar), ou seja, não significa piratear, vandalizar ou vender serviços criminosos, sentido este que tem sido utilizado massificamente pela mídia.

Casacuberta, responsável pelo dicionário de CiberDireitos, conceitua o termo da seguinte forma:

> "[...] a princípio, o termo (*hacker*) vem do verbo inglês "hack", que é usado normalmente no contexto dos lenhadores, no sentido de cortar alguma coisa em pedaços ou no sentido de dar pontapés. Segundo a lenda, o primeiro uso não "tradicional" do termo se deveu a alguém que sabia dar o pontapé ("hack") no ponto exato de uma máquina de refrigerantes para dessa forma conseguir uma lata (ou garrafa) gratuitamente. Seja nesse sentido, seja no sentido de cortar algo em pedaços, o certo é que é o primeiro uso genuíno de *hackear* no mundo da informática foi de alguém que conhecia de modo muito detalhado um sistema operacional (havia "cortado-o" em pedaços, por assim dizer) a ponto de poder obter desse o que quisesse (como o senhor da lenda urbana a respeito da máquina de refrigerantes). Deste modo, originalmente, um *hacker* é simplesmente alguém que conhece os sistemas operacionais (e, logo, os computadores) como a palma de sua mão, uma vez que, em princípio, para poder entrar num computador sem permissão, é imprescindível que se tenha grandes conhecimentos (ainda que não seja necessariamente essa a realidade). Rapidamente o termo se difundiu com uma nova acepção: a de intruso ou violador informático que acessa o controle de uma máquina na rede, sem permissão".

LEON (2000) atribui o início da utilização do termo *hacker* na área tecnológica, derivada dos *phreakers* que tiveram sua origem nos primórdios da invenção do telefone. Essa técnica foi automatizada e desencadeou um crescimento considerável de mecanismos e meios de manipulação do sistema telefônico americano. Enquanto *phreakers* espalhavam-se, estudantes do MIT passavam horas nas salas de informática em busca de conhecimentos sobre os caríssimos sistemas de computadores, querendo aprender independente dos meios utilizados. Durante a década de 80, houve muitas perseguições a *hacker-phreaker*, a ponto de criarem leis rigorosas contra essas atividades.

A relação do *hacker* e do *phreaker* vai mais além, no início os *phreakers* lutavam contra as grandes companhias telefônicas, as grandes corporações e os governos; não

---

[1] Em http://www.tuxedo.org/jargon (site do Jargon File), diversas definições do termo estão listadas, a maioria delas diz respeito à aptidão técnica e prazer em resolver problemas e superar limites.

possuíam nacionalidade e não defendiam uma etnia, eram livres. Invadiam sistemas telefônicos, ensinavam truques para fazer ligações gratuitas, etc. Os *hackers* trilharam o mesmo caminho, unindo forças e lutando pelos mesmos ideais, desafiando os mesmos inimigos e divulgando o que consideravam como maior bandeira: **a liberdade de informação**.

Para RAYMOND (2000*b*), há pessoas que aplicam a atitude *hacker* em outras coisas, como eletrônica ou música – na verdade, você pode encontrá-la nos níveis mais altos de qualquer ciência ou arte. Os *hackers* da informática entram nos sistemas sem a intenção de prejudicar, apenas deixam mensagens para provar que são capazes de invadi-lo. Em contrapartida, existem os que buscam lucros em benefício próprio, estes são os *crackers*, buscam dinheiro, ações, seguros de vida, números de cartões de crédito, registros de propriedades, tudo que representa valor está registrado, normalmente, num computador. O sonho do criminoso moderno é justamente entrar dentro desses valiosos arquivos e transferir as informações para o seu nome.

Embora haja um conflito entre os diversos autores ao tentarem, historicamente, resgatar a origem da palavra, uma coisa pode-se perceber: o termo *hacker*, em sua originalidade, não é sinônimo de vandalismo ou crime, tradução que, conforme discutido no item 2.2, tem sido amplamente divulgada.

2.4 HISTÓRIA DOS *HACKERS*

Existe uma comunidade, uma cultura compartilhada, de programadores *experts* e *gurus* de rede cuja história remonta a décadas atrás, desde os primeiros minicomputadores de tempo compartilhado e os primeiros experimentos na ARPANET. Dentre os nomes bem lembrados, está o de Richard Stallman, fundador da Free Software Foundation (FSF), um exemplo genuíno de um *hacker*. De acordo com RAYMOND (1999), em 1971, Stallman ingressou no laboratório de Inteligência Artificial do MIT, junto com outros colaboradores, dentre eles Eric S. Raymond, onde criaram diversos programas de código aberto. A atitude em disponibilizar o código fonte para o usuário partiu justamente de Stallman, o qual coordena até hoje o projeto GNU (Gnu is Not UNIX), composta por uma comunidade de desenvolvedores espalhada pelo mundo inteiro, onde cada integrante contribui no desenvolvimento de softwares, sem esperar retorno financeiro.

GEEK (2000) e RAYMOND (1999) relatam que nos anos 50 à 70, os *hackers* ocupavam-se do desenvolvimento de compiladores, programas e *debugs* para área de informática, no momento, recém criada. Nessa época surgiu também o movimento ideológico e ético, porém ficou restrito a estudantes sonhadores, pois o acesso a computadores era restrito, devido ao alto custo.

Nos anos 80, *hackers* como Clive Sinclair e Steve Wozniak, revolucionaram a forma de trabalhar com a informática, difundindo o uso de computadores não apenas nas grandes empresas, mas também às pessoas comuns (GEEK, 2000). Também surgiram grandes projetos, como o GNU, criado em 1984, com filosofia de código aberto, ou seja, a disponibilização do código fonte ao usuário, filosofia esta que vem gradativamente conquistando mais adeptos.

Os anos 90 vêm marcado pela intensa preocupação em fortalecer a cultura digital, democratizando a Internet e a Informação; formando movimentos *hackers* mais organizados, onde conferências anuais de sucesso têm sido realizadas, como a *DefCon* nos EUA e a *Hack Meeting* na Itália.

De forma contrária à ideologia *hacker*, há alguns anos a mídia o vem denominando como um criminoso, uma pessoa má, capaz de destruir tudo que puder tocar, e que, supostamente, pode tocar em tudo o que quiser. Com isso, os próprios vilões do mundo virtual vêm se auto-designando *hackers*, prejudicando ainda mais a imagem que um dia pertenceu a uma pequena elite de pessoas extremamente inteligentes e capazes, que, na maioria das vezes, passavam todo seu tempo criando facilidades e aprimorando tecnologias, aperfeiçoando sistemas e fazendo a informática avançar em prol da comunidade.

2.5 A IDEOLOGIA DO *HACKER*

A cultura *hacker* é baseada na reputação; em quão interessantes são os problemas a serem resolvidos e suas soluções, onde somente iguais ou superiores tecnicamente são capazes de avaliar. Antropologistas definem a cultura como "cultura de doação", onde o status e reputação estão ligados ao fato de doar coisas, como sua criatividade e os resultados de sua habilidade, e não para dominar outras pessoas ou por possuir coisas que

os outros desejam.

Os *hackers* interagem com os demais dentro de uma estrutura anarquista, normalmente se associam em pequenos grupos, trocam informações entre si, mas agem de forma isolada. Os líderes exercem funções apenas de organizador, caso contrário violaria premissas de descentralização e antiautoritarismo.

Existe um código de responsabilidade não escrito que, de certa forma, molda os parâmetros do *modus operandi* dos *hackers*, criada no MIT nos anos 60 (GEEK, 2000):

- o acesso à Internet e aos computadores deve ser ilimitado e completo;
- todas informações devem ser compartilhadas, pois isso as tornam um bem potente para o crescimento da democracia e contra o controle político da elite tecnocrata;
- a invasão é considerada eticamente correta, caso seja com a intenção de explorar e se divertir, não podendo ser utilizada para furtos, destruição, vandalismo ou qualquer ação que cause dano ao sistema de informações;
- o computador e a Internet devem ser ferramentas para transformação de uma nova realidade.

Hackers cultivam a filosofia do código aberto, onde código de programas estão amplamente disponíveis para serem compartilhados, podendo qualquer programador utilizar ou reutilizar, ou seja, acrescentar a este código o que achar conveniente ou necessário.

A filosofia do código aberto teve como embrião Richard Stallman, o qual relata que um dos impasses que mais o inquietava com os *softwares* protegidos comercialmente, era quando alguém descobria um erro no software e a única coisa que poderia fazer era avisar ao fabricante sobre o erro e esperar a boa vontade da empresa em lançar uma próxima versão, e pior do que isso era pagar novamente pela versão corrigida. (INTERNET WORLD, 1998)

Indignado com essas atitudes, Stallman convidou a legião de amigos para juntos desenvolverem substitutos livres para os softwares comerciais, *drivers*, editores de texto, planilhas, etc. Nascendo então o projeto FSF – Free Software Foundation.

Muitas pessoas têm desejo de se tornar um *hacker*. Para tanto, um bom começo é colaborar no desenvolvimento de algum, dentre os vários softwares de código fonte aberto que são desenvolvidos de forma cooperativa na Internet. Antes disso, RAYMOND (2000*b*) recomenda que um candidato a *hacker* deve ter, essencialmente, as seguintes atitudes:

- adotar o Unix como sistema operacional, pois o DOS e Windows possuem códigos fontes fechados, além do Unix ser o sistema operacional original da Internet (hoje temos o Linux também);
- dedicar-se a programação – conhecer, principalmente, HTML, C, *Phython* e *Perl*, além de possuir capacidade de aprender uma nova linguagem em poucos dias;
- colaborar em projetos de código aberto com testes e depuração;
- praticar valores provindos da comunidade *hacker* de forma expressiva.

Além das atitudes ora apontadas, Raymond aponta alguns quesitos são essenciais para o indivíduo, como a curiosidade. Não se contentam ao ver algo funcionando pura e simplesmente, são tão curiosos que querem saber por que e como aquilo funciona. São bons leitores. *Hackers* resolvem problemas e constróem coisas, acreditam na liberdade e na ajuda mútua voluntária. Para ser aceito como um *hacker*, você tem que se comportar de acordo com essa atitude. E para se comportar de acordo com essa atitude, você tem que realmente acreditar nessa atitude. Neste contexto, o próximo item delineia a respeito do perfil dos *hackers*.

2.6 PERFIL DO *HACKER*

Os *hackers* têm uma característica muito própria. O perfil (*profile*) que será aqui relatado é a essência de uma reflexão dos comentários de uma pesquisa feita com centenas de correspondentes da Usenet, pelo JARGON FILE (1995):

- **Aparência Geral**

As características mais comuns são: inteligência, curiosidade mortal, e facilidade com abstrações intelectuais. Quase todos os *hackers* são estimulados pela novidade (especialmente a novidade intelectual). A maioria são relativamente individualistas e anti-conformistas.

Embora seja comum, alta inteligência não é condição *sine qua non* para ser *hacker*. A capacidade de se deixar mentalmente absorto, reter e fazer referências a detalhes "insignificantes", acreditando na experiência para dar contexto e significado. Uma pessoa

de inteligência analítica meramente mediana, que tenha esta característica, pode virar um bom *hacker*. Porém, um gênio criativo não será passado para trás por gente que devora conteúdo de manuais de referencia, toda semana.

Contrariamente ao estereótipo, *hackers* usualmente não são bitolados. Eles tendem a se interessar por qualquer assunto que pode prover estimulo intelectual, e podem, quase sempre, discutir sabiamente e até de forma interessante sobre assuntos obscuros – se conseguir fazê-los falar longe do alcance do computador.

Hackers são fascinados por controle, em uma forma que não tem nada a ver com as conotações autoritárias ou coercitivas do termo. Da mesma forma que uma criança se delicia em fazer modelos de trem irem para frente e para trás, *hackers* amam colocar máquinas complicadas como computadores fazerem 'coisas inúteis' para eles. Não curtem as tarefas chatas do cotidiano. Tendem a ser ordeiros com suas vidas intelectuais, e caóticos no resto. Seu código (de programa) será lindo, mas sua mesa de trabalho vai, provavelmente, ter muito lixo.

Hackers são, geralmente, pouco influenciados por recompensas como aprovação social ou dinheiro. Eles tendem a ser atraídos por brinquedos interessantes, e julgar o interesse do seu trabalho, ou outras atividades em termos de desafios oferecidos pelos brinquedos.

– **Forma de vestir**

Casual, vagamente pós-hippie; camiseta, *jeans*, tênis, sandálias ou pés descalços. Cabelo comprido, barbas e bigodes são comuns. Alta incidência de camisetas com *slogans* (tipo vá ao teatro, mas não me chame, etc.). Uma minoria substancial prefere roupas de *camping* – coturnos, jaquetas militares e etc. *Hackers* vestem para conforto, funcionalidade e problemas mínimos de manutenção, ao invés de aparência (alguns levam isso a sério e negligenciam higiene pessoal). Eles têm um índice de tolerância baixo a jaquetas e outras roupas de negócios; é até comum largarem um emprego ao invés de se conformar com uma roupa formal. *Hackers* do sexo feminino tendem a nunca usar maquiagem visível. A maioria não usa.

– **Hábitos de leitura**

Usualmente com grandes quantidades de ciência e ficção científica. *Hackers* normalmente tem uma capacidade de leitura de coisas tão diferentes que impressiona gente

de vários gêneros. Tem, porém, a tendência a não comentar muito isso. Muitos *hackers* gastam lendo o que outros gastam assistindo TV e sempre mantêm estantes e estantes de livros selecionados em casa.

- **Outros interesses**

Alguns *hobbies* são bastante partilhados e reconhecidos como tendo a ver com cultura: ficção científica, música, medievalismo (na forma ativa praticada por Grupos que se isolam da sociedade e organizações similares) xadrez, gamão, jogos de guerra e jogos intelectuais de todos os tipos. Radio Amadorismo. Alguns até são lingüistas ou fazem teatro.

- **Atividade física ou esportes**

Muitos não seguem nenhum esporte e são anti-exercício. Aqueles que fazem, não curtem bancar o espectador. Esporte seria algo que se faz, não algo que se vê os outros fazerem. Também evitam esportes de grupo como se fossem a "peste", com possível exceção de *voleyball*. Os esportes dessa "raça" são, quase sempre, os que envolvem competição individual e auto-superação, alguns envolvendo concentração e habilidade motora.

- **Educação**

Quase todos os *hackers* acima da adolescência são portadores de diploma ou educados até um nível equivalente. O *hacker* que aprendeu sozinho é sempre considerado (pelo menos para os outros *hackers*) como mais motivado, e pode ser mais respeitado que o seu equivalente com o canudo. As áreas incluem (alem da obvia ciência da computação e engenharia elétrica) física, matemática, lingüística e filosofia.

- **Coisas que os *hackers* detestam e evitam**

IBM *mainframes*. *Smurfs*, Duendes e outras formas de "gracinhas". Burocracias. Gente estúpida. Música fácil de ouvir. Televisão (exceto pelo velho *Star Trek* e os *Simpsons*). Ternos. Desonestidade. Incompetência. Chateação. COBOL. BASIC e correlatos.

- **Religião**

Agnóstica. Ateísta. Judeu não praticante. Neo-pagão. Mais comum, três ou quatro desses aspectos combinados. Crentes são raros, mas não desconhecidos. Mesmo os *hacker* que se identificam com uma religião, tendem a ser meio relaxados sobre isso. Hostis quanto a uma religião organizada em geral, e todas as formas de bate-papos religiosos.

- **Ética *hacker***

A ética *hacker* pode ser definida numa frase: **"Veja tudo, aprenda tudo, não toque em nada".**

Analisando o perfil *hacker* acima descrito, muitos podem interpretá-lo como um perfil de indivíduos que fazem parte de uma seita com ares macabros. De outro lado, verifica-se que tais anseios e tal filosofia são características de indivíduos ecléticos que buscam cultivar apenas o que há de melhor em suas atitudes.

3 AMEAÇAS E VULNERABILIDADES NAS REDES

Um usuário, ou um prestador de serviços Internet, estando conectado à rede, geralmente quer saber o quão vulnerável está a ataques. Uma máquina conectada na Internet, mesmo sem oferecer qualquer tipo de serviço, está vulnerável à pelo menos dois tipos de ataques:

1. pela própria vulnerabilidade do Sistema Operacional, devido a implementação de diversos protocolos da pilha TCP/IP;
2. pela conexão de rede, pois em qualquer tipo de conexão, seja Ethernet, via *modem*, etc., sempre haverá um meio de transmissão que tem uma capacidade limitada de banda. No entanto, esta banda (ou *link*) pode ser inundada com um número excessivo de solicitações de conexão. Um exemplo deste tipo de ataque é o DDoS (*Distributed Denial of Service*), o qual está descrito no capítulo 4, seção 4.5.6.

Mensurar quão vulnerável se está é, também, medir o quanto se tem de segurança. Para isso, da mesma forma que ocorre na segurança na vida real, vários são os fatores que estão inseridos e devem ser considerados, dentre os quais destaca-se o valor agregado ao bem. Assim, a vulnerabilidade a ataques será maior ou menor em função do valor associado ao bem que, geralmente, é a informação.

O fornecimento de algum tipo de serviço, seja WEB, Mail ou FTP, é um fator a ser considerado. Dependendo da forma como o serviço é configurado para ser disponibilizado, pode afetar o servidor como um todo, não apenas o serviço oferecido.

Outro fator a considerar é a pretensão do atacante. Os prejuízos causados por um *hacker* podem ser distintos dos prejuízos causados por um *cracker*, conforme mencionado no capítulo 2, seção 2.2.

A importância ou respaldo da empresa atacada é outro fator que contribui para aumentar a vulnerabilidade a ataques, principalmente se ocorrerem danos ou interrupções nas atividades da empresa. Portanto, mecanismos de segurança devem ser pensados e tratados de forma diferenciada.

Para BERNSTEIN et al. (1997), o projeto de um sistema seguro é tão importante

quanto se ter uma política de segurança eficaz. E destaca que, para medir a confiabilidade do sistema, as ameaças devem ser conhecidas.

As ameaças que exploram pontos fracos de um sistema, normalmente estão relacionadas à tecnologia ou a política de operação. Para tanto, BERNSTEIN et al. (1997) classifica as fraquezas relacionadas à tecnologia em duas categorias: as causadas por deficiências inerentes a mecanismos e produtos, ou seja, são falhas da tecnologia, e as resultantes de configurações incorretas em sistemas operacionais e programas aplicativos.

3.1 VULNERABILIDADES NAS TECNOLOGIAS

Nesta sessão serão abordadas as vulnerabilidades inerentes à tecnologia TCP/IP, bem como as ameaças existentes na rede local, a qual, inadvertidamente, muitas vezes não é considerada vulnerável.

3.1.1 Vulnerabilidade na Rede Local

Para se estar vulnerável, não é preciso oferecer qualquer tipo de serviço. A simples conexão da rede local apresenta certas vulnerabilidades inerentes à sua própria natureza. Ataques que exploram estas vulnerabilidades são normalmente restritos às redes locais, mas também podem ser usadas para sobrepujar protocolos de mais alto nível na hierarquia TCP/IP.

As deficiências da tecnologia Ethernet, tecnologia esta que constitui a maioria das redes locais, expõem ainda mais as fragilidades da Internet. Os principais problemas desta tecnologia estão relacionados com:

- facilidade de realizar grampo;
- falso mapeamento entre endereço de rede (IP) e endereço físico (ARP).

A realização de grampo é possível na tecnologia Ethernet pelo fato desta tratar o meio físico como compartilhado, sendo então possível configurar a interface de rede de uma máquina em modo "promiscuo", e assim esta máquina receber todos os quadros transmitidos na rede. Em sistemas UNIX, esta facilidade está disponível somente ao super-

usuário, mas em sistemas operacionais como DOS, Windows e MacOS, não existe nenhuma restrição de acesso à interface de rede, podendo assim um usuário comum, com más intenções, obter informações privilegiadas, como por exemplo senhas. Além disso, é possível usar a "promiscuidade" das redes Ethernet para implementar ataques mais sofisticados, tal qual "sequestro de sessão".

O falso mapeamento entre endereço IP (rede) e endereço Ethernet (físico) é assegurado pelo protocolo ARP (*Address Resolution Protocol*), protocolo que mapeia um endereço IP em endereço Ethernet, enviando para isso uma mensagem do tipo *broadcast*, perguntando qual o endereço Ethernet da interface que possui o endereço IP conhecido. A máquina que tiver o endereço IP procurado, ou alguma outra agindo em nome daquela, responde com o par: endereço IP – endereço Ethernet. Uma máquina mal intencionada pode então enviar respostas falsas, desviando todo o tráfego para si, tendo como objetivo personificar uma máquina, ou mais sutilmente, modificar os dados que estiverem sendo transmitidos entre duas outras máquinas. *Firewalls* e redes locais fisicamente inseguras devem mapear endereços físicos de maneira estática, evitando o uso do protocolo ARP. De qualquer forma, pessoas mal intencionadas tanto em conexões internas como externas, certamente irão explorar a insegurança das redes Ethernet.

3.1.2 Vulnerabilidade na Arquitetura TCP/IP

A Internet (família de protocolos TCP/IP) é um resultado de um projeto desenvolvido nos anos 70 pela Agência de Projetos de Pesquisa Avançada – ARPA (*Advanced Research Projects Agency*), do Departamento de Defesa dos Estados Unidos – DoD (*Department of Defense*). No entanto, a fragilidade encontrada na família TPC/IP advém do fato destes protocolos terem sido desenvolvidos para conectar a Arpanet (em tempos de guerra fria), onde o principal objetivo era o de manter a rede em funcionamento mesmo que parte dela fosse destruída. Para isso, o protocolo foi dotado da capacidade do emissor do pacote estabelecer a rota a qual deve percorrer; assim, mesmo que partes da rede não estejam ativas, caminhos alternativos podem ser estabelecidos para manter a comunicação. (COMER, 1991)

Grande parte dos problemas relacionados à segurança nas redes provém, principalmente, das deficiências inerentes aos protocolos de comunicação. Os protocolos

TCP (*Transmission Control Protocol*) e o IP (*Internet Protocol*) são os mais populares da hierarquia de protocolos Internet. BELLOVIN (1989) e COMER (1998) afirmam que o conjunto de protocolos TCP/IP é intrinsecamente inseguro, pois os dados que trafegam nos pacotes IP não passam por nenhum mecanismo que garanta integridade, privacidade, autenticidade e não-repúdio. Tais atributos dizem respeito à confiabilidade de um sistema. No entanto, mais recentemente surge uma nova necessidade: a disponibilidade dos sistemas, a qual apresenta maior dificuldade em ser mantida. Cada um destes termos é brevemente abordado na seqüência:

- Integridade: assegurar a veracidade e plenitude da informação;
- Privacidade: assegurar restrições no acesso à informação;
- Autenticidade: assegurar a identificação da autoria de determinada ação, sem equívocos;
- Não-repúdio: assegurar que uma mensagem que foi enviada ou recebida por uma entidade não possa ser negada no futuro;
- Disponibilidade: assegurar que a informação está disponível sempre que for necessário.

PALMIERI (2000) também observa que a arquitetura TCP/IP foi concebida para ser tolerante a falhas de hardware, e não a ataques intencionais. O principal risco é o fato da arquitetura permitir que usuários remotos acessem dados e arquivos de outros equipamentos conectados à rede, não protegendo a privacidade dos dados. Uma máquina pode monitorar todo o tráfego na rede a qual está conectada, independente de seu destino. Como a Internet inteira funciona como uma grande rede TCP/IP, é possível, caso precauções não sejam tomadas, ganhar acesso á qualquer máquina localizada em qualquer ponto do globo.

Neste contexto, o protocolo TCP/IP representa um risco de segurança simplesmente porque permite que usuários remotos acessem arquivos e dados de outros equipamentos. Quando foi projetado, as pessoas envolvidas não tinham idéia de quão utilizado ele seria; além disso, o custo elevado das memórias, assim como qualquer recurso, forçou o desenvolvimento de um protocolo *clean*, possibilitando então, que atacantes explorem suas características para conseguir acesso não-autorizado em sistemas de computação. Muitos desses acessos são derivados de falhas no mecanismo de

autenticação.

3.1.3 O protocolo IP

O protocolo IP (*Internet Protocol*) é como um campo intermediário comum para a Internet. Ele pode ter muitas camadas diferentes abaixo dele, como Ethernet, Token Ring, FDDI, PPP, dentre outros. De forma análoga, o IP pode ter muitos outros protocolos dispostos em camadas acima dele, sendo TCP, UDP e ICMP os mais comuns. Caso medidas preventivas não sejam tomadas, muitos problemas de segurança poderão advir deste protocolo.

Desde 1982, está em uso a versão quatro (IPv4) do protocolo IP. As crescentes necessidades de endereçamento e segurança impulsionaram o desenvolvimento da próxima versão, o IPv6 ou IPng (*next generation*), onde esforços tem sido despendidos para incluir mecanismos de segurança.

No protocolo IP, os dados são transmitidos em blocos denominados datagramas ou pacotes. Cada pacote contém um cabeçalho fixo (*header*) e um campo de dados (*contents*) (Figura 1.1). Do ponto de vista de vulnerabilidades, o cabeçalho IP contém quatro fragmentos de informações interessantes: O endereço de origem; endereço de destino; o Protocolo que está encapsulado no pacote IP[2]; e o campo de Opções IP (veja seção 3.1.3.1)

Bits

0 – 4	4 – 8	8 – 12	16 – 20	20 – 24	24 – 28	28 – 31
Versão	IHL	Tipo de serviço	Comprimento total			
Identificação			*Flags*	Deslocamento do Fragmento		
Tempo de vida		Protocolo	*Checksum* do cabeçalho			
Endereço de origem						
Endereço de destino						
Opções IP (se houver)					*Padding*	
Dados						

Figura 3.1: Formato do datagrama da Internet, a unidade básica de transferência em uma interligação em redes TCP/IP. Fonte: COMER (1998)

[2] O pacote IP pode conter um pacote TCP ou um pacote UDP. O UDP é uma alternativa de baixo *overhead* pelo fato de não oferecer nenhum garantia de confiabilidade (entrega, ordenação e não duplicação), garantias que são oferecidas pelo TCP. Além disso, pode ser um pacote ICMP.

A maioria das redes tem um limite sobre o comprimento máximo de um pacote, que é menor que o limite imposto pelo IP. Para lidar com esse conflito, o IP pode dividir um pacote muito grande para cruzar uma determinada rede em uma série de pacotes menores chamados fragmentos. A fragmentação de um pacote não muda sua estrutura na camada IP (os cabeçalhos são duplicados em cada fragmento), mas pode significar que o corpo contém apenas uma parte de um pacote na próxima camada (veja seção 3.1.4).

Depois que os pacotes alcançam seu destino, eles são remontados, para formar novamente um fluxo contínuo de dados; o processo de fragmentação e remontagem é transparente ao usuário. Pelo fato de geralmente existirem diversas rotas entre origem e destino, cada pacote pode percorrer uma rota diferente e levar um tempo distinto para alcançar seu destino. Contudo, não existe nenhum mecanismo implícito de segurança, além disso o conteúdo do pacote é visível em todos os computadores pelos quais o pacote circula. (WEBER, 1997)

3.1.3.1 Opções IP

Os cabeçalhos IP incluem um campo de opções, que em geral está vazio. Em seu projeto, o campo de opções do IP foi planejado como um local para informações especiais ou tratamento de instruções que não tinham um campo específico no cabeçalho.

A opção comumente utilizada para ataques é a rota de origem IP. Esta opção determina na origem a rota que um pacote deve tomar até seu destino. Esta opção é útil para contornar roteadores com tabelas de roteamento rompidas ou incorretas. Entretanto, o roteamento na origem tem sido usado por atacantes que estão querendo desviar os pacotes por caminhos conhecidos (a fim de obter o conteúdo do mesmo) ou então tentando desviar os caminhos que mantêm medidas de segurança.

3.1.4 Fragmentação IP

Uma das características do IP é sua capacidade de dividir um grande pacote que em pacotes menores (fragementos) que, caso contrário, não conseguiria atravessar algum link de rede (devido às limitações de tamanho de pacotes ao longo desse link).

Um dos problemas com a fragmentação é que apenas o primeiro fragmento conterá as informações de cabeçalho de protocolos de nível mais alto (como TCP) de que o

sistema de filtragem de pacotes necessita para decidir se deve ou não deixar passar o pacote completo (Fig.3.2). Originalmente, permitia-se a passagem de quaisquer fragmentos que não fossem os primeiros e fazia-se a filtragem apenas no primeiro fragmento, pois sem o mesmo, o sistema de destino não seria capaz de reunir o restante dos fragmentos no pacote original. Porém, se um atacante repassar todos os fragmentos que não sejam os primeiros, o host de destino guardará os fragmentos na memória por algum tempo, esperando para ver se recebe o pedaço que falta; isso permite o uso de pacotes fragmentados para um ataque de negação de serviço.

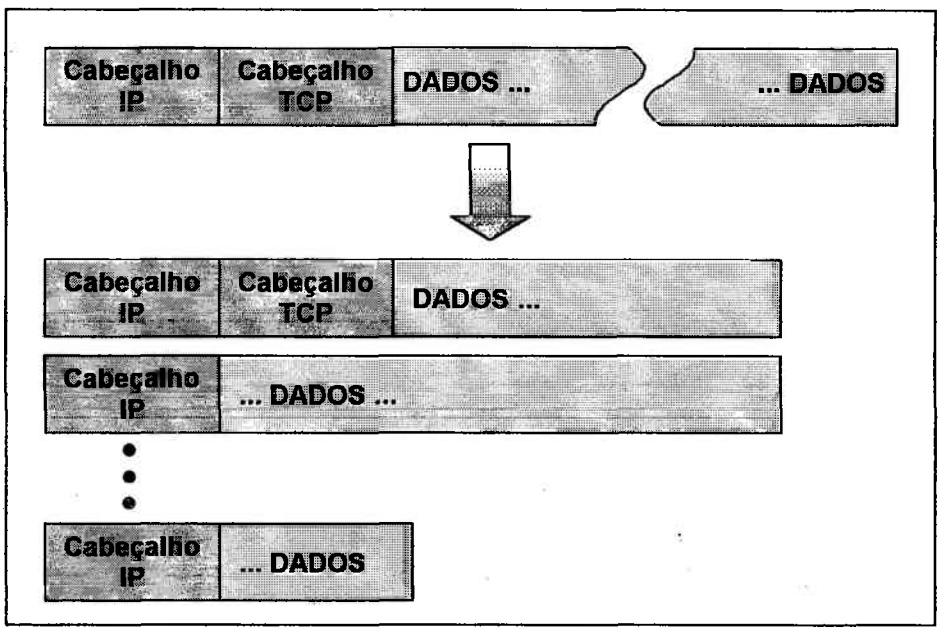

Figura 3.2: Fragmentação de dados. Fonte: ZWICKY et al. (2001)

Além disso, os atacantes podem usar pacotes fragmentados de modo especial para ocultar dados. Cada fragmento contém informações de onde começam e terminam os dados nele contidos. Normalmente, cada um começa após o término do último. Porém, um atacante pode construir pacotes onde os fragmentos realmente se superpõem e contém os mesmos endereços de dados.

ZWICKY et al. (2001) enumera três tipos de ataques que são explorados por meio de fragmentos superpostos:

- Ataques simples de negação de serviço contra hosts com respostas pobres a fragmentos superpostos;
- Ataques de ocultação de informações. Se um atacante souber que detectores de vírus,

sistemas de detecção de intrusos ou outros sistemas que prestam atenção ao conteúdo dos pacotes estão em uso e puder determinar que métodos de montagem os sistema usam para fragmentos superpostos, o atacante poderá construir fragmentos superpostos que irão ocultar o conteúdo dos sistemas de inspeção;

- Ataques que obtêm informações para porta que, caso contrário, estariam bloqueadas. Um atacante pode construir um pacote com cabeçalhos aceitáveis no primeiro fragmento, mas depois superpor o próximo fragmento de modo que ele também tenha cabeçalhos TCP em fragmentos que não sejam os primeiros, eles não o filtrarão, e os cabeçalhos não precisam ser aceitáveis. A figura 3.3 mostra fragmentos superpostos.

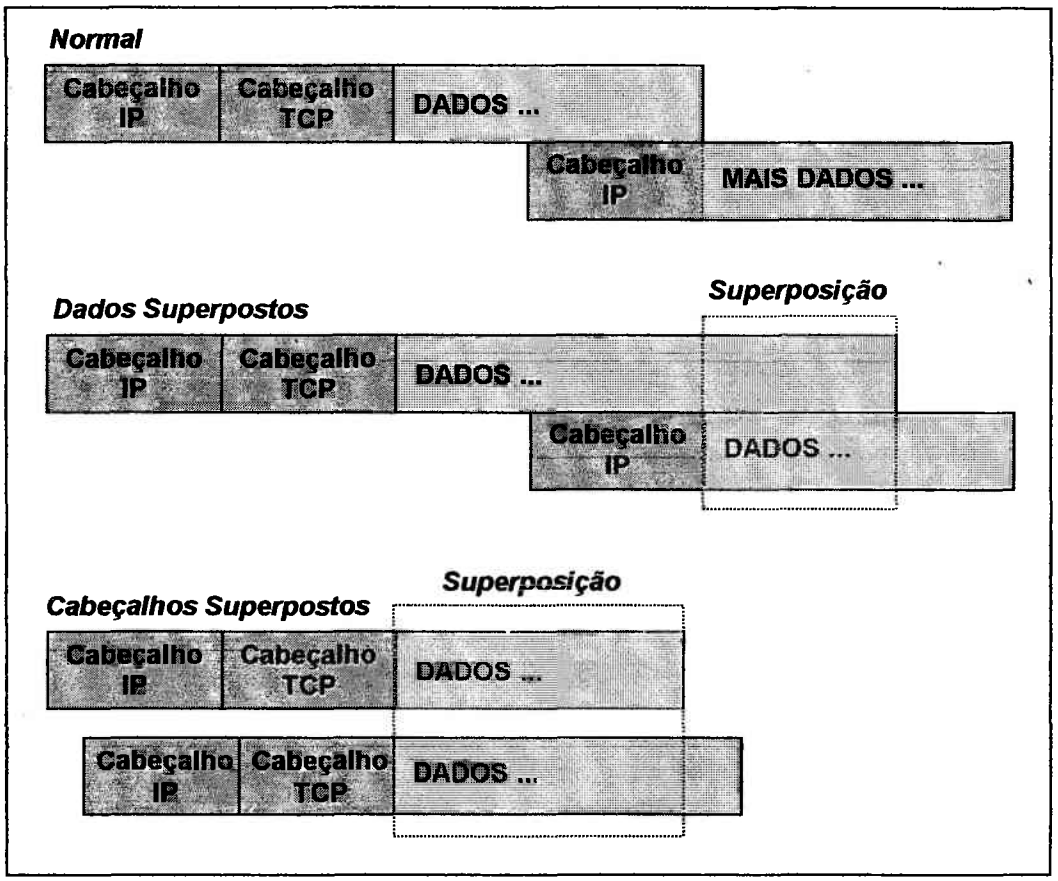

Figura 3.3: Fragmentos superpostos. Fonte: ZWICKY et al. (2001)

3.1.5 O protocolo TCP

O protocolo TCP fornece uma comunicação segura, ordenada e confiável entre duas máquinas. Cada conexão TCP utiliza duas portas (*ports*), identificadas por números de 16 bits. Assim, cada conexão é identificada por dois números de 32 bits (números IP de

origem e de destino) e dois números de 16 bits (as portas na origem e no destino).

Para bloquear uma conexão TCP, basta bloquear o primeiro pacote da conexão. O primeiro contém as informações corretas de inicialização da conexão e, sem esse primeiro pacote, quaisquer pacotes adicionais não serão remontados em um fluxo de dados pelo receptor, e a conexão nunca será feita. Esse primeiro pacote pode ser reconhecido porque o bit ACK em seu cabeçalho não está ativo; todos os outros pacotes na conexão, independente do sentido em que estão transitando, terão o bit ACK ativo. Outro bit, chamado SYN, também desempenha um papel de negociação da conexão; ele deve estar ativo no primeiro pacotes, mas não pode ser usado para identificar o primeiro pacote, pois ele também poderá estar ativo nos demais pacotes (ZWICKY, 2001)

3.1.5.1 Opções do TCP

O bit ACK é apenas uma das opções que podem ser definidas. Segue a lista inteira na ordem em que aparecem no cabeçalho:

- URG (urgente)
- ACK (confirmação)
- PSH (empurrão)
- RST (reinicialização)
- SYN (sincronizar)
- FIN (terminar)

URG e PSH são usados para identificar dados particularmente críticos; PSH informa ao receptor para parar a bufferização e oferecer os dados a algum programa, enquanto URG demarca dados que o transmissor considera de importância especial. Na prática, não há garantia que essas opções, caso utilizadas, serão executadas, pois o roteador, ou o *firewall* pode ignorá-las.

As opções RST e FIN são modos de fechar uma conexão. RST é um fechamento deselegante, enviado para indicar que algo saiu errado (por exemplo, não existe nenhum processo escutando na porta, ou parece haver algo estranho com o pacote que chegou). FIN é parte de um desligamento elegante, onde ambos extremos enviam FIN um ao outro para efetivar a desconexão.

Por fim, as opções ACK e SYN, as quais são freqüentemente lembradas quando o

tema é segurança, pelo fato de fazerem parte do estabelecimento de uma conexão TCP. Juntas, as opções ACK e SYN formam o *handshake* de três vias do TCP, assim denominado porque toma três pacotes para configurar uma conexão. A Figura 3.4 mostra como o ACK e SYN são definidos em pacotes que fazem parte de uma conexão de TCP.

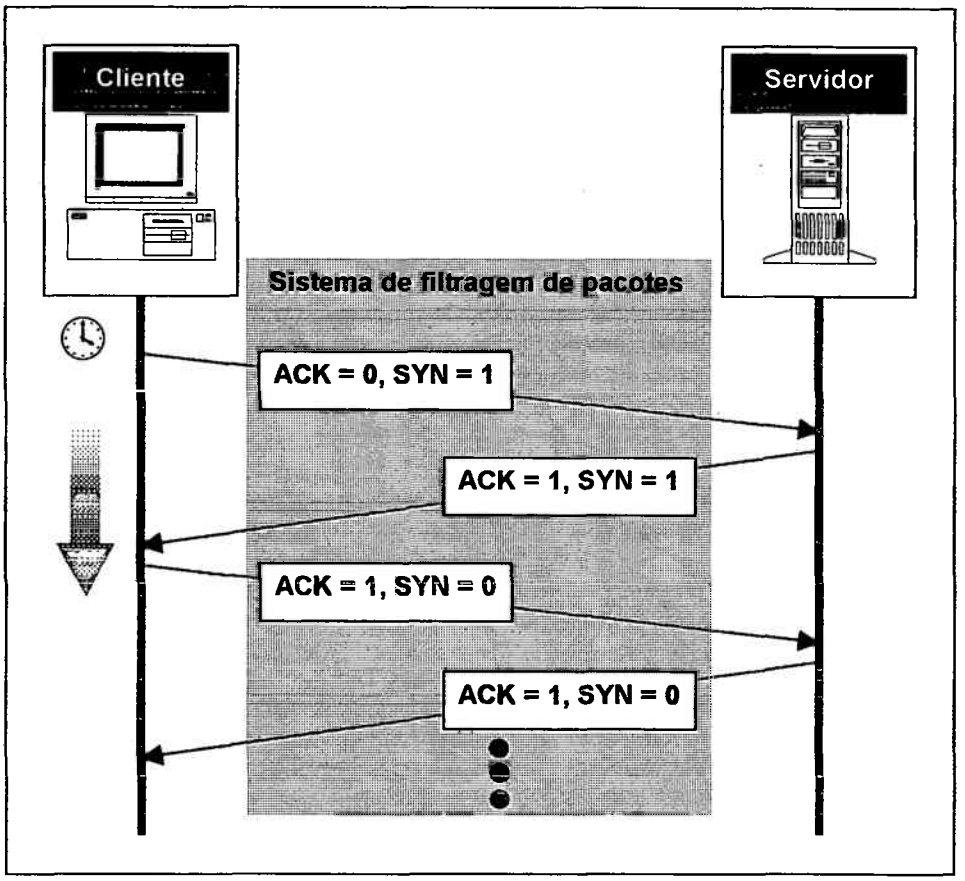

Figura 3.4: Conexão TCP requer um *handshake* de três vias. Fonte: ZWICKY et al. (2001)

Quando o cliente deseja iniciar uma nova conexão, ele envia uma pacote com ACK desligado (porque ele não é resposta a nada) e SYN ligado (a fim de dar ao próximo pacote um número de seqüência para confirmação). O servidor responde ao pedido enviando um pacote com ambos bits ligados, e o cliente responde agora com ACK ligado e SYN desligado. Todos os demais pacotes possuem ACK ligado.

O handshake de três vias é útil quando se deseja implementar *firewalls* baseadas em filtros de pacotes, mas também permitem a um intruso detectar o início de uma sessão e eventualmente capturar senhas ou até mesmo toda a sessão. Além disso, o servidor TCP tem um limite de requisições para conexão que ele pode ficar em aberto aguardando confirmação. No entanto, um falso cliente pode emitir pacotes de requisições, sem a intenção de completar o *handshake*, que excedam o limite de conexões reservadas para

aquele serviço do servidor, ficando este sem condições de receber conexões de clientes verdadeiros. (WEBER, 1997)

3.1.5.2 Número de seqüência do TCP

O TCP fornece uma garantia aos aplicativos de que eles sempre receberão dados na ordem correta. Porém, para implementar tal garantia, o TCP numera cada um dos pacotes, chamados de números de seqüência. No início de uma conexão, cada extremidade escolhe, de forma independente, um número inicial aleatório[3], e esse número é transmitido no *handshake* quando o SYN está ativado. Após o SYN, para cada pacote, o número é simplesmente incrementado de acordo com o número de bytes de dados no pacote. Se o primeiro número de seqüência é 200 e o primeiro pacote de dados contém 80 bytes de dados, ele terá um número de seqüência igual a 280. O ACK é acompanhado pelo número do próximo fragmento de dados esperado (o número de seqüência mais uma unidade, ou 281 neste caso). (ZWICKY, 2001)

O capítulo 4 – Modus Operandi, seção 4.5.1, detalha como são implementados ataques através dos número de seqüência.

3.1.6 O protocolo UDP

Os pacotes UDP são muitos semelhantes em estrutura a pacotes TCP. Um cabeçalho UDP contém número de portas UDP de origem e destino, exatamente como os números de portas TCP de origem e destino. Porém, um cabeçalho UDP não contém nenhum dos números sinalizadores ou números de seqüência que o TCP utiliza. No entanto, não existe nenhum modo de um roteador de filtragem de pacotes, simplesmente examinando o cabeçalho do pacote, perceber a entrada do pacote UDP. (ZWICKY, 2001)

Esta característica do protocolo UDP permite que ele seja utilizado para implementar diversos ataques, como a falsificação de endereços IP (IP *spoofing*).

---

[3] Algumas implementações TCP/IP fixam o número de seqüência, deixando a rede mais vulnerável.

3.1.7 O protocolo ICMP

O ICMP é usado para mensagens de *status* e controle do IP. Os pacotes ICMP são transportados no corpo de pacotes IP, da mesma forma que pacotes TCP e UDP. Os exemplos de mensagens ICMP incluem:

- Echo request (solicitação de eco)
- Echo response (resposta de eco)
- Time exceeded (tempo excedido)
- Destination unreachable (destino inacessível)
- Redirect (redirecionar)

O tipo específico de mensagem utilizada estabelece a interpretação do pacote ICMP. A mensagem *Destination unreachable* transporta códigos individuais que dizem o porquê o destino está inacessível. Uma destas mensagens é o código "*Fragmentation needed and Don't Frangment set*" usado para descoberta de caminho MTU. Porém, outros campos podem ser usados para varrer redes a fim de verificar quais hosts podem ser atacados.

A maioria dos pacotes ICMP tem pouca ou nenhuma informação significativa no corpo do pacote e, portanto, eles devem ser bem pequenos. Porém, ataques de negação de serviço usando pacotes ICMP superdimensionados (pacotes de eco, como o ping da morte) já foram explorados.

A mensagen *Redirect* também é utilizada pelos roteadores para informar à origem a existência de um caminho melhor, sugerindo assim uma nova rota. Através desta mensagem é possível inserir uma rota falsa para enviar os pacotes ao atacante.

As falhas de protocolos como RIP, DNS, ARP, dentre outros, serão detalhados no capítulo 4 (*Modus Operandi* – Ataques e contramedidas), pois, em sua maioria, a insegurança existente nestes protocolos, devem-se à utilização dos protocolos que estão no coração da arquitetura Internet (o TCP, UDP, IP e ICMP) ou às deficiências das próprias redes locais.

3.2 VULNERABILIDADE NA CONFIGURAÇÃO

Os problemas de má configuração geralmente estão associados ao fato do tamanho dos sistemas e sua grande complexidade de configuração.

Normalmente os pontos fracos na configuração dos sistemas estão em (BERNSTEIN et al., 2000):

- contas de usuários inseguras (como *logins* de convidados ou contas de usuários expiradas);
- contas de sistema com senhas originais conhecidas e não alteradas;
- serviços da Internet mal configurados;
- parâmetros básicos inseguros nos produtos.

Para uma correta configuração dos serviços é necessário entender seu funcionamento e estar ciente de seus pontos críticos. Entre todos os serviços suportados pelos protocolos TCP/IP, existem aqueles que devem merecer atenção especial por parte dos administradores, visto que são mais visíveis por usuários externos ao domínio e, portanto, mais expostos a ataques; como exemplo o DNS, FTP, HTTP e SMTP. Existem outros que são empregados somente no interior do domínio, sendo seu impacto na segurança global menor, porém, como freqüentemente os problemas de segurança são ocasionados por usuários confiáveis (funcionários, por exemplo), deve-se escolher cautelosamente os protocolos que poderão trafegar pela rede.

3.3 SERVIÇOS E SUAS VULNERABILIDADES.

3.3.1 Telnet

O telnet permite serviços de terminal virtual, ou seja, usuários podem acessar sistemas remotamente como se estivessem conectados diretamente a ele. Porém não apresenta interface gráfica, sendo a utilização em forma de linhas de comando. (GLOBAL,

2000b).

O propósito do protocolo de Telnet é permitir um método padrão de interfacear dispositivos terminais e processos orientados para terminal entre si. (ANONYMOUS, 1998). No que diz respeito a segurança, o Telnet apresenta falhas em conseqüência dos dados trafegarem sem nenhuma forma de cifra, inclusive as senhas são transferidas em forma de texto simples. (WEBER,1997; BERNSTEIN et al., 1997).

Desta forma, a utilização do Telnet sem a utilização de mecanismos extras (como criptografia), não é aconselhável, pois o fluxo de dados pode passar por inúmeras redes intermediárias não confiáveis.

3.3.2 FTP – *File Transfer Protocol*

O FTP é o serviço padrão da Internet para a transferências de arquivos de um sistema para outro. Tendo como objetivo a transferência de arquivos de modo rápido e eficaz; permitindo que usuários internos acessem arquivos remotos e também, usuários remotos acessem dados internos.

Acesso a arquivos externos não representa grandes riscos, exceto pela possibilidade de aquisição de arquivos com vírus. O problema está na abertura de acesso a dados internos. Existem dois tipos de acesso FTP : *user ftp* e *anonymous ftp.* O *user ftp* no estabelecimento de login solicita ao usuário identificação e senha, enquanto o *anonymous ftp* permite o acesso a dados em uma área restrita do sistema sem necessidade de identificação. (WEBER,1997).

A utilização do FTP pode ser considerada como um método seguro para transferência de arquivos, mas não dispões de criptografia na transferência de senhas, assim como o Telnet.

Alguns servidores FTP permitem acesso anônimo possibilitando a qualquer usuário o download de arquivos e dados contidos em sites da Internet, no servidor sem fornecer nenhuma forma de autenticação.

O sistema de arquivos de um servidor FTP é normalmente restrito a um ramo específico da árvore de diretórios, porém alguns servidores FTP anônimo permitem que o usuário navegue por toda estrutura da árvore de diretórios. Dessa forma um atacante tem acesso a dados sensíveis como `/etc/passwd`.

A decisão de instalação de um FTP anônimo deve levar em consideração pelo menos dois pontos vulneráveis intrínsecos: maior quantidade de vias de ataque; pois quanto mais serviços estiverem sendo executados, maiores as chances de existência de bug ou configurações incorretas e upload, o qual a permissão abre pontos para ataques que necessitam de instalação de arquivos nas máquinas de destino. Uma forma de limitar esse risco é não possibilitar que usuários leiam ou executem esses arquivos após a transferência (BERNSTEIN, 1997).

Em geral o FTP anônimo é um serviço útil que, sendo necessário, deve ser cuidadosamente configurado, como qualquer outro serviço.

3.3.3 Finger

O Finger é um serviço para obter informações sobre usuários na Internet. Através dele pode-se (BERNSTEIN, 1997):

- verificar se um endereço eletrônico está correto.
- obter o nome de um usuário, seu último logon, e em alguns casos, descobrir qual foi a última vez que leu ou recebeu mensagens.
- Obter outras informações pessoais que o usuário tenha deixado disponível.

Usuários mal intencionados podem utilizar essa lista de informações detalhadas de usuários, obtidas através do uso do Finger, para investirem em tentativas de adivinhação de senhas e ataques de engenharia social (LOPES, 2000).

Como o Finger possibilita a verificação de dados sobre o último logon, atacantes podem identificar contas inativas ou pouco utilizadas. Estas são alvos considerados fáceis, pois normalmente passam desapercebidas por um usuário final ou um administrador de sistemas.

3.3.4 TFTP

O TFTP (Trivial File Transfer Protocol) é semelhante ao FTP e visa, principalmente, a transferência de arquivos de configuração durante a inicialização do sistema. Permite que hosts sem discos sejam inicializados a partir da rede, devido sua

implementação ser simples e caber em uma memória ROM, o que o torna ideal para boot remoto (ANONYMOUS, 2000).

O protocolo TFTP é baseado em UDP e ouve na porta 69. Na questão de segurança é deficiente; se um servidor TFTP apresentar uma configuração incorreta, um atacante pode obter a transferência de uma cópia do arquivo /etc/passwd para seu sistema, com facilidade. O servidor deve ser configurado de modo a restringir o acesso a diretórios específicos, para impedir que atacantes consigam extrair arquivos sensíveis da configuração do sistema.

Versões atuais do protocolo permitem implementação de limitações a determinados diretórios.

3.3.5 Gopher

O Gopher é um sistema de procura e transferência de informações orientado a títulos de documentos, que permite ao usuário localizar artigos relacionados a busca na Internet (ANONYMOUS, 1998).

A interface de Gopher foi projetada para assemelhar-se com sistemas de arquivos, pois é um bom modelo para localização de documentos e serviços, no qual o software de cliente Gopher apresenta uma hierarquia de itens e diretórios (GLOBAL, 2000*a*).

O modelo Gopher é baseado em cliente-servidor, sendo que o usuário não necessita informar seu logon em nenhum momento. O cliente envia uma mensagem para o servidor Gopher, que retorna os documentos disponíveis.

Um dos pontos de vulnerabilidade encontrado em aproximadamente 10% dos servidores Gopher é um bug antigo que permite a qualquer usuário ganhar o acesso irrestrito a conta. (MCCLURE, SCAMBRAY e KURTZ, 2000).

3.3.6 Correio Eletrônico

O correio eletrônico tem sido um dos serviços da Internet mais utilizados atualmente. Inicialmente foi projetado para servir como uma ferramenta de comunicação entre pessoas, mas diversas empresas têm direcionado seu uso como parte de importantes processos e como forma de comunicação regular (WEBER,1997).

Os serviços de correio eletrônico se baseiam no SMTP (Simple Mail Transfer Protocol), o qual não apresenta problemas de segurança, porém as implementações de seus servidores, sim, como o sendmail.

O servidor SMTP mais utilizado em sistemas Unix é o *sendmail*, considerado um dos programas mais perigosos em uso. Ele é extensível, altamente configurável e complexo. Diversas vulnerabilidades relacionadas a estouro de buffer remoto e ataques de validação de entrada já foram identificadas no programa; além de ser possível explorar suas funcionalidades para ganhar acesso privilegiado (MCCLURE, SCAMBRAY e KURTZ, 2000).

Segundo BERNSTEIN (1997) as maiores desvantagens do SMTP refletem os problemas comuns na maioria dos serviços da Internet, ou seja, a falta de confiabilidade – o correio eletrônico é vulnerável a interceptações, não sendo, portanto, um meio de comunicação adequado a dados confidenciais; a falta de autenticidade – não existe a garantia de que remetente ou destinatário das mensagens estejam fornecendo identidades corretas; e, por fim, a falta de integridade, pois não há como garantir que uma mensagem recebida seja a mesma que foi enviada.

4 *MODUS OPERANDI* – ATAQUES E CONTRAMEDIDAS

Ataque é qualquer procedimento de tentativa de acesso não autorizado, que tem por finalidade esconder, danificar, incapacitar dados e comprometer serviços ou o funcionamento de um sistema.

Para STALLINGS (1999) ataques de segurança de um sistema de computador ou rede é caracterizado pela visualização da função de um sistema de computador como provedor de informação, onde há um fluxo de informação, como um arquivo ou uma região da memória principal, da origem para um destino, tal como outro arquivo ou um usuário. Este fluxo normal é descrito na primeira parte da figura 4.1. As demais partes da figura apresentam as possíveis formas de ataques, a saber:

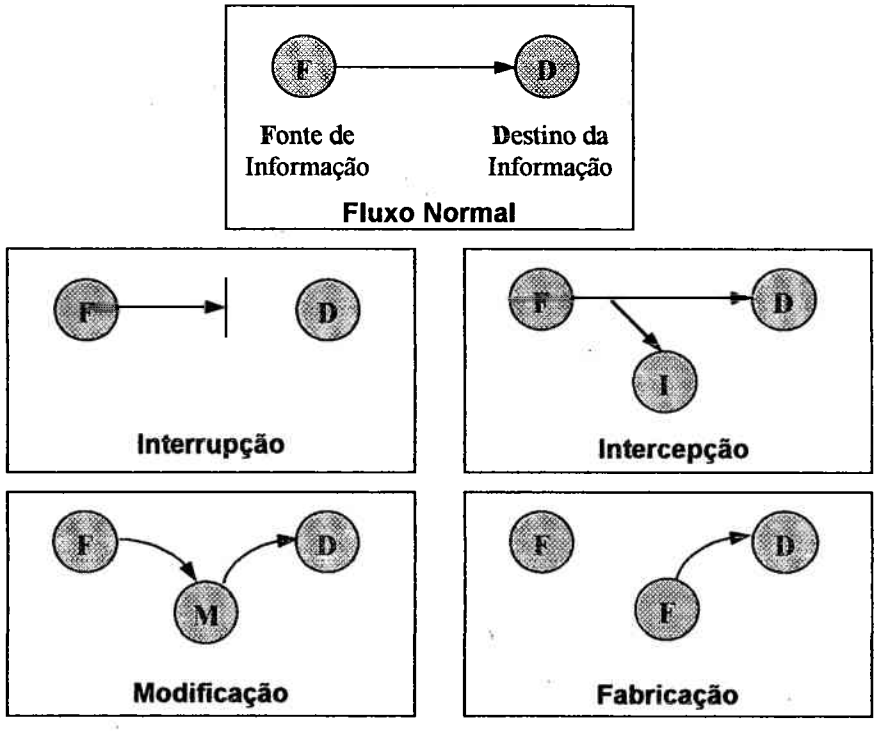

Figura 4.1: Ameaças de Segurança. Fonte: STALLINGS (1999)

- **Interrupção**: A parte ativa de um sistema é destruída ou torna-se indisponível. Este tipo de ataque ameaça a **disponibilidade** do sistema. Exemplos incluem destruição de disco rígido, corte de uma linha de comunicação ou a desativação do sistema de gerenciamento de arquivos.

- **Intercepção**: Uma parte não autorizada ganha o acesso do sistema. Este tipo de ataque ameaça a **confiabilidade** do sistema. A parte não autorizada pode ser uma pessoa, um programa ou um computador. Exemplos incluem grampos para capturar dados em uma rede e a cópia ilícita de arquivos e programas.
- **Modificação**: Uma parte não autorizada não somente ganha acesso para leitura dos dados, mas também os altera. Este é um ataque que ameaça a **integridade** dos dados. Exemplos incluem a mudança de valores em registros de arquivos de dados, alteração de programas para que executem de forma anormal e modificação do conteúdo das mensagens que estão sendo enviadas pela rede.
- **Fabricação**: Uma parte não autorizada envia objetos falsos no sistema. Este tipo de ataque é uma ameaça a **autenticidade**. Exemplos incluem a inserção de mensagens espúrias na rede ou a adição de registros em arquivos.

De acordo com STALLINGS (1999), tais ataques têm características ativas e passivas. Os ataques de interrupção, modificação e fabricação são caracterizados como ativos, enquanto que o ataque de intercepção é caracterizado como passivo, pois apenas observa as informações que trafegam no sistema. No entanto, um ataque passivo pode permitir que um ataque ativo ocorra, pois na observação dos dados, senhas de acesso, por exemplo, podem ser obtidas.

Vários ataques foram criados explorando as deficiências existentes na arquitetura TCP/IP. O grande problema desta arquitetura diz respeito a atenticação, onde os serviços e protocolos baseiam-se no endereço IP para tal autenticação. No entanto, sabe-se que pacotes IP podem ser forjados, ou seja, o cabeçalho IP pode ser alterado da forma que o atacante quiser. Desta forma é possível para um atacante personificar qualquer máquina na rede. Esta técnica é chamada de IP *spoofing* e serve como base para diversos outros tipos de ataques uma vez que, entre outras coisas, ela permite esconder a origem do ataque. Somam-se a estes ataques outros que usam bugs de segurança nas implementações dos serviços, protocolos TCP/IP e falhas nos softwares em geral. Analisaremos neste capítulo os ataques conhecidos e explorados.

4.1 VÍRUS

Vírus são programas ou comandos que podem ter origem casual (falhas internas de um programa) ou intencional. Associados a softwares podem se multiplicar a cada vez que estes são executados, alterando clandestinamente softwares instalados em uma máquina e causar danos.

A denominação vírus faz analogia ao vírus biológico, pois possuem fases semelhantes: multiplicam-se, necessitam de hospedeiro, aguardam o momento para ataque e tentam não serem detectados para manterem-se vivos. Stephen Hawhing se referiu ao vírus de computador como uma forma de vida construída pelo homem.

Os vírus são ativados quando é executado um programa infectado, fazendo que o código do vírus também seja executado e tente infectar outros programas do computador. Usualmente é um programa muito pequeno, utilizando dessa característica para passar "despercebido"; o pequeno tamanho, ajuda na sua dissimulação, o que pode potencializar sua atuação. Alguns dos mais prejudiciais vírus são extremamente pequenos. Por exemplo: 191, 200, 403, 512 levam tais nomes pelo fato de serem estes os respectivos tamanhos em bytes! Apesar do tamanho, são perigosos e provocam danos à estrutura lógica de discos, com perda de informações. Na mesma situação encontra-se o vírus WWT (67bytes), por outro lado, o vírus WHALE (baleia) – dependendo da versão – o tamanho dele pode ser de 9.216 bytes a 11.442 bytes; sendo que ele apenas corrompe arquivos, não afetando a estrutura lógica do disco.

Normalmente um vírus tem um período de "incubação" aonde, durante este período, vai se auto-duplicando, procurando ampliar o número de programas ou disco contaminados. Após o período de "incubação", o vírus começa sua atividade realmente danosa. O fato que determina o final do período de incubação, que pode ser tanto a ocorrência de uma data ou período específico, um determinado número de contaminações efetuadas, um determinado número de teclas acionadas, etc. O período de incubação varia para cada vírus, dependendo a forma como foi programado. O vírus Jerusalém ou 6° feira 13, como a própria data indica, deleta os programas contaminados que forem executados numa 6° feira 13. Antes dessa data, porém, ele vai apenas contaminando os programas.

Existem vírus em que o período de incubação é inexistente, como o Stoned, que apresenta a mensagem "*Your PC is now stoned*". Este vírus, ao mesmo tempo em que se manifesta, atacando o setor de boot ou programas, procura contaminar outros discos e

programas.

Porém os vírus ganharam um grande poder de devastação com a velocidade de propagação que a Internet proporciona. Um desses exemplos ocorridos este ano, foi o ILOVEYOU, desenvolvido por um filipino, atingiu milhões de pessoas no mundo todo, e ocasionou prejuízos na ordem de bilhões de dólares.

Na verdade o ILOVEYOU não é um vírus, e sim um worm, pois é um script em Visual Basic, ou seja, um programa pequeno que não precisa ser compilado. O conteúdo do VBScript do worm espalha-se primariamente da mesma forma que o Melissa, se auto-enviando por email para os registos do livro de endereços da vítima. Contudo, apresenta uma vertente mais perigosa, pois na tentativa de fazer carregar e executar o programa WIN-BUGSFIX.exe, de acordo com alguns relatórios, tenta roubar as "passwords" em cache da vítima. (VIANA, 2000)

Figura 4.2: Dados e Modus operandi do worm ILOVEYOU. Fonte: VIANA (2000).

O worm infiltrava as organizações por email (um canal geralmente aberto). Ao não espalhar o trojan WIN-BUGSFIX.exe diretamente, o autor era capaz de passar os mecanismos de defesa que verificam attachments executáveis na entrada de mensagens de email. Uma vez executado, o VBScript usava um canal para fora (geralmente aberto) para se ligar à sua "base", carregando o trojan WIN-BUGSFIX.exe através de um pedido que se parecia como uma ligação típica de navegação pela Web. Por causa do nome do executável e porque era automaticamente carregado pelo Internet Explorer na inicialização do programa, era provável que os usuários executassem o trojan, pensando ser parte da funcionalidade de auto-atualização do Internet Explorer.

O ILOVEYOU foi efetivo ao mostrar as ameaças de agentes maliciosos. Contudo, tecnologia e potencial destrutivo, são apenas a ponta do iceberg.

4.2 CAVALOS DE TRÓIA

O termo Cavalo de Tróia (*Trojan Horses*) está relacionado diretamente com a Guerra de Tróia, na qual gegros presentearam os troianos com um grande cavalo de madeira, porém os troianos não sabiam que dentro dele havia muitos soldados gregos, que esperaram apenas adentrar na fortaleza para então saírem do cavalo e surpreenderem os troianos, vencendo assim a batalha.

Trojan Horses são programas que ocultam seus objetivos reais sob a forma de programas comuns. Tais programas apresentam-se de diferentes formas, como jogo, fotos ou texto, aparentemente sem qualquer ameaça, mas trazem ocultamente consigo códigos maliciosos, que contém funções não autorizadas e normalmente indesejáveis (ANONYMOUS, 2000)

Um trojan possui várias funções, porém no que diz respeito a segurança na Internet, ele pode realizar e conciliar funções que revelem informações vitais e privilegiadas sobre determinado sistema, podendo comprometê-lo.

Algumas classes de trojan como PC Cyborg Aids Trojan e AOL causam danos a vítima, realizando tarefas como criptografia e formatação de disco.

Apesar de muitos considerarem trojans uma espécie de vírus, eles apresentam particularidades (SPLITNET, 2000).

- não possuem auto_replicação;

- são programas autônomos, não necessitam infectar outras entidades (programas, setores de boot) para serem executados (ainda que possam estar agregados a eles);
- não existe uma preocupação primordial de auto-preservação, uma vez que não visam a auto-replicação.

Trojan horses não são comuns por causa da sua limitada capacidade de disseminação. Como não se replicam, costumam permanecer indefinidamente no PC ou se auto-destruir juntamente com os dados que visa apagar ou corromper. A sua propagação se dá apenas por meio canais de distribuição como Internet, e normalmente são colocados à disposição como um programa muito útil e até mesmo milagroso, assim, voluntariamente são copiados por usuários enganados quanto aos reais efeitos do programa, e a partir deles eventualmente propagam para outros usuários.

Os trojans mais conhecidos e utilizados atualmente são o Net Bus e Back Orifice, os quais são analisados na seqüência.

4.2.1 Net Bus

O Net Bus foi criado pelo sueco Carl Friedrick Neikter e disponibilizado na Internet em março de 1998, com o objetivo de utilizá-lo como ferramente de gerencia remota. Para tanto, o Net Bus é constituído por duas partes: um cliente (`netbus.exe`) e o servidor (`patch.exe`) (UNICOR, 2000).

A utilização do Net Bus tem sido voltada principalmente por piratas cibernéticos com a intenção de obterem controle de outras máquinas sem serem notados, e isso tem tornado-o um dos maiores trojans na Internet.

O Net Bus permite amplo controle sobre o servidor, apresenta as seguintes características (BIDO, 2000):

- é auto-instalável e auto-executável;
- permite executar/terminar qualquer aplicativo;
- permite desligar o servidor e desconectar usuários;
- consegue capturar tela do servidor;
- não aparece no Task list do Windows.

O Net Bus utiliza TCP/IP para estabelecer o envio/recebimento de pacote e dados,

permanecendo ativo nas portas 12345 e 12346 aguardando por conexões clientes (BLACK, 2000).

O ataque funciona da seguinte forma:

1. O arquivo patch.exe é enviado à vítima, com o nome alterado, passando-se por outro software, e aguarda até que seja executado para realizar sua instalação, mas alguns fazem uso de instaladores que automatizam esse processo.
2. Após a instalação, basta conseguir o número IP da vítima e a máquina estará aberta para invasões.

4.2.2 Back Orifice

O Back Orifice , BO, é um programa desenvolvido pelo grupo de crackers de elite chamado *Cult of Dead Cow* (Culto da Vaca Morta), com o objetivo de mostrar as fragilidades do sistema operacional Windows, atingindo usuários do Windows (95/98/NT). Hoje o Back Orifice encontra-se também disponível para outras plataformas, como o Unix e MacOS (GOMES, 2000).

Similarmente ao Net Bus, o programa constitui-se de duas partes: BO Server e o próprio Back Orifice. É um sistema de controle remoto de microcomputadores, via conexão TCP/IP, que permite ao cliente monitorar e administrar a máquina em que está sendo executado o servidor.

Com o programa instalado em uma máquina, qualquer cliente BO que possua a mesma versão e senha do servidor pode se conectar a máquina em questão e efetuar diversas tarefas como:

- Controle de Sistema: fornece dados como informações sobre o usuário, drivers, tipo de CPU, versão do sistema operacional, senhas, e outras;
- Controle de Arquivos de Sistema: possibilita que arquivos sejam renomeados, alterados, lidos, copiados compactados, descompactados e deletados;
- Controle de Rede: verificar todos os recursos de rede acessíveis, todas as conexões de entrada e saída, listas e pode criar e excluir conexões de rede;
- Controle Multimídia: permite trocar arquivo .wav e capturar cenas da tela;
- Acesso ao Registro: lista as funções do registro, permite adicionar funções e apagá-las.

Como o BO tem sido difundido sem senha, qualquer pessoa pode tomar o controle

de uma máquina infectada.

A troca de dados entre cliente e servidor faz-se através de pacotes UDP criptografados.

Para verificação da existência do BO em uma máquina com Windows, deve-se verificar se a porta 31337, está em listening (UDP), pois é a porta por default utilizada pelo trojan.

No prompt do MS-DOS, digite: C:\>netstat –na, se nas linhas resultantes constarem a porta 31337 (UDP) em listening, ou seja, em estado de espera por conexões externas, certamente o backdoor default está instalado. Mas deve-se verificar se existem outras portas abertas, pois pode alterar facilmente a porta de conexão.

Outra maneira de detectar é através do editor de registro, acessando `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices`, se verificando a existência de serviços que não foram intencionalmente instalados ou que possam despertar alguma suspeita.

Para remover totalmente o Back Orifice, remova o registro e apague o arquivo server. Se possível, faca um backup de todos os dados, formate a máquina e instale novamente o sistema. Existem programas específicos para a eliminação do BO como Antigen e BOShiel que detectam e desabilitam o BO a cada vez que este tentar se instalar em uma máquina.

Existem vários programas disponíveis que detectam tentativas de uso de Back Orifice (BO) para diversas plataformas, um deles é o NOBO, utilizado para plataformas Microsoft® Windows. O NOBO abre a porta do Back Orifice e fica esperando pacotes TCP/IP vindos de clientes BO. Quanto algum pacote é recebido, o NOBO registra a informação, com dados do remetente (endereço IP e nome da máquina), podendo ignorar o pacote ou mesmo responder com uma mensagem padrão.

Toda atividade – todo pacote do BO recebido – pode ser gravada em um arquivo de "log", ou seja, de registro de eventos. O NOBO não elimina o Back Orifice e nem impede que o BO entre em ação, ele apenas faz com que o BO não consiga entrar em ação se tentar ouvir na porta que o NOBO esteja instalado, mas deixa livre para utilizar outras portas.

4.3 ENGENHARIA SOCIAL

Ataques de engenharia social é um método que consiste em enganar as vítimas através do emprego de informações adquiridas por pesquisa. O atacante se provê destas informações para induzir a execução de ações indevidas e perigosas.

O intruso pode utilizar-se de vários meios para conseguir lançar um ataque de engenharia social, sendo telefonemas e mensagens de correio eletrônico falsificadas os mais comuns.

4.4 SNIFFERS

A ação de capturar informações destinadas a uma outra máquina é chamada *sniffing*, técnica utilizada geralmente para captura de senhas.

"*Sniffers* são dispositivos que capturam pacotes de rede. Seu propósito legítimo é analisar tráfego de rede e identificar áreas potenciais de preocupação". (ANONYMOUS, 2000, p. 255).

Em uma rede local cada estação de trabalho tem um endereço de hardware que a identifica de maneira exclusiva na rede, e em circunstâncias normais, todas máquinas podem "ouvir" o tráfego, mas tratará apenas informações cujo endereço de destino seja o seu endereço de hardware ou um endereço de *broadcast*. A maioria das interfaces de rede permitem o modo promíscuo (veja seção 3.1.1), dessa maneira é gerada uma interrupção para CPU da máquina sempre que a interface recebe um pacote, independente para qual do endereço esteja destinado. (WEBER, 2000)

Os sniffers podem capturar todos pacotes que trafegam na rede, porém atacantes normalmente utilizam apenas os primeiros 300 bytes de cada pacote, devido ao grande volume de dados e além das informações mais visadas encontram-se nessa parte do pacote.

Snnifers podem representar um grande risco em um ambientes de rede que trafegam muitos dados sensíveis sem nenhum tipo de cifragem nos dados, onde informações confidenciais, *username* e *password* podem ser capturadas, além de ser possível abrir brechas na segurança de redes vizinhas ou ganhar acessos de alto nível.

(ANONYMOUS, 2000).

Um sniffer pode ser encontrado em qualquer ponto, mas normalmente atacantes utilizam-se de pontos estratégicos, como um lugar adjacente a uma máquina ou rede que recebe muitas senhas ou em uma rede local Ethernet com a interface de rede no modo promíscuo (em máquinas usando sistema operacional Linux, usar a interface no modo promíscuo só é permitido com privilégios de root).

A detecção de sniffers é difícil, principalmente se estiver sendo utilizados como programas passivos, ou seja, coletam dados e não respondem a nenhuma informação, dessa forma não deixam trilha de auditoria, sendo necessário percorrer fisicamente todas as conexões de rede. Outra maneira de detectar-se um *sniffer* em operação é através dos imensos arquivos gerados por ele.

4.5 EXPLORAÇÃO DE PROTOCOLOS

O protocolo TCP/IP foi desenvolvido sem a análise de diversos aspectos de segurança, hoje encontra-se na quarta versão, porém ainda traz grande parte das fragilidades do projeto original. Assim muitos ataques exploram essas brechas existentes, conseguindo dessa forma personificar máquinas, capturar pacotes em tráfego e conseguir acesso não autorizado a sistemas de computação.

4.5.1 TCP *Sequence Number Prediction Attack* (SNPA)

O ataque TCP *Sequence Number Prediction* é um dos ataques mais fascinantes e complexos, onde através dele um atacante pode inserir-se entre uma conexão existente entre servidor e um usuário.

O protocolo TCP utiliza uma seqüência de números associada aos pacotes para garantir que os pacotes cheguem ao destino na ordem em que foram enviados. O ataque TCP *Sequence Number Preditiction* consiste em descobrir o próximo número de seqüência do pacote trafegado e assim forjar pacotes com a numeração esperada pela máquina alvo (ANONYMOUS, 1998).

A compreensão desse ataque se faz necessário o conhecimento da a função do *sequence number*, o qual garante que os pacotes chegarão ao destino, e na ordem correta.

Para o que host A comunique-se com o host B, A envia o primeiro SYN para B no início da conexão, informando o número de seqüência inicial – ISN (*Initial Sequence Number*) que será usado por A. O host B responde com a segunda fase do *handshake* (SYN, ACK), informando o número de seqüência inicial usado por ele e o número de seqüência de A (ACK). Para cada byte transmitido, o número de seqüência é incrementado em uma unidade.

A → B: SYN (ISNA)

B → A: SYN (ISNB), ACK (ISNA)

A → B: ACK (ISNB)

<Conexão estabelecida>

A → B: dados <e/ou>

B → A: dados

Quando a máquina é inicializada, o ISN é setado para 0. A cada segundo ele é incrementado no valor de 128.000 e a cada conexão solicitada em 64.000. Este será o valor do sequence number, a partir daí ele é incrementado em 1 para cada byte transmitido (STRAUCH, 2000).

Assim funciona uma conexão normal, a seguir será descrito o funcionamento do ataque:

O atacante, host C, irá inserir-se na conexão entre os usuários A e B, fazendo se passar por A.

Primeiro o host A é tirado do ar, através um ataque como *SYN Flooding*. Em seguida o host C envia uma requisição de conexão ao host B, com o endereço IP de A; o host B envia a confirmação de conexão ao host A, o qual encontra-se fora do ar devido ao ataque sofrido e não irá responder a B.

C → B: SYN (ISNC)

B → A: SYN (ISNB), ACK(ISNC)

Agora cabe ao host C confirmar a conexão, e para isto precisa enviar um ACK com o número de seqüência enviado pelo host B. Somente com este número é que o host B considerará a conexão válida.

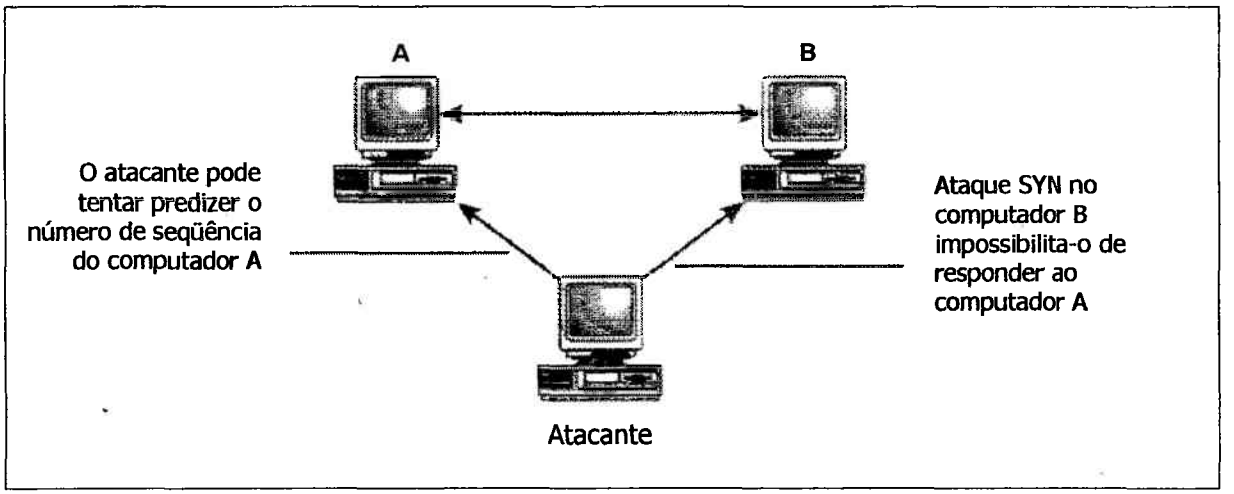

Figura 4.3: Obtenção do ISN para o ataque *sequence number*. Fonte: NORTHCUT (1999)

Para descobrir o valor do ISN, o atacante pode conectar-se a algum outro serviço do host B e analisar qual o ISN fornecido. Pelo fato de conhecer como o ISN é incrementado, o atacante pode ter uma idéia aproximada de qual o ISN que foi enviado ao host A.

O número calculado é enviado, na resposta ACK, ao host B. Caso o número esteja correto, a conexão será estabelecida entre os hosts B e C, sendo que o primeiro pensa que estabeleceu uma conexão com o host A.

C → B: ACK (ISNB)

Se o número enviado for menor que o esperado, ele será considerado um pacote duplicado e será descartado, e se o número for maior o servidor imaginará que alguns pacotes intermediários se perderam e espera a retransmissão. Nos dois casos o ataque não funcionará.

4.5.2 *Spoofing*

Spoofing é uma técnica sofisticada de autenticar uma máquina para outra forjando pacotes de um endereço de origem confiável. (ANONYMOUS, 2000).

O ataque *spoofing* utiliza-se dos conceitos de confiança e autenticação entre máquinas. A confiança e autenticação geralmente têm um relacionamento inverso, quanto mais alto o nível de confiança entre máquinas, menor será a autenticação requerida. Muitas

comunicações entre computadores na Internet se baseiam em "parceiros" confiáveis. Um computador X pode manter uma comunicação com um computador Y de forma que não seja necessária a constante verificação de autenticidade entre eles. (MENDES, 1999)

Assim o atacante se disfarça, informando ao host X que é o host Y (uma máquina confiável do alvo) , fazendo que o computador X vai aceite seus comandos.

Existem diversas técnicas de *spoofing*: IP, DNS, ARP e RIP estão entre as mais conhecidas.

4.5.2.1 IP Spoofing

O IP *Spoofing* não é exatamente uma forma de ataque, mas sim uma técnica que é utilizada na grande maioria dos ataques para esconder a identidade do atacante.

A técnica IP *Spoofing* consiste em simulação, onde ocorre a troca do número IP da máquina por outro, para assim poder passar-se por outro host. Isto pode ser feito através de manipulação direta dos campos do cabeçalho (MCCLURE, SCAMBRAY e KURTZ, 2000).

Quando um host A quer se conectar ao B, a identificação é feita através do número IP que está no cabeçalho IP, por isto, se o número IP do cabeçalho enviado pelo host A for falso (IP de um host C), o host B, por falta de outra forma de identificação, acredita estar se comunicando com o host C.

Apesar da falha na autenticação para estabelecimento de conexão, que é baseada unicamente no endereço de origem de IP, não é suficiente apenas forjar o endereço de origem da máquina para o ataque ser realizado com êxito; é necessário executar outra técnica para manter um diálogo de seqüência com o alvo, o Sequence Number. Dessa forma o atacante consegue acesso a máquinas que confiam no IP que foi falsificado, capturam assim conexões e burlam filtros de pacotes dos *firewalls* que bloqueiam o tráfego baseado apenas em endereços de origem e destino.

4.5.2.2 DNS *Spoofing*

O DNS (*Domain Name Service*) é o responsável pela conversão de nomes em endereços IP. Pelo fato do DNS ser uma base de dados distribuída, onde em cada servidor

apenas se encontra uma parte de toda a informação da base de dados, um pedido de conversão pode ser satisfeito de duas formas:

- *authoritative*: situação em que se obtém uma resposta do servidor que faz a gestão do correspondente domínio;
- *non-authoritative*: quando a resposta é proveniente de um servidor que tenha na sua cache a equivalência pretendida.

No caso de a resposta não ser encontrada localmente, existem duas formas de obtê-la, mediante especificação do cliente:

- recursiva: o servidor local remete o pedido a um servidor externo que com grande probabilidade, possui uma resposta *authoritative* a enviar ao cliente, guardando uma cópia na cache;
- iterativa: o servidor retorna ao cliente o endereço externo ao qual ele remeteria o pedido, deixando a cargo do cliente a realização efetiva do pedido.

O ataque DNS *Spoofing* consiste em alterações de nomes, fazendo com que máquinas não confiáveis (do atacante), passem por máquinas confiáveis. Para isso o atacante obtém o controle do servidor DNS e realiza alterações nas tabelas de endereço de IP e nome de hosts, gravando essas alterações em um banco de dados na tabela de tradução no servidor DNS. Realizada essas alterações, o intruso obtém acesso livre a serviços que realizam autenticação baseada em nome.

A cada pedido atendido, é associado um TTL, configurado pelo gestor do domínio, que permanece válido nas caches dos servidores. Dessa forma, se um servidor estiver sob o controle de um atacante, as repostas authoritative obtidas desse servidor, manterão ativas de acordo com as configurações atribuídas ao TTL, podendo ser um período de horas, sendo tempo suficiente para realizações de diversos tipos de ataques.

Uma forma de tentar evitar o ataque é através de uma contraprova, ou seja, uma vez obtido o endereço IP desejado, efetuar um novo pedido ao DNS para conversão do IP no nome. Este tipo de pedido designa-se *reverse lookup*, o qual só é eficaz se o atacante do servidor apenas alterar um dos dois diretórios que compõem cada base de dados local de DNS. Uma boa medida de segurança será manter em diferentes máquinas os servidores primários para os diretórios de name lookup e reverse lookup.

Outra forma de minimizar os efeitos de um ataque ao DNS é manter em

exploração uma versão recente do programa servidor de DNS.

A maioria dos novos sistemas possui métodos contra este tipo de ataque, utilizando para isto uma técnica conhecida com *cross-check*. Nela, o nome retornado pela consulta é submetido novamente ao serviço de nomes. Se o endereço utilizado para a conexão é diferente do retornado pelo *cross-check*, a conexão é abortada e uma violação de segurança é apontada. O *cross-check* pode ser implementado no servidor de nomes ou nos servidores dos serviços com autenticação baseada em nomes. Entretanto, existem variantes do *DNS Spoofing* onde o intruso contamina o cache das respostas do DNS na máquina alvo ou inunda o servidor de DNS com pedidos, com o objetivo de enganar o *cross-check*.

4.5.2.3 RIP – Rounting Information Protocol

Um roteador tem como função identificar a melhor rota que um pacote deve seguir, para a comunicação entre máquinas em rede distintas. Inicilamente um roteador possui apenas as configurações locais das redes as quais está diretamente conectado, sendo que as demais devem ser definidas por rotas estáticas ou por protocolos de roteamento.

Rotas estáticas são caminhos nas tabelas de roteamento inseridas manualmente, enquanto os protocolos de roteamento trocam informações automaticamente e todas as mudanças das rotas são atualizadas dinamicamente.

Um dos protocolos mais utilizados para trocas de tabelas de roteamento entre roteadores é o RIP (*Routing Information Protocol*), cujas atualizações são realizadas em intervalos de tempos fixos e pré-definidos.

O RIP envia a cada intervalo de tempo, a sua tabela de roteamento utilizando broadcast em UDP na porta 520. Os roteadores que recebem estes pacotes reorganizam sua tabela de roteamento dinâmico de acordo com os dados recebidos.

Se o roteador for um participante passivo do RIP, onde apenas recebe os broadcast e atualiza as tabelas dinâmicas, torna-se simples a introdução de entradas falsas nessas tabelas, basta que qualquer computador de um IP confiável envie as tabelas através da porta 520, aos roteadores até o alvo. Uma máquina com um IP confiável é fácil de conseguir, basta o atacante personificar uma máquina que o alvo confie, alterando o endereço IP. Assim com as informações de roteamento falsas, a rota constante na tabela passa a ser do alvo para o atacante.

Porém, após a alteração das rotas, a máquina personificada pelo intruso não

conseguirá fazer qualquer comunicação com o alvo, pois as respostas estarão sendo redirecionadas para o atacante, podendo este ser descoberto. Para tanto, o intruso deverá personificar uma máquina que, de preferência, esteja desligada ou inoperante, devido a algum ataque de negação de serviço.

Em uma variante sutil e perigosa deste ataque, o atacante personifica uma máquina que está ativa e recebe os pacotes do alvo; visualizando-os ou alterando-os, e após a utilização envia para o destino correto através de IP *source address routing*, não precisando da inoperabilidade da máquina alvo. Além disto, outro ponto deste ataque é que conexões requisitadas pelo alvo também passam pelo atacante, que pode analisar os pacotes que, ocasionalmente, poderão conter senhas de usuários ou outras informações relevantes.

As principais formas de evitar este tipo de ataque são:

- Sempre que possível estabelecer rotas estáticas;
- Não utilizar RIP passivo;
- Se o gateway for um computador, utilizar um daemon de roteamento no qual seja possível configurar os endereços IP para origens confiáveis de RIP.

Hoje já existem roteadores que aceitam o protocolo RIP versão 2, que especifica um melhoramento em diversos aspectos em relação ao RIP, incluindo autenticação nas mensagens de atualização. Entretanto ele não torna os sistemas de roteamento completamente seguros; apenas torna o trabalho do atacante um pouco mais difícil.

4.5.2.4 ARP – Address Resolution Protocol

O protocolo ARP permite realizar uma equivalência entre os endereços IP e os endereços de hardware. Quando um pacote IP se encontra pronto para ser enviado para a rede, é necessário associar-lhe um endereço de hardware de destino, e nesta altura duas situações podem ocorrer: o endereço IP corresponde à rede local, e nesse caso o endereço de hardware de destino será o da interface de destino; caso contrário, o endereço de hardware de destino será o da interface do *gateway* da rede local ou sub-rede.

Cada host mantém uma tabela ARP com as equivalências correspondentes aos outros hosts com os quais manteve contato recentemente. Cada entrada nesta tabela tem um tempo de vida (TTL) de alguns minutos, findos os quais um novo pedido ARP é

enviado para revalidar a equivalência. Se alguém pretende receber indevidamente pacotes que tenham como destino um determinado endereço IP, poderá atribuir à sua própria máquina esse endereço IP e esperar alguns minutos, durante os quais deverá interromper a ligação da rede ao computador com o endereço verdadeiro. A interrupção poderá ser provocada de diversas formas, consideradas acidentais, ou mesmo sem que tal seja notado. Desta forma, a tabela ARP passa a ter um diferente destino para os pacotes enviados para o endereço IP sob ataque. A melhor forma de evitar este ataque é através da utilização de tabelas ARP estáticas. As equivalências estáticas não possuem um TTL.

4.5.3 ICMP – *Internet Control Message Protocol*

O ICMP é um protocolo utilizado para fazer a manutenção de conexões TCP/IP. Ele também é usado para gerar uma grande quantidade de ataques. Entres as mensagens ICMP mais exploradas por atacantes estão a *redirect* e *destination unreachable*.

A *redirect* é utilizada por gateways para avisar máquinas sobre melhores rotas. Os ataques produzidos com esta mensagem subvertem os sistemas de roteamento, se assemelhando aos ataques no RIP. Um ataque viável utilizando a *redirect* é configurar uma falsa rota para um host confiável *A*, através de um *gateway* secundário, já de posse do atacante. Para isto, seguem os seguintes passos:

- o atacante envia para o alvo um pacote de requisição de conexão falsificando o endereço de *A*.
- o alvo, dando seqüência ao *handshake*, envia seu próprio pacote para *A*, através do roteador primário.
- enquanto o pacote estiver em trânsito, o atacante falsifica uma mensagem *redirect* afirmando ser do gateway primário e referindo-se á falsa conexão.

Esta mensagem é enviada para o alvo, que a aceita como sendo um controle legítimo. Caso o alvo faça as mudanças ditadas pela *redirect* em suas tabelas globais de roteamento, existirá então um caminho entre o alvo e o intruso, que poderá ser utilizado para conexões legítimas. Na realidade, não é necessário que o atacante disponha de um *gateway* secundário. É suficiente que ele detenha a posse de qualquer máquina na rede local do alvo, utilizando-a como sendo o *gateway*.

A mensagem *destination unreacheable* é utilizada para informar a um host que o destino da sua conexão não pode ser encontrado. Esta mensagem pode ser enviada por um roteador, caso nenhuma rota para o destino esteja disponível, ou se a máquina destino for inexistente ou se está inativa. A *destination unreacheable* pode, também, ser enviada pelo próprio host destino. Neste caso, o host está afirmando que a porta a que a conexão se refere não pode ser encontrada. A *destination unreacheable* pode ser utilizada para terminar conexões específicas existentes e, deste modo, promover ataques do tipo *denial-of-service*. Para isto, basta que o intruso conheça os números da porta local e remota de uma conexão TCP e falsifique um pacote ICMP. Estas informações são, as vezes, disponíveis através do serviço *netstat*.

4.5.4 Ataque de Fragmentação

Um ataque de fragmentação tenta fazer com que serviços TCP de máquinas protegidas por um filtro de pacotes possam ser acessados mesmo que o filtro não permita seu acesso.

A maior parte dos filtros de pacotes tradicionais filtra apenas o primeiro fragmento de um pacote, já que sem este fragmento não é possível para a máquina cliente montar o pacote completo, deixando os demais fragmentos passarem. E nos casos em o primeiro fragmento não é um pacote de abertura de conexão a autorização de passagem é concedida.

O ataque consiste em se fragmentar o pacote TCP de modo que o segundo fragmento contenha parte do cabeçalho TCP, mais especificamente as portas e os flags do protocolo, ou seja, o estabelecimento de conexão TCP. Este segundo fragmento possuirá os flags para uma abertura de conexão. O primeiro fragmento consiste em um pacote que não é de abertura de conexão.

Da forma com que ambos os fragmentos foram montados, eles serão aceitos pelo filtro de pacotes e atingirão a máquina destino. Uma vez na máquina destino, os pacotes serão montados, produzindo um pacote de abertura de conexão, que se estivesse sido enviado inteiro teria sido bloqueado no filtro de pacotes.

4.5.5 DoS – *Denial of Service*

Os ataques DoS são bastante conhecidos no âmbito da comunidade de segurança de redes. Estes ataques, através do envio indiscriminado de requisições a um computador alvo, visam causar a indisponibilidade dos serviços oferecidos por ele.

Estes ataques na verdade não "invadem" o site, ele interrompe ou nega completamente serviço a usuários legítimos, redes sistemas ou outros recursos. E devido a facilidade normalmente encontrada na interrupção de serviços, por brechas deixadas pelo protocolo TCP/IP, de alguns sistemas operacionais em suas pilhas de rede e interface amigável que muitas técnicas apresentam, esse forma de ataque tem se tornado comum. (MCCLURE, SCAMBRAY e KURTZ, 2000).

Existem diferentes tipo de ataque DoS, MCCLURE, SCAMBRAY e KURTZ (2000) classificam em quatro: consumo de largura de banda, inanição de recursos, ataques de roteamento e DNS e ataque DoS genérico. Na seqüência, tais métodos são brevemente analisados.

4.5.5.1 Consumo de Largura de Banda

Ataque no qual o atacante consome toda a largura de banda da rede do alvo. Pode ocorrer basicamente em duas situações:

1. O atacante consegue inundar a conexão da rede da vítima quando possuir mais largura de banda disponível do que o alvo ou vítima;
2. O atacante utiliza múltiplas instalações para conseguir inundar a conexão de rede da vítima, quando esta possuir largura maior.

4.5.5.2 Inanição de Recursos

Um ataque de inanição de serviços visualiza consumir recursos do sistema, normalmente ciclos de CPU, memória e quotas de sistemas de arquivos.

4.5.5.3 Ataques de Roteamento e DNS

Para efetuar ataques DoS de roteamento, atacantes manipulam entradas de tabelas

de roteamento, sendo executados onde existam protocolos com autenticação fraca ou inexistente, como o RIP e BGP. Assim o tráfego da vítima é roteado para a rede do atacante ou para uma rede que não exista (buraco negro).

Nos ataques DNS, o servidor DNS é comprometido e suas tabelas são explicitamente mudadas (IP e hostnames), sendo gravadas estas alterações nas tabelas de banco de dados no servidor, fazendo que um cliente receba um endereço IP falso como resposta, a uma solicitação. (veja figura 4.4)

4.5.5.4 Ataques DoS Genéricos

Os ataques DoS implementam ferramentas que realizam funções com a tentativa de um flood (inundação) em uma rede; tentativa de rompimento de conexões entre duas máquinas; tentativa de impedimento de acesso a um serviço por parte de um usuário e tentativa de romper serviços a um sistema específico, ou a um usuário.

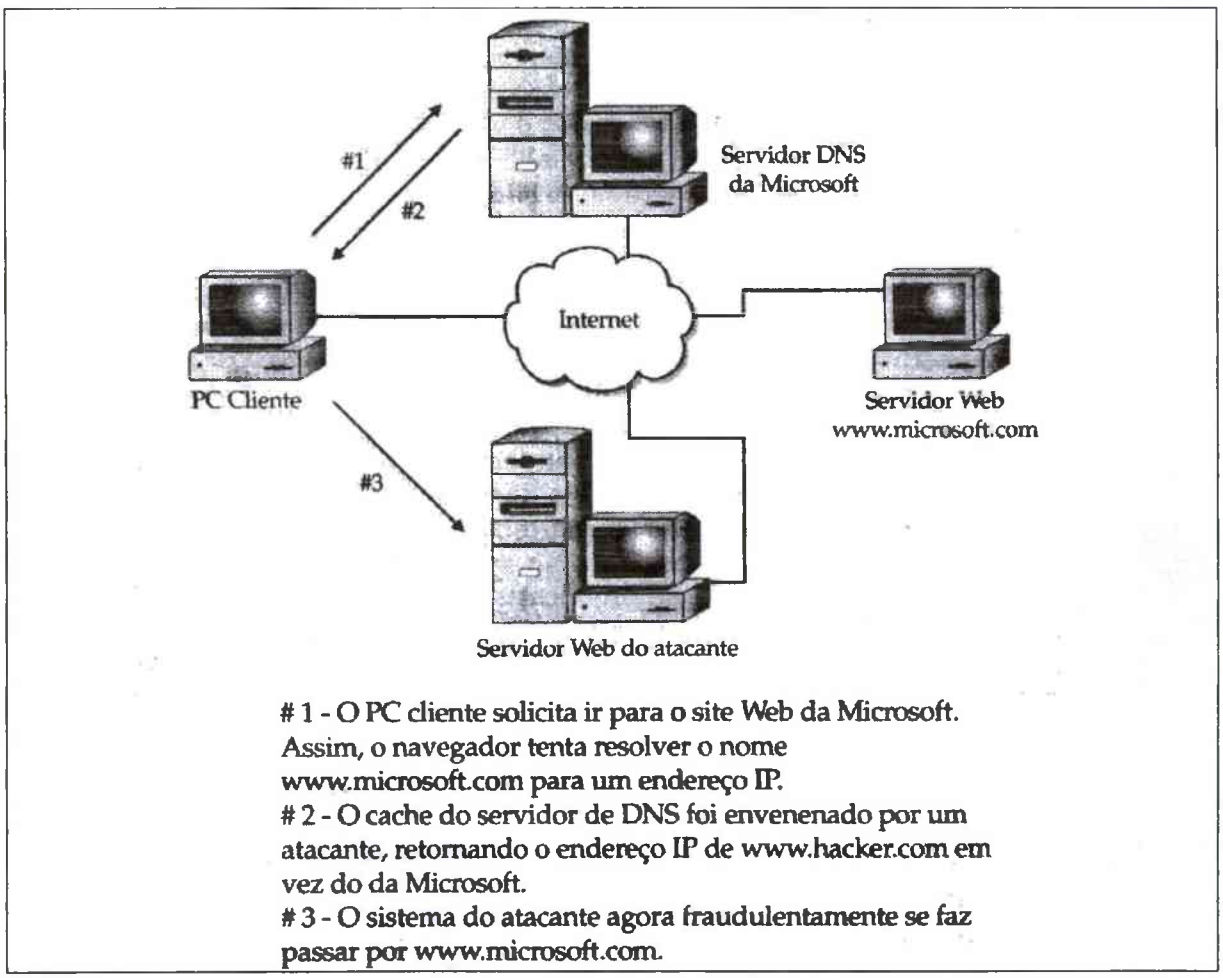

Figura 4.4: Alterações na cache do DNS. Fonte: McCLURE , SCAMBRAY e KURTZ (2000)

4.5.5.5 SMURF

O ataque SMURF é um dos ataques DoS mais poderosos, devido ao seu efeito de amplificação, pois envolve centenas de computadores, sendo necessário três elementos para realização: o atacante, uma rede amplificadora e a vítima.

O smurf manda uma solicitação de ICMP ECHO falsificada para o endereço de broadcast da rede amplificadora. Com o endereço da vítima como origem, simulando a procedência da solicitação.

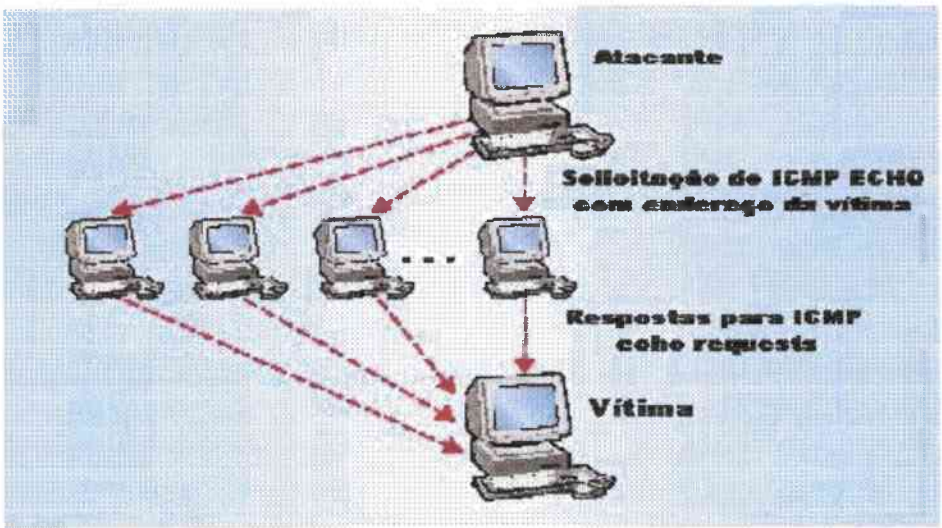

Figura 4.5: Ataque de negação de serviço – smurf . Fonte: GARBER (2000)

Cada sistema integrante da rede amplificadora responderá o pedido de ECHO, fazendo que a vítima receba todas as respostas. Se o pacote ICMP for enviado a uma rede que possua 100 sistemas que respondam a ping broadcast, o ataque DoS terá um fator de multiplicação igual a 100. A razão da amplificação é devido às redes responderem aos pacotes enviados.

O *fraggle* é um variante dessa forma de ataque, diferenciando apenas por utilizar o UDP como protocolo de transporte. Ele causa uma "inundação" ao enviar um pacote UDP para uma lista de endereços de broadcast. Normalmente a porta utilizada para a execução do ataque é a porta 7. (STRAUCH, 2000).

4.5.5.6 SYN FLOOD

SYN Flood utiliza-se de uma brecha existente no estabelecimento de conexão no protocolo TCP/IP, para realizar operações de negação de serviços, pois, embora a maioria dos sistemas consegue suportar centenas de conexões concorrentes para uma porta específica, tais sistemas só podem tratar aproximadamente uma dezena de solicitações de conexão potenciais antes de exaurir todos os recursos alocados para o estabelecimento de conexões (GARBER, 2000).

O processo DoS ocorre quando o atacante inicia o processo de estabelecimento de conexão (envio do pacote SYN) à máquina da vítima, por exemplo de A para B, fazendo uso de um endereço falso. B enviará um pacote SYN/ACK para este endereço, que caso exista, retornará uma mensagem de RST, pois não solicitou conexão com A; porém se o atacante fizer uso de um sistema não existente, B enviará o pacote SYN/ACK e não receberá resposta do sistema A, deixando a conexão em potencial no estado SYN_RECV, na fila de espera. O sistema permanece comprometido em estabelecer conexão, sendo as conexões em potencial removidas somente quando esgotar o tempo de conexão, que pode variar de 75 segundos a 23 minutos (MCCLURE , SCAMBRAY e KURTZ, 2000).

Como a fila de conexão é normalmente pequena, para o ataque ser bem sucedido, basta o envio de pacotes SYN a cada 10 segundos para colocar uma porta específica totalmente fora de ação.

O SYN Flood não é apenas uma forma de ataque que pode ser utilizada separadamente, mas também uma técnica utilizada em ataques mais elaborados, onde o atacante precisa parar uma máquina para que a mesma não atrapalhe o seu ataque.

4.5.6 DDoS - *Distributed Denial of Service*

Os ataques DdoS utilizam centenas ou até milhares de computadores desprotegidos e ligados na Internet para lançar coordenadamente o ataque.

Este tipo de técnica foi amplamente divulgado pela imprensa em fevereiro de 2000, quando gigantes do mundo virtual como Yahoo, Ebay, Amazon, CNN, foram "forçados" a saírem do ar por algumas horas, devido a ataques sofridos por piratas cibernéticos, onde na maioria dos casos os prejuízos não puderam ser totalmente avaliados. Ainda no mesmo mês os ataques ocorreram no Brasil contra os sites da UOL, Globo On e

IG (ROÇAS, 2000).

Uma forma de ataque do DoS refere-se ao chamado "flood de pacotes", o envio de grande quantidade de dados sem significado tomando todos os recursos do servidor, esses *floods* disparados de um único host, não tem "poder" suficiente para derrubar um site com grande capacidade de serviços. Porém, utilizando vários hosts simultaneamente para a realização do *flood*, multiplica-se a capacidade de estourar o link do alvo.

As ferramentas principais para ataques DDoS são: Trinoo, TFN, TFN2K e Stacheldraht. Os agentes são geralmente instalados em hosts Linux e Unix, no entanto, já existem versões para plataforma Windows.

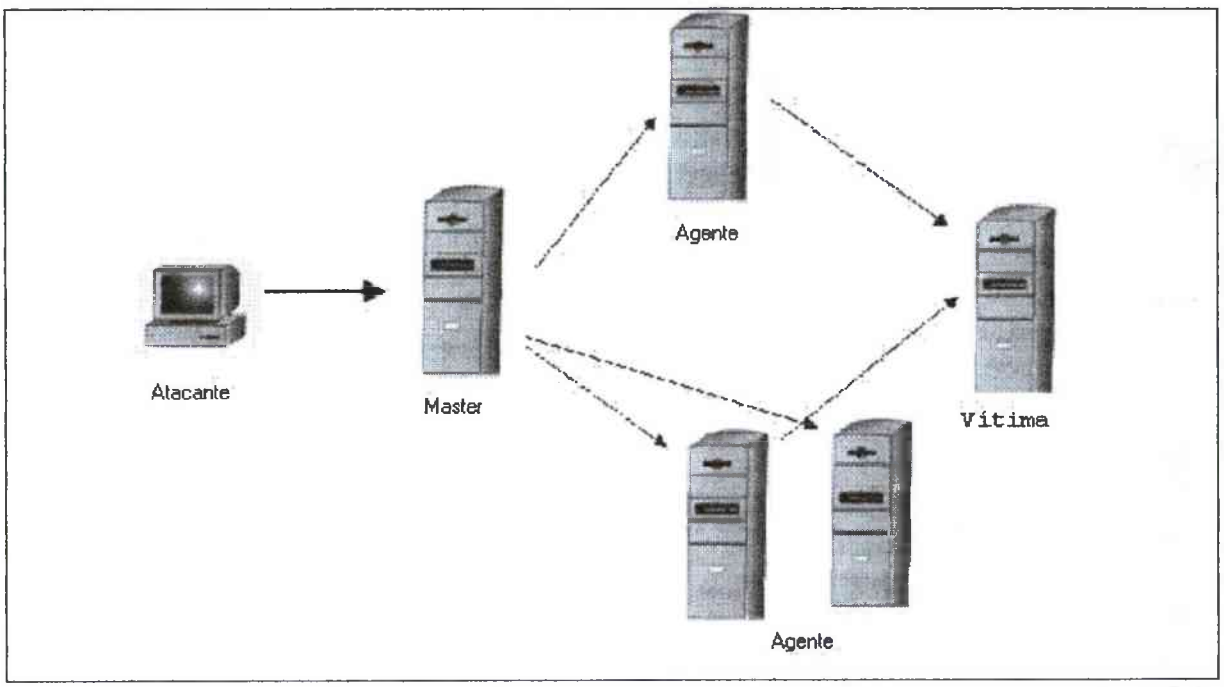

Figura 4.6 O funcionamento de um ataque DDoS

Para análise do funcionamento do ataque DDoS, é necessário a identificação de cada uma das partes envolvidas, a saber:

- **Atacante**: O coordenador do ataque.
- **Master**: Máquina que recebe os parâmetros para o ataque e comanda os agentes.
- **Agente**: Máquina que efetivamente concretiza o ataque DoS contra uma ou mais vítimas, conforme for especificado pelo atacante.
- **Vítima**: Alvo do ataque (máquina "inundada" por um volume enorme de pacotes).
- **Cliente**: Aplicação que reside no master e que efetivamente controla os ataques enviando comandos aos daemons.

- **Daemon**: Processo que roda no agente, responsável por receber e executar os comandos enviados pelo cliente.

O ataque DDoS é efetuado basicamente em três fases: fase de "intrusão em massa", na qual ferramentas automáticas são usadas para comprometer máquinas e obter acesso privilegiado (acesso de boot). A segunda, o atacante instala software DDoS nas máquinas invadidas com o intuito de montar a rede de ataque. E na última fase, são lançados floods de pacotes contra uma ou mais vítimas, consolidando o ataque.

Cada uma das fases é detalhada na seqüência, sendo que as fases 1 e 2 são executadas imediatamente uma após a outra de maneira altamente automatizada (acompanhe na figura 4.6):

Fase 1: Intrusão em massa

Esta primeira fase consiste basicamente nos seguintes passos:

- É realizada uma varredura nas portas e vulnerabilidades em redes consideradas "interessantes", como redes com conexões de banda-larga e com baixo grau de monitoramento;
- Explora-se as vulnerabilidades reportadas, para obter acessos privilegiado nessas máquinas. Devido a existência de sniffers e rootkits, máquinas Solaris e Linux, são as preferidas;
- É criada uma lista com IPs de máquinas invadidas, as quais serão utilizadas no ataque.

Fase 2: Instalação de software DDoS

- É escolhida uma conta de usuário para ser utilizada como repositório para as versões compiladas de todas as ferramentas de ataque DDoS.
- Uma vez que a máquina é invadida, os binários das ferramentas de DDoS estarão instalados nestas máquinas, permitindo o controle remoto. Essas máquinas comprometidas que desempenharão os papéis de masters ou agentes. A definição de máquinas agente ou master fica a critério do atacante. Normalmente o perfil das máquinas master é de máquinas não manuseadas freqüentemente por administradores e tão pouco monitoradas, enquanto as agentes são selecionadas de acordo com as velocidades dos links.

- Uma vez instalado e executado o daemon DDoS nos agentes, eles anunciam sua presença aos masters e ficam à espera de comandos (status "ativo"). O programa DDoS cliente, que roda nos masters, registra em uma lista o IP das máquinas agentes ativas. Esta lista pode ser acessada pelo atacante.
- A partir da comunicação automatizada entre os masters e os agentes, organizam-se os ataques.
- Opcionalmente, visando ocultar o comprometimento da máquina e a presença dos programas de ataque, são instalados rootkits.

Fase 3: Disparando o ataque

O atacante controla uma ou mais máquinas master, as quais, por sua vez, podem controlar um grande número de máquinas agentes, e a partir destes agentes que é disparado o flood de pacotes que consolida o ataque. Os agentes ficam aguardando instruções dos masters para atacar um ou mais endereços IP (vítimas), por um período específico de tempo.

Assim que o atacante ordena o ataque, uma ou mais máquinas vítimas são bombardeadas por um enorme volume de pacotes, resultando não apenas na saturação do link de rede, mas principalmente na paralização dos seus serviços.

4.5.6.1 Ferramentas de DDoS

Ao contrário do que se pensa, os ataques DDoS não são novos. A primeira ferramenta conhecida com esse propósito surgiu em 1998: a Fapi. Desde então, foram diversas as ferramentas de DDoS desenvolvidas, cada vez mais sofisticadas e com interfaces mais amigáveis. A seguir, são listadas algumas ferramentas na ordem em que surgiram:

1. Fapi
2. Blitznet
3. Trinoo (ou Trin00)
4. TFN
5. Stacheldraht
6. Shaft

7. TFN2K

8. Trank

9. Trin00 win version

Na seqüência, apenas algumas das técnicas de funcionamento são analisadas. Em MIRANDA (2000), MCCLURE, SCAMBRAY e KURTZ (2000) e SPYMAN (1999) são encontradas explicações de forma mais detalhada.

Técnicas como Trin00 e o TFN e suas variantes, funcionam de forma semelhante: o atacante deve invadir uma série de máquinas para montar a chamada infra-estrutura de ataque. A figura 4.7 ilustra esta estrutura:

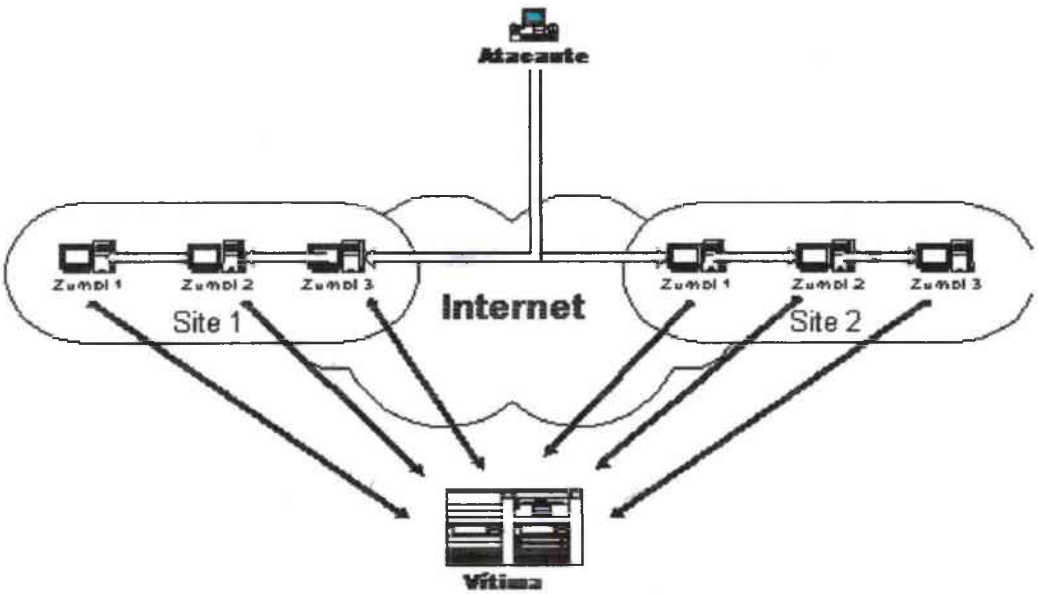

Figura 4.7: Método de ataque das ferramentas DDoS. Fonte: BERTHOLDO e ANDREOLI (2000).

TRIN00

O Trin00 é uma ferramenta distribuída usada para lançar ataques DoS coordenados, especificamente, ataques do tipo UDP flood.

Uma rede Trinoo é composta por um número pequeno de masters e um grande número de agentes.

Para "infectar" uma máquina com um componente Trinoo, o agressor necessita ter acesso administrativo ao sistema. Um subconjunto de máquinas consideradas vulneráveis quanto à segurança, são comprometidas e utilizadas para a rede Trin00, e os componentes

"cliente" e "servidor" dos programas são instalados a partir de um *script* automático de instalação.

Depois da rede Trinoo instalada, o atacante pode controlar os componentes de servidor, também conhecidos como daemons, ao manipular remotamente os componentes "cliente" do programa, também conhecidos por mestres. A ligação ao sistema mestre pode ser estabelecida através do protocolo Telnet, ao ligar a uma porta específica e fornecer as devidas credenciais de "login". O controle de acesso baseado em "password" é usado para comunicações entre todos os nós da rede Trinoo para impedir tanto a aproximação de investigadores, como atacantes rivais de usurpar os agentes. Se uma tentativa de ligação for feita ao mestre durante uma sessão decorrente, o sistema envia um aviso para o utilizador previamente autenticado, dando ao atacante uma oportunidade para fazer a limpeza e retirar-se.

Uma vez ligado ao mestre, o atacante pode controlar um exército de daemons Trinoos emitindo comandos que iniciam ou param ataques DoS contra anfitriões específicos. Agindo por parte do atacante, o mestre Trinoo comunica com os seus daemons através de um protocolo proprietário de texto que usa o UDP como protocolo de transporte subjacente. O ataque termina normalmente depois de um período de tempo pré-determinado ou quando o atacante envia o comando "mdie" ao mestre, que depois envia o comando "dle" aos seus daemons, indicando-lhes que se devem desligar.

A comunicação entre o master Trinoo e os agentes é feita via pacotes UDP na porta 27444 ou via pacotes TCP na porta 1524. A senha padrão para usar os comandos é "l44adsl" e só comandos que contêm a substring "l44" serão processados.

A comunicação entre os agentes e o master Trinoo também é através de pacotes UDP, mas na porta 31335.

TFN – TRIBE FLOOD NETWORK

O TFN é uma ferramenta distribuída usada para lançar ataques DoS coordenados a uma ou mais máquinas vítimas, a partir de várias máquinas comprometidas. Além de serem capazes de gerar ataques do tipo UDP flood como o Trinoo, uma rede TFN pode gerar ataques do tipo SYN flood, ICMP flood e Smurf/Fraggle. Neste tipo de ataque é possível forjar o endereço origem dos pacotes lançados às vítimas, o que dificulta qualquer processo de identificação do atacante.

No caso específico de se fazer uso do ataque Smurf/Fraggle para atingir a(s) vítima(s), o flood de pacotes é enviado às chamadas "redes intermediárias" que consolidarão o ataque, não diretamente às vítimas.

O controle remoto de uma master TFN é realizado através de comandos de linha executados pelo programa cliente. A conexão entre o atacante e o cliente pode ser realizada usando qualquer um dos métodos de conexão conhecidos, tais como: rsh, telnet, etc. Não é necessária nenhuma senha para executar o cliente, no entanto, é indispensável a lista dos IPs das máquinas que têm os daemons instalados.

O mecanismo empregue pelos servidores TFN para comunicarem com os seus clientes é particularmente interessante porque é puramente baseado em pacotes ICPM com resposta de eco ("echo reply"). Tais pacotes são normalmente gerados em resposta a mensagens ICMP de pedido de eco ("echo request") que são produzidas pelo comando "ping" para verificar o acesso à rede de uma máquina.

Algumas versões da aplicação cliente usam criptografia (Blowfish) para ocultar o conteúdo desta lista.

A comunicação entre o cliente TFN e os daemons é feita via pacotes ICMP_ECHOREPLY. Não existe comunicação TCP ou UDP entre eles. Tanto a aplicação cliente (comumente encontrada sob o nome de tribe) como os processos daemons instalados nas máquinas agentes (comumente encontrados sob o nome de td), devem ser executados com privilégios de usuário root.

STACHELDRAHT

Baseado no código do TFN, o Stacheldraht é outra das ferramentas distribuídas usadas para lançar ataques DoS coordenados a uma ou mais máquinas vítimas, a partir de várias máquinas comprometidas. Como sua predecessora TFN, ela também é capaz de gerar ataques DoS do tipo UDP flood, TCP flood, ICMP flood e Smurf/fraggle.

Funcionalmente, o Stacheldraht combina basicamente características das ferramentas Trinoo e TFN, mas adiciona alguns aspectos, tais como: criptografia da comunicação entre o atacante e o master; e atualização automática dos agentes.

A idéia de criptografia da comunicação entre o atacante e o master surgiu exatamente porque uma das deficiências encontradas na ferramenta TFN era que a conexão entre atacante e master era completamente desprotegida, obviamente sujeita a ataques TCP

conhecidos (hijacking, por exemplo). O Stacheldraht lida com este problema incluindo um utilitário "telnet criptografado" na distribuição do código.

A atualização dos binários dos daemons instalados nos agentes pode ser realizada instruindo o daemon a apagar a sua própria imagem e substituí-la por uma nova cópia (Solaris ou Linux). Essa atualização é realizada via serviço rpc (514/tcp).

Uma rede Stacheldraht é composta por um pequeno número de masters onde rodam os programas clientes (comumente encontrados sob o nome de mserv), e um grande número de agentes, onde rodam os processos daemons (comumente encontrados sob o nome de leaf ou td). Todos devem ser executados com privilégios de root.

Como foi mencionado anteriormente, o controle remoto de um master Stacheldraht é feito através de um utilitário "telnet criptografado" que usa criptografia simétrica para proteger as informações que trafegam até o master. Este utilitário se conecta em uma porta TCP, comumente na porta 16660.

Diferencialmente do que ocorre com o Trinoo, que utiliza pacotes UDP na comunicação entre os masters e os agentes, e do TFN, que utiliza apenas pacotes ICMP, o Stacheldraht utiliza pacotes TCP (porta padrão 65000) e ICMP HOREPLY.

TFN2K - TRIBE FLOOD NETWORK 2000

A ferramenta Tribe Flood Network 2000, mais conhecida como TFN2K, é mais uma ferramenta de ataque DoS distribuída. O TFN2K é considerado uma versão sofisticada do seu predecessor TFN. Ambas ferramentas foram escritas pelo mesmo autor, Mixter.

Da mesma forma que ocorre no TFN, a vítimas pode ser atingida por ataques do tipo UDP *flóod*, TCP *flood*, ICMP *flood* ou *smurf/fraggle*. O daemon pode ser instruído para alternar aleatoriamente entre estes quatro tipos de ataque.

O controle remoto do master é realizado através de comandos via pacotes TCP, UDP, ICMP ou os três de modo aleatório. Estes pacotes são criptografados usando o algoritmo CAST. Deste modo, a filtragem de pacotes ou qualquer outro mecanismo passivo, torna-se impraticável e ineficiente.

Diferentemente do TFN, esta ferramenta é completamente "silenciosa", isto é, não existe confirmação (ACK) da recepção dos comandos, a comunicação de controle é unidirecional. Ao invés disso, o cliente envia 20 vezes cada comando confiando em que, ao menos uma vez, o comando chegue com sucesso.

De um modo geral, os binários das ferramentas DDoS têm sido comumente encontrados em máquinas com sistema operacional Solaris ou Linux. No entanto, o código fonte dos programas pode ser facilmente portado para outras plataformas.

Ainda em relação às ferramentas, vale lembrar que a modificação do código fonte pode causar a mudança de certas propriedades da ferramenta, tais como: portas de operação, senhas de acesso e controle, nome dos comandos, etc. Isto é, a personalização da ferramenta é possível.

4.5.6.2 Prevenção e detecção de ataques de negação de serviços

Não existe uma "solução mágica" para evitar os ataques DDoS, o que pode ser feito é aplicar certas estratégias para amenizar o ataque, como:

- **Incrementar a segurança do host:** Uma das características principais do DDoS é a formação de uma rede máquinas comprometidas atuantes como masters ou agentes. Aumentar o nível de segurança das máquinas é uma das maneiras para dificultar a formação dessas redes.
- **Aplicar filtros anti-*spoofing***: Nos ataques DDoS, geralmente atacantes escondem seus endereços IP através de técnicas de *spoofing* para dificultar a identificação da origem do ataque. Para evitar essas incidências, podem ser implementados filtros anti_*spoofing* em:
    a) Provedores de acesso, na entrada dos roteadores, de modo que ele garanta que as redes dos seus clientes não coloquem pacotes forjados na Internet.
    b) Redes conectadas à Internet, na saída dos roteadores de borda garantindo assim que eles próprios não enviem pacotes forjados na Internet.

- **Limitar banda por tipo de tráfego**: Alguns roteadores permitem limitar a banda consumida por tipo de tráfego na rede. Algumas técnicas de ataque DDoS lançam um flood de pacotes ICMP ou TCP SYN, por exemplo, o sistema pode ser configurado para limitar a banda que poderá ser consumida por esse tipo de pacotes. Assim, mesmo que ocorra negação para alguns tipos de serviços, a máquina alvo não ficará totalmente comprometida e o administrador poderá tomar as medidas para encerrar o processo de intrusão.
- **Evitar que a rede seja usada como amplificadora**: Sendo que algumas das

ferramentas DDoS podem lançar ataques smurf (ou fraggle), que utilizam o mecanismo de envio de pacotes a endereços de broadcasting, recomenda-se que sejam implementadas em todas as interfaces dos roteadores diretivas que previnam o recebimento de pacotes endereçados a tais endereços. Isto evitará que sua rede seja usada como "amplificadora".

- **Planejamento prévio dos procedimentos de resposta**: Nesse ataque o tempo é crucial, para isso um prévio planejamento e coordenação são críticos para garantir uma resposta adequada no momento que o ataque está acontecendo. Este planejamento deverá incluir necessariamente procedimentos de reação conjunta com o provedor de backbone.

4.5.6.3 Ferramentas de detecção específicas

Foram desenvolvidas muitas ferramentas para detectar o uso de ferramentas para ataque DDoS, que eventualmente sejam instaladas nos sistemas, dentre elas:

O NIPC (National Infraestructure Protection Center) disponibilizou uma ferramenta de auditoria local chamada "find_ddos" que procura no filesystem os binários do cliente e daemon das ferramentas de Trinoo, TFN, Stacheldraht e TFN2K.

Dave Dittrich, Marcus Ranum e outros desenvolveram um script de auditoria remota, chamado "gag" que pode ser usado para detectar agentes Stacheldraht rodando na sua rede local. Dave Dittrich, Marcus Ranum, George Weaver e outros desenvolveram a ferramenta de auditoria remota chamada "dds" que detecta a presença de agentes Trinoo, TFN e Stacheldraht.

O uso do mecanismo de *spoofing* nos ataques DDoS dificulta em muito a identificação do atacante. Assim, o momento em que se pode fazer um rastreamento e chegar ao verdadeiro responsável é no exato momento em que está ocorrendo o ataque, portanto é imprescindível a comunicação rápida com os operadores de rede do seu provedor de acesso/backbone.

O *modus operandi* dos invasores já é vastamente conhecido pela mídia, o que tem permitido a prevenção da maioria dos ataques. No entanto, a DDoS é uma técnica que ainda não foi encontrada uma forma garantida de precaução, pois as solicitações vêm de lugares diferentes e aparentemente confiáveis, impossibilitando a distinção de clientes verdadeiros de solicitações falsas, provenientes de um ataque DDoS.

5 TECNOLOGIAS DE PREVENÇÃO

A Web foi projetada sem muitas preocupações com segurança, o objetivo principal era disponibilizar informações de uma forma mais amigável que os recursos disponíveis na época.(FIGUEIREDO, 2000).

Hoje a Web é um conjunto da integração de banco de dados, intercâmbio eletrônico de dados – EDI (Eletronic Data Interchange), comércio eletrônico e até projetos colaborativos via videoconferência.

Com o potencial das tecnologias da Web, muitas empresas empregam esta tecnologia para capacitar parceiros comerciais, clientes e funcionários, colocando cada vez mais informações valiosas. Com isso, empresas obtêm reduções de custo, aumento de receita e vantagens competitivas, em contrapartida, tem se tornado alvo de constantes ataques e a segurança passa a ser um ponto de crucial importância.

Existe uma série de conceitos que são essenciais para a segurança de dados em redes de computadores. Conceitos que, de uma forma geral, tornaram-se uma espécie de requisitos dos novos mecanismos de segurança associados à problemática da segurança (MENEGHEL, 2000; IXPRESS, 2000; FIGUEIREDO, 2000):

Integridade dos dados: A integridade dos dados garante que dados não sofreram modificações não autorizadas, ou foram corrompidos durante a transmissão.

Ataques a integridade consistem em alterações maliciosas de dados, programas, mensagem e até informações armazenadas na memória.

Para a garantia da integridade dos dados podem ser usadas técnicas de detecção de modificação, normalmente associadas a detecção de erros em bits, em blocos, ou erros de seqüência introduzidos por enlace e redes de comunicação. Porém cabeçalhos e fechos com informações de controle devem ser protegidos contra modificações, para que a análise da integridade possa ser efetivada.

Confidencialidade: É o processo que protege as informações contra acesso de pessoas não autorizadas, desconhecidas ou terceiros, mesmo que tenham obtido acesso de forma ilícita ao canal de comunicação. Essa garantia deve ser fornecida não somente aos dados, mas também aos cabeçalhos, pedidos e respostas.

Dados secretos devem ser protegidos quando armazenados ou transmitidos, podendo utilizar métodos de criptografia para esse fim, uma vez que o requisito privacidade é particularmente importante quando estão em jogo transações comerciais que envolvem dados de caráter pessoal e confidencial.

Autenticação: A autenticação garante que os usuários sejam realmente quem afirmar ser, e que o servidor e os sistemas host utilizados também sejam autênticos, para isso pode ser utilizadas diferentes técnicas. A escolha do mecanismo de autenticação depende do ambiente onde se dará a autenticação:

- parceiros e meios de comunicação confiáveis: a identificação pode ser efetuada por senhas.
- parceiros confiáveis e meios de comunicação não confiáveis: a proteção pode ser fornecida por emprego de métodos de criptografia.
- parceiros e meios de comunicação não confiáveis: esse caso exige técnicas que impeçam que uma identidade negue que tenha enviado ou recebido determinados dados, podendo ser utilizados mecanismos de assinatura digital, ou que envolvam o compromisso de um terceiro confiável.

A autenticação do usuário é o processo de verificação da identidade proclamada pelo usuário. Embora uma solicitação de um cliente origine-se de um usuário específico, é importante verificar o pedido antes de determinar o acesso às permissões. Quando um servidor Web recebe uma solicitação de um Web browser, o Web server vê apenas uma Uniform Resource Locator (URL). Quando a URL estiver solicitando acesso a uma página crítica que contenha informações comerciais de acesso restrito, será de suma importância autenticar o usuário para determinar se a sua identificação é válida.

A maioria dos servidores Web suporta níveis simples de autenticação via Identificação e senha, porém na maioria das vezes não é suficiente frente aos riscos.

Autorização: É o processo de atribuir os acessos permitidos para cada usuário. A permissão de acessos inclui uma especificação, tal como, se o usuário possui permissão para ler, escrever, ou alterar um dado diretório ou arquivo.

Controle de acessos: O controle de acesso é normalmente implementado através da utilização de *firewall*; especificado pelo administrador do sistema ou pelo dono do recurso.

Negação de Serviço: Esta é uma das mais sérias ameaças na Web e

provavelmente a mais difícil de prevenir. Este tipo de ataque consiste em ações que consomem recursos de um determinado serviço que se encontra disponível com solicitação falsas, deixando-os indisponíveis a solicitações reais.

Segurança Física e de Pessoal: Procedimentos operacionais devem ser definidos para delinear responsabilidades do pessoal que interage com um dados sistema. A segurança de qualquer sistema depende, em última instância, da segurança física dos seus recursos e do grau de confiança do pessoal que opera o sistema. Ou seja, não adianta utilizar mecanismos sofisticados de segurança se os intrusos puderem acessar fisicamente os recursos do sistema.

Registro de Eventos (arquivos de logs): A manutenção de registro de eventos facilita a detecção e investigação de violações da segurança ocorridas em um sistema, tornando possível a realização de auditorias de segurança.

Apesar de serem pontos crucias para disponibilização do serviço Web, muitos não são analisados corretamente, sendo um dos maiores problemas de segurança; assim como com outros serviços de rede, o mau gerenciamento. Sistemas distribuídos com uma má configuração pode significar um desastre. Sistemas crescem e também crescem em sua complexidade. Se não se tem uma boa configuração torna-se difícil o emprego de uma política de gerenciamento satisfatória.

5.1 POLÍTICA DE SEGURANÇA

O componente mais importante de uma estratégia geral de segurança é ter uma política de segurança robusta. Uma política de segurança define o que se considera aceitável em termos de segurança e o que será feito no caso de se detectar violações do que for estabelecido. (CERT-RS, 2000)

Em termos gerais, políticas são diretivas de gerenciamento que estabelecem metas comercias de organização, fornecem uma estrutura de implementação para alcançar objetivos dessas metas e atribuem responsabilidades e domínios ao processo. Deve incluir regras detalhadas definindo como as informações e recursos da organização devem ser manipulados, o que é, e o que não é permitido em termos de segurança, durante a operação de um dado sistema. (BERNSTEIN et al.,1997).

A política de segurança deve prover gerenciamento dos riscos que a empresa incorre ao buscar objetivos comerciais.

O primeiro passo a ser tomado na implantação de uma política de segurança é definir as metas, conferindo ou quais restrições impor. Para uma política de segurança sólida, é essencial a avaliação de determinados critérios como: (CERT-RS, 2000; BERNSTEIN et al., 2000; ANONYMOUS, 1997):

- Flexibilidade: a medida em que a empresa evoluir, é preciso rever e atualizar esta política de modo a refletir as novas aplicações e tecnologias bem como as novas comunidades de usuários.
- Pertinência: reflexão quanto aos objetivos comercias da empresa.
- Aplicabilidade: elaboração deve estar baseada nas realidades do ambiente computacional.
- Atualidade: a política deve manter sempre sofrendo atualizações, constantemente brechas são descobertas.
- Facilidade de uso: O sistema mais fácil de usar permitiria acesso a qualquer usuário e não requereria senha; isso quer dizer, não haveria nenhuma segurança. Requerer senhas torna o sistema um pouco menos conveniente, mas mais seguro.
- Custo/Beneficio: existem diversos custos para implantação de segurança: o monetário, (custo de comprar hardware de segurança e software como *firewalls* e geradores de senhas descartáveis); desempenho (cifragem de dados consome tempo); e facilidade de uso. Porém há diversos riscos: perda de privacidade (a leitura de informação por indivíduos sem autorização), perda de dados (a corrupção ou deleção de informação) e a perda de serviço (por exemplo, o preenchimento do espaço de armazenamento de dados, uso de recursos computacionais, e negação de acesso a rede). Devendo ser importante a avaliação tanto do custo contra cada tipo de perda.
- Integração: a política de segurança na Internet deve ser bem integrada com a política geral de proteção aos sistemas de informação.
- Serviços oferecidos: Cada serviço oferecido apresenta um próprio risco de segurança. O administrador deve avaliar e decidir se o risco que o mesmo apresenta excede em valor o beneficio do serviço, desabitando-o ou mantendo-o ativo.

5.2 CUIDADOS COM O SERVIDOR WEB

Os servidores Web são projetados para receber requisições anônimas de computadores não autenticados e para responderem de maneira eficaz. Assim, tais servidores estão sujeitos a atacantes que buscam falhas no código fonte dos servidores para conseguirem acesso ao sistema. Por sua vez os programas de servidores Web são complicados e geralmente têm o código fonte disponível, possibilitando assim ao atacante achar vulnerabilidades e atacar o sistema.

A escolha do servidor influenciará diretamente sobre o tipo de segurança que poderá ser provida. De um modo geral, quanto mais recursos o servidor possuir, maiores serão as probabilidades da existência de brechas de segurança.

Após a escolha do servidor Web, deve ser efetuada as configurações. Um dos maiores problemas de segurança é o mau gerenciamento; que podem resultar em grandes desastres. Sem uma boa configuração, torna-se difícil o emprego de uma política de gerenciamento satisfatória.

Alguns tópicos irão referir-se a servidores Unix/Linux, por ser a plataforma mais utilizada atualmente.

5.2.1 Estrutura de Diretórios

A estrutura hierárquica de diretórios de um servidor Web é composta de dois diretórios:

- Raiz do servidor: tem informações de controle do servidor, como: arquivos de configurações, aplicações adicionais, etc.
- Raiz de Documentos: mantém o conteúdo "público" (informações disponibilizadas via conexões HTTP). Geralmente este é um sub-diretório do diretório raiz do servidor.

Ao se "disparar" um servidor Web, todas as informações abaixo da raiz do diretório de documentos poderão ser disponibilizadas publicamente a menos que se tenha alguma restrição de acesso. Um dos erros mais comuns é se executar o servidor Web como "root", o que trás vulnerabilidades para seus sistemas. Uma solução muitas vezes adotada é rodar o servidor Web como usuário "nobody", que também trás vulnerabilidades. Algumas

aplicações também podem estar sendo executadas como "nobody" e, portanto, o servidor Web passa a ter seus mesmos privilégios. Uma boa estratégia é criar um usuário qualquer (e também um grupo) específico para o servidor Web (exemplos: web, www, webserver, etc.). Assim, acessos ao sistema de arquivo serão restringidos a este usuário, e também eventuais ataques feitos ao servidor também seriam afetados ao usuário Web específico do servidor.

5.2.2 CGI Scripts

Páginas interativas geralmente requerem dados do servidor web para serem manipulados em conjunto com as entradas dos usuários. Para isto, devem ser utilizadas diferentes ferramentas, por exemplo, o CGI (*Commom Gateway Interface*), que é uma das formas mais convenientes de manipular dados no servidor web. CGI é um tipo de middleware, que permite a interoperabilidade requeridada por este conteúdo. Esse protocolo age como uma linguagem de programação ou *script*, e o servidor Web. (ANONYMOUS,1997).

Em muitos servidores web os CGIs são escritos por diferentes pessoas com vários níveis de experiência. Para programadores UNIX experientes é relativamente fácil encontrar furos em CGI através do protocolo HTTP. Para facilitar a administração e diminuir os riscos potenciais de segurança deve ser definido somente um diretório de CGI executáveis (em muitos servidores UNIX ele é chamado de cgi-bin), é uma idéia interessante para minimizar os riscos de segurança.

O processo CGI funciona da seguinte maneira: (BERNSTEIN et al., 1997)

1. O usuário, executando um navegador da Web, seleciona um item que ativa um script CGI. O cliente envia ao servidor o nome do script e os dados associados a ele.
2. O servidor configura o ambiente de operação adequado e chama o programa CGI. O ambiente permite a troca de dados entre o servidor e o programa CGI, que pode ou não estar no mesmo host. O servidor envia os dados de entrada do cliente para o servidor nesse momento.
3. O programa CGI é executado e retorna a saída ao servidor, este por sua vez, passa as informações para o cliente.

Aplicações CGI escritas sem cuidados com a segurança podem causar sérios problemas a vulnerabilidades de servidores Web:

- Um programa legítimo pode ter um furo de segurança que permita a um invasor executar comandos não autorizados, ou descobrir informações sobre o sistema.
- Se o upload de arquivos for permitido, invasores poderão incluir seus próprios programas CGI malignos no servidor.

Todas entradas de usuários nos scripts CGI devem ser examinadas, para detectar presença de técnicas malignas nos campos, incluídas por invasores.

Um exemplo da situação acima seria nos servidores Unix, nos quais comandos podem ser escritos em uma mesma linha utilizando o separador ponto-e-vírgula, o atacante pode inserir um comando mal intencionado em um campo qualquer, como: badguy@malicious.com; mail badguy@malicious.com </etc/passwd. Se o CGI não foi bem desenvolvido, poderá passar a linha diretamente para o shel do Unix, e o arquivo de senha ser enviado o atacante (BERNSTEIN et al., 1997).

Um CGI script sendo executado sob o mesmo UID do servidor Web não é necessariamente um fato ruim, mas se alguma aplicação CGI possui um furo de segurança que permite que um atacante execute programas sob UID do servidor Web, isso pode acarretar um problema bastante sério ao site.

Uma maneira de contornar este problema é via "wrappers", isto é, programas que envolvem outros programas a fim de alterar a maneira que estes operam. Assim, em ambientes onde usuários escrevem independentemente aplicações CGI, é uma boa estratégia isolá-los um dos outros, isto é, implementar mecanismos em seu servidor de maneira que acessos de scripts de um usuário não venham a interferir em dados de outros usuários. Existem outras ferramentas que também tratam este problema fazendo com que aplicações CGIs sejam executadas sob o UID do próprio usuário, isto é, o dono da aplicação CGI.

5.2.3 SSI – *Server-Side Include*

As *server-side includes* são diretivas que você coloca em uma página HTML e que o servidor *web* interpreta antes de enviá-la ao *browser* do usuário, permitindo dar muito mais vida e recursos que facilitem a manutenção de um web site e implementação de

serviços especiais.

SSIs são bastante úteis, mas podem também ser "caros" computacionalmente, acabando com a portabilidade e podendo abrir sérios furos de segurança. Se esta funcionalidade não é necessária para o seu servidor, é melhor desabilitá-la.

Diversos ataques podem ser realizados inserindo código SSI em um campo avaliado como um documento HTML pelo servidor Web, permitindo que o atacante execute comandos localmente e ganhe acesso ao próprio servidor Essa brecha pode ser solucionada utilizando um script pré-analisador que leia todos arquivos HTML e remova as linhas SSI antes de passar ao servidor.

Quando um servidor retorna um arquivo contendo comandos SSI, é necessário ler cada linha do arquivo à procura da sintaxe especial de comandos SSI, ou seja, é realiza uma análise do arquivo. Os comandos SSI podem estar localizados em qualquer parte dos arquivos HTML. Isso significa que o servidor precisa realizar um esforço extra para localizar os eventuais comandos contidos no arquivo da HTML e que arquivos SSI são retornados mais lentamente ao cliente da web que os arquivos comuns da HTML. Quanto maior for o número de arquivos SSI que o servidor precise processar, maior a carga de trabalho fazendo que o servidor trabalhe com uma menor velocidade.

Se o servidor utilizar programas CGI e forem seguidas as mesmas restrições para utilização de SSI, não haverá riscos adicionais.

5.3 MECANISMOS DE PROTEÇÃO

5.3.1 *Firewall*

Um *firewall* é um sistema ou grupo de sistemas através do qual flui o tráfego de dados entre duas redes distintas de computadores, permitindo que se implemente uma política de segurança que determine o que pode ou não passar de uma rede para outra. Os *firewalls* podem oferecer proteção contra ataques a protocolos ou aplicações individuais, protegem contra ataques *spoofing* e têm relativa flexibilidade de configuração, oferecendo várias restrições para diferentes tipos de tráfego (GONÇALVES,1997).

O *firewall* é um dispositivo de hardware dotado de duas placas de redes (uma ligada à rede corporativa e outra ligada à Internet), rodando software específico de análise

e roteamento de pacotes. Como todo pacote enviado de uma rede a outra passa obrigatoriamente pelo sistema, o *firewall* tem chance de analisá-lo, verificando se apresenta risco, podendo descartá-lo antes que alcance o destino.

Os *firewalls* podem ser vistos como um par de mecanismos: um para bloqueio de tráfego e outro permitir o tráfego. Alguns *firewalls* dão maior ênfase ao bloqueio de tráfego, enquanto outros enfatizam a permissão do tráfego, o que deve ser sempre observado é uma configuração do *firewall* de acordo com a política de segurança da organização que o utiliza, estabelecendo o tipo de acesso que deve ser permitido ou negado. (LIMA, 2000).

Existem basicamente duas políticas para implementação de um *firewall* :

- *Default Permti* – Padrão permitir: Qualquer host ou protocolo que não esteja dentro do conjunto de regras que irão resultar no bloqueio de dados, será permitido por *default*; e
- *Default Deny*: Essa política ao contrário da anterior, permite a passagem ou acesso a subrede somente os especificados na política.

Em geral as empresas utilizam três tipos de *firewalls* (BERNSTEIN et al., 1997):

- filtros de pacotes baseados no roteador e no host;
- filtros "inteligentes" baseados no host; e
- gateways de aplicações baseadas no host.

A seguir, cada um dos três tipos de firewalls serão analisados.

5.3.1.1 Filtros de pacotes

Filtragem de pacotes é o processo de selecionar pacotes que podem trafegar entre duas redes, baseando-se nas informações obtidas nos cabeçalhos dos pacotes e em um conjunto de regras de filtragem. A figura abaixo apresenta uma implementação, *screening router*, onde é utilizado um roteador com filtros de pacotes.

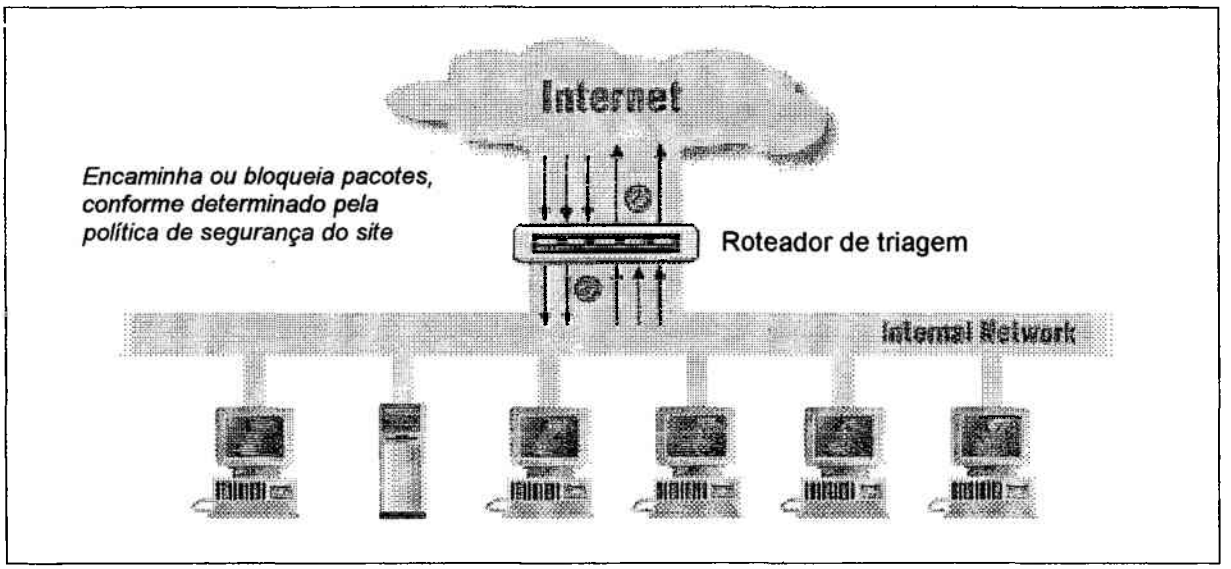

Figura 5.1: Implementação de um Screening Router. Fonte: SWICKY (2001)

A maioria dos roteadores disponível no mercado já possui filtros de pacotes incorporados nos softwares, e dispõem das seguintes informações:

- direção do tráfego;
- interface na qual o tráfego foi recebido ou para qual se destina;
- tipo de protocolo (IP, ICMP, TCP, UDP, etc)
- endereços de origem e destino;
- a porta TCP ou UDP de origem e destino;
- informação sobre "estado" do TCP.

Os filtros de pacotes podem ser considerados inseguros, pois eles não fazem nenhuma filtragem na parte de dados dos pacotes, assim como não isolam a rede interna, já que pacotes vindos de hosts na Internet podem chegar, caso sejam permitidos pelas regras de filtragem no filtro de pacotes, aos hosts dentro da rede interna. Dentre outras limitações dos filtros de pacotes é não suportarem log de tráfego, não sendo possível identificar os pacotes que atravessaram e os abandonados pelo filtro, e também não possuem capacidade de desempenhar autenticação do usuário.

Ao usar um *firewall* de filtragem de pacotes, é possível controlar que clientes tenham acesso ao Web site ou ao sistema back-end, mas não se pode fazer este controle por usuário. Os filtros de pacotes não têm o conceito de usuário; são eficazes como uma linha de frente de defesa, mas não oferecem o nível de segurança mais forte.

5.3.1.2 Filtros Inteligentes

Filtros Inteligentes são filtros baseados em hosts que desempenham as mesmas funções gerais que filtros de pacotes padrão, porém, possuem maior funcionalidades e estão livres de algumas limitações presentes no filtro padrão.

Os chamados filtros inteligentes são aplicações executadas em, por exemplo, computadores ligados ao roteador e à rede interna. O tráfego de um lado para outro se dá (ou não) conforme as regras estabelecidas nas aplicações. Apesar desta solução requerer um equipamento extra, ela nos dá uma série de vantagens sobre os filtros baseados em roteador, principalmente no que diz respeito à monitoração de acesso.

Roteadores, quando possuem algum tipo de log, não guardam informações muito precisas sobre as tentativas de conexão na (ou da) rede interna, enquanto os filtros inteligentes possuem vários níveis de logs, nos quais é possível (e bastante recomendável) perceber os tipos de tentativa de acesso, e até definir certas ações caso um evento em especial relacionado à segurança aconteça.

Outra característica dos filtros inteligentes é a tentativa de implementar um controle de pacotes UDP, guardando informações sobre eles e tentando "improvisar" o flag ACK. Montando-se uma tabela de pacotes UDP que passam, pode-se comparar os pacotes UDP que retornam e verificar se eles são uma resposta ou se são uma tentativa de novo contato.

5.3.1.3 Servidor Proxy

O servidor proxy não permite tráfego direto entre duas redes; a comunicação requer o estabelecimento de duas conexões, uma entre o remetente proxy e a outra entre o proxy e o destinatário. O proxy correlaciona as duas comunicações através do número de sessão TCP (que está presente em todos os pacotes) e ecoa os dados que vêm da conexão *x* para a conexão *y* e vice-versa. Antes dos pacotes serem ecoados de uma conexão para outra, o servidor proxy faz uma análise verificando se há abusos de segurança no protocolo utilizado na porta específica e decide se o pacote deve passar ou ser descartado. (INVARSORPAGE, 2000).

A principal vantagem dos servidores proxy é o fato de esconderem o host interno

do servidor de destino, útil para empresas que não desejam que a rede interna seja visível ao mundo externo.

Os servidores proxy compreendem o conceito de usuários, mas são sistemas especializados que oferecem grande segurança às custas de transparência. Em outras palavras, um filtro de pacotes é completamente transparente aos usuários, mas os usuários percebem quando estão passando por um servidor proxy. Um servidor proxy tipicamente requer autenticação dos usuários antes de determinar se eles podem se comunicar com os sistemas solicitados.

O uso de um servidor proxy para executar a autenticação de usuários e o controle de acesso a um site Web é, normalmente, efetuado através do emprego de um programa especial executando em um servidor proxy denominado proxy HTTP. Este software possibilita a colocação de um servidor proxy *firewall* no lado não confiável do seu servidor Web. Graças às grandes características de segurança do proxy HTTP é possível garantir que qualquer usuário que consiga passar por este servidor e acessar o seu servidor Web seja um usuário conhecido e digno de confiança.

Tabela 5.1: Comparação entre filtros de pacotes e servidor proxy

Filtragem de Pacotes	**Servidor Proxy**
Nenhum controle de nível de usuário	Possibilidade de autenticação do usuário
Conexão direta entre redes	Sem conexão direta entre redes
Capacidade limitada de logging	Grande capacidade de logging
Software agregado a outro dispositivo de rede	Sistema de segurança dedicado
Configuração extremamente simples	Configuração possivelmente complexa

Uma prática comum e eficiente é utilizar a combinação de filtragem de pacotes juntamente com o servidor de proxy, pois, se comparadas as duas técnicas, (tabela 5.1) percebe-se que as mesmas se complementam.

5.4 CRIPTOGRAFIA

A palavra criptografia tem origem grega (kriptos = escondido, oculto e grifo = grafia) e define a arte ou ciência de escrever em cifras ou em códigos, utilizando um

conjunto de técnicas que torna uma mensagem incompreensível, chamada comumente de texto cifrado, através de um processo chamado cifragem, permitindo que apenas o destinatário desejado consiga decodificar e ler a mensagem com clareza, no processo inverso, a decifragem. (TRINTA e MACEDO, 2000)

Uma das formas de se evitar que informações confidenciais sejam acessadas indevidamente é através da codificação ou cifragem de dados, conhecida como criptografia, fazendo com que apenas as pessoas às quais estas informações são destinadas, consigam compreendê-las.

Existem diversas técnicas de criptografia utilizadas como meio eficaz de proteção a informações suscetíveis a ataques, estando elas armazenadas em um computador ou trafegando pela rede; as quais visam prover uma comunicação segura, serviços básicos de autenticação, privacidade e integridade dos dados. (TRINTA e MACEDO, 2000)

A cifra é um método utilizado criptografar mensagens; técnica na qual o conteúdo da mensagem é cifrado através da mistura e/ou substituição das letras da mensagem original. O processo de deciframento é obtido fazendo-se o processo inverso ao ciframento. Os principais tipos de cifras são analisados nos itens a seguir.

5.4.1 Cifras de substituição

Esse método consiste na troca um caracter ou um bloco de caracteres por outro na mensagem cifrada, de acordo com uma tabela de substituição. A decifragem é feita realizando-se a substituição inversa, de forma a restaurar os caracteres do texto original. (PISTELLI,2000).

- **Substituição monoalfabética**: cada caracter é substituído por outro, de acordo com uma tabela ou uma regra simples. A cifra mais antiga conhecida é a chamada Cifra de César, onde cada caracter é substituído por três caracteres adiante do alfabeto. Assim, A é substituído por D, B por E, etc, até W por Z, X por A, Y por B e Z por C. Para a decifragem, cada caracter é retrocedido de três posições.

  Pode-se usar outros valores ao invés de 3 posições de deslocamento, que constitui a chave de ciframento. Como existem apenas 26 chaves, é um método que visa proteger textos com pequeno grau de sigilo, este método ainda é empregado nos dias de hoje, pelo

algoritmo ROT13 (por exemplo na Usenet News), onde os caracteres avançam 13 posições.

Uma melhoria que pode ser empregada neste método é fazer um mapeamento individual de cada caracter, ao invés de utilizar um deslocamento constante, porém exige o uso de uma tabela, com uma entrada para cada caracter do alfabeto.

- **Substituição monofônica**: opera como o método anterior, mas agora cada caracter da mensagem original pode ser mapeado para um ou vários caracteres na mensagem cifrada, evitando a linearidade. Por exemplo, a letra A seria substituída pelos números 5, 13, 25 ou 56; a letra B por 7, 19, 31 ou 42, e assim por diante.
- **Substituição polialfabética**: realiza uma combinação várias substituições monoalfabéticas, de acordo com algum critério ou chave, em que letras da mensagens são rodadas seguidamente, porém com valores distintos. Por exemplo, poderiam ser usadas 4 tabelas em alternância a cada quatro caracteres: a primeira para os caracteres localizados nas posições 1, 5, 9, 13, etc; a segunda nas posições 2, 6, 10, 14, etc; a terceira nas posições 3, 7, 11, 15, etc, e a quarta nas posições 4, 8, 12, 16, etc.
- **Substituição por polígramos**: é o método que durante a substituição que utiliza grupos de caracteres em vez de trabalhar com caracteres individuais. Assim, por exemplo, se forem considerados trigramas, "ABA" poderia ser substituído por "RTQ", "ABB" por "KXS", etc. Contudo as tabelas de substituição aumentam rapidamente (26^2 para digramas, 26^3 para trigramas, e assim por diante).
- **Substituição por deslocamento:** as letras não são trocadas sempre por uma letra n posições a frente no alfabeto, cada letra tem um valor associado para a rotação através de um critério. Por exemplo, no caso da chave ser igual a 12345, a primeira letra é trocada pela letra que está 1 posição a frente no alfabeto, a segunda pela que está 2 posições a frente, e assim por diante, repetindo a chave se necessário.

Os métodos de cifras de substituição podem ser quebrados se analisado a freqüência de cada caracter no texto cifrado e compará-las as freqüências que apresentam em um texto normal em um determinado idioma. As vogais têm maior freqüência que as consoantes e alguns caracteres possuem freqüência baixíssima em relação aos demais. (PISTELLI, 2000).

A utilização de várias tabelas de cifragem é uma tentativa de minimização do problema citado anteriormente, levando em consideração também que quanto maior a chave mais seguro é o método.

5.4.2 Cifras de transposição

Nestes métodos, não ocorre substituição do caracter, ele permanece, o que sofre alterações é ordem como aparece disposto, sendo alterado a mensagem de acordo com alguma regra ou função (que também podem estar baseadas em alguma chave). A palavra CIFRAS poderia ser cifrada e ser retornada como SIRAFC.

5.4.3 Máquinas de Cifragem

Um código trabalha com grupos de caracteres de tamanho variável, ao contrário da cifra. Cada palavra é substituida por outra. Quebrar um código equivale a quebrar uma gigantesca substituição monoalfabética onde as unidades são as palavras e não os caracteres. Para isso deve-se usar a gramática da língua e analisar a estrutura das frases. Máquinas de cifragem baseiam-se em engrenagens que têm tamanhos diferentes e que giram a velocidades diferentes, obtendo uma substituição polialfabética com chave de 26n, onde n é o número de engrenagens. (PISTELLI,2000)

5.5 O USO DE CHAVES CRIPTOGRÁFICAS

Os algoritmos de criptografia fazem uso de uma chave para controlar a encriptação e decriptação. As chaves são elementos fundamentais em um criptosistema, que interagem com os algoritmos, como a ilustra a figura.

Uma chave é uma cadeia aleatória de bits utilizada em conjunto com um algoritmo. Cada chave distinta faz com que o algoritmo trabalhe de forma ligeiramente diferente. (MAIA, 2000)

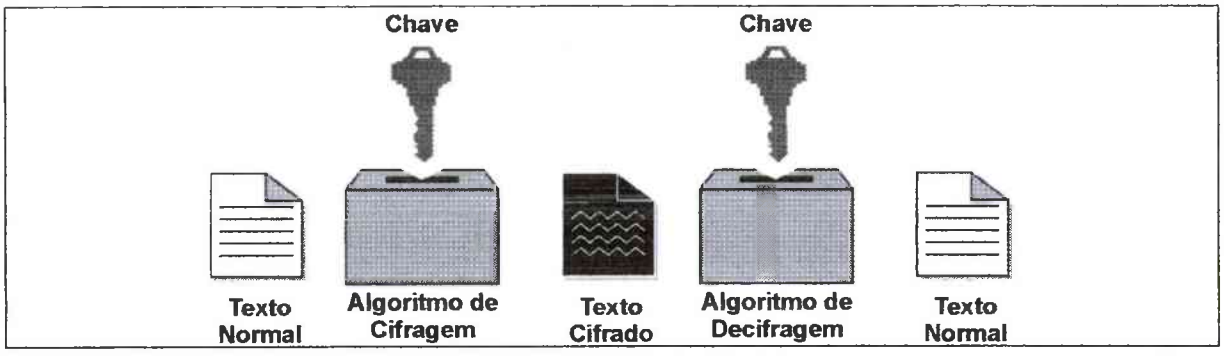

Figura 5.2: Utilização de chaves em algoritmos criptográficos Fonte: TRINTA e MACEDO (2000)

O uso de chaves na criptografia refere-se ao acesso ou não à informação cifrada; devendo o usuário possuir a chave correta para poder decifrar as mensagens.

Na criptografia moderna, as chaves são longas seqüências de bits. Visto que um bit pode ter apenas dois valores, 0 ou 1, uma chave de três dígitos oferece 8 possíveis valores para a chave. Sendo assim, quanto maior for o tamanho da chave, maior será o grau de confidencialidade da mensagem.

As chaves de encriptação e decriptação não precisam ser iguais, separando os algoritmos baseados em chaves em duas classes: simétrico (ou chave privada) e assimétrico (ou chave pública).

5.5.1 Algoritmo de chave simétrica

Este método de criptografia utiliza uma única chave para criptografar e decifrar os dados e consiste em substituir as letras de uma mensagem pela n-ésima letra após a sua posição no alfabeto. Assim o texto criptografado produzido para um mesmo texto normal pode varia de acordo com o valor de n, que é a chave utilizada nos processos de codificação e decodificação do método.

Apesar de sua simplicidade, existem alguns problemas na criptografia simétrica:

- Como cada par necessita de uma chave para se comunicar de forma segura, para um uma rede de n usuários seria necessário algo na ordem de n^2 chaves, um fator dificultador para a gerência das chaves;
- A chave deve ser trocada entre as partes e armazenada de forma segura, o que nem sempre é fácil de ser garantido;

– A criptografia simétrica não garante a identidade de quem enviou ou recebeu a mensagem (autenticidade e não-repudiação).

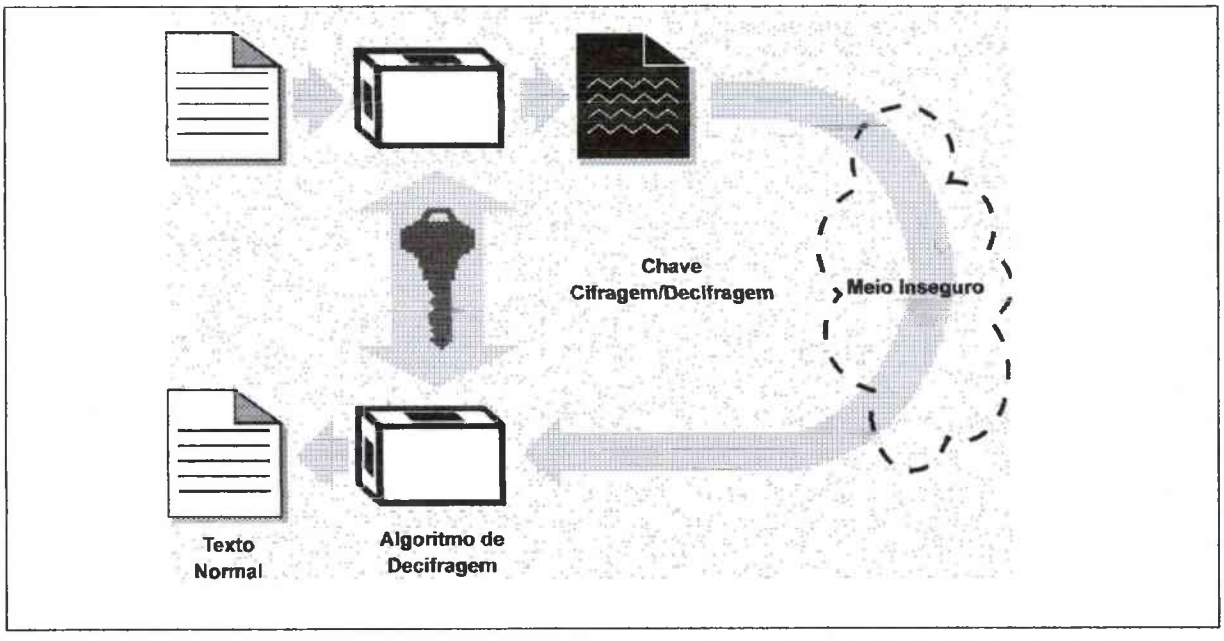

Figura 5.3: Funcionamento do algoritmo da chave privada Fonte: TRINTA & MACEDO (2000)

O exemplo mais difundido de cifrador computacional de chave única é o DES (*Data Encryption Standard*), desenvolvido pela IBM em 1977. O DES cifra blocos de 64 bits (8 caracteres) usando uma chave de 56 bits mais 8 bits de paridade (o que soma 64 bits).O algoritmo inicia realizando uma transposição inicial sobre os 64 bits da mensagem, seguida de 16 passos de ciftagem e conclui realizando uma transposição final, que é a inversa da transposição inicial. Para os 16 passos de ciftagem usam-se 16 sub-chaves, todas derivadas da chave original através de deslocamentos e transposições.

Um passo de cifragem do DES tem dois objetivos básicos: a difusão e a confusão. A difusão visa eliminar a redundância existente na mensagem original, distribuindo-a pela mensagem cifrada. O propósito da confusão é tomar a relação entre a mensagem e a chave tão complexa quanto possível.

O DES apesar de permitir cerca de 72 quadrilhões de combinações (256), seu tamanho de chave (56 bits) é considerado pequeno, pode ser quebrado pelo método da "força bruta", tentando-se todas as combinações possíveis de chave.

5.5.2 Algoritmo de chave assimétrica

A criptografia assimétrica, também chamada de chave pública, está baseada no conceito de par de chaves: uma chave privada e uma chave pública. Neste método, torna-se pública e acessível a todos a chave de ciframento, sem que haja quebra na segurança. Dessa forma cada usuário tem uma chave de ciframento, de conhecimento público, e outra de deciframento, secreta. (GARFINKEL e SPAFFORD, 2000).

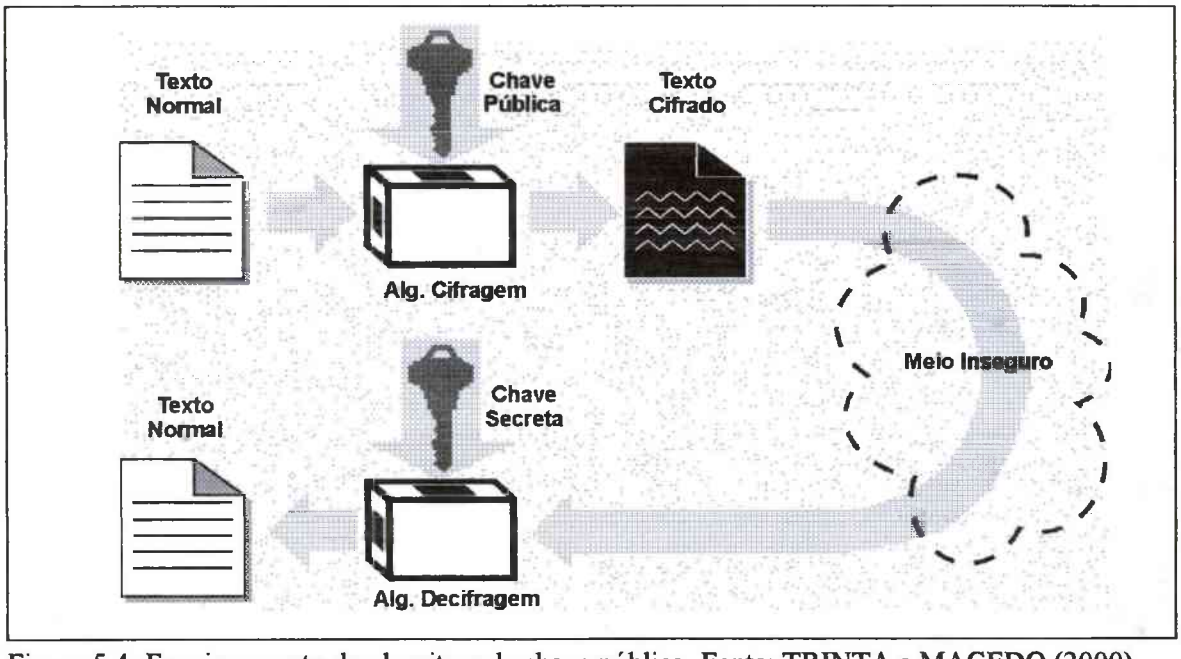

Figura 5.4: Funcionamento do algoritmo da chave pública. Fonte: TRINTA e MACEDO (2000)

Esta técnica é muito utilizada para o envio de mensagens onde se deseja que somente o destinatário, portador da chave privada, consiga ler a mensagem. O emissor da mensagem utiliza a chave pública para criptografar a mensagem e a envia, quando o receptor receber a mensagem utilizará a chave privada para decifrar a mensagem. Este esquema provê a confidencialidade dos dados e também autenticação pois somente o proprietário da chave privada será capaz de decifrar a mensagem.

A grande vantagem deste sistema é permitir que qualquer um possa enviar uma mensagem secreta, utilizando apenas a chave pública de quem irá recebê-la. Como a chave pública está amplamente disponível, não há necessidade do envio de chaves como é feito no modelo simétrico. A confidencialidade da mensagem é garantida, enquanto a chave privada estiver segura. Caso contrário quem possuir acesso á chave privada terá acesso às mensagens.

Um exemplo de algoritmo simétrico é o RSA inventado por Ron Rivest, Adi Shamir e Len Adleman, 1977 no MIT. É, atualmente, o algoritmo de chave pública mais amplamente utilizado, além de ser uma das mais poderosas formas de criptografia de chave pública conhecidas até o momento. O RSA utiliza números primos.

A premissa por trás do RSA baseia-se na dificuldade de fatorar números primos muitos grandes, onde é fácil multiplicar dois números primos para obter um terceiro número, mas muito difícil recuperar os dois primos a partir daquele terceiro número. Gerar a chave pública envolve multiplicar dois primos grandes; qualquer um pode fazer isto. Derivar a chave privada a partir da chave pública envolve fatorar um grande número. Se o número for grande o suficiente e bem escolhido, então ninguém pode fazer isto em uma quantidade de tempo razoável.

Uma chave RSA de 512 bits foi quebrada em 1999 pelo Instituto Nacional de Pesquisa da Holanda, com o apoio de cientistas de mais de 6 países, levou cerca de 7 meses e foram utilizadas 300 estações de trabalho para a quebra.

5.6 ASSINATURAS DIGITAIS

Dentre os questionamentos feitos por todos que acessam a Internet e que por esse meio fazem negócios ou estabelecem relações de qualquer nível, a segurança é a que mais preocupa, pois como qualquer outro compromisso ele pode ser desvirtuado e comprometer as partes envolvidas. Por isso da preocupação em resguardar os meios de segurança dos documentos e a necessidade do meio técnico absolutamente pessoal para o sucesso dessas relações.

É bem verdade que mesmo no mundo real, assinaturas são falsificadas e documentos são forjados, porque o ser humano é falho e será sempre assim, tanto no campo real como no campo virtual.

Existem sistemas de proteção para todo o tipo de fraudes nos documentos materiais e a legislação, tanto civilista quanto penalista, dispõe de normas inibidoras e repressoras para defender a sociedade, como deve ser; mas no mundo virtual esse é o novo desafio e esta é uma pequena abordagem sobre esse assunto que urge ser estudado e discutido para que as novas relações possam alcançar o fim esperado, ou seja, a globalização completa e segura (BRASIL, 2000).

Um dos mecanismos utilizados na Internet para transações comerciais é a assinatura digital, que tem como objetivo garantir a autenticidade de um documento e evitar a repudiação pela pessoa que o assinou. Portanto uma vez verificada a assinatura é possível posteriormente provar para um terceiro (juiz em tribunal) que só o proprietário da chave primária poderia ter gerado a mensagem.

O Congresso norte-americano aprovou projeto de lei que confere validade legal a assinaturas digitais, permitindo seu uso em transações via Internet. A nova lei não estabelece quais tecnologias podem ser utilizadas para garantir a veracidade das assinaturas, permitindo que as pessoas e empresas envolvidas em cada transação escolham o método que julgar mais confiável. No Brasil, há projetos para uma lei nacional de comércio eletrônico que está tramitando no Congresso. A promessa da Comissão Especial de Comércio Eletrônico é que este ano deva se chegar a um denominador comum e que se aprove uma lei para beneficiar as transações eletrônicas. (COMPUTERWORLD, 2001)

As assinaturas digitais utilizam funções *hash* e algoritmos de chave pública. Hash são funções que, para uma string digital de tamanho qualquer, calculam um identificador digital confiável de tamanho fixo, usualmente de 16 ou 20 bytes (VICECONTI, 1999). Qualquer alteração da string original, por menor que seja, gera uma alteração significativa no valor do "hash" correspondente. Deve ser impossível gerar intencionalmente uma string digital, diferente da original, que resulte no mesmo valor hash.

A assinatura é obtida aplicando uma função de hash na mensagem, assim obtém-se um valor hash para esta; o qual é criptografado e utilizando uma chave privada do autor, é adicionado ao documento e o enviado. Em seguida, criptografa a mensagem junto com sua assinatura, utilizando a chave pública do receptor. Este, ao receber a mensagem, deve, primeiramente, decifrá-la com sua chave privada, o que garante sua privacidade. Em seguida, "decifrá-la" novamente, para verificar a assinatura utilizando a chave pública do autor, garantindo assim sua autenticidade.

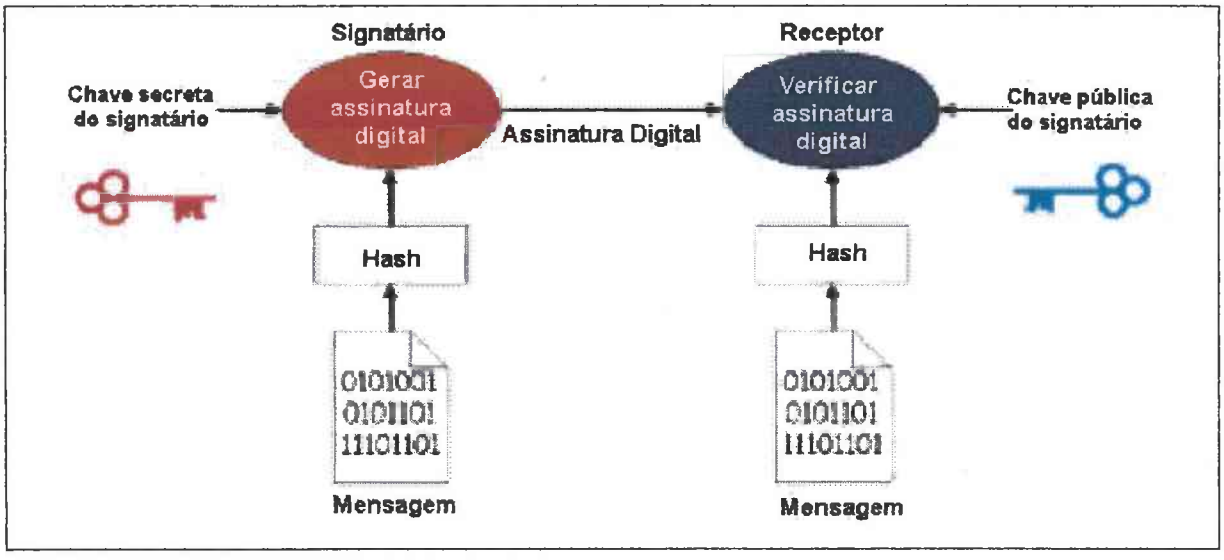

Figura 5.5: Geração de uma assinatura digital

Propriedades da assinatura digital:

- A assinatura é autêntica: quando um usuário usa a chave pública de A para decifrar uma mensagem, ele confirma que foi A e somente A quem enviou a mensagem;
- A assinatura não pode ser forjada: somente A conhece sua chave secreta:
- O documento assinado não pode ser alterado: se houver qualquer alteração no texto criptografado este não poderá ser restaurado com o uso da chave pública de A;
- A assinatura não é reutilizável: a assinatura é uma função do documento e não pode ser transferida para outro documento;
- A assinatura não pode ser repudiada: o usuário *B* não precisa de nenhuma ajuda de *A* para reconhecer sua assinatura e *A* não pode negar ter assinado o documento.

5.7 CERTIFICADOS DIGITAIS

Certificado de Identidade Digital, também conhecido como Certificado Digital, associa a identidade de um titular a um par de chaves eletrônicas (uma pública e outra privada) que, usadas em conjunto, fornecem a comprovação da identidade, ou seja garantem que uma chave pública, relativa a uma assinatura digital, pertença a uma determinada entidade ou pessoa.

Serve como prova de identidade, reconhecida diante de qualquer situação onde

seja necessária a comprovação de identidade.

O Certificado de Identidade Digital é emitido e assinado (chancelado) por uma Autoridade Certificadora Digital (Certificate Authority), como a Thawte (certificadora da ArtNET), que emite o Certificado. Para isso, esta autoridade usa as mais avançadas técnicas de criptografia disponíveis e de padrões internacionais (norma ISO X.509 para Certificados Digitais), para a emissão e chancela digital dos Certificados de Identidade Digital.

Um certificado contém três elementos:

Informação de atributo: informações sobre o objeto que é certificado. No caso de um usuário, pode ser incluído: nome, nacionalidade e e-mail, entre outros dados.

Chave de informação pública: chave pública da entidade certificada. O certificado atua para associar a chave pública à informação de atributo, podendo a chave pública ser qualquer chave assimétrica.

Assinatura da Autoridade em Certificação (CA): necessária para adicionar credibilidade ao certificado. Quem recebe o certificado verifica a assinatura e acreditará na informação de atributo e chave-pública associados, isso se confiar na Autoridade em Certificação.

Os certificados podem ser: certificados de autoridades, certificados de servidores, certificados pessoais e certificados para distribuição de software.

O Certificado Digital tem como objetivos principais a transmissão de dados de forma confiável e a chave pública do proprietário a terceiros; permite que terceiros transmitam arquivos, mensagens, etc. criptografados ao proprietário do Certificado Digital e permitir que terceiros verifiquem as assinaturas digitais geradas e anexadas nas mensagens pelo proprietário do Certificado Digital com o uso da sua chave privativa de criptografia

5.8 DADOS BIOMÉTRICOS

Os dados biométricos estão sendo trabalhados como uma maneira confiável de autenticação, reconhecer a identidade de uma pessoa através de características fisiológicas como impressões digitais, imagens da retina, identificação da voz , imagens térmicas ou em base em alguma coisa do seu comportamento como o padrão dos toques de digitação

ou escrita manual. (STRAUCH, 2000)

A autenticação através de dados biométricos tem alto custo de implantação, porém apresenta um alto grau de proteção e precisão, variando de acordo com técnica adotada. Segundo a empresa LG, o índice de erros em sistema baseado na leitura de íris é de 0,0008%, a impressão digital em torno de 5% e o reconhecimento facial entre 5% a 10%. O cálculo feito para margem de erro baseia-se em dois fatos: o sistema rejeitar um usuário legítimo ou autenticar um usuário falso. (GOYA, 2000).

Já se encontra disponível no mercado algumas técnicas de identificação por dados biométricos, como:

Impressão digital: Uma das mais simples e custo mais baixo entre as técnicas de autenticação biométrica; necessitando de um scanner de alto grau de precisão para capturar as impressões digitais e um programa que realize o tratamento da imagem capturada.

Algumas soluções utilizam no processo de reconhecimento, a criação de uma string com tamanho entre 250 e 300 bytes; com informações das distâncias entre as linhas da impressão digital capturada; criando dessa forma uma identidade única e exclusiva para cada usuário. (NETWORK COMPUTING,2000).

Reconhecimento da face: O reconhecimento é feito com base no mapeamento da geometria e proporções da face; registrando diversos pontos delimitadores, definindo proporções, distâncias, tamanhos e formas de elementos do rosto como o nariz, olhos, queixo e orelhas.

Identificação pela íris: a precisão do reconhecimento pela íris é maior que pela face ou impressão digital, pois é praticamente imutável. Para implementação é necessária a utilização de uma câmera digital e o programa de reconhecimento. A câmera, pode ser monocromática, pois as cores não são significativas para a identificação, que pode ser feita mesmo com lentes e óculos não muito escuros.

Reconhecimento da voz: Esse processo de identificação faz uma análise dos padrões harmônicos da voz capturada por um microfone ou telefone. A presença de ruídos no ambiente e o estado emocional momentâneo da pessoa, podem comprometer a precisão do reconhecimento, podendo o usuário ser rejeitado em casos de rouquidão, fatiga ou exaltação.

Reconhecimento da assinatura manuscrita: Assim como a assinatura pessoal, aceita em cheques e diversos documentos, o reconhecimento da assinatura pode ser utilizado para autorização do usuário. Esse sistema, também conhecido como sistema de

verificação de assinatura dinâmico, pode fazer a análise de diversas características como: pressão da caneta, velocidade, identificação dos movimentos da caneta no ar e os pontos em que a caneta é levantada do papel; dificultando tentativas de fraude através de cópias.

O reconhecimento da assinatura manuscrita necessita uma prancheta digitalizadora pequena e/ou uma caneta especial, para captura do comportamento da escrita, e transcrição para um modela matemático para identificação.

Reconhecimento da dinâmica da digitação: Essa técnica verifica a forma de autenticação do usuário, analisando a velocidade, tempo entre o acionamento de cada tecla, intensidade de pressão, tempo que se mantém a tecla pressionada da digitação de usuário e senha no logon. Como não requer nenhum hardware alternativo, é uma técnica barata e que pode ser adotada em redes corporativas, de forma natural, devido sua transparência para o usuário.

Tecnologias futuras: Existem técnicas que ainda estão em fase de pesquisas e desenvolvimento; possivelmente no futuro existirão sistemas de autenticação biométrica baseados na análise de DNA, odores e salinidade do corpo, formato de orelhas e padrões das veias por imagens térmicas do rosto ou punho.

As técnicas de reconhecimento por dados biométricos podem ser implementadas de duas formas: verificação e identificação. A primeira, o usuário apresenta-se e o sistema confere a veracidade da informação; a identificação se faz através da busca em um banco de dados, comparando os registrados até que encontre um idêntico.

Um dos maiores obstáculos para crescimento de adoção de sistemas biométricos é o alto custo de tecnologia, embora venha caindo continuamente. Outra barreira é a resistência de alguns usuários, pela análise de características pessoais.

5.9 FERRAMENTAS DE PREVENÇÃO

5.9.1 Wrappers

Os wrappers são programas que tem como objetivo o aumento de segurança dos serviços da Internet, sendo uma ferramenta de geração de log e gerenciamento de conexão.

O aumento de segurança proporcionado pelo wrapper varia desde uma maior capacidade de log, verificações se um dado host não esta pretendendo ser outro host

através de falsificação de IP (IP Spoofing), falsificações de nomes de host (*Host Name Spoofing*), até aumento das restrições de cada um dos serviços.

Como principal exemplo de wrapper temos o TCP-Wrapper, o qual é uma camada que se encontra antes do servidor, podendo colocar restrições de quem poderá se conectar. TCP Wrappers é uma biblioteca onde são *linkadas* aos programas (servidores), possibilitando que estes tenham controle de acesso ao serviços. Este controle de acesso é baseado em domínio, ou seja, deve-se definir quais domínios podem, ou não, se conectar a um determinado serviço.

Esta técnica de proteção, como qualquer outra, não deve ser utilizada isoladamente. Ainda fica sujeito do ataque de *spoofing*, caso falsificarem o IP no seu domínio. É um ataque um pouco mais sofisticado, se a pessoa conhece alguma máquina que está autorizada a se conectar a determinado tipo de serviço, tem formas de personificar este domínio autorizado e fazer uma conexão.

Características do TCP Wrappers (ANONYMOUS, 1998):

- Controla os acessos;
- *Loga* os acessos autorizados;
- Permite que se configure reações: se alguém de domínio não autorizado tentar acessar, pode-se configurar para tomar alguma medida reação, como por exemplo: enviar um e-mail para o administrador notificando, ou então fazer um *finger* nesta máquina tentando descobrir mais informações desta pessoa. Um problema que pode ocorrer caso a pessoa esteja falsificando o endereço IP.

O processo wrapper fica completamente transparente para os diversos serviços de rede, não interage com o server nem com o cliente. Desta forma o wrapper fica independente de aplicação, permitido que funcione com diversos tipos e versões de programas de rede. O fato de estar sendo executado apenas no momento inicial da conexão do server com o cliente torna insignificante a carga por ele gerada no servidor. Finalmente, isto também o torna invisível para os usuários externos, que nem saberão que ele existe.

5.9.2 PGP – *Pretty Good Privacy*

O PGP destina-se à comunicação segura via e-mail, para isto utiliza-se de criptografia de chave pública/privada de até 1024 bits.

A mensagem enviada é criptografada com a chave pública do destinatário. Para que o usuário possa decifrar a mensagem ele decifra com a sua chave secreta a qual somente ele possui. Desta forma, qualquer pessoa que intercepte esta comunicação não conseguirá decifrar a mensagem (HOLSCHUH, 2000).

No PGP dispõe da facilidade de poder assinar uma mensagem, assim a mensagem vai cifrada com a chave secreta do remetente e o destinatário decifra com a chave pública do remetente. Uma mensagem pode ir cifrada e assinada.

PGP é um criptsistema híbrido que combina algoritmos de chaves públicas (assimétricas) com algoritmos convencionais (simétricos), com a vantagem de utilizar a velocidade da criptografia convencional e a segurança da criptografia por chaves públicas. As chaves públicas são mantidas em arquivos que contém a identificação do usuário (i.e. o nome da pessoa), a hora (timestamp) da geração do par de chaves e as chaves propriamente ditas. São usados dois arquivos (key rings) diferentes, um para chaves públicas e outro para as secretas, que podem conter uma ou mais chaves cada um (SWANCH,2000).

As chaves públicas são internamente referenciadas por uma Key JD, que é uma abreviação dessa chave (os 64 bits menos significativos). Enquanto muitas chaves podem ter a mesma identificação do usuário (User JD), nenhuma chave pode ter a mesma Key JD.

PGP faz uso de "message digest" para realizar as assinaturas. "Message digest é o nome que se dá a um conjunto de 128 bits fortemente cifrados, função da mensagem. É algo análogo ao checksum ou ao CRC, que é um código verificador de erros, e representa compactamente a mensagem, usada para detectar mudanças em seu conteúdo. Diferentemente do CRC, entretanto, é computacionalmente impraticável a qualquer pessoa descobrir uma outra mensagem que produza uma mesma "message digest", que ainda é criptografada pela chave secreta para formar a assinatura digital. (SWANCH, 2000).

Os documentos são autenticados por um prefixo que contém o Key JD da chave secreta que foi usada para assiná-lo, o "message digest" do documento devidamente criptografado pela chave secreta do remetente e a hora (*timestamp*) de quando foi realizada a assinatura. O Key ID é utilizado pelo destinatário para relacioná-lo com a chave pública do remetente, a fim de checar a assinatura. O software automaticamente procura a chave pública e a identificação do usuário remetente no arquivo de chaves públicas (SWANCH, 2000).

Arquivos cifrados são prefixados pelo Key ID da chave pública usada para cifrá-la. O receptor usa essa informação para relacionar a correspondente chave secreta que

decifra a mensagem. Da mesma forma, o software do destinatário automaticamente localiza essa chave secreta no arquivo de chaves secretas. Esses dois tipos de arquivos são o principal método de armazenamento e gerenciamento das chaves públicas e privadas. (PISTELLI, 2000).

5.9.3 SSL – *Secure Socket Layer*

O protocolo SSL foi desenvolvido pela Netscape Communications para garantir a segurança de quaisquer dados que estejam em trânsito na Internet, desde que ambos, o servidor e o cliente, apoiem o protocolo.

Para um site poder fazer uso do SSL precisará de um certificado de autenticação "assinado" por uma entidade certificadora, permitindo que o cliente conecte ao Web Site e, de forma transparente, criando um canal de comunicação seguro entre o Site e o Cliente, garantindo autenticação, confidencialidade e integridade dos dados, sendo planejado para autenticar o servidor e opcionalmente o cliente (OLIVEIRA, 1999).

O SSL usa como protocolo de transporte o TCP, e provê uma transmissão e recepção segura dos dados. Uma vez que o SSL reside no nível *socket*, ele é independente das aplicações de mais alto nível, sendo assim considerado um protocolo de segurança independente do protocolo aplicacional (RIBEIRO, 2000).

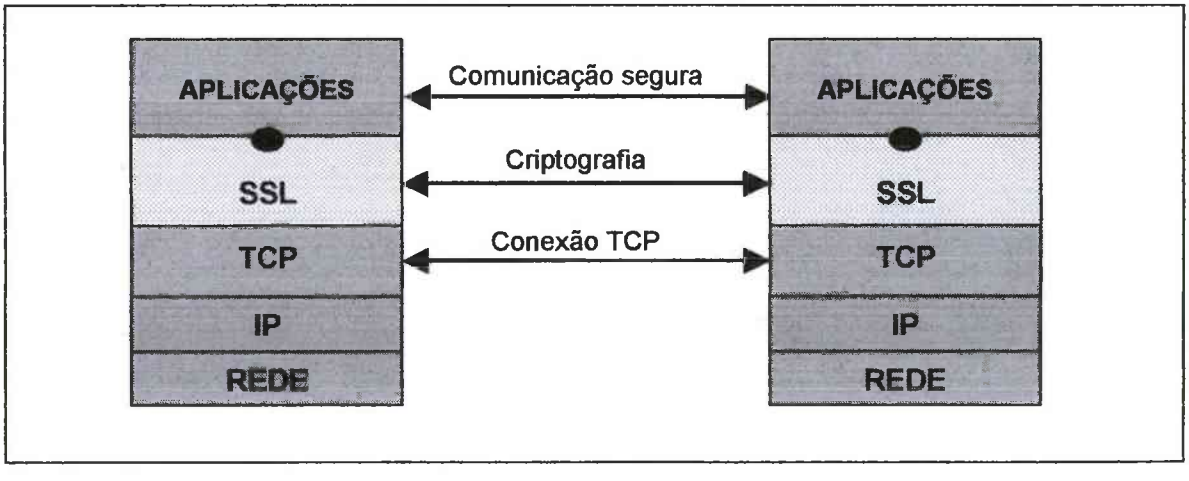

Figura 5.6: Implementação do protocolo SSL

O protocolo SSL atua entre o nível de aplicação e transporte da arquitetura Internet. Possui duas camadas: *SSL Record Protocol*, responsável pelo encapsulamento de outros protocolos de alto nível e a *SSL Handshake Protocol*, recebe os dados a serem

cifrados/decifrados, além de ser responsável pela autenticação do cliente e/ou servidor; negociação do algoritmo criptográfico e suas chaves antes da aplicação receber ou enviar qualquer byte de dados (INFORESTUD@NTE, 2000).

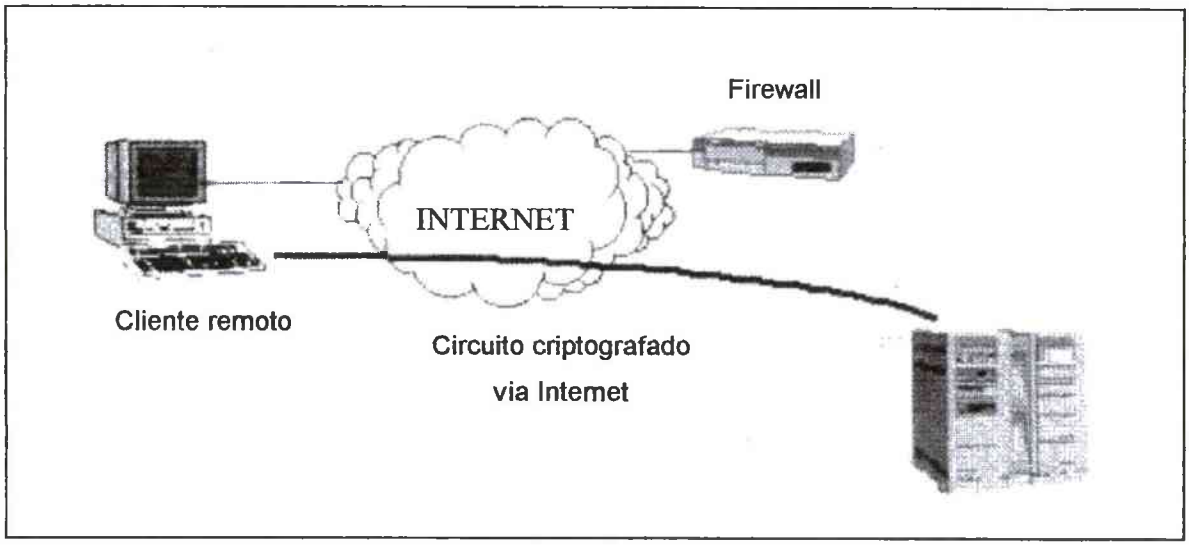

Figura 5.7: Criptografia SSL

5.10 CONSIDERAÇÕES FINAIS

Um ditado muito popular, nos meios de segurança, diz que a resistência de uma corrente é medida pelo seu elo mais fraco. Muitas vezes, redes corporativas com projetos de segurança muito bem elaborados acabam esbarrando em falhas de usuários que expõem o sistema inadvertidamente. E por meio de operações que eles sequer desconfiam que podem estar sendo utilizadas para uma tentativa de invasão. (SOUZA, 1998)

Algumas dicas simples, mas que tornam o sistema menos vulnerável, são dadas a seguir (ANONYMOUS, 2000):

1. Somente coloque o que for necessário na máquina;
2. Nunca coloque compiladores na máquina servidora, isto irá facilitar a vida do invasor;
3. Deve haver servidores de logs específicos e seguros;
4. Nunca tenha apenas um ponto de falha;
5. Use ferramentas que "batem a foto" da forma que o seu servidor está;
6. Analise os logs, não adiante tê-los sem nunca ter os olhado; e
7. Atualize-se, listas de discussões são boas sugestões, dados estatísticos são fornecidos

em diversos *sites*.

Firewalls, VPNs, *sniffers*, filtros de pacotes nos roteadores e tantas outras ferramentas são essenciais, mas certamente ineficazes sem o acompanhamento constante do administrador. E quando os usuários não são instruídos de maneira adequada sobre procedimentos que devem adotar ou evitar para que a segurança passe a ser encarada como conjunto de ferramentas e normas de conduta.

A visão de que a segurança é uma meta estática é errônea. A segurança deve ser vista como resultado de uma evolução constante onde vulnerabilidades podem ser corrigidas, antes que descobertas ou exploradas por intrusos em potencial.

6 O *HACKER* INSERIDO NO DESENVOLVIMENTO TECNOLÓGICO

Um difuso grupo de pessoas freqüentemente chamadas de *hackers* tem sido caracterizado como não ético, irresponsável, e como um perigo para a sociedade. Este capítulo busca resgatar a imagem do *hacker*, identificando o papel do mesmo perante a sociedade e o papel da sociedade perante esta classe. Para isso, contribuições e preocupações tanto da sociedade como do *hacker* serão identificadas, bem como a influência da sociedade sobre o *hacker*, e vice-versa.

Para tal, este capítulo primeiramente relata contribuições dos *hackers*, como: a metodologia de desenvolvimento de *software* de código aberto, advinda da filosofia *hacker*; a exigência dos *hackers* quanto à qualidade de produtos e de serviços; a habilidade *hacker* na identificação de falhas de segurança; a imagem que a sociedade tem perante os *hackers*; e por fim, é discutida a necessidade de resgatar esta classe, enfocando o papel da sociedade para que tal resgate ocorra.

6.1 A FILOSOFIA DO *OPEN SOURCE*

De acordo com BUYS et al. (2001), o impulso inicial para a história do *software* livre foi dado em 1969, quando Ken Thompson, pesquisador do Bell Labs, criou a primeira versão do Unix, o qual era utilizado pelos mainframes que existiam na década de setenta em universidades e grandes empresas. O Unix era distribuído gratuitamente para as universidades e centros de pesquisa, com seu código-fonte aberto. A sigla OSS (*Open source Software*) é a que designa esse tipo de programa, cuja estrutura pode ser modificada por qualquer usuário com conhecimentos em informática. A partir daí foram surgindo novas versões do Unix, igualmente abertas e compartilhadas pelo meio acadêmico.

Em 1971, Richard Stallman, do MIT (*Massachusetts Institute of Technology*), inaugurou o movimento *Open Source*, produzindo, no Laboratório de Inteligência Artificial do MIT, diversos programas com código-fonte aberto. Em 1979, quando a empresa AT&T anunciou seu interesse em comercializar o Unix, a Universidade de Berkeley criou a sua versão do sistema, o BSD Unix. A AT&T se juntou a empresas como

IBM, DEC, HP e Sun para formar a *Open source* Foundation, que daria suporte ao BSD. Em 1984, Stallman criou o Projeto GNU (GNU's Not Unix), com o objetivo de desenvolver uma versão do Unix com código-fonte aberto acompanhada de aplicativos e ferramentas compatíveis igualmente livres. Em 1985, ele publicou o Manifesto GNU e um tratado anti-*copyright* intitulado *General Public License* (GPL), esse tratado criava a *Free Software Foundation*, explicando a filosofia do *software* livre. (BUYS et al., 2001). No Manifesto GNU, Stallman defende que todos os *softwares* deveriam ser livres, ou seja, que seus usuários tenham o direito de acesso ao código fonte do programa, isto não significa que não possa cobrar pelo *software*.

A ideologia de Stallman, embora extremista, fora o princípio para o desenvolvimento de diversos *softwares* livres, os quais, atualmente, são largamente utilizados, não apenas devido ao fácil acesso, mas, principalmente, pela robustez que apresentam. Alguns exemplos de *softwares* livres que vêm incomodando a indústria de *software* são: Linux, Apache, Sendmail, Mozilla, Perl, Python, PHP, os quais detêm uma grande legião de usuários.

Para MANHÃES (2001), o movimento *Open Source* não se trata de algo anarquista anti-*business*, mas de uma alternativa ao modelo de negócio para a indústria de *software*. Ressalta o autor que o valor da liberdade é supremo, o benefício econômico pode ser amplificado pela cooperação e, socialmente, distribuído, estando a eficácia deste modelo, na esfera virtual, plena e fartamente comprovada pela história: o TCP/IP, o SMTP, o HTTP e outros protocolos são frutos de cooperação livre que produziu padrões abertos de *facto*, como também programas pioneiros que testaram, depuraram e validaram tais padrões.

A postura do código fonte aberto é, sem dúvidas, uma abordagem ousada, que contribui não só para o avanço da informática, mas também para o crescimento do indivíduo que participa. Uma declaração neste sentido é dada por Alfredo Kojima, desenvolvedor da interface gráfica WindowMaker, com relação ao Linux, à revista GEEK (1999), onde coloca que no desenvolvimento de um software livre existe o retorno "moral", na forma de gratidão dos usuários; além de servir como vitrine para os autores, onde, a partir do trabalho reconhecido, tem-se melhores oportunidades de emprego.

6.1.1 A Catedral e o Bazer de Raymond

Para esclarecer melhor o conceito de *software* de código fonte aberto (open-source *software*), vale analisar o artigo *The Cathedral and the Bazaar*, de Eric S. RAYMOND (1998)[4], onde o mesmo revela a dicotomia entre os modelos de desenvolvimento de *software* "catedral" e "bazar". O modelo catedral é utilizado no desenvolvimento da maioria dos *softwares* comerciais e o modelo bazar, utilizado por Linus Torvalds no desenvolvimento do Sistema Operacional Linux.

Raymond difere os dois modos de desenvolvimento, a saber:

Catedral: este é o modelo antigo, tradicionalmente usado em empresas de *software* fechado. O controle é extremamente centralizado e dependente do líder: há casos em que o projeto morre com a saída do líder. O processo geralmente é muito demorado.

Bazar: este é o novo modelo de desenvolvimento de *software*. Pode ser comparado a uma feira, onde os comerciantes "vendem" as suas idéias. O bazar deve ter um objetivo claro e praticável. Deve haver um líder, que funciona como catalisador da comunidade de usuários e/ou desenvolvedores, e não um centralizador de poder. Portanto, o projeto é quase independente do líder, ou seja, uma troca de líder não deve abalar o sucesso do projeto. Deve haver um meio de comunicação que una a comunidade (a Internet é um bom exemplo). O processo ocorre muito rapidamente.

Para Raymond, o mais ousado e bem sucedido exemplo de produto desenvolvido no modelo bazar é o sistema operacional Linux, o qual, desde a liberação de seu código na Internet em 1991, vem sendo aprimorado e já está na sua versão 2.4 com cerca de 280 distribuições diferentes (MANHÃES, 2001).

Raymond declara que o estilo de Linus Torvalds de desenvolvimento – libere cedo e freqüentemente; delegue tudo que você possa; esteja aberto a ponto da promiscuidade – veio como uma surpresa, pois, até então, acreditava-se que grandes projetos, como um sistema operacional, se teria dificuldades em ser realizado no modelo bazar. No entanto, não há uma catedral calma e respeitosa para Linus – ao contrário, a comunidade Linux pareceu assemelhar-se a um grande e barulhento bazar, de diferentes agendas e aproximações (adequadamente simbolizada pelos repositórios do Linux, que

---

[4] Este item é baseado no artigo The Cathedral and de Bazaar, de RAYMOND (1998), logo, buscando facilitar a leitura do texto, o ano de publicação será omitido.

aceitaria submissões de qualquer pessoa) de onde um sistema coerente e estável poderia, aparentemente, emergir somente por uma sucessão de milagres.

A experiência de Raymond com o desenvolvimento bazar vinha desde meados de 1980, época em que contribuiu para o projeto GNU, co-desenvolvendo diversos programas (Emacs em modo VC e GUD, xlife, e outros). A experiência adquirida por Raymond o fez diagnosticar que **"Dados bastantes olhos, todos os erros são triviais"**; teoria esta que, de forma natural, permite qualidade dos *softwares* desenvolvidos no modelo bazar, o que resulta também em maior segurança, pois, conforme fora enfatizado no Capítulo 3, os problemas de segurança decorrem, em grande parte, devido a erros no *software*, seja no projeto e/ou na implementação.

Raymond acompanhou o crescimento do Linux e, objetivando testar o modelo bazar através de uma experiência prática, em meados de 1996, liderou um projeto aberto que, inicialmente, chamava-se "popclient", e posteriormente veio a se chamar "fetchmail". O sucesso desse projeto levou-o a constatar 19 teorias (lições) para o desenvolvimento de sofwares, a saber: (RAYMOND, 1998)

1. *Todo bom trabalho de software começa colocando o dedo na ferida de um programador.*
2. *Os programadores bons sabem o que escrever. Os grandes sabem o que reescrever (e reusar).*
3. *Planeje jogar algo fora; você irá, de qualquer maneira. (Teoria de Fred Brooks, "The Mythical Man-Month", Capítulo 11)*
4. *Se você tem a atitude certa, problemas interessantes irão encontrá-lo.*
5. *Quando você perde interesse em um programa, sua última obrigação a fazer com ele é entregá-lo a um sucessor competente.*
7. *Libere cedo. Libere freqüentemente. E ouça seus fregueses.*
8. *Dada uma base grande o suficiente de 'beta-testers' e co-desenvolvedores, praticamente todo problema será caracterizado rapidamente e a solução será óbvia para alguém.*
9. *Estrutura de dados inteligentes e código burro trabalham muito melhor que o contrário.*
10. *Se você tratar seus 'beta testers' como seu recurso mais valioso, eles irão responder tornando-se seu mais valioso recurso.*
11. *A melhor coisa depois de ter boas idéias é reconhecer boas idéias dos seus usuários. As vezes a última é melhor.*
12. *Freqüentemente, as soluções mais impressionantes e inovadoras surgem ao se perceber que o seu conceito do problema estava errado.*
13. *A perfeição (em projetar) é alcançada não quando não há mais nada a adicionar, mas quando não há nada para jogar fora.*
14. *Qualquer ferramenta deve ser útil da maneira esperada, mas uma ferramenta verdadeiramente boa leva ela própria a usos que você nunca esperou.*
15. *Quando estiver escrevendo um software gateway de qualquer tipo, faça tudo para perturbar o conjunto de dados o menos possível – e nunca jogue fora informação a*

não ser que o destinatário force você a isto!

16. *Quando sua linguagem não está perto de um Turing completo, açúcar sintático pode ser seu amigo.*
17. *Um sistema de segurança é tão seguro quanto é secreto. Esteja atento a pseudo-segredos.*
18. *Para resolver um problema interessante, comece achando um problema que é interessante para você.*
19. *Contanto que o coordenador do desenvolvimento tenha uma mídia pelo menos tão boa quanto a Internet, e saiba como liderar sem coerção, muitas cabeças são inevitalmente melhores do que uma.*

Uma das grandes forças do método bazar está no fato de que existe uma enorme quantidade de participantes, fazendo com que os bugs sejam rapidamente eliminados do *software*. Raymond acredita que, no futuro, o *software* de código aberto irá pertencer a pessoas que saibam como jogar o jogo do Linus, pessoas que deixam para trás a catedral e abraçam o bazar. Isto não quer dizer que uma visão individual e brilhante não irá mais ter importância; ao contrário, o estado da arte do *software* de código aberto irá pertencer a pessoas que iniciem de uma visão individual e brilhante, amplificando-a através da construção efetiva de uma comunidade voluntária.

A genialidade de Linus se deu, principalmente, pelo fato de ter sido mediador (e por acreditar) no desenvolvimento de um sistema operacional no modelo bazar, sendo que até então se acreditava que um S.O. deveria ser construído como uma catedral, desenvolvido com extremo esmero por mágicos ou pequenos grupos de magos trabalhando em esplêndido isolamento, com nenhum beta para ser liberado antes do tempo.

O movimento *Open source*, segundo MANHÃES (2001), vem sendo atacado pela classe de analistas de tecnologia, com a justificativa de que tal modelo agride a propriedade intelectual do criador. Linus Torvalds contesta a necessidade de cobrar pela propriedade intelectual, citando os trabalhos desenvolvidos por pessoas como Isaac Newton, Einstein, Rutherford, Bohr, Leonardo da Vinci e uma série de outros gênios que trouxeram benefícios incalculáveis para a humanidade, sem cobrarem pelas propriedades intelectuais.

6.2 DIAGNÓSTICO *HACKER* DA QUALIDADE DE PRODUTOS E SERVIÇOS

Um *hacker*, entrevistado por DENNING (1990), relata que a facilidade de arrombar um sistema revela a falta de preocupação por parte do gerente de sistema em proteger o usuário e os recursos da companhia, ou o fracasso por parte de fornecedores para advertir os gerentes sobre a vulnerabilidade de seus sistemas. Tal *hacker* calculou a taxa de sucesso de invadir em 10-15%, e isso sem gastar mais do que uma hora em qualquer sistema alvo. Outro *hacker* diz que vê mensagens de fornecedores que notificam as falhas aos gerentes, e estes não reagem a tal notícia.

Dentre os fatores que contribuem para que invasões ocorram, lembram FARMER e WIETSE (1994), é o fato de que muitos administradores de segurança não estão adequadamente preparados; outros inferem numa falsa proteção contra os invasores, acreditando que, se tal ato é ilícito, existirá um amparo legal. Ora, sabe-se que roubar é ilícito, mas não por isso deixam-se bens sem resguardo, simplesmente pelo fato de existir o amparo legal.

Muitos profissionais discutem questões morais dos invasores fazendo uma analogia do mundo real com o virtual (digital). A exemplo, MARTIN (1989), para justificar seu julgamento contra a ação de um *hacker* como sendo não ética, faz diversas analogias neste sentido. Martin diz que arrombar um sistema é o mesmo que arrombar uma casa; que fazer download de dados, usar recursos de computação e serviços de telecomunicações é comparado a roubar bens tangíveis. Ora, quando alguém arromba uma casa, o objetivo é roubar bens que são insubstituíveis, e a propriedade, geralmente, é danificada no processo. Em contrapartida, quando um *hacker* invade um sistema, o objetivo é aprender, e o mesmo evita causar danos. Fazer download de informação é fazer cópia, não roubo, pois ainda existe o sistema original. Além disso, informação, tradicionalmente, não é considerada como propriedade. DIBBEL (1990) diz que quando as indústrias de *software* e companhias de telefone reivindicam perdas de bilhões de dólares para pirataria, eles não estão falando sobre bens que desaparecem das estantes, e que poderiam ter sido vendidos. No entanto, não é possível simplesmente transcrever as leis que são aplicadas no mundo real para o mundo digital. É imprescindível que se inicie uma vasta discussão com relação aos crimes digitais, para que uma legislação coerente e específica seja elaborada.

Para FRAIA (2001), os *hackers* surgiram de pessoas curiosas que quiseram provar que eram capazes de quebrar a inteligência da máquina e, por vezes, a humana, conseguindo ter acesso aos dados sem, no entanto, danificá-los. Para tanto, essas pessoas não precisavam apenas de curiosidade, era necessário muito conhecimento da forma utilizada no armazenamento dos dados e da linguagem usada para criar tudo isso. É aí que entram as linguagens de programação, importantes para um verdadeiro *hacker*. Um *hacker* cria muito, explora muito, mas não danifica nada. Em contrapartida, a maioria das pessoas os vem como causadores da falta de segurança de modo geral.

Para o criminologista Jim Thomas, muitas pessoas têm uma visão simplificada do mundo, onde tudo cresce das forças da luz (nós) ou da escuridão (*hackers*), embora muitas das ações de administradores têm semelhanças com as ações de *hackers* (por exemplo, monitorando a comunicação sem autorização e mentindo para adquirir informação), geralmente os administradores rejeitam qualquer tipo de sugestão proveniente de *hackers*. DENNING (1990) relata o depoimento de um *hacker* que tentou colaborar com um administrador, avisando-o da existência de furos de segurança em sua rede, e como gratidão, tal *hacker* fora ameaçado em ser denunciado à polícia, pois o administrador entendera que, como o indivíduo não fora contratado pela empresa, não tinha permissão de perceber tal falha. Ora, se um vizinho avisar que sua porta está aberta, certamente irá agradecê-lo pela gentileza prestada. No entanto, muitos administradores de redes entendem que tais pessoas fazem isso apenas com intuito de extorquir dinheiro, onde, na realidade, tais indivíduos buscam identificar falhas para chamarem a atenção e exporem suas habilidades, atitudes típicas de adolescentes.

Deste fato, pode-se perceber que as pessoas não estão mais acostumadas a serem auxiliadas sem terem que pagar para isso. Tal atitude é agravada devido a concorrência e pela apreensão do indivíduo em perder o emprego. Porém, agindo de forma arredia, as pessoas estão afastando potenciais aliados e amigos, e permitindo que o inimigo se aproxime, e em muitos casos, conquistando mais inimigos.

Um exemplo típico de falha na segurança é citado por BERGLIND (2001). A forma como os arquivos são criptografados no sistema de arquivos (*Encrypted File System*) do Windows 2000, deixa-os à mercê de um ataque. Quando um usuário marca um arquivo para criptografia, outro, de *backup*, chamado `efs0.tmp` é criado; sendo que o original é, então, excluído e recriado, na forma criptografada, a partir do arquivo `efs0.tmp`. Depois, após a criação bem-sucedida do arquivo criptografado, o arquivo temporário (que nunca é

exibido para o usuário) também é excluído. Desse modo, o único arquivo remanescente é o criptografado. Porém, a falha está justamente na exclusão desse arquivo temporário, que é feita como a de qualquer outro arquivo: a entrada no `$mft` é marcada como vazia e os *clusters*, nos quais o arquivo foi armazenado, serão marcados no `$Bitmap` como disponíveis. Entretanto, o arquivo (bem como as informações que o contém) não é fisicamente excluído. As informações presentes no arquivo original, que o usuário criptografou, continuarão existindo no arquivo de *backup* `efs0.tmp`, sob a forma de texto puro (*plain-text*), na superfície do disco, podendo ser recuperadas através de editor de discos, por exemplo: o utilitário dskprobe.exe, que é uma das ferramentas de suporte do Windows 2000.

Na tentativa de evitar a pirataria de *softwares*, técnicas inconvenientes ainda são implementadas nos *softwares*, aumentando a possibilidade de que os usuários tenham uma justificativa moral para romper a proteção. Um exemplo disso, relatado por LETTICE (2001), é o método de proteção do sistema operacional Whistler da Microsoft, o qual, além de ser inconveniente, é arcaico. O método, criado pela Microsoft, consiste em uma combinação de chave de CD e de um código, gerado a partir do *hardware* da máquina que está rodando o *software*; estes códigos geram outro, que é validado pela Microsoft via telefone ou pela Web, e o usuário recebe, então, outra chave que "destrava" o *software*. O usuário não pode usar o mesmo código em duas máquinas diferentes, e, se ele alterar o *hardware* do computador e precisar reinstalar o produto, a chave torna-se inválida. Ou seja: este método dificultará o manuseio para o usuário, e será extremamente complicado para administradores de sistemas, que desejarem realizar várias instalações simultaneamente.

Os exemplos ora citados motivam os *hackers* a reivindicarem produtos e serviços que estejam de acordo com a evolução da informática e que primem pela qualidade, pois empresas, na usura de ganhar mercado, vendem a usuários inadvertidos produtos imaturos e sem qualidade.

6.3 A PERÍCIA *HACKER* NA IDENTIFICAÇÃO DE FALHAS DE SEGURANÇA

Não são raras as notícias como a publicada por HARRISON (2001), o qual noticia que um bretão (natural da Grã-Bretanha) de 20 anos, que invadiu os web sites de dois

bancos mercantis na primeira semana de abril de 2000, jamais poderá ser processado. Chris McNab, que diz ter "um pé no submundo e outro no mundo corporativo", afirma ter um dos melhores empregos da indústria de Tecnologia de Informação (TI). Ele é um *hacker* ético da MIS Corporate Defence Solutions. Seu trabalho, cujo título oficial é o de "analista de segurança de redes", consiste em invadir os web sites de bancos importantes e de multinacionais. McNab declara: "Na primeira semana de abril de 2000 invadi dois bancos mercantis. Este trabalho é fantástico!".

Por £850 ao dia (aproximadamente três mil reais), a empresa MIS testa os sites de companhias para aferir o montante dos prejuízos que uma invasão poderia causar — os testes duram três ou quatro dias, geralmente. Então, a MIS oferece *firewalls* ou sistemas de segurança Web aos seus clientes. "Estou lá para emular os *hackers* da Internet. Preciso estar a par de todas as técnicas usadas por eles, e tenho de identificar todos os caminhos existentes nos sites", diz McNab. Ele afirma poder acessar a maioria dos sites em menos de meia hora. O invasor profissional, que deixou a escola aos 17 anos com nota 10 no exame GCSE[5], trabalhou na administração de sistemas antes de obter o trabalho na MIS, em janeiro de 2000. Quando não está quebrando códigos comerciais de alta segurança, McNab passa o tempo pesquisando as últimas técnicas para *hackers*.

Muitas empresas dos E.U.A. confiam aos *hackers* a tarefa de identificar as falhas de segurança de seus sistemas de computadores. Um dos mais famosos grupos de *hackers* dos E.U.A. – chamado Legion Of Doom – entrou no ramo de segurança de computadores após a formação de sua companhia, a Comsec Security. DEVOST (2001) afirma: "Os garotos do Legion são, agora, armas digitais de aluguel. Se sua companhia possuir um Calcanhar de Aquiles, e você puder pagar pelo serviço, os mais famosos *hackers* da América aparecerão em sua porta e colocarão sua casa digital em ordem. É garantido."

Alguns alegam que este tipo de serviço é mera extorsão, mas os indivíduos não estão dizendo "pague, senão entraremos em seu sistema"; eles estão, simplesmente, oferecendo suas habilidades para proteger as empresas de outros invasores digitais. *Hackers* podem ser usados como um meio de proteção da 'casa digital' dos E.U.A., sendo assim, todos os esforços devem ser empreendidos para não aliená-los do novo mundo digital do qual estão emergindo. (DEVOST, 2001)

Até mesmo os governantes se rendem às habilidades dos *hackers*. Após os ataques

---

5 General *Certificate of Secondary Education* (Certificado Geral de Conclusão do Secundário)

em sites famosos como America Online, Yahoo!, e-bay, AT&T, Intel, Cisco, Sun, etc. ocorridos no início de fevereiro de 2000, Clinton reuniu-se na Casa Branca com uma equipe de peritos de segurança informática, dentre eles um *hacker* conhecido por Mudge, para discutir a (in)segurança na Internet.

A exemplo dos E.U.A., o Brasil também está contratando *hacker* para cuidar das "casas digitais". Em junho deste ano, o *hacker* Ribeiro fora convidado pelo Ministério Público para formar um grupo que ajudará o governo a melhorar o nível de segurança na rede (COSTA, 2001). Em janeiro, o Ministério Público do Rio de Janeiro contratou o ex-*hacker*, Wanderley José de Abreu Júnior[6], para ajudar nas investigações eletrônicas do Estado, especialmente para atuar na Operação Catedral, onde foram investigados e encontrados mais de 50 computadores, no Rio, suspeitos de pedofilia na Internet. Wanderley, com 22 anos, fala com orgulho que aos 17 anos criou um sistema de segurança para uma multinacional do setor de petróleo. Em janeiro deste ano, ele foi nomeado chefe dos Assistentes Técnicos da Coordenadoria de Investigações Eletrônicas, para prestar serviços ao Ministério Público, juntamente com sua empresa. (CONJUR, 2001)

A empresa Excite@Home reforçou sua segurança depois que um *hacker* revelou, em abril deste ano, uma vulnerabilidade que permitia acessar a rede interna da companhia e expor ao público, aproximadamente, 3 milhões de registros de suporte. O *hacker*, conhecido como Adrien Lamo, contatou a empresa após descobrir um servidor que poderia ser usado para invadir partes de sua rede. Dentre os dados, que poderiam ser acessados, estavam um banco de dados de usuários para apoio a clientes, e as configurações de equipamentos e endereços. Depois do ocorrido, a empresa reuniu-se com Lamo, que os auxiliou no projeto de segurança, instalando *firewalls*, restringindo o acesso à rede, implantando programas para evitar ataques e acrescentando *hardware* e *software* projetados para prevenir ameaças à segurança. (ANDRADE, 2001)

Um fato, lembrado por TITTON (1996), confirma a habilidade dos *hackers*. Em 1995, um norte-americano de 19 anos foi contratado por aquela polícia para auxiliar em uma investigação sobre abuso sexual. O trabalho infrutífero de um mês, por parte da polícia, foi resolvido em 45 minutos pelo *hacker*.

Se os *hackers* não encontrassem as vulnerabilidades nos sistemas, tais furos

---

6 Publicado no Diário Oficial do Estado do Rio de Janeiro - Poder Executivo - Parte I - Fls. 27, de 19 de Fevereiro de 2001.

seriam utilizados por grandes criminosos, causando danos maiores.

6.4 IMAGEM PÚBLICA DOS *HACKERS*

Hackers demonstram preocupação sobre sua negativa identidade e imagem pública, os quais vêm sendo retratados como imorais e irresponsáveis.

Para HAFNER (apud DENNING, 1990), a identidade pública de um indivíduo ou de um grupo é generalizada por algumas ações que vão contra a ideologia da sociedade em geral. Por exemplo, fora da comunidade *hacker*, o ato de quebrar sistemas é considerado como crime, e basta isto para delinear a imagem deste grupo. O uso de palavras pejorativas como vândalos, patifes e criminosos conduzem a conotações de alguém mal; *hackers* dizem que eles não são criminosos neste sentido. Hafner observa que Robert Morris, o qual foi condenado pelo lançamento do *worm* na Internet em 1988, fora comparado a um terrorista, embora não tenha destruído dados.

A maioria das invasões que ocorrem, suspeitam-se premeditadamente de invasões de piratas eletrônicos. No entanto, a Módulo, empresa especializada em segurança, apurou que a fonte mais comum de ataques é advinda dos próprios funcionários, com 19%. Em segundo lugar estão os *hackers*, responsáveis por 14% dos incidentes constatados (GREGO, 2000). Outra pesquisa publicada por ROGERS (1999), relata que as estatísticas mostram que 70% das atividades são ocasionadas pelos ex-empregados, e geralmente é por vingança.

Um exemplo específico é relatado por FERNANDES (2001), onde o perito contábil Marcelo Alcides Gomes, especializado em investigar funcionários que roubam a própria firma, descobriu um sofisticado caso de fraude digital. O empregado de uma multinacional montou um esquema para desviar dinheiro da companhia, e o diretor já desconfiava disso. Só que não conseguia provas. Gomes contratou um *hacker* para rastrear a movimentação eletrônica no computador do suspeito e, em pouco tempo, conseguiu as provas que faltavam.

A imagem destrutiva dos *hackers* é também constatada por HUTHEESING e ROSS (1999), onde relatam que as organizações que sofrem algum dano, concluem precipitadamente que foram alvos de *hackers*. No entanto, após perícia, na maioria dos casos, descobria-se que os danos eram oriundos ou por inadvertência ou por má intenção

de funcionários ou ex-funcionários da empresa.

Isso ratifica que não basta apenas utilizar as diversas tecnologias de segurança existentes; mais do que isso, é preciso acompanhar continuamente as pessoas qué norteiam o sistema, permitindo a percepção de comportamentos não costumeiros.

Hackers declaram que os gerentes de sistemas os tratam como inimigos e criminosos, ao invés de ajudantes potenciais na tarefa de compor a segurança dos sistemas. DENNING (1990) relata depoimentos de *hackers* que gostariam de ajudar os gerentes a deixarem seus sistemas mais seguros. Citados na mesma obra, LANDRETH (1989) e GOODFELLOW (1983) também defendem a aproximação entre gerentes de sistemas e *hackers*, tornando-os aliados. Drake sugere oferecer a *hackers* acesso livre em troca de auxílio na segurança, pois a atitude de tratá-los como inimigos não é conducente, e depreciando-os, mais problemas são causados.

Vários *hackers* entrevistados por DENNING (1990), declaram que gostariam de estar em uma posição onde eles têm permissão para hackear sistemas. Eles gostam de pesquisar sobre segurança de computador e gostariam de ensinar como proteger-se contra as vulnerabilidades.

GOODFELLOW (1983) sugere a contratação de *hackers* para trabalhar em "times de tigre" que são comissionados para localizar vulnerabilidade em sistemas por prova de penetração. Baird Info-Systems Safeguards, Inc., uma empresa de consultoria em segurança, reporta que eles têm contratado *hackers* em várias tarefas. Declaram que o *hackers* não tem violado confiança deles nem dos clientes, e que os mesmos têm atuado de maneira excelente. Neste contexto, nota-se que as vulnerabilidades de sistemas podem ser melhor identificados por pessoas que exploraram sistemas.

Segundo DENNING (1990), muito se fala da imaturidade dos *hackers* e da maturidade dos mais velhos, onde estes já sabem discernir o certo do errado, pois preservam a ética. Porém, em certos pontos, é difícil concluir quais dos lados são éticos. Por exemplo, um *hacker* escreveu: "Eu concordo que está moralmente errado copiar qualquer *software* proprietário, porém, eu também penso que é moralmente errado cobrar $6000 por um programa com apenas 25K de tamanho". Percebe-se que não é um caso simples de bem ou maduro (adultos) contra o ruim ou imaturo (*hackers*). As diferenças éticas refletem uma diferença em filosofia sobre a informação e informação que manipula recursos; considerando que *hackers* defendem compartilhamento, nós (adultos) parecemos estar defendendo posse de propriedade. As diferenças também representam uma

oportunidade para examinar nosso próprio comportamento ético e nossas práticas para compartilhamento e proteção de informação.

DENNING (1990) relata que algumas pessoas pedem penalidades mais rígidas para *hackers*, inclusive condições severas de prisão, a fim de dar uma mensagem rigorosa de impedimento aos *hackers*. John Draper, que fora encarcerado nos 1970, argumenta que esta prática agrava o problema. Draper declarou que quando estava encarcerado, fora forçado, sob ameaça, em ensinar para outros presos o conhecimento dele de sistemas de comunicações. Ele acredita que sentenças de prisão só servirão para disseminar o conhecimento dos *hackers* para os criminosos de carreira. Relata que, fora da prisão, nunca havia se aproximado de criminosos, mas no período em que lá esteve, os criminosos tiveram controle sobre dele.

Existem penalidades que estão fora da proporção para os atos cometidos por computador. Goldstein diz que se Kevin Mitnick tivesse cometido crimes semelhantes aos que ele cometeu, mas sem um computador, ele teria sido classificado como um fabricante de dano e talvez teria sido multado em cem dólares pela infração; ao invés, foi pósto em prisão sem fiança (GOLDSTEIN, 1989).

Diante das declarações citadas, constata-se que a imagem dos *hackers* está denegrida de tal forma que qualquer evidência de perdas de dados digitais, acima de qualquer suspeita encontram-se os *hackers*.

6.4.1 Confusão de sentimentos: admiração e rejeição presentes nas notícias/reportagens

GONÇALVES (2001) diz que os termos *hacker* e *cracker* mantêm diferenças básicas, a saber: "[...] o *hacker* como aquele especialista em programação que tenta descobrir falhas em sistemas e programas, sem intenção de gerar danos para as empresas ou sites invadidos. Já o *cracker* faz uso do seu conhecimento para atacar, destruir ou mesmo roubar informações, sendo considerado um criminoso". A definição ora citada realmente é fato e está de acordo com diversos autores. Porém, mais adiante o autor considera que "[...] todo e qualquer tipo de invasão deveria ser considerada crime. Mesmo em diferentes níveis, *hackers*, que apenas encaram a invasão ou a descoberta de falhas como um desafio, e *crackers*, que procuram tirar vantagem de sua habilidade, estão no mesmo lado nessa guerra".

A mídia freqüentemente tem conduzido as notícias de forma tendenciosa e

simplista, provocando um conflito entre leigos e *hackers*.

Neste contexto, poder-se-ia dizer que tais reportagens são assim conduzidas com intuito de simplificar a solução do problema, já que as soluções simplificadas são melhores aceitas, além de serem consideradas mais elegantes. Porém, o que tem ocorrido na busca da simplicidade é a distorção dos fatos; ou seja, neste processo há uma alteração na essência do conteúdo. E isso vem acontecendo não somente nos termos aqui mencionados, mas também em todo e qualquer tipo de notícia que, muitas vezes, confundem os leitores, pois ora são notícias que vangloriam os *hackers* ora os denigrem.

A exemplo disso, COSTA (2001) publica com honras a notícia de que o Brasil é tricampeão em um concurso de *hackers* (para quebra de segurança na rede), promovido pela IDNet durante uma feira mundial de segurança em informática, organizada pela SANS (*System Administration, Networking and Security*), em Baltimore, no Estado de Maryland. Pela terceira vez consecutiva, o carioca Rinaldo Ribeiro vence o concurso. Ribeiro levou meia hora para invadir o sistema, o qual era defendido por uma rede de técnicos. Ele se aproveitou de um defeito no servidor IIS da Microsoft, tendo com isso acesso irrestrito à página do IDNet na Web. A falha no servidor permitia penetrar em qualquer banco de dados, como senhas e saldos bancários.

Pethia da CERT (*Computer Emergency Response Team*) reporta que são raros os casos de invasões maliciosas causadas pelos *hackers*, mas que toda invasão ocasiona desordem e insegurança, pois tanto usuários bem como administradores querem estar seguros de que nada foi danificado. HARVEY (1986) diz que *hackers* causam danos aumentando a paranóia que, em troca, conduz a controles de segurança extremamente restritos, que diminuem a qualidade de vida dos usuários. *Hackers* respondem a estes pontos dizendo que são a favor a bodes expiatórios de sistemas que não são protegidos adequadamente. Eles dizem que a paranóia é gerada por medos infundados e pela distorção da mídia, que a segurança não precisa ser opressiva para evitar a intrusão de *hackers*; e que isso é garantido, principalmente, com boas senhas e sistemas bem escolhidos. (DENNING, 1990).

O termo *hacker* vem sendo amplamente utilizado na mídia sem preconceitos, pois jamais alguém se ofendeu por ter sido assim chamado. Com isso, o termo tornou-se genérico para identificar tanto um indivíduo que entenda profundamente de segurança em sistemas, como também indivíduos que causam danos ou invasões nas redes.

6.4.2 O caso John Draper (*Captain Crunch*)[7]

John Draper é uma das personalidades mais conhecidas no submundo digital, juntamente com Kevin Mitnick. Draper tornou-se popular em 1971, quando descobriu que o apito de brinquedo na caixa de cereal '*Captain Crunch*' podia enganar a rede telefônica e dar-lhe ligações gratuitas[8].

Na ocasião, os fundadores da Apple, Steve Jobs e Stephen Wozniak (que era estudante da Universidade da Califórnia em Berkeley), ficaram tão impressionados com o artigo que procuraram pelo *phreaker*. Após Wozniak entrar em contato com Draper, reza a lenda que ele foi até o dormitório de Wozniak usando estranhos óculos com bigode e chifres, que anunciavam "Sou Eu!" – uma típica entrada exibicionista de Draper, que é bem conhecido por sua insistente curiosidade.

De acordo com o *International Herald Tribune*, Draper ensinou Wozniak e Jobs a arte de construir suas próprias "Caixas Azuis" ('*blue boxes*') — dispositivos eletrônicos que permitem que os usuários ganhem acesso gratuito (e ilegal) à rede telefônica. A lenda diz que os dois aprendizes de empresários saíram vendendo as caixas azuis de porta em porta no campus de Berkeley, muitos anos antes de fundar a Apple Computer.

Entretanto, o artigo da *Esquire* também chamou a atenção do FBI. Ele foi detido e condenado à prisão em várias ocasiões, sob a acusação de *telephone phreaking* (isto é, obter ligações telefônicas gratuitas). Entretanto, Draper não perdeu seu tempo na prisão, e lá projetou o *Easy Writer*, um processador de textos para máquinas Apple que foi distribuído com o primeiro IBM PC em 1981, vencendo Bill Gates na corrida para fazê-lo.

Agora, Draper voltou novamente ao cenário, reunindo-se com alguns parceiros para fundar uma empresa de segurança de *software* e consultoria, denominada ShopIP.

Draper, agora com 57 anos, descreve-se como um *White hat hacker* (*hacker* de chapéu branco), e vê a criação da ShopIP como um modo de pagar à sociedade por suas condutas indiscretas do passado; isto é o que ele afirma em uma entrevista com o *International Herald Tribune*, na qual descreve os problemas que teve no passado. Draper declara: "Eu não sou uma pessoa má, mas estou sendo tratado como uma raposa tentando tomar conta do galinheiro".

---

[7] Relatado por LEYDEN (2001).

[8] As atividades de Draper como um *phreaker* estão documentadas na edição de outubro de 1971

6.5 CONDUZINDO OS *HACKERS* PARA USAREM SUAS HABILIDADES EM PROL DO DESENVOLVIMENTO TECNOLÓGICO

De acordo com a originalidade do termo, os *hackers* são indivíduos apaixonados pela profissão, que se doam completamente por paixão no que fazem, consequentemente, garantem a qualidade em tudo que fazem. No entanto, é notória a validade de resgatar o termo *hacker* para manter-se atos benéficos em suas atitudes, incentivando-os a fazer o que gostam.

Para isso, HARVEY (1986) afirma que os jovens estudantes devem ser estimulados com desafios saudáveis. Harvey acredita que disponibilizar apenas recursos triviais para jovens estudantes, como disquetes e linguagens como Basic e Pascal, é inadequado para o desafio intelectual. Sua recomendação é que estudantes devem ter acesso aos computadores, e devem ser orientados no uso destes recursos superiores, mostrando as suas responsabilidades. HARVEY (1986) relata sobre um projeto que implantou em uma universidade pública de Massachusetts no período de 1979 a 1982. Na época fora instalado um PDP-11/70 e disponibilizado para que estudantes e professores realizassem a administração do sistema. Harvey acreditou que deixando a responsabilidade, dos problemas de usuários mal intencionados, para os próprios estudantes, existiria um esforço potencial de educação. Ele também notou que os estudantes com habilidade e interesse em quebrar senhas eram desencorajados, desta atividade, a fim de resguardar a confiança neles depositada, pois com boas atitudes poderiam conquistar o acesso livre de superusuário.

HARVEY (1986) faz uma analogia interessante entre ensinar computação e ensinar karate. Na instrução do karate, estudantes aprendem técnicas poderosas, as mesmas que são ensinadas aos adultos. Eles dão acesso a uma poderosa arma fatal e, ao mesmo tempo, disciplinas e princípios são ensinados com o intuito de não abusar da arte. Harvey acredita que a razão pela qual os estudantes não usam de forma indevida estas potencialidades, é porque sabem que algo importante está lhes sendo confiado, e querem demonstrar que são capazes.

Sabendo conduzir os jovens que são vistos como rebeldes resultados positivos e qualitativos são obtidos. A maioria dos interessados em se tornar *hackers* são adolescentes

---

da revista *Esquire*, em um artigo denominado "Segredos da Pequena Caixa Azul".

ou jovens, que estão constantemente em busca de reconhecimento e atenção.

A *Secure Music Digital Initiative* (SMDI)[9] ofereceu, em setembro de 2000, um prêmio de 11600 euros (hoje aproximadamente 50 mil reais) ao *hacker* que conseguisse remover a marca de água ou outros sistemas de proteção dos seus arquivos de música. Segundo TEK (2001*b*), além dos *hackers* poderem mostrar publicamente suas habilidades, as empresas envolvidas aproveitam para testar seus *softwares* de proteção.

O Instituto de Ciência e Tecnologia Coreano (*Korea Advanced Institute of Science and Technology* – KAIST) também explorou as habilidades dos *hackers* para testar a segurança de um de seus sites, o Olymfair. Os prêmios aos vencedores corresponderam a 50 mil dólares para o primeiro prêmio, 20 mil para o segundo e 10 mil para o terceiro. O concurso internacional fora organizado pelo KAIST junto com o *Hacker* Lab., empresa de segurança que é destinada a promover a troca de tecnologias de segurança entre especialistas mundiais. A idéia é treinar jovens na tecnologia e orientá-los positivamente, aumentando a consciência geral para as questões de segurança. (TEK, 2001*a*)

Segundo DEVOST (2001), o submundo digital deve ser visto como um patrimônio. Os invasores têm usado meios ilegais para satisfazer sua curiosidade sobre o funcionamento da tecnologia de computadores, pois o sistema nega a eles outros meios de acessar o reino digital que eles tanto adoram.

O professor de Direito da Universidade de Harvard, Laurence H. Tribe, (citado por DEVOST, 2001), sustenta a idéia de que o acesso à tecnologia pode ser um ideal da sociedade democrática. Tribe afirma que certas tecnologias podem tornar-se socialmente indispensáveis, de tal modo que o acesso à tecnologia pode ser um ideal constitucional tão importante quanto o acesso à educação elementar. Isso significa (ou deveria significar) que as obrigações constitucionais do governo devem, muitas vezes, assegurar o acesso à informação em vez de simplesmente reforçar proibições negativas contra determinadas maneiras de invasão.

---

[9] A SDMI é constituída por 175 empresas da indústria eletrônica e da música, foi formada em 1999 com o objetivo de conceber uma forma segura de disponibilizar música digitalmente.

7 CONCLUSÃO

A Internet está propiciando um novo tipo de realidade econômica, a qual tem reduzido o número de intermediários no processo de compra e venda. Já é possível adquirir inúmeros produtos ou serviços via Web; aplicar em fundos de investimentos ou ações, solicitar documentos e até registrar ocorrências policiais com o auxílio da Internet. Para tanto, criar um ambiente seguro tornou-se uma tarefa crítica; e qualquer usuário, para utilizar as funcionalidades da Internet, precisa sentir-se seguro de que suas informações estão protegidas contra alteração e divulgação não autorizada.

Porém, o número de ocorrências de problemas de segurança na Internet está crescendo a cada ano. Muitos destes problemas são decorrentes da fragilidade que a Internet apresenta desde o seu projeto inicial. Na tentativa de corrigir tais falhas, inúmeros mecanismos têm sido criados; no entanto, estes novos mecanismos lançados no mercado continuam a apresentar falhas de projeto e implementação. Além disso podem ser inadequadamente configurados pelos administradores de rede, permitindo acesso de pessoas não autorizadas.

Neste contexto, os piratas eletrônicos têm desfrutado de tais falhas para, principalmente, mostrar a fragilidade que a Internet ainda apresenta, evidenciando as fragilidades do comércio eletrônico.

A disseminação da informação na Internet tornou-a uma das maiores armas para atacantes, pois muitas brechas de segurança têm sido transformadas em ferramentas automatizadas de ataques como os vírus e os cavalos de Tróia. Tais ferramentas estão acessíveis a qualquer usuário, deixando a Rede cada vez mais vulnerável. Usuários mal intencionados fazem uso destas ferramentas para invadir sites, com o objetivo de denegrirem ou roubarem informações. As medidas de prevenção contra essas ameaças são: evitar o download de arquivos de procedência suspeita, não abrir arquivos anexados em e-mails e a utilização de um antivírus atualizado.

Uma técnica de ataque que mais tem crescido na Internet é o DDoS, técnica que tem deixado grandes sites fora do ar durante horas, acumulando enormes prejuízos. Pode-se constatar que a falta de conscientização e descrença dos administradores, unido à pressa

na construção de novas funcionalidades em todo o ambiente, têm deixado as portas abertas não somente para novos ataques, mas também para ataques amplamente conhecidos e que poderiam ser evitados.

Embora não exista nenhuma forma de extinguir todas possibilidades de ataques de negação de serviço, há diversas medidas que podem ser adotadas no sentido de dificultar a realização destes ataques, dentre as quais estão a implantação de filtros inteligentes e a utilização de rotas estáticas e, nos casos em que forem utilizados protocolos de roteamento, não configurá-los em modo passivo. Além disso, deve-se evitar que sua rede não seja utilizada como amplificadora em um ataque.

Dentre as ameaças apontadas contra a segurança da informação, estão a integridade, a transmissão de dados e a repudiação. Assim, a criptografia tem sido utilizada para garantir a confiabilidade dos dados tanto armazenados quanto durante uma transmissão. E, para garantir a não-repudiação, a assinatura digital é considerada confiável na autenticação segura das transações comerciais eletrônicas.

As ferramentas que possibilitam autenticações por dados biométricos, devido ao alto custo de implantação, ainda não estão muito difundidas. Contudo, é um dos métodos que garantem um maior nível de confiabilidade, apresentando porcentagem de falhas próximo a zero. Esforços têm sido despendidos para tornar o método mais acessível.

No que diz respeito a garantia da segurança, uma afirmação pode ser feita: não existe sistema totalmente seguro, e é muito provável que nunca existirá, principalmente nos casos em que a segurança depende da intervenção ou confiança do ser humano. O que se consegue é reduzir os riscos. A exemplo da segurança de uma agência bancária, existem os seguranças, as câmeras, as portas giratórias com alarmes, enfim, todo aparato para dificultar o acesso, o qual não garante 100% de segurança.

Os hackers têm contribuído de forma significativa no aprimoramento de novas tecnologias, principalmente na exigência de produtos de melhor qualidade, descobrindo falhas de segurança nos sistemas e, consequentemente, reportando suas descobertas aos desenvolvedores. Os *hackers* também são os responsáveis pelo enorme patrimônio intelectual em software livre que circula pelo mundo, do qual todos que estão engajados na revolução digital estão se beneficiando. A abordagem do código fonte aberto resgatou atitudes singulares como o cooperativismo, o esforço mútuo e a doação, valores que estavam se perdendo e estão sendo praticados nas ações dos *hackers*.

No entanto, o termo *hacker* tornou-se genérico para designar tanto indivíduos

invasores como também os "gênios" da informática; ou seja, um termo que exprime ao mesmo tempo admiração e repulsa. Neste conflito, o termo *hacker* tem sido utilizado como engodo para aumentar as vendas de notícias, principalmente nos períodos em que invasões ocorrem.

As empresas de segurança digital, o governo e a mídia, têm, inconscientemente, incentivado o crescimento das invasões nas redes. O anúncio equivocado e freqüente de que *hackers* são bandidos e, ao mesmo tempo, a contratação de seus serviços para implementarem a segurança, gera uma incoerência que estimula, principalmente os adolescentes a ações ilícitas para demostrarem suas capacidades, buscando provar algo para que sejam serem reconhecidos pelas suas habilidades.

É importante que os profissionais da informática, juntamente com o governo, somem esforços para reverter a visão distorcida que se tem em relação aos *hackers*. Ao invés do governo perdurar batalhas contra os *hackers*, meios para permitir o acesso a tecnologias deveriam ser implementados, sem atitudes contraditórias, promovendo a educação para estes indivíduos. O que deve ser percebido é que os *hackers* não são um problema tecnológico, e sim educacional, e o único modo de resolver isso, é ensinar as "crianças" o que devem fazer.

Se piratas eletrônicos são inimigos temerosos da Internet e da segurança em geral, então é supremo que eles sejam compreendidos melhor e não simplesmente rotulados como criminosos. Como Sun Tzu declara em seu livro A Arte da Guerra: "Se você conhece o inimigo e conhece a si mesmo, não precisa temer o resultado de cem batalhas".

Neste contexto, para minimizar a insegurança dos sistemas, é manter um conhecimento detalhado e atualizado das tecnologias existentes, característica inerente aos piratas eletrônicos. Assim, reconhecendo suas habilidades e incentivando-os a aprimorarem os sistemas de segurança, os esforços que hoje são dispendidos nestes sistemas, poderão ser tranferidos para outros desafios.

Para trabalhos futuros, uma vasta discussão sober trabalhos cooperativos realizados via Internet deve ser feita. O trabalho cooperativo tem sido pouco explorado, embora haja um campo ilimitado para tal discussão. Assim como o *open source* é um exemplo de sucesso, diversas produções qualitativas poderiam ser realizadas, tal como leis e a criação de estórias infantis. Outra sugestão é a elaboração de uma metodologia para viabilizar a realização de ataques legais, a fim de explorar as falhas dos sistemas, permitindo assim que os *hackers* demonstrem suas habilidades sem que haja conflitos.

8 REFERÊNCIAS

ANDRADE, Cris. **Hacker conserta falha de segurança na Excite@Home.** Publicada em 04/06/2001. Disponível em: <http://www2.uol.com.br/cgi-bin/anti-hackers/builder/builder.cgi>. Acesso em: 15 de junho de 2001.

ANONYMOUS. **Maximum Security** - A *Hacker*'s Guide to Protecting Your Internet Site and Network. 2nd. ed. Indianapolis : SAMS Publishing, 1998.

ANTI-HACKERS. **A Divisão Hierárquica.** Anti-Hackers Internet Informática Ltda. Disponível em:<http://www2.uol.com.br/cgi-bin/anti-hackers/builder/builder.cgi?sec=hackers&id=hackers>. Acesso em: 26 de agosto de 2000

BELLOVIN, Steven M. **Security Problems in the TCP/IP Protocol Suite**. ACM Computer Communications Review, Vol. 19, No. 2, pp. 32-48, April 1989.

BERGLIND, Rickard. **Encrypted File System Win 2000 flaw.** Help Net Security®. Publicado em 11/05/2001. Disponível em: <http://www.net-security.org/text/bugs/979957507,70469,.shtml>. Acesso em: 02 de junho de 2001.

BERNSTEIN, Terry; BHIMANI, Anish B.; SCHULTZ, Eugene; SIEGEL, Carol A. **Segurança na Internet**. Rio de Janeiro : Campus, 1997.

BIDO, Luiz Carlos. **Defendendo-se do Back Orifice.** Disponível em: <http://www.smo.com.br/manuais/backorifice.htm>. Acesso em: 24 de agosto de 2000.

BLACK, Frank. **Trojans e Back Doors: Back Orifice.** Disponível em: <http://kL.virtualave.net/ie/bo.htm>. Acesso em: 14 de abril de 2000.

BORTONE, João. Segurança na Era da Internet. **PC Master**. São Paulo, Ano 3, n. 10,

p. 10, março 2000.

BRASIL, Angela Bittencourt. **Assinatura Digital**. Disponível em: <http://jusnavigandi.com.br/doutrina/assidigi.html>. Acesso em: 30 de Outubro de 2000

BUYS, Bruno; EVANGELISTA, Rafael; CUNHA, Rodrigo; KANASHIRO, Marta. **DVD e Linux**: liberdade na era digital. Disponível em: <http://www.comciencia.br /reportagens/softliv/>. Acesso em: 10 de julho de 2001.

CERT-RS. **Identificando Recursos**. Disponível em: <http://www.cert-rs.tche.br /docs_html/segcheck.html>. Acesso em: 28 de setembro de 2000.

CHESWICK, Willian R.; BELLOVIN, Steven M. ***Firewalls* and Internet Security:** Repelling the wily hacker. Massachusetts: Addison Wesley, 1994

COMER, Douglas E. **Interligação em Rede com TCP/IP** – Princípios, Protocolos e Arquitetura. Vol. I. Rio de Janeiro : Campus, 1998.

COMPUTERWROLD. **Lei do comércio eletrônico**. Disponível em: <http://computerworld.terra.com.br/noticias/leiec/leiec.asp>. Acesso em: 18 de junho de 2001.

CONJUR. Ex-hacker no Judiciário: MP contrata especialista em investigações eletrônicas. **Revista Consultor Jurídico**. Publicado em 10/05/2001. Disponível em: <http://cf6.uol.com.br/consultor/view.cfm?numero=5216&print=yes>. Acesso em: 12 de junho de 2001.

COSTA, Maira da. O Brasil é Tricampeão. **Revista Exame**. São Paulo. n. 13, Ed. 743, p. 24. 17 jun. 2001.

CUNHA, Derneval R. R. da. **Barata Elétrica.** Diponível em: <barata0.htmlbarata0.html fim.htmlfim.htmlbarata02.htmlbarata02.html>. Acesso em: 11 de agosto de 2000.

DEVOST, Matthew G. **The Digital Threat: United States National Security and Computers.** Annual Meeting of the New England Political Science Association, Massachusetts, April 24, 1994. Disponível em: <http://www.oss.net/Proceedings/ossaaa/aaa3/aaa3ae.htm>. Acesso em: 28 de maio de 2001.

EINDHOVEN, Vicious Fishes. **Invadindo seu site para melhorar a segurança**. Disponível em: <http://www.absoluta.org, 26 de março de 2000>. Acesso em: 11 de agosto de 2000.

FARMER, Dan; VENEMA, Wietse. **Improving the Security of Your Site by Breaking Into It**, 1994. Disponível em: <http://bau2.uibk.ac.at/matic/secu.htm>. Acesso em: 15 de julho de 2001.

FERNANDES, Manoel. Surgem os investigadores especializados em desvendar e evitar crimes eletrônicos. **Veja.** 26 nov. 1997.

FIGUEIREDO, Antonio. **Administração de Sistemas e Segurança.** n. 06. Disponível em:<httl://www.revista.unicamp.br/infotec/admsis/admsis6-1.html>. Acesso em: 14 de setembro de 2000.

GARBER, Lee. **Denial-of-Service Attacks Rip the Internet**. Disponível em: <http://www.computer.org/computer/articles/April/TechNews_400.htm>. Acesso em: 17 de julho de 2000.

GARFINKEL, Simon; SPAFFORD, Gene. **Web Security & Commerce**: Risks, Technologies and Strategies. USA : O'Reilly & Associates, 1997.

GEEK. Nasce o Software Livre. **Geek.** n. 05, p. 34-35. São Paulo, Abr. 1999.

_____. Rugido da Contra Cultura. **Geek**. n. 07, p. 10-49. São Paulo, 2000.

GLOBAL Network Solutions Tec. **GOPHER.** Disponível em: <http://www.gns.com.br/gopher.htm>. Acesso em: 15 de setembro de 2000*a*.

GLOBAL Network Solutions Tec. **Telnet (execução remota).** Disponível em: <http://www.gns.com.br/telnet.htm>. Acesso em: 23 de setembro de 2000*b*.

GOLDSTEIN, Emmanuel. **Hackers in Jail.** 2600, Vol. 6, n. 1, Spring 1989.

_____. **Two Views of Hacking**. Disponível em: <http://cnn.com/TECH/specials /*hackers*/qands/>. Acesso em: 12 de março de 2000.

GOMES, Olavo José A. **Segurança Total – Protegendo-se contra os hackers** São Paulo : Makron Books, 2000.

GONÇALVES, Kelli. Anti-Hackers. **PC World.** 15/03/2001. Disponível em: < http://pcworld.terra.com.br/pcw/testes/internet/0040.html>. Acesso em: 22 de maio de 2001.

_____, Marcus. **Segurança na Internet – Protegendo seu Web Site com *Firewalls*** Rio de Janeiro : Axcel Books, 1997.

GOYA, Denise H. Biometria : Diversas tecnologia permitem identificar pessoas pelas características físicas. **PC World**, São Paulo p.54-62, agosto de 2000.

_____, Denise H. *Hackers* não são a única ameaça à segurança. **PC World**. São Paulo, n. 93, p.33-48 , mar. 2000.

GREGO, Maurício. O submundo dos hackers. **Info Exame**. São Paulo, n. 173, p.46-58, ago. 2000.

HARRISON, Linda. **Ethical hacker reveals secrets of underground world**. Publicado em 12/04/2000. The Register®. Disponível em: <http://www.theregister.co.uk/content /archive/10323.html>. Acesso em: 6 de março de 2001.

HARVEY, Brian. **Computer Hacking and Ethics.** University of California, Berkeley. Versão modificada do "Panel on Hacking", da ACM, abr. 1986. Disponível em: <http://www.cs.berkeley.edu/~bh/hackers.html>. Acesso em: 14 de jun. 2001.

HOLSCHUH, Henrique. **Pretty Good Privacy (PGP)**. Disponível em: <http://www.dca.fee.unicamp.br/pgp/pgp.shtml>. Acesso em: 06 de novembro de 2000.

HUTHEESING, Nikhil; ROSS, Philip E. Um negócio chamado Hackerfobia. **Info Exame**. São Paulo, Ano 13, n. 146, p. 14-15, maio 1998.

InforEstud@nte. **SSL-Secure Socket Layer – Apresentação.** Disponível em:. <http://www.fct.uc.pt/Inforestudante/introduÃ§aoSSL.htm>. Acesso em: 13 de setembro de 2000.

IXPRESS. **O IXpress e a Segurança Web.** Disponível em: <http://www.consist.com/br/ixpress/ixp-secw.htm#a4>. Acesso em: 09 de agosto de 2000.

JARGON FILE. Appendix B. **A Portrait of J. Random Hacker.** Publicado em: 11 July, 1995. Disponível em: <http://www.dpmms.cam.ac.uk/~gjm11/jargon /jargappB.html>. Acesso em: 18 de agosto de 2000*b*.

LEON, Nash. **Unsekurity Team**. Disponível em: <http://unsekurity.virtualave.net/txts /prog_basica.txt>. Acesso em: 15 de agosto de 2000.

LETTICE, John. **Copy protection on Whistler easily cracked.** Publicado em 19/01/2001. The Register®. Disponível em: <http://www.theregister.co.uk/content/4/16223.html>. Acesso em: 22 de março de 2001.

LEYDEN, John. **Captain Crunch sets up security firm.** Publicado em 01/02/2001. The Register®. Disponível em: <http://www.theregister.co.uk/content/archive/16536.html>. Acesso em: 08 de março de 2001.

LIMA, Marcelo. *Firewalls* – Uma Introdução à segurança. **Revista do Linux**, Curitiba n. 2, jan./fev. 2000.

LOPES, Rodrigo. **Finger**. Disponível em: <http://www.artnet.com.br/~lopes/index.htm>.

Acesso em: 20 de agosto de 2000.

MAIA, Luiz Paulo. **Criptografia e Certificado Digital.** Disponível em: <http://www.training.com.br/pub_seg_cripto.htm>. Acesso em: 26 de setembro de 2000.

MANHÃES, Marcos Martins. **O que é Open Source?** Publicado em: 07/05/2001. Disponível em: <http://www.olinux.com.br/artigos/322/print_preview.html>. Acesso em: 12 de julho de 2001.

McCLURE, Stuart.; SCAMBRAY, Joel; KURTZ, George. **Hacking Exposed:** Network Security Secrets and Solutions. California : Osborne/McGraw-Hill, 1999.

MENDES, Wayne Rocha. **Linux e os hackers** – Proteja seu sistema: ataques e defesas. Rio de Janeiro : Ciência Moderna, 1999.

MENEGHEL, Luciana. **Segurança – Overview**. Disponível em: <http://www.dca.fee.unicamp.br/courses/IA368F/1s1998/Monografias/luciana.html>. Acesso em: 27 de setembro de 2000.

MIRANDA, André F. **Ataques de DDoS: Não seja o vilão.** Fev/2000. Disponível em: < http://www.modulo.com.br/empresa/noticias/artigo_entrevista/>. Acesso em: 14 de setembro de 2000.

NETO, Silva; MORAES, Amaro. **Resgatemos os Hackers.** Disponível em: <http://www.advogado.com/internet/zip/hackers.htm>. Acesso em: 25 de outubro de 2000.

NETWORK COMPUTING. Chega de senha. **Network Computing**, ano 1, n. 11, p.53, jan. 2000.

NORTHCUTT, Stephen. **Network Intrusion Detection**: An Analyst's Handbook. Indianapolis : New Riders, 1999.

OLIVEIRA, Wilson José. **Hacker Invasão e Proteção.** Florianópolis : Visual Books, 1999.

_____. **Técnicas para Hackers.** Lisboa : Centro Atlântico, 2000.

PALMIERI, Sílvio A. **Segurança na Internet.** Disponível em: <http://www.barretos.com.br/usuarios/py2gea/diversos/seguranÃ§a%20na%20internet.html>. Acesso em: 22 de setembro de 2000.

PISTELLI, Daniela. **Criptografia**. Disponível em: <http://www.nucc.pucsp.br/novo/cripto/cripto.html>. Acesso em: 26 de setembro de 2000.

RAYMOND, Eric S. **A Brief History of Hackerdom**. Publicado em 1999. Disponível em: <http://www.tuxedo.org/~esr/writings/*hacker*-history/index.html>. Acesso em: 17 de setembro de 2000.

_____. **Jargon File Resources**. Disponível em: <http://www.tuxedo.org/~esr/jargon/>. Acesso em: 14 de julho 2000*a*.

_____. **How To Become A Hacker**. Disponível em: <http://www.tuxedo.org/~esr/faqs/*hacker*-howto.html>. Acesso em: 11 de maio de 2000*b*.

_____. **The Cathedral and the Bazaar.** Publicado em 1998. Disponível em: <http://www.tuxedo.org/~esr/writings/cathedral-bazaar/index.html>. Acesso em: 13 de julho de 2000*c* (1998).

REZENDE, Pedro Antonio Dourado de. **Tópicos em Segurança de Dados**. Universidade de Brasília – UNB. Disponível em: <http://www.cic.unb.br/docentes/pedro/pedro.html>. Acesso em: 18 de junho de 2000.

RIBEIRO, Andréa et al. **Segurança na Internet**. Faculdade de Ciência e Tecnologia. Universidade de Lisboa. Disponível em: <http://students.fct.unl.pt/users/clsc/gsi/>. Acesso em: 02 de novembro de 2000.

ROÇAS, Christian. Credibilidade em cheque. **Internet Business**, São Paulo, p. 48-55. Fev. 2000.

ROGERS, Marc. **Psicology of Hackers:** Steps Toward a New Taxonomy. Dept. of Psychology University of Manitoba. Publicado em 02/12/1999. Disponível em: <http://www.infowar.com/hacker/99/HackerTaxonomy.shtml>. Acesso em: 21 de junho de 2001.

SOUZA, Aylton de. Segurança na nova era da informação. **Network World Telecom**. n. 02, p. 60, set. 1998.

SPITZNER, Lance E. **Técnicas adotadas pelos crackers que tentam entrar em redes coorporativas e Redes Privadas.** Disponível em: <http://vexxor.virtualave.net /artigos/seguranca/crackcops.html>. Acesso em: 26 de março de 2000.

SPLITNET. Virus & Antirus Info Site. **O que são Trojans Horses?** Disponível em: <http://www.splitnet.com/index3.html>. Acesso em: 15 de agosto de 2000.

SPYMAN. **Manual Completo do *Hacker*.** 3. ed. Rio de Janeiro : Book Express, 1999.

STALLINGS, Willian. **Cryptography and Network Security:** Principles and Practice. 2nd. ed. Ney Jersey : Prentice-Hall, 1999.

STRAUCH, Suzana. **Ataques que exploram falhas na implementação**. Disponível em: <http://expand.virtualave.net/ntexto/seguranca/ataques5.html>. Acesso em: 15 de agosto de 2000.

TANENBAUM, Andrew. **Computer Networks,** 3rd edition. New Jersey : Prentice Hall, 1996.

TEIXEIRA JÚNIOR, José; SUAVÉ, Jacques; MOURA, José A.; TEIXEIRA, Suzana. **Redes de Computadores:** Serviços, Administração e Segurança. São Paulo, Makron Books, 1999.

TEK. **Concurso de *hackers* na Coreia.** Publicado em 20/06/2000. Disponível em: <http://tek.sapo.pt/470/117786.html>. Acesso em: 24 de fevereiro de 2001*a*.

_____. **Prêmio para o melhor hacker.** Publicado em 14/09/2000. Disponível em: <http://tek.sapo.pt/470/126239.html>. Acesso em: 24 de fevereiro de 2001*b*.

TZU, Sun. **A Arte da Guerra.** 24. ed. Rio de Janeiro: Record, 2001.

TITTON, Luiz Antônio. **Hackers – Administração do Previsto**. Disponível em: <http://www.sit.com.br/SeparataNET007.ht>. Acesso em: 02 de março de 2001.

TRINTA, Fernando A Mota; MACÊDO, Rodrigo C. **Um Estudo sobre Criptografia e Assinatura Digital**. Disponível em: <http://www.ufpe.br>. Acesso em: 30 de outubro de 2000.

UNICOR. **Cuidando com o Net Bus.** Disponível em: <http://www.unincor.br/provedor/Navegantes/netbus/index.htm>. Acesso em: 25 de agosto de 2000.

VASCONCELLOS, Márcio José Acciolli de. **A Internet e os Hackers – Ataques e Defesas**. 4. ed. São Paulo: Chantal, 1999.

WEBER, Raul. Segurança na Internet. In: ESCOLA REGIONAL DE INFORMÁTICA DA SBC REGIÃO SUL, V, 1997. Florianópolis - SC, Maringá - PR e Santa Maria - RS, **Anais...** Santa Maria : Pallotti, 1997. 209p., p. 101-131.