

**LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO
DO RIO GRANDE DO NORTE**

SEGURANÇA NA COMUNICAÇÃO DE DADOS: UM ESTUDO SOBRE
CRIPTOGRAFIA E FERRAMENTAS DE SEGURANÇA

Mouragida é apresentada ao Curso de
Graduação em Engenharia de Informática da
Faculdade Natalense para o Desenvolvimento
do Rio Grande do Norte como trabalho parcial
para obtenção do grau de Bacharel em
Engenharia de Informática.

SEGURANÇA NA COMUNICAÇÃO DE DADOS: UM ESTUDO SOBRE
CRIPTOGRAFIA E FERRAMENTAS DE SEGURANÇA

Autógrafa Mouragida de Sousa Farias, MSc.

Rômulo Murdock

UNIVERSIDADE DE VARSÓVIA
RÔMULO MURDOCK
FACULDADE NATALENSE PARA O DESENVOLVIMENTO
DO RIO GRANDE DO NORTE

SEGURANÇA NA COMUNICAÇÃO DE DADOS: UM ESTUDO SOBRE
CRIPTOGRAFIA E FERRAMENTAS DE SEGURANÇA

Monografia apresentada ao Curso de
Graduação em Sistemas de Informação da
Faculdade Natalense para o Desenvolvimento
do Rio grande do Norte como requisito parcial
para obtenção do grau de Bacharel em
Sistemas de Informação

Orientador:

Aluizio Ferreira da Rocha Neto, Mst.

RÔMULO MURDOCK

SEGURANÇA NA COMUNICAÇÃO DE DADOS: UM ESTUDO SOBRE
CRİPTOGRAFIA E FERRAMENTAS DE SEGURANÇA

RB-13.757

M
004.7.

M 949

2002

FARN

BIBLIOTECA

Nº Reg. M000233

11/05/05

Murdock, Rômulo

Segurança na comunicação de dados: um estudo
sobre criptografia e ferramentas de segurança/ Rômulo
Murdock. ____ Natal: FARN, 2002.

54p

Monografia (Graduação em Bel. em Sistemas de
Informação) – Faculdade Natalense para o
Desenvolvimento do Rio Grande do Norte.

Orientador: Aluizio Ferreira da Rocha Neto, Mst.

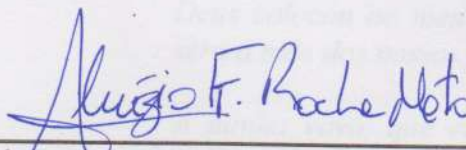
1. Redes de Computadores – Monografia
2. Segurança da Informação através da criptografia.
I. Título

**LIGA DE ENSINO DO RIO GRANDE DO NORTE
FACULDADE NATALENSE PARA O DESENVOLVIMENTO
DO RIO GRANDE DO NORTE**

**SEGURANÇA NA COMUNICAÇÃO DE DADOS: UM ESTUDO SOBRE
CRIPTOGRAFIA E FERRAMENTAS DE SEGURANÇA**

Rômulo Murdock

Monografia apresentada e aprovada em 18 de 12 de 2002 pela banca examinadora
composta pelos seguintes membros

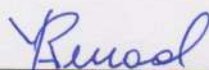


Aluizio Ferreira da Rocha Neto, Mst.

Orientador



Antônio de Pádua de Miranda Henriques



Ytalo Rosendo do Amaral

Natal, 18 de DEZEMBRO de 2002.

AGRADECIMENTOS

Primeiramente agradeço ao senhor Pai, Jesus Senhor Jesus Cristo, que me permitiu a vida e este plano e que sempre me dá a luz e que hoje, sempre me dá uma direção na minha jornada da vida.

Agradeço a todos os meus professores que foram verdadeiros mestres em dando-me a difícil arte da computação.

Em especial, um abraço para meus professores Luis Augusto Martins Silva, José da Paçoca, Lúcia Maria Martins, Felipe Araújo, Alisson, Artur da Paçoca, e Maria Patrícia, Alexandre Mendes, Luis Damasceno, Nidia, e todos os outros que me ajudaram a chegar até aqui.

Adilson amigo e orientador, um abraço especial por sua paciência e dedicação, sempre me ajudando a chegar até aqui.

Agradeço aos meus amigos de classe, especialmente os que me ajudaram a chegar até aqui, pelos momentos de alegria e apoio.

Em especial, um agradecimento para minha mãe, Maria da Paçoca, e meu pai, José da Paçoca, por me terem criado e me terem dado a oportunidade de estudar.

Agradeço a minha avó, Zuleide de Carvalho Santos, por me dar a oportunidade de estudar e por me ter criado com tanto amor e carinho.

Agradeço a minha irmã, Rosane, por me dar a oportunidade de estudar e por me ter criado com tanto amor e carinho.

Agradeço a minha filha, Minnie, por me dar a oportunidade de estudar e por me ter criado com tanto amor e carinho.

Agradeço a todos os que me ajudaram a chegar até aqui, especialmente os que me deram a oportunidade de estudar.

Agradeço a Deus por tudo o que me deu e por me ter criado com tanto amor e carinho.

À minha amada mãe dedico este trabalho, como resultado de meu esforço e também do apoio que dela recebi em todos os momentos da minha vida; fossem estes bons ou ruins.

À minha querida e amada Marta, a mulher que Deus colocou no meu caminho e que um dia será a mãe dos nossos filhos.

À minha vovó, que eu amo muito; obrigado pelos anos de apoio, incentivo, e por ter nos colocado nesse mundo; pela criação exemplar que deu a sua filha e que hoje se reflete na minha educação.

À minha doce irmã Rosane.

À minha filha Minnie, pelos grandes momentos de alegria que temos passado.

A mim mesmo.

AGRADECIMENTOS

Primeiramente agradeço ao nosso Pai, nosso Senhor Jesus Cristo, que me permitiu a vinda a este plano e que somente graças a Ele, é que hoje, cumpro mais uma etapa na minha jornada da vida.

Agradeço a todos os meus professores que foram verdadeiros mestres, ensinando-nos difícil arte da computação.

Em especial, um abraço para meus professores Luiz Augusto Mendes Filho, Antônio Pádua, Livia Maria Martins, Felliipe Araújo Aleixo, Andréia Rodrigues, Robson Pacheco, Alessandra Mendes, Luiz Damasceno, Nadja Rodrigues, que deles recebi ensinamentos que vão muito além de cálculos e algoritmos.

Ao meu amigo e orientador, um agradecimento especial, Prof. Aluizio Rocha pela paciência, destreza, companheirismo, atenção neste trabalho.

Agradeço aos meus colegas de classe pelo companheirismo, pelas horas de estudo, pelas baladas, pelos namoros, pela paciência, pelo carinho a mim dispensados.

Em especial, um agradecimento para meus amigos João Maria, Rodrigo Pires, Luciano Britto, Ana Angélica, Sandro Galvão, Danniele Lobo e José Nicodemus. Nunca esquecerei vocês.

Agradeço a nossa comissão de formatura que vem empenhando um trabalho fenomenal e se hoje teremos um baile, será graças a eles.

Agradeço a minha chefe Zenilda de Carvalho Nunes que me deu a oportunidade de por meus ensinamentos em prática em uma instituição como o Ministério da Saúde.

Finalmente, gostaria de agradecer a todos que contribuíram direta ou indiretamente na conclusão deste trabalho.

Obrigado, e que Deus os abençoe!

RESUMO

Nos dias de hoje, usar a Internet já faz parte de uma grande maioria dos brasileiros. Usamos a Internet para, dentre outras coisas, pagar nossas contas, ler jornais ou livros, fazer nossa agenda, conversar com parentes e fazer compras.

Diante deste grande mercado em evolução, não param de crescer os sites de comércio eletrônico. Os sites de comércio eletrônico já fazem parte da nossa vida; e a maioria das grandes empresas, migraram ou estão migrando para a Internet.

Porém, o que a grande maioria dos empresários e Internautas não sabem, é que para fazer e-bussiness não basta apenas colocar o site no ar, anunciar os produtos e vendê-los. É preciso toda uma infra-estrutura de segurança capaz de garantir ao consumidor que os seus dados sigilosos - como número de cartão de crédito, senhas - não serão violados. A segurança, hoje, é um item crítico na agenda de quem quer utilizar a rede para fazer negócios.

O seguinte trabalho, tem como objetivo de apresentar alguns métodos de como proteger os dados na Internet, visando esclarecer o usuário e o administrador de redes de como fazer uso destas ferramentas para aumentar a confiabilidade do seu cliente no seu negócio. E apresenta também o uso destas ferramentas.

O trabalho abrange o estudo da Criptografia como padrão mundial para envio de dados de forma segura, demonstrando através de algoritmos o funcionamento das principais soluções de criptografia na atualidade; métodos de ataque e defesa a dados criptografados e algumas soluções em ferramentas de segurança para a Internet.

ABSTRACT

In the days today, to use Internet is already part of a great majority of the Brazilians. We used Internet for, among other things, to pay our bills, to read newspapers or books, to do our calendar, to talk with relatives and to do purchases.

Before this great market in evolution, they don't stop increasing the sites of e-bussiness. The sites of e-bussiness are already part of our life; and most of the great companies, they migrated or they are migrating for Internet.

However, which the entrepreneurs' great majority and Internautas don't know, it is that to do e-bussiness it just is not enough to put the site in the air, to announce the products and to sell them. It is necessary an entire infrastructure of safety capable to guarantee to the consumer that their secret data - as credit card number, passwords - they won't be violated. The safety, today, is a critical item in the calendar of who wants to use the net to do businesses.

The following work, has as objective of presenting some methods of how to protect the data in Internet, seeking to explain to the user and the administrator of nets of how to do use of these tools to increase your customer's reliability in your business. And it also presents the use of these tools.

The work includes the study of the cryptography as world pattern for sending of data in a safe way, demonstrating at the present time through algorithms the operation of the main cryptography solutions; attack methods and defense to given criptografados and some solutions in tools of safety for the internet.

SUMÁRIO

1	Aspectos de Segurança nas Empresas.....	10
1.1	Definindo um plano de Análise de Risco	11
1.2	Política de Segurança.....	11
1.3	Organização do Trabalho.....	12
2	Criptografia.....	13
2.1	O Que é Criptografia	13
2.2	Algoritmos Simétricos ou Algoritmos de chave secreta (Burnett, 2002).....	15
2.3	Data Encryption Standard – DES.....	16
2.3.1	Funcionamento do DES	16
2.4	Algoritmos de Chave Pública – Assimétricos (Terada, 2000).....	18
2.5	Os Problemas da Chave	21
3	Fundamentação Matemática.....	23
3.1	Aritmética Módulo m (Terada, 2000).....	23
3.1.1	Adição Módulo m.....	24
3.1.2	Multiplicação Módulo m.....	24
3.2	Congruência	25
3.3	Álgebra de Boole – Funções Lógicas (Carvalho, 2001).....	25
3.3.1	Função “OU” ou “OR”	26
3.3.2	Circuito OU-Exclusivo	26
3.4	Máximo Divisor Comum (MDC) (Prazeres, 2000).....	27
3.5	Números Primos (Agrawal, 2002).....	28
3.5.1	Números Primos Relativos.....	29
3.5.2	Geração de Números Primos.....	29
3.6	Pequeno Teorema De Fermat (Carvalho, 2001)	30
3.7	Função Φ de Euler (Função TOTIENT) (Prazeres, 2000)	30
3.8	Funções Hash	31
4	Quebra e Ataques em Criptografia.....	33
4.1	O problema da Chave	33
4.2	Ataque à Chave de Criptografia.....	33
4.3	Ataques por Chaves Conhecidas (Prazeres, 2000).....	36
4.4	Ataques Por Dicionário (Prazeres, 2000)	36
4.5	Ataques ao Algoritmo (Prazeres, 2000).....	36
4.5.1	Ataques de texto simples (legível) conhecido.....	37
4.5.2	Ataque por Só-Texto-Ilegível.....	37
4.5.3	Ataque por Texto Legível Escolhido	37
4.5.4	Ataque Adaptativo Por Texto Legível Escolhido.....	37
4.5.5	Ataque por Texto Ilegível Escolhido	38
4.5.6	Ataque Adaptativo por Texto Ilegível Escolhido.....	38
5	Ferramentas de Segurança.....	39
5.1	Firewall	39
5.2	Virus Scanner	39
5.3	Intrusion Detection System – IDS.....	40
6	Exemplo de Caso – A Semente do Netscape.....	42
7	Conclusão	43
8	Referências Bibliográficas.....	44
9	Bibliografia Consultada.....	45

ÍNDICE DE TABELAS E FIGURAS

Tabela 1: Cifragem do alfabeto com Chave $ch=3$	14
Figura 1.1. Envio de mensagem segura através de linha insegura	15
Figura 1.2 Funcionamento do DES.....	17
Tabela 3.1: Tabela Verdade de uma função OU ou OR.....	26
Tabela 3.2: Tabela Verdade de circuitos ou-exclusivo.....	26
Tabela 4.1: Um cenário pior do que a pior das situações: quanto tempo um ataque de força bruta levaria quanto aos vários tamanhos de chave (Burnett, 2002).....	35

1 ASPECTOS DE SEGURANÇA NAS EMPRESAS

De acordo com a oitava pesquisa Módulo (Módulo, 2002) em segurança da informação, 80% dos altos executivos de 547 empresas brasileiras entrevistadas acreditam que os problemas com segurança vão aumentar neste ano e 45% deles consideram vital a proteção de dados no ambiente corporativo.

De acordo com a mesma pesquisa, 72% das empresas brasileiras que sofreram algum tipo de invasão não souberam quantificar o seu prejuízo. Das que conseguiram, 19% tiveram perdas inferiores a R\$ 50 mil, 8% tiveram prejuízos entre R\$ 50 mil e R\$ 1 milhão e 1% acima de R\$ 1 milhão.

Outro estudo, do Gartner Group (Barbosa, 2002) mostrou que as fraudes em lojas on-line chegaram a 700 milhões de dólares. E muitas delas, só pensam seriamente em segurança depois que são “hackeadas”.

Isso aliado ao fato de até o novembro de 2002 43% das empresas brasileiras tinham sido vítimas de algum tipo de ataque e 31% das empresas entrevistadas não souberam responder se já tinham sido vítimas de algum tipo de ataque (Módulo, 2002).

A pesquisa Módulo (2002) mostrou ainda, que 66% dos usuários deixaram de comprar pela internet por causa da sensação de falta de segurança.

A maioria das empresas acredita que basta ter um *firewall* e um antivírus que a empresa esta segura. Mas, sabe-se que não adianta ter um antivírus e não atualizá-lo e ter o melhor *firewall* do mundo sem saber como configurá-lo.

Para obter sucesso na proteção dos dados, faz necessário investir em uma política de segurança séria e eficiente. Definido o que é prioridade, como proceder em casos de invasão e como se proteger das invasões. A isto é chamado Análise de Risco.

1.1 *Definindo um plano de Análise de Risco*

O primeiro passo na elaboração de um plano de Análise de Risco eficiente, consiste em saber o que proteger, pois, ao entrar na Internet, a imagem, os dados e até mesmo a continuidade dos negócios da companhia podem ser ameaçados. Sem conhecer o que realmente está em jogo, as empresas correm o risco de sofrer prejuízos ainda que utilizem as soluções de segurança mais avançadas do mercado.

Fernando Santos, gerente-geral da Chek Point no Brasil, aponta alguns tópicos que ajudam a medir o risco da empresa (Valim, 2002):

- Qual é o valor das informações para as operações da companhia se forem perdidas?
- Quanto valem se forem roubadas?
- Quanto pode custar uma hora de interrupção nas operações da companhia?
- Qual será o meu prejuízo de imagem junto a clientes se uma invasão hacker se tornar publicamente conhecida?

Trata-se de um levantamento árduo e complexo variando, é claro, do tamanho da empresa em questão. Mas o fato é que se a sua empresa esta na Internet você esta exposto a maior rede hacker do mundo e logo poderá perder o seu negócio.

Neste caso, ainda não são estudadas como se irá fazer a segurança do sistema, pois para isso, precisa saber primeiro o que vai ser protegido; o que realmente é importante. Só depois de coletados todos os dados, e realizado o estudo de fragilidade do sistema, é que partirá então para a política de segurança do sistema eletrônico.

1.2 *Política de Segurança*

Um plano de política de segurança irá implementar os recursos para prevenir a invasão. Fatores como: sistema operacional do servidor, sistema operacional das estações, servidor de banco de dados, tipo de criptografia a ser utilizado, uso de tokens, a escolha do antivírus, a escolha do *firewall*, tudo isso é levado em consideração.

Neste ponto é que entrará a escolha do método a ser utilizado para a proteção dos dados no sistema. Todos os aspectos são estudados, analisados e se aprovados implementados.

É importante frisar que a uma política de segurança não entra só dados de software e hardware. O funcionário também entra aqui, pois muitas vezes, o próprio funcionário é quem abre a porta para hackers ou ele mesmo é o próprio hacker.

Um outro estudo da Módulo Consultoria (Fortes, 2002), realizado com 165 executivos no Brasil, mostrou que em 2001 os funcionários foram a maior causa dos problemas de segurança, com 31% dos casos. Neste ponto, o que se pode fazer é a orientação e o diálogo com o seu funcionário. Nem sempre o erro é intencional. É enormemente comum o uso de “senhas frágeis” como 1234, datas de nascimento, números de telefones, nome de filhos, facilitando a entrada no sistema de hackers. E quando não, sempre se pode encontrar um “lembrete” na mesa ou mesmo grudado no monitor.

Mas para os funcionários mais “curiosos”, existem iniciativas que vem dando certo nas empresas. Os chamados “hackers éticos” são funcionários treinados pela própria empresa, para fazer parte do time de segurança da empresa. Eles têm a função de invadir o sistema e sair. Feito isso, apresentam relatórios detalhados de onde se encontram as brechas e falhas.

1.3 Organização do Trabalho

Este trabalho está organizado da seguinte forma.

O capítulo 2 trata da fundamentação teórica que irá introduzir o leitor nos principais conceitos da criptografia moderna.

O capítulo 3 trata da fundamentação matemática que dá a base para os estudos dos algoritmos pesquisados.

O capítulo 4 aborda as várias técnicas de ataques em criptografia.

O capítulo 5 apresenta algumas ferramentas de segurança utilizadas para proteção de redes de computadores.

O capítulo 6 apresenta um exemplo de caso que visa comprovar a eficácia de sistemas criptográficos na utilização de redes seguras.

Finalmente, o capítulo 7 faz as considerações finais sobre este trabalho.

2 CRIPTOGRAFIA

Estima-se que a criptografia seja, aproximadamente, tão antiga quando a escrita (Carvalho, 2001). O que não é de se espantar se considerarmos que a natureza humana. “Pessoas quase sempre tem algo a esconder” (Carvalho, 2001 p.19). Uma vez que o homem começou a escrever, começou a pensar em como esconder em certas ocasiões o que foi escrito.

“À medida que a Internet torna-se cada vez mais presente na vida diária, a necessidade da segurança eletrônica torna-se ainda mais crítica. A utilização eficaz de técnicas criptográficas está no núcleo de várias dessas estratégias de gerenciamento de risco” (Burnett, 2002 p.21).

A criptografia forte, outrora tratada como material bélico é hoje tratada como mais flexibilidade permitindo o seu intercâmbio e conseqüentemente o seu aprimoramento.

“O setor de tecnologia de segurança passou por um crescimento explosivo em um curto período de tempo, com várias novas opções emergindo com relação à instalação de técnicas de segurança eletrônica baseadas na criptografia. Variando desde novos desenvolvimentos em hardware criptográfico até o uso de cartões inteligentes pessoais em infra-estruturas de chave pública, a indústria continua a aumentar as opções disponíveis para resolver da segurança eletrônica” (Burnett, 2002 p. 23).

2.1 O Que é Criptografia

“Arte de escrever em cifra, em código” (Ferreira, 1994 p.499).

“A criptografia converte dados legíveis em algo sem sentido, com a capacidade de recuperar os dados originais a partir desses dados sem sentido” (Burnett, 2002 p.15).

“Algoritmos criptográficos basicamente objetivam ‘esconder’ informações sigilosas de qualquer pessoa desautorizada a lê-las, isto é, de qualquer pessoa que não conheça a chamada chave secreta de criptografia” (Terada, 2000 p.15).

De modo algum a criptografia é a única ferramenta necessária para assegurar a segurança dos dados, nem resolverá todos os problemas de segurança. É um instrumento entre vários outros. Além disso, a criptografia não é a prova de falhas, sobretudo se for implementada incorretamente, ela não agrega nenhuma segurança real (Burnett, 2002).

Durante décadas a criptografia ficou calçada nas técnicas de substituição e permutação de símbolos. Um dos mais conhecidos e antigos é o sistema de César. O imperador romano Júlio César usava o seguinte sistema para enviar uma mensagem cifrada.

O sistema consiste em pegar um número entre 1 e 26 (Chave, ch) e utilizá-lo para mover as letras para a direita para empreitar.

Por exemplo, seja uma chave $ch=3$. Veja a tabela:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Tabela 1: Cifragem do alfabeto com Chave $ch=3$

Agora digamos que queremos cifrar a palavra BACHAREL utilizando a mesma chave, veja como ficaria:

BACHAREL

EDFLDUHO

Matematicamente falando, o sistema funciona assim:

Para criptografar:

$$L' \leftarrow (L + Ch) \text{ MOD } 26$$

Para descriptografar:

$$L \leftarrow (L' - Ch) \text{ MOD } 26$$

O Problema principal desse algoritmo se dá no tamanho da chave Ch que vai de 1 a 26, ou seja, existem só 26 possibilidade de cifrar uma letra. Logo o sistema se mostra extremamente frágil e em hipótese nenhuma deverá ser utilizado nos dias de hoje. Apresento o sistema aqui apenas para fins históricos e comparativos.

2.2 Algoritmos Simétricos ou Algoritmos de chave secreta (Burnett, 2002)

A criptografia de chave simétrica vem para resolver os problemas na proteção de dados que devem permanecer sigilosos. Essa proteção se dá através de uma chave secreta (ch) que o usuário informa para criptografar e descriptografar a sua mensagem (daí o nome simétrica – o mesmo dos dois lados).

Se um invasor tentar ou mesmo obter parte ou todo o texto cifrado e se o algoritmo for bem projetado, isto é, seguro, será computacionalmente difícil para ele recuperar o texto original sem o conhecimento da chave, mesmo conhecendo o algoritmo que o usuário utilizou para cifrar o texto (Terada, 2000).

Tomemos como exemplo a seguinte situação: Alice deseja enviar uma planilha de projeção de vendas para Beto. Então Alice com uma chave K , criptografa o texto legível x obtendo-se um outro texto ilegível y .

$$f_k(x) = y$$

O texto y é enviado para Beto através de uma linha de comunicação insegura que conhece a chave k e irá utilizar o algoritmo inverso $f_k^{-1}(y)$ obtendo-se x .

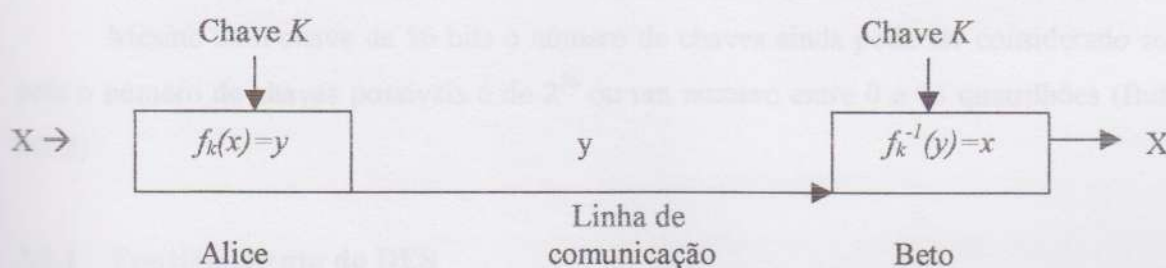


Figura 1.1. Envio de mensagem segura através de linha insegura

Terada (2000) aponta estes algoritmos como solução para os seguintes problemas:

1. Como esconder informações sigilosas, como o número de cartão de crédito das pessoas que controlam as linhas de comunicação e os computadores intermediários como X e Y ; ou seja, Alice só deseja que Beto consiga ler a informação enviada – isto é chamado de autenticação de destino;

2. Como evitar que alguma pessoa desautorizada leia as informações enviadas e armazenadas no computador de Beto;
3. uma variação do problema anterior é como evitar que alguma pessoa desautorizada leia e altere as informações já enviadas e armazenadas no computador de Beto – isto é chamado também de detecção de integridade de informação.

2.3 Data Encryption Standard – DES

A história de sucesso do DES remota o início da década de 70. Naquela época estava claro que os algoritmos não eram mais seguros o suficiente para manter os dados sigilosos; tinham fraquezas e eram difíceis de serem implementados.

Diante disso, pesquisadores da IBM decidiram desenvolver um novo algoritmo para a era dos computadores e criaram um esquema chamado de Lúcifer. Um algoritmo criado por Horst Feistel. O Lúcifer possuía entrada e saída de 64 bits e chave de 128 bits.

Para aprimorar o Lúcifer, os pesquisadores da IBM contaram com o apoio da National Security Agency – NSA americana. Nessa parceria surgiu o DES. O DES utiliza entrada e saída de 64 bits, porém a chave foi encurtada para 56bits. “Acredita-se que o encurtamento da chave tenha sido proposital a pedido da NSA” (Terada, 2000 P.42).

Mesmo com chave de 56 bits o número de chaves ainda pode ser considerado seguro pois o número de chaves possíveis é de 2^{56} ou um número entre 0 e 73 quatrilhões (Burnett, 2002).

2.3.1 Funcionamento do DES

Seja $B = 64$ bits de um bloco x de texto puro. Este texto é submetido a uma primeira permuta $P1$ com $1 \leq P_i \leq x \forall i(1 \leq i \leq n)$ resultando em um bloco intermediário $X1$ de 64 bits.

Em seguida são efetuadas 16 interações idênticas exceto pelo uso de 16 subchaves $K1, K2, \dots, Kn$ que são geradas a partir da chave principal K de 56 bits através da seguinte função:

$$fK_j(D_j) \text{ XOR } fK_j(D_j) = 0$$

Após as 16 interações há uma troca das metades esquerda e direita de X_{17} entre si, e o resultado X_{17}^m é submetido ao inverso da Permutação Inicial IP^{-1} resultando o bloco final com o texto cifrado y de 64 bits.

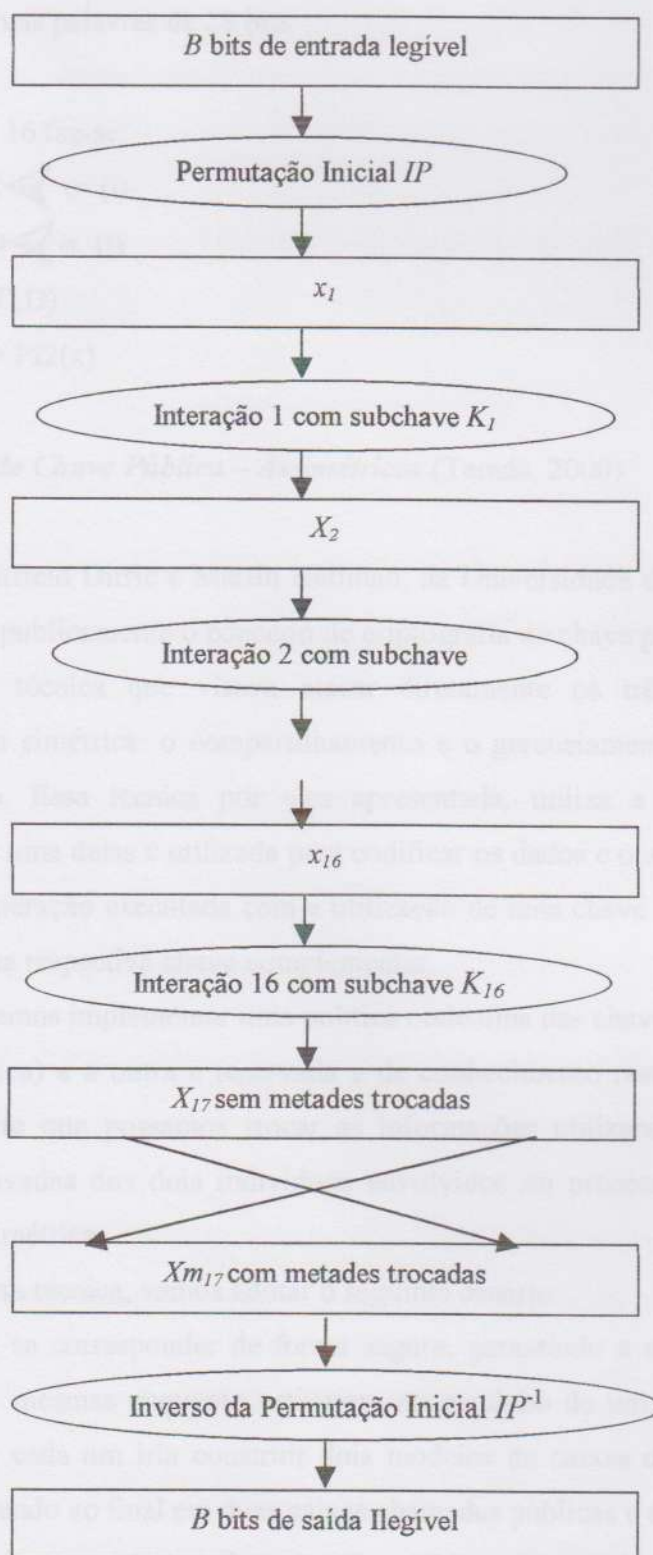


Figura 1.2 Funcionamento do DES

A seguir, um passo do DES:

1. Seja X uma palavra de 56 bits
2. $X \leftarrow PI_1(ch)$
3. Sejam C e D duas palavras de 28 bits
4. $(C,D) \leftarrow X$
5. Para i de 1 até 16 faz-se:
 - a. $C \leftarrow C \triangleleft \sigma(i)$
 - b. $D \leftarrow D \triangleleft \sigma(i)$
 - c. $X \leftarrow (C,D)$
 - d. $Ch_i \leftarrow PI_2(x)$

2.4 Algoritmos de Chave Pública – Assimétricos (Terada, 2000)

Em 1976, Whitfield Diffie e Martin Hellman, da Universidade de Stanford – Estados Unidos, introduziram publicamente o conceito de criptografia de chave pública ou assimétrica implementando uma técnica que visava atacar diretamente os três pontos falhos da criptografia por chave simétrica: o compartilhamento e o gerenciamento das chaves, assim como a autenticação. Essa técnica por eles apresentada, utiliza a filosofia de chaves complementares onde uma delas é utilizada para codificar os dados e outra distinta é utilizada para decodificar. A operação executada com a utilização de uma chave só pode ser revertida com a utilização da sua respectiva chave complementar.

Com isto, podemos implementar uma política onde uma das chaves é de conhecimento de todos (chave pública) e a outra é reservada e de conhecimento restrito (chave privada). Esta estratégia permite que possamos trocar as informações utilizando a combinação de chaves públicas e privadas dos dois indivíduos envolvidos no processo e resolver os três problemas da chave simétrica.

Para ilustrar esta técnica, vamos adotar o seguinte cenário:

Alice e Beto desejam se corresponder de forma segura, garantindo a autoria das cartas e a confidencialidade das mesmas enquanto estiverem no caminho de um para o outro. Então, eles combinaram que cada um iria construir dois modelos de caixas de madeira com duas chaves distintas resultando ao final em duas caixas chamadas públicas e duas caixas chamadas privadas. Cada um seria o proprietário de uma caixa pública e uma caixa privada com as suas respectivas chaves:

Caixa Pública de *Alice* = C_{uj}

Chave Pública de *Alice* = K_{uj}

Caixa Privada de *Alice* = CR_j

Chave Privada de *Alice* = KR_j

Caixa Pública de *Beto* = C_{um}

Chave Pública de *Beto* = K_{Um}

Caixa Privada de *Beto* = CR_m

Chave Privada de *Beto* = KR_m

Onde a quantidade de todas as caixas e das chaves públicas é infinita e apenas *Alice* e *Beto* possuem as respectivas chaves privadas.

Feito isto, a comunicação utilizando a criptografia assimétrica entre os dois pode ser implementada da seguinte forma:

- *Alice* escreve uma carta e coloca em uma caixa privada dele (CR_j), como só *Alice* tem a chave para a sua caixa privada, temos a garantia de que foi *Alice* quem criou esta caixa e portanto é o responsável pelo seu conteúdo:

$$\text{Carta} \approx KR_j \rightarrow \boxed{CR_j}$$

- *Alice* pega a sua caixa CR_j e utiliza a chave pública de *Beto* K_{Um} para gerar uma caixa pública dela (C_{Um}) e que por tanto, só poderá ser aberta por ela provendo a segurança no meio de transporte que irá entregá-la:

$$\boxed{CR_j} \approx K_{Um} \rightarrow \text{Carta}$$

- *Beto* recebe a C_{Um} enviada por *Alice* e utiliza a sua chave privada (KR_m) para abrir a caixa e pegar o seu conteúdo que é a caixa privada de *Alice* (CR_j):

$$\text{Carta} \approx KR_m \rightarrow \boxed{CR_j}$$

- De posse da caixa privada de *Alice*, *Beto* utiliza a chave pública de *Alice* (K_{Uj}) para abrir a caixa e pegar a carta. Ao utilizar esta chave e obter êxito, *Beto* tem certeza de que foi *Alice* quem construiu a caixa e portanto é o responsável pela carta nela contida.

$$\boxed{CR_j} \approx K_{Uj} \rightarrow \text{Carta}$$

Esta nova forma de implementar a criptografia, foi a verdadeira grade revolução desta tecnologia, apenas ressaltando que Diffie e Hellman não chegaram a implementar esta técnica mas deram as diretrizes para tal, levantando os itens que deveriam se contemplados para que houvesse sucesso.

A estrutura que tornaria possível a implementação de uma criptografia assimétrica ou de chave pública deveria ser essencialmente se calcar num algoritmo onde:

- Fosse computacionalmente fácil para qualquer das partes gerar o seu par de chaves (K_U e K_R)
- Fosse fácil para o emissor A, conhecendo a chave pública do receptor B (K_{Ub}) e a mensagem a ser codificada (M), gerar o dado cifrado (C), ou seja:

$$M \approx K_{Ub} \rightarrow C$$

- Fosse fácil para o receptor B, decifrar a mensagem codificada (C) usando a sua chave privada, ou seja:

$$C \approx K_{Rb} \rightarrow M$$

- Fosse computacionalmente inviável para um intruso, conhecendo as chaves públicas de B (K_{Ub}), determinar a chave privada de B (K_{Rb}), ou seja:

$$K_{Ub} \nrightarrow X \nleftarrow K_{Rb}$$

- Fosse inviável para um intruso, conhecendo a chave pública de B (K_{Ub}) e o dado codificado (C), chegar até o dado original (M), ou seja:

$$K_{Ub} \approx C \nrightarrow X \nleftarrow M$$

Tal implementação só seria possível com a utilização de funções *one-way*. Uma função *one-way* é toda aquela que mapeia o seu domínio em uma faixa onde para qualquer valor da função exista um único inverso, com a condição de que o cálculo da função seja fácil ao passo que o cálculo da inversa seja inviável.

$$Y = f(X) \rightarrow \text{deve ser fácil}$$

$$X = f^{-1}(Y) \rightarrow \text{deve ser inviável}$$

O outro recurso que tornou possível a implementação dos algoritmos de criptografia por chave assimétrica foi a função *trap-door one way*, que é uma função one-way que pode ser facilmente reversível a partir do conhecimento de alguma informação sobre a chave.

Para ser considerada uma função *trap-door one way*, a função deve atender às seguintes condições:

$Y = f_k(X) \rightarrow$ cálculo fácil, se K e X são conhecidos;

$X = f_k^{-1}(Y) \rightarrow$ cálculo fácil, se K e Y são conhecidos;

$X = f_k^{-1}(Y) \rightarrow$ cálculo inviável, se Y é conhecido, mas K não é.

Como 'fácil' geralmente definimos os problemas que podem ser resolvidos num tempo que tem relação polinomial com o tamanho da entrada. Isto significa que sendo n o número de bits de entrada e a uma constante, o tempo de processamento para esta função será proporcional a n^a .

Para os problemas 'inviáveis' consideramos aqueles em que o esforço para a solução cresce mais rapidamente que o tempo polinomial em função do tamanho da entrada. Temos como exemplo uma função cuja entrada tem tamanho n e o tempo de solução é proporcional a 2^n .

2.5 Os Problemas da Chave

Em ambos os tipos de criptografia vimos um consenso: ambos precisam de uma chave. A chave é a porta de entrada e saída dos seus dados sigilosos, assim como a chave da sua casa abre e fecha a porta do seu lar. Então entra um problema: como proteger a chave do inimigo?

Na criptografia de chave simétrica, vimos que a mesma chave é utilizada para encriptar e desencriptar. O problema é como enviar a chave através de um canal inseguro para o destinatário da mensagem?

Burnett (2002) sugere uma solução para este problema: encriptar a chave e utilizar uma senha para recuperá-la. Acredita-se que criptografar uma chave criptográfica é a melhor maneira de proteger uma chave. Na literatura esta chave é conhecida como KEK (Key Encryption Key).

Burnett (2002) explica que outro problema está no tamanho da chave. Se sua chave for muito pequena o tempo para um invasor “quebrar” esta chave é muito pequeno. Digamos que se utilize uma chave de 30 bits que dá aproximadamente um número entre 0 e cem bilhões. Burnett explica que, na maioria das vezes, pouco mais de 50% da chave é explorada até ser quebrada. Utilizando um computador que calcula as chaves a uma razão de 1 bilhão de chaves por segundo, se levará 50 segundos para pesquisar 50% do tamanho da chave e possivelmente encontrar a chave correta.

2.1 Aritmética Modular (Tenen, 2004)

A aritmética modular m é baseada no resto da divisão de uma operação entre dois números inteiros por m . Esta operação pode ser a soma ou o produto entre dois inteiros. Quando a operação for a soma, temos a adição de inteiros módulo m e quando for o produto temos a multiplicação de inteiros módulo m .

Na aritmética modular m , todos os números são representados em m por um, ou seja, a representação de números inteiros positivos. Os números negativos são representados em m por $1, \dots, m-1$. Um número é representado pelo resto da sua divisão por m .

Por exemplo, o conjunto de inteiros positivos representados na aritmética modular 4 é:

$$0 \rightarrow 0 \bmod(4)$$

$$1 \rightarrow 1 \bmod(4)$$

$$2 \rightarrow 2 \bmod(4)$$

$$3 \rightarrow 3 \bmod(4)$$

$$4 \rightarrow 0 \bmod(4)$$

$$5 \rightarrow 1 \bmod(4)$$

$$6 \rightarrow 2 \bmod(4)$$

3 FUNDAMENTAÇÃO MATEMÁTICA

Para o melhor entendimento dos algoritmos que serão abordados posteriormente, segue-se um apanhado geral dos fundamentos matemáticos nos quais eles estão sedimentados. Como o objetivo aqui não é o estudo profundo das bases matemáticas da criptografia, o estudo foi feito apenas para os fundamentos necessários ao entendimento da funcionalidade dos algoritmos.

3.1 Aritmética Módulo m (Terada, 2000)

A aritmética módulo m é baseada no resto da divisão de uma operação entre dois números inteiros por m . Essa operação pode ser a soma ou o produto entre dois inteiros. Quando a operação for a soma, temos a adição de inteiros ao módulo m e quando for o produto temos a multiplicação de inteiros módulo m .

Na aritmética módulo m , todos os números são superiores ou iguais a 0 por tanto, ele é uma aritmética de números inteiros positivos. Os números utilizados pertencem ao conjunto $\{0, 1, \dots, m-1\}$. Um número é representado pelo resto da sua divisão por m .

Por exemplo, o conjunto de inteiros positivos representados na aritmética módulo 4 escreve-se:

$$0 \rightarrow 0 \bmod (4)$$

$$1 \rightarrow 1 \bmod (4)$$

$$2 \rightarrow 2 \bmod (4)$$

$$3 \rightarrow 3 \bmod (4)$$

$$4 \rightarrow 0 \bmod (4)$$

$$5 \rightarrow 1 \bmod (4)$$

$$6 \rightarrow 2 \bmod (4)$$

3.1.1 Adição Módulo m

Sendo $m > 1$ um inteiro e indiquemos por $\mathbf{R}_m$ o conjunto completo de restos mínimos positivos, módulos m : $\mathbf{R}_m = \{0, 1, 2, 3, \dots, m-1\}$. Tomamos x, y como pertencentes a $\mathbf{R}_m$ ($x, y \in \mathbf{R}_m$), teremos que a soma módulo m de x com y é igual ao resto da divisão de $x + y$ por m . A notação que será usada para indicar a soma módulo m de x com y será: $(x + y) \bmod m$. Como exemplo:

$$(5 + 6) \bmod 4 = 3$$

pois na divisão de 11 (resultante de $5 + 6$), por 4 o resto é 3.

Como a soma de dois inteiros quaisquer é sempre outro inteiro, o resto da divisão de um inteiro qualquer por m está sempre em $\mathbf{R}_m$, então a adição módulo m de dois inteiros sempre estará em $\mathbf{R}_m$.

3.1.2 Multiplicação Módulo m

Para cada par de elementos $x, y \in \mathbf{R}_m$, indicaremos por $(x \cdot y) \bmod m$ como o produto módulo m de x por y . $(x \cdot y) \bmod m$ é igual ao resto da divisão de xy por m . Como exemplo:

$$(5 \cdot 3) \bmod 2 = 1$$

Como o produto de dois inteiros quaisquer é sempre outro inteiro, e o resto da divisão de um inteiro qualquer por m está sempre em $\mathbf{R}_m$, então a multiplicação módulo m de dois inteiros sempre estará em $\mathbf{R}_m$.

3.2 Congruência

Sejam a , b , e m números inteiros, $m > 0$. Dizemos que a é congruo a b , módulo m , se a e b , quando divididos por m , tem o mesmo resto. A representação gráfica é: $a \equiv b \pmod{m}$.

Por exemplo:

$$7 \equiv 15 \pmod{8}, \text{ pois}$$

$$7/8 = 0 \text{ com resto } 7 \text{ e}$$

$$15/8 = 1 \text{ com resto } 7;$$

Uma das propriedades da congruência nos diz que todo número inteiro é congruo a ele mesmo, desde que o módulo seja maior que 0 (zero). Ou seja:

$$a \equiv a \pmod{m} \quad \forall a \pmod{m} > 0 \text{ e para } \forall m > 0.$$

3.3 Álgebra de Boole – Funções Lógicas (Carvalho, 2001)

A principal característica das funções lógicas é que neste universo teremos apenas dois estados possíveis:

- o estado 0 (zero) e
- o estado 1 (um)

O estado zero (0) representará, por exemplo: portão fechado, aparelho desligado, ausência de tensão, chave aberta, “não”, etc; o estado um (1) representará, então: portão aberto, aparelho ligado, presença de tensão, chave fechada, “sim”, etc.

Note que, se representarmos por zero (0) uma situação, representaremos por um (1) a situação contrária.

3.3.1 Função “OU” ou “OR”

A função OU é aquela que assume o valor um (1) quando uma ou mais variáveis da entrada forem iguais a um (1) e assume valor zero (0) se, e somente se todas as variáveis forem iguais a zero (0). A sua representação gráfica usual é o símbolo “+”, ou seja, $S=A+B$.

	A	B	S
1º	0	0	0
2º	0	1	1
3º	1	0	1
4º	1	1	1

Tabela 3.1: Tabela Verdade de uma função OU ou OR

Na tabela 2.1, temos todas as combinações possíveis dos valores de A e B com os respectivos valores S que a função OU assume para cada combinação.

3.3.2 Circuito OU-Exclusivo

Um circuito ou-exclusivo é um circuito combinacional, mas podemos considerá-lo também como sendo um bloco lógico básico. A função que ele executa consiste em fornecer o valor 1 (um) à saída quando as variáveis de entrada forem diferentes entre si. Para representarmos uma combinação que utilize um circuito ou-exclusivo, utilizamos o símbolo “ $\oplus$ ”, ou seja $S=A \oplus B$.

	A	B	S
1º	0	0	0
2º	0	1	1
3º	1	0	1
4º	1	1	0

Tabela 3.2: Tabela Verdade de circuitos ou-exclusivo

3.4 Máximo Divisor Comum (MDC) (Prazeres, 2000)

Para introduzir o conceito de Máximo Divisor Comum, vamos remeter ao conceito de divisibilidade considerando o conjunto dos números inteiros, onde tomamos a e b sendo $b \neq 0$.

Dizemos que “ b divide a ” (ou que “ b é divisor de a ” ou que “ a é múltiplo de b ”) quando existe um $n \in \mathbb{Z}$ tal que $a = nb$. Representamos esta propriedade pelo símbolo “ $b|a$ ” (“ b divide a ”). Representamos o fato de b não dividir a por “ $b \nmid a$ ” (“ b não divide a ”).

Neste contexto, é considerado a e $b \in \mathbb{Z}$ e não nulos, definimos o máximo divisor comum (MDC) de a e de b , denotado por “ (a, b) ” ou “ $\text{mdc}(a, b)$ ”, como sendo o maior número inteiro que divide a e b simultaneamente.

Como exemplo podemos calcular o MDC de 8 e 12 listando os divisores positivos de cada um deles e encontrando o maior número comum nas duas listagens. Sendo assim, temos:

Divisores de 8 = 1, 2, 4, 8.

Divisores de 12 = 1, 2, 3, 4, 6, 12.

Logo o maior número comum às duas listagens é o 4, sendo ele por tanto o Máximo Divisor Comum de 8 e 12. Ou seja, $\text{mdc}(8, 12) = 4$.

O algoritmo de Euclides calcula o MDC de dois inteiros de forma simples, através da estrutura abaixo:

Entrada: Dois números inteiros não negativos a e b , sendo $a \geq b$.

Saída: O máximo divisor comum de a e b .

Algoritmo:

Enquanto $b \neq 0$, faça o seguinte:

Atribua a r o valor de $a \bmod b$; ($r \leftarrow a \bmod b$)

Atribua a a o valor de b ; ($a \leftarrow b$)

Atribua a b o valor de r . ($b \leftarrow r$)

Quando $b = 0$, devolva o valor atual de a .

3.5 Números Primos (Agrawal, 2002)

Alguns estudiosos consideram os números primos como sendo os números mais importantes de todos por serem estes os “átomos” da matemática, dado ao fato de que qualquer outro número inteiro pode ser criado a partir da multiplicação de combinações de números primos. Sendo assim, podemos considerar os números primos como sendo “os blocos de construção numérica”.

Isto posto, definimos um número primo como sendo todo número inteiro de $p \in \mathbb{Z}$, sendo $p > 1$, cujos únicos divisores sejam 1 e p . Os números não primos são chamados de “compostos” pois podem ser fatorados num produto $a = bc$, sendo $a \in \mathbb{Z}$ e $a > 1$.

Como citação histórica, podemos relatar que os gregos antigos não consideravam o algarismo 1 (um) como sendo um número, posto que este é o “símbolo da unidade”, sendo então uma referência para os números e não um deles. De forma análoga, o 0 (zero) também não era referenciado como o número por ser a representação do nulo. Sendo assim, o 0 (zero) e o 1 (um) não são classificados nem como primos nem como compostos.

O Teorema Fundamental da Aritmética nos dá conta que, para todo número natural $a > 1$ existem números primos $p_1, p_2, \dots, p_r$ ($r \geq 1$), de maneira que $a = p_1 \cdot p_2 \cdot \dots \cdot p_r$. Além disso, se também $a = q_1 \cdot q_2 \cdot \dots \cdot q_s$ ($s \geq 1$), onde os q_i são igualmente primos, então $r = s$ e cada p_i é igual a algum dos q_j . Ou seja, qualquer inteiro positivo pode ser fatorado em uma combinação única de números primos. Esta forma fatorada é chamada de decomposição em fatores primos e é claro que, numa decomposição $a = p_1 \cdot p_2 \cdot \dots \cdot p_r$, nem sempre todos os fatores serão diferentes entre si. A reunião de possíveis fatores iguais nos leva à expressão

$$a = p_1^{a_1} p_2^{a_2} \dots p_s^{a_s}$$

onde $1 \leq s \leq r$, $p_i \neq p_j$ sempre que $i \neq j$ e $a_i \geq 1$ ($i = 1, 2, \dots, s$). Se, além disso, impusermos $p_1 < p_2 < \dots < p_s$, teremos a chamada “decomposição canônica” ou como “forma padrão” de a . Por exemplo:

$$91 = 7 \cdot 13,$$

$$60 = 2^2 \cdot 3 \cdot 5,$$

$$126 = 2^2 \cdot 3^2 \cdot 7,$$

$$11011 = 7 \cdot 11^2 \cdot 13.$$

3.5.1 Números Primos Relativos

Dizemos que dois números são primos relativos quando eles não possuem nenhum fator primo em comum, ou seja, o único fator comum entre eles é o 1. Isto equivale a dizer que, considerando a e $b \in \mathbb{Z}$ e diferentes de 0, a e b são primos relativos se $\text{mdc}(a, b) = 1$. Podemos citar como exemplo os números 8 e 15 como sendo primos entre só, pois:

Divisores de 8 = 1, 2, 4 e 8

Divisores de 15 = 1, 3, 5, 15.

Observa-se que o único fator comum entra as duas listagens é o 1.

3.5.2 Geração de Números Primos

Atualmente, não existe uma fórmula ou técnica usual para a geração de números primos grandes, sendo este fato um dos pilares para a sustentação dos algoritmos como o RSA. O procedimento que geralmente é utilizado para encontrarmos números primos grandes, consiste em pegar um número ímpar aleatório e testar se ele é primo. Mesmo com estes testes, ainda não teremos a certeza de que encontramos realmente um número primo, já que os teste realizados são probabilísticos.

Um dos métodos para testar números primos passa pelo “pequeno teorema” de Fermat que diz se um inteiro n é primo, então, $n|(2^n - 2)$, ou seja, $2^n - 2$ for divisível por n . Se o resultado desta operação para um dado número for falso, o número não pode ser primo, entretanto, se for verdadeiro ainda não teremos a garantia de que o número é realmente primo. Como exemplo podemos citar o número 341 que não é primo, porém, $(2^{341} - 2)$ é divisível por 341.

O algoritmo de Miller-Rabin (Terada, 2000) é um dos mais eficientes e populares algoritmos para teste de números primos. Neste algoritmo, um número n é testado envolvendo alguns cálculos com um outro inteiro a , escolhido aleatoriamente. Se n perde no teste com a , então n é um número composto. Se n passa no teste com a , selecionamos um novo a aleatório e repetimos o teste. A cada novo teste passado, aumentamos a probabilidade de que n seja realmente um número primo. Se n passar em muitos teste com diversos a 's, temos uma probabilidade muito grande de termos atingido um número primo verdadeiro.

Este tipo de teste pode parecer tedioso e exaustivo, mas, na verdade, não são necessárias infintas tentativas até encontrarmos um número primo. Um resultado da teoria dos números, conhecida como a teoria dos números primos, afirma que o primo próximo a n está separado por um espaço de $\log_n$ números inteiros. Sendo assim, deverão ser testados $\ln n$ inteiros antes que se encontre o primeiro número primo. De fato, como os números pares são imediatamente rejeitados, a figuração correta é $\ln (n/2)$. Por exemplo, se um primo da ordem de magnitude de 2^{200} fosse testado, seriam necessárias algo em torno de $\ln (2^{200}/2) = 70$ tentativas para encontrar um primo.

3.6 *Pequeno Teorema De Fermat* (Carvalho, 2001)

Com o objetivo de encontrar uma fórmula que gerasse números primos (Carvalho, 2001), o estudioso Pierre de Fermat desenvolveu o teorema onde afirmava que:

$$a^{p-1} \equiv 1 \pmod{p}$$

sendo a e p primos relativos. Ou seja, se a é um número inteiro primo relativo de p e p é um número primo, então $(a^{p-1}) - 1$ é múltiplo de p .

3.7 *Função Φ de Euler (Função TOTIENT)* (Prazeres, 2000)

Em 1760, Leonard Euler desenvolveu uma generalização do teorema de Fermat (Terada, 2000) e pôde estabelecer uma função que nos retornasse um número indicador da quantidade de números positivos e primos relativos a n e menores que n . Ou seja, a função $\Phi : \mathbb{N}^* \rightarrow \mathbb{N}^*$ que associa a cada $n \in \mathbb{N}^*$ o número de elementos do conjunto

$$\{k \in \mathbb{N}^* \mid 1 \leq k \leq n \text{ e } \text{mdc}(k,n) = 1\}$$

Esta função é conhecida como a Função Φ (Phi) de Euler ou Função Totient de Euler e é representada por $\Phi(n)$ ou $\varphi(n)$.

Como exemplo, podemos encontrar $\Phi(8) = 4$, pois o conjunto W , que representa o intervalo fechado entre 1 e 8, existem 4 números – conjunto $\text{Pr}(8)$ – primos relativos a 8 menores que ele próprio.

$$W = \{1, 2, 3, 4, 5, 6, 7, 8\}$$

$$\text{Pr}(8) = \{1, 3, 5, 7\}$$

$$\#\text{Pr}(8) = 4$$

Considerando p um primo é fácil concluir que de 1 a p , o único elemento não primo relativo a p é próprio p , então temos que $\Phi(p) = p - 1$.

Como uma forma prática de calcular $\Phi(n)$, vamos considerar p e q como sendo dois primos tal que $n = pq$. Assim sendo, podemos afirmar que:

$$\Phi(n) = \Phi(pq) = \Phi(p) \cdot \Phi(q) = (p-1) \cdot (q-1).$$

3.8 Funções Hash

Chamamos de função hash (Terada, 2000), toda função, matemática ou não, que gera uma saída de tamanho fixo – chamada de “valor hash” – a partir de um valor de entrada de tamanho variável – chamado de pré-imagem. O valor hash é a assinatura da pré-imagem e deve ser único para cada uma delas. Sendo assim, conhecendo-se o valor hash e uma determinada pré-imagem, podemos afirmar se foi ela ou não que gerou este valor.

A função Hash mapea strings de tamanho finito e aleatório em strings de tamanho fixo tendo de ser livre de colisão, ou seja, uma pré-imagem diferente da que gerou o valor hash não pode gerar o mesmo valor hash. Para um domínio D e uma faixa R com $h: D \rightarrow R$ e $|D| > |R|$ temos uma função que é de muitos para um, o que implica na existência inevitável de colisões. Se restringirmos h para um domínio de entrada com t -bits ($t > n$) e se este mesmo h for aleatório, onde todas as saídas são equiprováveis, então 2^{t-n} entradas irão mapear cada saída, e a probabilidade de duas entradas gerarem a mesma saída é 2^{-n} (independente de t).

Uma função hash baseia-se no princípio de mapear grandes domínios em uma faixa menor. Em uma função hash ideal, o valor hash serve como representação compacta da string de entrada, podendo ser utilizada como uma identificação única da string. A garantia desta característica requer que a função hash:

- Possa ser usada em mensagens de qualquer tamanho;
- Produza um resultado de tamanho fixo;
- $f(x)$ seja relativamente fácil de calcular;
- Resistência à pré-imagem, deve ser inviável encontrar uma pré-imagem x a partir de y tal que $f(x)=y$, isto é, a partir de um valor de saída de uma função hash deve ser difícil obter o valor de entrada;

4 QUEBRA E ATAQUES EM CRIPTOGRAFIA

4.1 O problema da Chave

Ao longo deste trabalho, muito se falou em chave de criptografia. Mas afinal, o que é chave de criptografia e para que serve?

Burnett (2002) explica que o termo “chave” vem do mesmo princípio de uma chave que usamos para abrir a porta da nossa casa. É um padrão que utilizamos para criptografar ou descriptografar mensagens.

Em criptografia chaves são geralmente seqüências de números, letras e símbolos aleatórios que são informados pelo usuário.

Para Burnett (2002), o uso da chave se faz necessário pela segurança que ela oferece. Mesmo que o invasor saiba qual foi o algoritmo utilizado para encriptar a mensagem, ele não conseguirá resgatá-la sem o conhecimento da chave.

Além disso, é mais fácil proteger uma pequena chave do que um documento inteiro.

4.2 Ataque à Chave de Criptografia

“O conjunto de métodos matemáticos para quebrar um algoritmo ou um texto ilegível é chamado ataque ou criptanálise” (Terada, 2000. P.19).

Na criptografia, um invasor é conhecido como criptanalista (Terada, 2000). O criptanalista irá desenvolver ataques a dados criptografados em busca da chave secreta de criptografia.

Burnett (2002) explica que:

“Todos os algoritmos podem ser ‘quebrados’; os melhores são simplesmente aqueles algoritmos suficientemente fortes para suportar um ataque pelo maior tempo possível, fazendo com que a quebra chegue ‘tarde demais’. Portanto, o trabalho de um criptanalista é encontrar as fraquezas que possam ajudar a alguém a quebrar o algoritmo mais rapidamente. O criptógrafo desenvolve sistemas de criptografia; o criptanalista procura por suas fraquezas.”

(Burnett, 2002 P. 15)

Uma das formas de ataque mais comuns e difundidas de ataque à chave criptográfica é conhecida como ataque de força bruta. Este método consiste em tentar todas as chaves possíveis até que a correta seja identificada (Burnett, 2002).

O tempo que o invasor levará para quebrar a chave vai depender do tamanho utilizado da chave para criptografar o texto.

Essencialmente, pouco mais de 50% da chave é pesquisada até ser encontrada (Burnett, 2002).

O tamanho da chave “ideal” vai depender de quanto tempo se deseja que o segredo fique seguro. Os padrões atuais de chave simétrica, encontram-se, hoje, em 128 bits.

Sobre o tamanho da chave, Terada (2000), explica:

A cada bit adicionado a chave, dobra-se o número de chaves possíveis. Tomemos como exemplo, uma chave com $ch = 3$ bits:

000 001 010 011 100 101 110 111

Agora com 4 bits:

0000 0001 0010 0011 0100 0101 0110 0111

1000 1001 1010 1011 1100 1101 1110 1111

Assim como adicionar mais um bit, dobra o número da chave possíveis, dobra-se também o tempo médio que um ataque de força bruta levará para encontrar a chave correta.

Em 1997, os laboratórios RSA propuseram alguns desafios de teste de tempo de um ataque de força bruta. Nesta experiência, uma chave de 40 bits foi quebrada em três horas e uma chave de 48 bits em 280 horas. No mesmo ano, foi publicado um desafio de 64 bits. Até dezembro de 2000, ele ainda não havia sido resolvido.

É interessante frisar que em todos esses casos, centenas ou até milhares de computadores operavam conjuntamente para quebrar as chaves. Um invasor não seria capaz de reunir a força de centenas de computadores e talvez não possuía uma máquina específica para quebrar um algoritmo em particular.

Para fins de entendimento, vamos imaginar uma situação exageradamente pior: examinar 1% do espaço de uma chave de 56 bits em um segundo e 50% em um minuto:

Bits	1% do espaço da chave	50% do espaço da chave
56	1 segundo	1 minuto
57	2 segundos	2 minutos
58	4 segundos	4 minutos
64	4,2 minutos	4,2 horas
72	17,9 horas	44,8 dias
80	190,9 dias	31,4 anos
90	535 anos	321 séculos
108	140.000 milênios	8 milhões de milênios
128	146 bilhões de milênios	8 trilhões de milênios

Tabela 4.1: Um cenário pior do que a pior das situações: quanto tempo um ataque de força bruta levaria quanto aos vários tamanhos de chave (Burnett, 2002)

De acordo com Burnett (2002), mesmo com o avanço rápido da tecnologia, é quase certo que nunca precisará utilizar uma chave maior que 512 bits (64 bytes).

Suponha que cada átomo do universo conhecido (aproximadamente 2300) fosse um computador e que cada um desses computadores pudessem verificar 2300 chaves por segundo. Isso levaria cerca de 2162 milênios para pesquisar 1% do espaço de uma chave de 512 bits. De acordo com a teoria do Big Bang, o tempo decorrido desde a criação do universo é de 224 milênios (Burnett, 2002).

Dadas essas dificuldades, os criptanalistas buscam outros tipos de ataques a chave que não os de força bruta. Terada (2000), explica que ao invés de tentar reproduzir a chave o criptanalista pode tentar reproduzir o PRNG e a semente que foi utilizada para criar a chave.

Um PRNG (Pseudo-random number generator) ou gerador de números pseudo-aleatórios são algoritmos que produzem o que é chamado de números pseudo-aleatórios. Pseudo-aleatórios pois os números são repetíveis (Burnett, 2002) independente de onde se instalar o programa ou o tempo de execução do programa.

A utilização desses PRNG's por si só, não apresentam muita segurança. Por isso que no seu funcionamento são agregados dados aleatórios de entrada para, então, gerar um número. A esses dados aleatórios, chamamos de semente.

“Na prática, essa semente pode ser várias coisas: a hora do dia em milissegundos, medidas em constate alteração do estado de um computador, entradas de usuários e outros valores” (Burnett 2002 p.23).

Se o criptanalista conhece um PRNG específico e o método de coleta da semente utilizada, ele pode alimentar o PRNG com a semente e criar uma mesma chave.

4.3 *Ataques por Chaves Conhecidas* (Prazeres, 2000)

Neste tipo de ataque, o criptanalista conhece algumas chaves já usadas e utiliza tal conhecimento para deduzir chaves novas

4.4 *Ataques Por Dicionário* (Prazeres, 2000)

Este tipo de ataque, muito parecido com o ataque de força bruta, é muito usado para deduzir senhas quando o criptanalista tem acesso ao arquivo de senhas legítimas criptografadas; o criptanalista calcula antecipadamente as senhas y correspondentes às senhas x mais comuns (como nome de pessoas combinadas com números de telefones, etc) e compara com y calculados e armazenados no arquivo de senhas legítimas; se encontrar um y igual no arquivo, ele já possui a senha x válida correspondente.

Neste tipo de ataque fica clara a importância da conscientização do usuário na hora de escolher as senhas ou chaves criptográficas; quanto mais frágil a senha, mais rápido o criptanalista terá êxito na quebra da chave.

4.5 *Ataques ao Algoritmo* (Prazeres, 2000)

Os ataques aos algoritmos de criptografia não são os mais comuns, mas alguns criptanalistas o usam para tentar decifrar a mensagem sem o conhecimento da chave. Listaremos a seguir, os principais.

4.5.1 Ataques de texto simples (legível) conhecido

Esse tipo de ataque ao algoritmo, funciona da seguinte maneira:

O Criptanalista possui e analisa pares (x,y) de legível e ilegível correspondentes. Neste ataque o criptanalista conhece o algoritmo utilizado, mas não a chave. Ele observa por exemplo, que cada 14º bit de um dado texto cifrado é o mesmo bit que cada 12º bit do seu texto simples e assim por diante, independentemente da chave.

Sendo assim, o criptanalista pode simplesmente examinar o texto cifrado e decifrar as partes do texto simples. Sem conhecimento da chave.

Em resumo, se o criptanalista puder computar a chave a partir de um pedaço de um texto cifrado e do texto simples correspondente, o restante das mensagens também poderá ser computado.

4.5.2 Ataque por Só-Texto-Ilegível

O criptanalista tenta adquirir o conhecimento útil a quebra analisando apenas um ou mais ilegíveis y . Se esse tipo de ataque for computacionalmente viável, o algoritmo em questão é considerado fraco e totalmente inseguro e inútil.

4.5.3 Ataque por Texto Legível Escolhido

Nesse ataque, o criptanalista pode escolher os legíveis x e obter os y correspondentes. Ele vai escolher x que apresente alguma característica estrutural que aumente o seu conhecimento do algoritmo e da chave em uso. Com o conhecimento adquirido, ele pode deduzir o legível correspondente a um ilegível novo.

4.5.4 Ataque Adaptativo Por Texto Legível Escolhido

Além do suposto no tipo anterior, a escolha de um novo x pelo criptanalista pode depender dos ilegíveis $\bar{y}$ analisados anteriormente.

Desta forma, a escolha de um novo x é condicionada ao conhecimento já adquirido pela análise dos $\bar{y}$ já analisados.

4.5.5 Ataque por Texto Ilegível Escolhido

O criptanalista escolhe inicialmente o ilegível y e então obtém o legível x correspondente. Supõem-se que o criptanalista tenha acesso apenas ao algoritmo de decryptografia (sem acesso a chave) e o seu objetivo é, mais tarde, sem ter mais acesso à decryptografia, ser capaz de deduzir x correspondente a um y novo.

4.5.6 Ataque Adaptativo por Texto Ilegível Escolhido

Além do suposto no tipo anterior, a escolha de um novo y pelo criptanalista pode depender dos ilegíveis $\bar{y}$ analisados anteriormente. Desta forma, a escolha de um novo y é condicionada ao conhecimento já adquirido pela análise dos $\bar{y}$ anteriores.

5 FERRAMENTAS DE SEGURANÇA

As ferramentas de segurança visam prestar uma proteção onde a rede encontra-se mais vulnerável.

A seguir veremos algumas das principais soluções para evitar, ou pelo menos minimizar, ataques com danos à rede.

5.1 Firewall

De acordo com Torres (2001), um *firewall* é um poderoso roteador interligando duas redes: a pública ou insegura e a privada ou segura.

Sua principal função é “escutar” todo o tráfego TCP/IP de entrada e saída. O *firewall* analisa os cabeçalhos dos pacotes IP e com base em uma tabela de regras, classifica como pacote permitido ou pacote negado.

5.2 Vírus Scanner

Muitos usuários têm seu primeiro contato com redes de computadores através do correio eletrônico (e-mail). É o serviço aplicativo mais utilizado na rede.

Trata-se de um serviço popular porque oferece um método rápido e conveniente de transferência de informações, além de acomodar, com um único mecanismo, de pequenas notas até grandes quantidades de memorandos. Não deve ser novidade que existem mais usuários enviando arquivos pelo correio eletrônico do que pelos protocolos de transferência de arquivos.

A partir dessa popularização desse serviço é que chegamos aos problemas na rápida disseminação dos vírus.

“Nove entre dez vírus vem por e-mail” (Fortes, 2002 p.64).

Com base nessa afirmação, fica fácil saber porque é tão importante softwares antivírus.

Os softwares antivírus ou *virus scanners* tem a função de varrer, detectar e eliminar os vírus de computador.

O seu funcionamento baseia-se em examinar o cabeçalho do arquivo em busca da assinatura do vírus, ou em busca de traços de código malicioso.

De acordo com Fortes (2002), 20 novos vírus surgem a cada dia infectando 113 máquinas para cada lote de mil computadores.

De posse desses dados, entende-se porque os softwares antivírus nem sempre eliminam todas as pragas.

Há soluções que, na dúvida, colocam o arquivo em quarentena e aguarda a criação de uma vacina para este.

5.3 *Intrusion Detection System – IDS*

Os *Intrusion Detection System* (Sistemas de Detecção de Intruso), segundo Fortes (2002), é um complemento do *firewall* atuando diretamente no tráfego da rede filtrando informações que passam pelo *firewall*. Assim, se um invasor furar um bloqueio, será detectado mais rapidamente.

De acordo com a Symantec (2002), os sistemas IDS visam fornecer soluções para a maioria dos ataques à redes consolidando em um pacote, as principais ferramentas para os problemas a seguir:

1. Ataques de Negação de Serviço (*Denial of Service – DOS*);

Esse tipo de ataque visa o travamento do servidor através de sobrecarregamento do tráfego de pacotes na rede (Spyman, 2000);

2. Verificações e validações de integridade (*Scanning & Probing*);

Esse tipo de ataque busca vulnerabilidades conhecidas, bem como verificar as portas TCP e UDP atrás de vulnerabilidades (Spyman, 2000);

3. Ataques a Senhas (*Passwords Attacks*);

Nesse tipo de ataque o invasor pode tentar conseguir a senha através de três modos: Roubo, adivinhação ou quebra. No primeiro, pode-se utilizar de um *scanner* e obter a senha; no segundo, senhas padrão são tentadas ou contas *guest*; e a caracteriza ataques de força bruta ou ataques de dicionário;

4. Obtenção de Privilégio;

Uma vez que invadiu o sistema, tenta ganhar acesso privilegiado. No Linux tenta se tornar o Root, no Windows NT o Administrador, e assim por diante (Spyman, 2000);

5. Inserção de Código hostil (*Trojans Horse, Virus, Back Doors*);

Ataques através de vírus, cavalos de tróia, *trojans*. Visam causar danos ou obter dados da rede que se deseja invadir (Spyman, 2000);

6. Vandalismo Cibernético (*Defacers*);

Caracterizado, principalmente, pela “pichação” de sites ou substituição das páginas originais por com outro conteúdo. Geralmente usadas pelos invasores para se vangloriarem da sua invasão (Spyman, 2000);

7. Roubo de Dados Proprietários (*Sniffers*);

De acordo com Spyman (2000), *Sniffers* são programas que, uma vez instalados, ficam “escutando” o tráfego na rede e enviando dados importantes para o invasor, que com base nessas informações, planeja o seu ataque à rede;

8. Fraude, Desperdício e Abuso;

Fraude envolve transferência ilegal de fundos, roubo de números de cartão de crédito; o desperdício e o abuso ocorrem por conta do usuário quando este utiliza recursos para tarefas que não sejam diretamente relacionadas com os objetivos da empresa, como: jogos, sites pornográficos, *chats*;

9. Adulteração das Trilhas de Auditoria;

Essa é a maneira que o invasor utiliza para apagar o seu rastro; esconder as pistas;

10. Ataque à Segurança da Infra-estrutura;

Esse tipo de ataque é mais caracterizado por invasores que já estão dentro da empresa; o invasor visa à modificação de *firewalls*, roteadores, contas de usuários.

Como visto, cada problema tem uma característica bem peculiar. A Symantec (2002), sugere como solução para esses problemas, uma vez que foram claramente identificados, um conjunto de estratégias de segurança através de sistemas IDS.

6 EXEMPLO DE CASO – A SEMENTE DO NETSCAPE

A criptografia de chave simétrica é um componente de SSL¹ que foi inventado por pesquisadores da Netscape. Após a versão 1.0 do seu navegador, a Netscape passou a oferecer implementação SSL.

Em algum ponto de uma sessão SSL, o código deve gerar uma chave. Para fazer isso, a implementação do Netscape utiliza um PRNG. Na versão 1.1 (distribuída em 1995), o código do PRNG coletava como semente a hora do dia, o ID do processo e o ID do processo pai.

Ian Goldberg e David Wagner apud Burnett (2002) decidiram testar a qualidade que uma semente para essas três fontes produziria. Eles descobriram que os ID's de processo eram fáceis de capturar se alguém tivesse acesso ao computador. Se ninguém tivesse acesso ao computador, bastava um pequeno teste de força bruta, pois cada ID tinha apenas 15 bits. E a data/hora do dia? Bem, o ano, o mês, o dia e até a hora e o minuto eram conhecidos; um invasor simplesmente teria de verificar quando ocorria a sessão de SSL. Quanto aos segundos, existem apenas 60 valores possíveis, já que a Netscape utiliza a hora do dia apenas em segundos e não em milisegundos.

Em 17 de setembro de 1995, Goldberg e Wagner informaram ao newsgroup Cypherpunks que eles poderiam encontrar a semente e, portanto a chave, em menos de um minuto. Se a chave fosse de 40 bits ou 128 bits, eles levariam apenas um minuto, já que o problema não era na chave mas no que gerou a chave.

A Netscape corrigiu o problema na versão 2.0 adicionando mais semente. Cada plataforma ainda tem fontes diferentes de sementes, mas entre as diversas plataformas dependentes de sementes, agora a Netscape utiliza a posição de cursor ou de mouse, o status da memória, o último pressionamento de tecla, o volume de áudio e vários outros parâmetros.

¹ SSL – Secure Socket Layer. Protocolo para comunicação segura na Internet desenvolvida pela Netscape (Netscape, 2002)

7 CONCLUSÃO E BIBLIOGRAFICAS

Após 11 de setembro de 2002, o mundo compreendeu que vivemos sob constante ameaça de algum tipo de ataque. A preocupação com esta nova realidade ultrapassou o âmbito dos governos e instituições financeiras e atingiu também as organizações que perceberam o significativo aumento das ameaças e vulnerabilidades a que estamos expostos, principalmente no que se refere a alta vulnerabilidade do ambiente digital.

A necessidade de proteger seus ativos e assegurar a continuidade das operações, tem levado as empresas a desenvolverem um sistema de gestão de segurança de informação a implementar controles baseados em análises dos riscos ao negócio e em requisitos legais e compatíveis com a natureza da sua atividade.

Por sua vez, os usuários de sistemas de informação ficaram mais alertas e cautelosos na hora de realizar operações on-line.

Estamos certos de que este trabalho contribuiu para entendermos as necessidades de segurança nas aplicações computacionais. Além disto, foram apresentados os principais conceitos de criptografia e os ataques mais comuns.

NETSCAPE – Secure Socket Layers. [online] Disponível na Internet via WWW. URL: <http://home.netscape.com/products/security/ssl/certformat.html> Arquivo capturado em 14 de novembro de 2002

PIAZZARES, Cassio Vinícius Sestini. Fundamentos teóricos da criptografia. 2000. Monografia (Graduação em Ciências da Computação) – Universidade de Salvador, Bahia.

SPYMAN. Manual completo do hacker, como ser e evitá-lo. 3ed., Rio de Janeiro: Book Express, 2000.

SYMANTEC. Sistema de detecção de intrusão para o ambiente corporativo: uma abordagem multi-nível. [online] Disponível na Internet via WWW. URL: <http://www.symantec.com/products/security/ida/index.html> Arquivo capturado em 14 de novembro de 2002.

TERADA, Raulo. Segurança de dados: criptografia em redes de computador. São Paulo: Edgard Blücher, 2000.

TORRES, Gabriel. Redes de computadores: curso completo. Rio de Janeiro: Axcel, 2001.

VALIM, Carlos Eduardo. Acesso negado. Revista Information Week Brasil. São Paulo: v.IV 4, n 74, p.36/48, jul. 2002.

8 REFERÊNCIAS BIBLIOGRÁFICAS

- AGRAWAL, Manindra. **Primes is in P**. 2002. Artigo – Indian Institute of Technology Kanpur – Índia.
- BARBOSA, Alexandre. **E-business com segurança**. Revista Internet Business. São Paulo: v.XVII, n195, jun. 2002., pg.25/37.
- BURNETT, Steve; PAINE, Stephen. **Criptografia e segurança: o guia oficial RSA**. Rio de Janeiro: Campus, 2002.
- CARVALHO, Daniel Balparda de. **Segurança de dados com criptografia: métodos e algoritmos**. 2ed., Rio de Janeiro: Book Express, 2001.
- COMER, Douglas E. **Redes de computadores e Internet**. 2ed., Porto Alegre: Bookman, 2001.
- FERREIRA, Aurélio Buarque de Holanda. **Novo dicionário da língua portuguesa**. 2ed., São Paulo: Nova Fronteira, 1994.
- FORTES, Débora. **Segurança: sua empresa está vulnerável?**. Revista Info Exame. São Paulo: v.XVII, n195, jun. 2002.
- MODULO, Security Magazine. **8ª Pesquisa nacional de segurança da informação**. [online] Disponível na Internet via WWW.URL: <http://www.modulo.com.br/comum/docs> arquivo capturado em 05 de novembro de 2002.
- NETSCAPE – **Secure Socket Layers**. [online] Disponível na Internet via WWW. URL: <http://home.netscape.com/products/security/ssl/certformat.html> Arquivo capturado em 14 de novembro de 2002.
- PRAZERES, Cássio Vinícius Serafim. **Fundamentos teóricos da criptografia**. 2000. Monografia (Graduação em Ciências da Computação) – Universidade de Salvador, Bahia.
- SPYMAN. **Manual completo do hacker: como ser e evitá-los**. 3ed., Rio de Janeiro: Book Express, 2000.
- SYMANTEC. **Sistemas de detecção de intruso para o ambiente corporativo: uma abordagem multi-nível**. [online] Disponível na Internet via WWW. URL: <http://www.symantec.com/products/security/ids/index.html> Arquivo capturado em 14 de novembro de 2002.
- TERADA, Routo. **Segurança de dados: criptografia em redes de computador**. São Paulo: Edgard Blücher, 2000.
- TORRES, Gabriel. **Redes de computadores: curso completo**. Rio de Janeiro: Axcel, 2001.
- VALIM, Carlos Eduardo. **Acesso negado**. Revista Information Week Brasil. São Paulo: v.IV 4, n.74, p.36/48, jul. 2002.

9 BIBLIOGRAFIA CONSULTADA

BERNSTEIN, Jerry. **Segurança na Internet**. Rio de Janeiro: Campus, 1997.

FAJARDO, Karine. **Lojas abrem suas portas com segurança**. Revista Internet World. Rio de Janeiro, v.III, n.35, p.38/43, jul. 1998.

GOMES, Ricardo Reis. **Assinatura digital X democracia**. [online] Disponível na Internet via WWW.URL:

<http://www.modulo.com.br/comum/docs/arquivo> capturado em 05 de novembro de 2002.

MODULO, E-Security. **Criptografia contemporânea**. Rio Grande do Sul [s.d].

O'BRIEN, James A. **Sistemas de informação: e as decisões gerenciais na era da internet**. 9ed., São Paulo: Saraiva, 2001.

PISTELLI, Daniela. **Criptografia**. [online] Disponível na Internet via WWW. URL:

<http://www.nucc.pucsp.br/novo/cripto/cripto.html> Arquivo capturado em 27 de março de 2002.

ROCKENBACH, Aléxis. **Easy-SET: um framework para o padrão SET**. [online] Disponível na Internet via WWW. URL:

<http://www.inf.ufrgs.br/pos/SemanaAcademica/Semana98/alexis.html> Arquivo capturado em 11 de abril de 2002.

SET Secure Electronic Transaction LLC. **FAQ SET**. [online] Disponível na Internet via WWW. URL:

<http://www.setco.org> última atualização em 14 de junho de 2002.

SILVA, Clayton Inácio. **E-Commerce**

[online] Disponível na Internet via WWW.URL:

<http://www.unifio.br/revista/rev2001/ecommerce.htm> arquivo capturado em 26 de agosto de 2002.

VERISIGN INC. - **Secure Socket Layers**. [online] Disponível na Internet via WWW. URL:

<http://www.verisign.com> última atualização em 14 de junho de 2002.