UNIVERSIDADE SÃO FRANCISCO

CURSO DE ENGENHARIA ELÉTRICA

CRIPTOGRAFIA E SEGURANÇA COMPUTACIONAL

Área de Engenharia Eletrica Modalidade Telecomunições

por

Rodrigo Silvio de Souza

Manoel Vitório Barbin, Prof. Mestrando
Orientador

Itatiba (SP), Novembro de 2004

UNIVERSIDADE SÃO FRANCISCO

CURSO DE ENGENHARIA ELÉTRICA

CRIPTOGRAFIA E SEGURANÇA COMPUTACIOANAL

Área de Engenharia Elétrica e Telecomunicações

por

Rodrigo Silvio de Souza

Relatório apresentado à Banca Examinadora do Trabalho de Conclusão do Curso de Engenharia Elétrica para análise e aprovação.
Orientador: Manoel Vitório Barbin, Prof. Mestrando

Itatiba (SP), Novembro de 2004

SUMÁRIO

LISTA DE ABREVIATURAS

AES	*Advanced Encryption Standard*
ANSI	*American National Standard Institute*
CA	*Certification Authority*
CN	*Common Name*
CRL	*Certificate Revocation List*
DES	*Data Encryption Standart*
DSS	*Digital Signature Standard*
FIPS	Padrão de Processamento de Informação Federal
http	*Hypper Text Transfer Protocol*
IBM	*International Business Machines*
IDEA	*Internacional Data Encryption Algorithm*
IP	*Internet Protocol*
IPSec	*Internet Protocol Security*
ITU	*Internation Telecomunication Union*
MD5	*Message Digest*
NBS	*National Bureou of Standards*
NIST	*National Institute for Standarts and Technology*
NSA	*National Security Agency*
OU	*Unidade Organizacional*
PKI	*Public Key Infrastructure*
PGP	*Pretty Good Privacy*
RSA	*Rivest, Shamir, Adleman*
SET	*Secure Electronic Transaction*
SMIME	*Secure Multipurpose Internet Mail Extension*
SMTP	*Simple Mail Transport Protocol*
SSL	*SSL - Secure Socket Layer*
TCC	Trabalho de Conclusão de Curso
TELNET	Protocolo de Terminal Virtual
TLS	*Transport Layer Security*
VPN	*Virtual Private Network*
USF	Universidade São Francisco

LISTA DE FIGURAS

RESUMO

SOUZA, Rodrigo Silvio. **Criptografia e Segurança Computacional.** Itatiba, 2004. 31 f. Trabalho de Conclusão de Curso, Universidade São Francisco, Itatiba, 2004.

Por incrível que pareça, desde que existe a criptografia também existiram os harckers de plantão. Durante a Idade Média, época de extrema paradeira e de uma desvairada perseguição aos "demônios" da criptografia na Europa, a civilização árabe-islamica deu uma enorme contribuição: foi o berço da criptoanálise. De 700 a 1200, incríveis estudos estatísticos e figuras de destaque como Al Khalil, Ibm Dunainir e Ibn Adlan marcaram época. Já que a criptoanálise se ocupa de quebrar sistemas, é obvio que um dos temas relevantes seja a segurança. É fácil intuir que, quanto mais seguro for um sistema, mais difícil será quebrá-lo. Porém como medir o nível de segurança? Existem sistemas a prova de hackers[1]? Como diria o "guru de segurança" da atualidade, Buce Schneier: "*O seguro morreu de bits*". [SCHNPA]

Este trabalho tem como objetivo apresentar a historia da criptografia. Como foi o surgimento dos padrões criptográficos até os dias atuais. Quais os tipos de chaves criptográficas, vantagens e desvantagens de cada modelo de chave de criptografia. Também serão abordados quais os protocolos utilizados pelas chaves criptográficas, a importância da criptografia para os métodos utilizados hoje para comunicação digital, envolvendo a assinatura digital e certificados digitais para garantir segurança nas transações. Também serão abordados os certificados digitais e a importância da função hashing.

[1] É aquela pessoa que possui uma grande facilidade de análise, assimilação, compreensão e capacidades surpreendentes de conseguir fazer o que quiser (literalmente) com um computador. Ela sabe perfeitamente que nenhum sistema é completamente livre de falhas, e sabe onde procurar por elas, utilizando-se de técnicas das mais variadas (aliás, quanto mais variado, mais valioso é o conhecimento do hacker).

ABSTRACT

SOUZA, Rodrigo Silvio. **Criptografia e Segurança Computacional.** Itatiba, 2004. 31 f. *Course Conclusion Paper*, Universidade São Francisco, Itatiba, 2004.

Amazingly, as old as the cryptography itself, the hackers are. During the middle age, considerate as an epoch with no human progression, the Arabian civilization gave its huge contribution; it was responsible for the cryptanalysis creation. From 700 to 1200 dc, outstanding statistical breakthroughs and brilliant people like Al Kalir and Ibn Adlan left their legacy. As long cryptography concentrates on breaking systems, it is obvious that security comes up as the most relevant matter. However, how can security level be measured? Would hacker-proof system exist? As the "security guru", Buce Schneier use to say: there is no safe bit-constituted system.

The aim of this paper is to present the cryptography history and the development of cryptographic standards up to nowadays. This paper will cover the cryptographic keys, advantages and drawbacks of each cryptographic key and the protocols that those keys make use. Moreover, the importance of implementing cryptographic on nowadays digital communication like digital signing and certificates, in order to assure secure transactions, will be spotted as well. To end with, the hashing function, used for digital signing and the most used digital certificates standards will be discussed.

***Keywords**: cryptography, cryptographic, keys, hashing, certificates, computer security*

1. INTRODUÇÃO

A necessidade da troca de mensagens sigilosas e a possibilidade de ler informações inimigas que podem determinar o vencedor em uma guerra, impulsionaram a evolução dos métodos criptográficos. Hoje em dia, com o advento da Internet, cresceu o interesse pela Criptografia que tem o objetivo de preservar o sigilo da correspondência em um ambiente seguro.

Existe uma busca constante pela mobilidade, bem como uma continua migração de varias tecnologias analógicas para digital, trazendo inúmeros benefícios. No entanto, surgem novos inconvenientes em busca dessa nova facilidade, como por exemplo, risco de fraudes até pouco tempo inexistentes. Neste sentido os mecanismos que provêem segurança aos sistemas de computação e seus dados são uma das questões principais para o sucesso das novas formas eletrônicas de interação entre as pessoas.

1.1. OBJETIVOS

1.1.1. Objetivo Geral

Estudar a origem da Criptografia, a necessidade de se utilizar a criptografia, certificados digitais, assinaturas digitais e algoritmos criptográficos.

1.1.2. Objetivos Específicos

Esse trabalho será sobre o assunto criptografia e a segurança computacional. Será descrito a historia da criptografia até os tempos atuais, os principais algoritmos assimétricos e simétricos, hashing criptográfico, protocolos e principais algoritmos para assinaturas digitais. Será descrito também o uso dos protocolos para autenticação, a negociação de chaves criptográficas e a segurança das aplicações.

?? Historia da Criptografia

?? Conceitos sobre algoritmos simétrico e assimétrico

?? Principais algoritmos criptográficos assimétricos e simétricos

?? Assinaturas digitais

?? Protocolos criptográficos

?? Certificados digitais

?? Segurança das Aplicações

1.2. METODOLOGIA

Pesquisas através de livros e artigos científicos disponíveis na Internet.

1.3. ESTRUTURA DO TRABALHO

No capitulo dois, será apresentada uma pequena fundamentação teórica, onde será discutida a recente evolução da criptografia e os modelos criptográficos mais usados.

O capitulo três descreve a história recente da criptografia computacional e a origem da palavra. Também é mostrada a necessidade do surgimento de um padrão para os algoritmos de criptografia, os requisitos que este padrão deveria obedecer e qual o órgão regulador desse padrão. Veremos também neste capitulo a escolha do padrão cifrador LUCIFER e como esse padrão foi adotado mundialmente.

Ainda no capitulo três, também descrevemos os modelos criptográficos simétricos e assimétricos, as vantagens e desvantagens de cada um, exemplos de algoritmos de criptografia simétrica e assimétrica e alguns protocolos criptográficos. Também será descrito o funcionamento das assinaturas digitais, os algoritmos utilizados para prove-las e a função *hashing* para assinatura digital.

Por fim, o capitulo três apresenta uma descrição do que é um certificado digital, quais os padrões mais adotados, servidores e distribuição de certificados digitais.

O capitulo quatro contem as considerações finais.

O Anexo I apresenta um artigo sobre o "herói da criptografia moderna", Phil Zimmermann, criador do difundido programa de criptografia PGP e por quê ele é necessário.

O Anexo II apresenta uma demonstração do algoritmo criptográfico RSA.

O Anexo III apresenta o padrão criptográfico AES.

O Anexo VI mostra o perfil de Bruce Schneier, considerado por muitos, o "guru" da segurança computacional moderna.

Por fim, no Anexo V, podemos ver um exemplo de um algoritmo escrito em liguagem C apresentando um exemplo de implementação do algoritmo criptográfico DES.

2. FUNDAMENTAÇÃO TEÓRICA

2.1. A EVOLUÇÃO DA CRIPTOGRAFIA

A criptografia para uso computacional teve seu desenvolvimento mais acentuadoo principalmente a partir da década de 70. No ano de 1976, o algoritmo LUCIFER foi adotado como padrão federal nos EUA, e publicado em 1977, com o nome de "*Data Encryption Standard*". Desde então tornou-se obrigatório o seu uso pela administração federal americana e somente em 1997, o NIST, orgão responsável pelos padrões de tecnologia, anunciou um concurso para o substituto do DES. O concurso foi chamado de AES, que teve como vencedor o algoritmo chamado de Rijndael (ver Anexo III).

2.2. CRIPTOGRAFIA SIMÉTRICA

A criptografia simétrica ou de chave secreta, foi o primeiro tipo de criptografia criado. Funciona transformando um texto em uma mensagem cifrada, através da definição de uma chave secreta, que será utilizada posteriormente para descriptografar a mensagem, tornando novamente um texto simples. Veremos os conceitos sobre os seguintes algoritmos de criptografia simétrica: DES, DESX, IDEA, 3DES. A figura 1 demonstra de forma simples a chave simétrica em funcionamento. A soma da mensagem mais chave gera uma mensagem criptografada, após a geração, ela é enviada através da rede, e ao chegar ao lado oposto, ela é descriptografada através da chave que está no destino (a mesma chave).

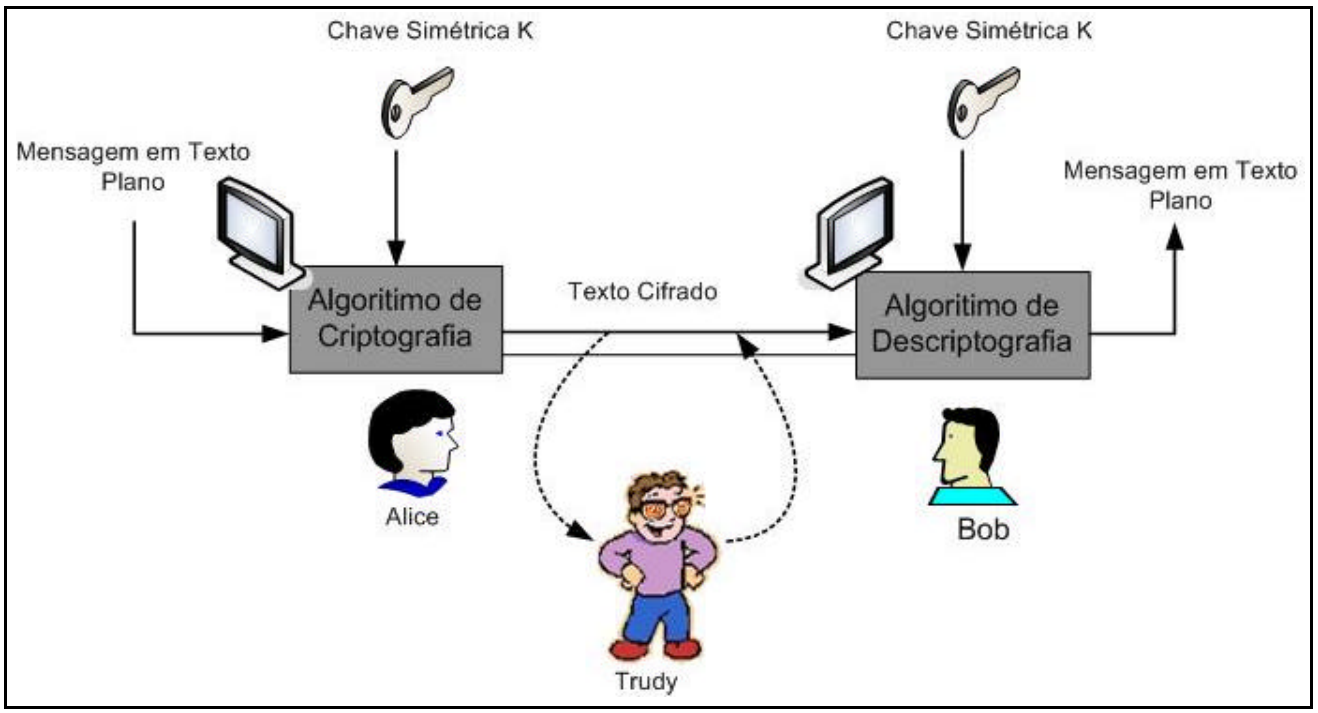

Figura 1. Sistema de criptografia simétrica ou de chave secreta
Fonte: Adaptado de Kurose (2003), página 610.

2.3. CRIPTOGRAFIA ASSIMÉTRICA

A criptografia assimétrica, também conhecida como de chave pública, utiliza duas chaves, uma para cifrar o texto ou mensagem, e outra para decifrar. Pode ser empregada para assinatura digital e autenticação.É possível combinar a criptografia simétrica com a assimétrica, somando segurança com a rapidez. Também veremos os seguintes algoritmos de criptografia assimétrica: Diffe-Helman, Elgamal, DSS e RSA.

Na figura 2, podemos ver o princípio da criptografia utilizando chave assimétrica, uma chave publica e uma chave privada.

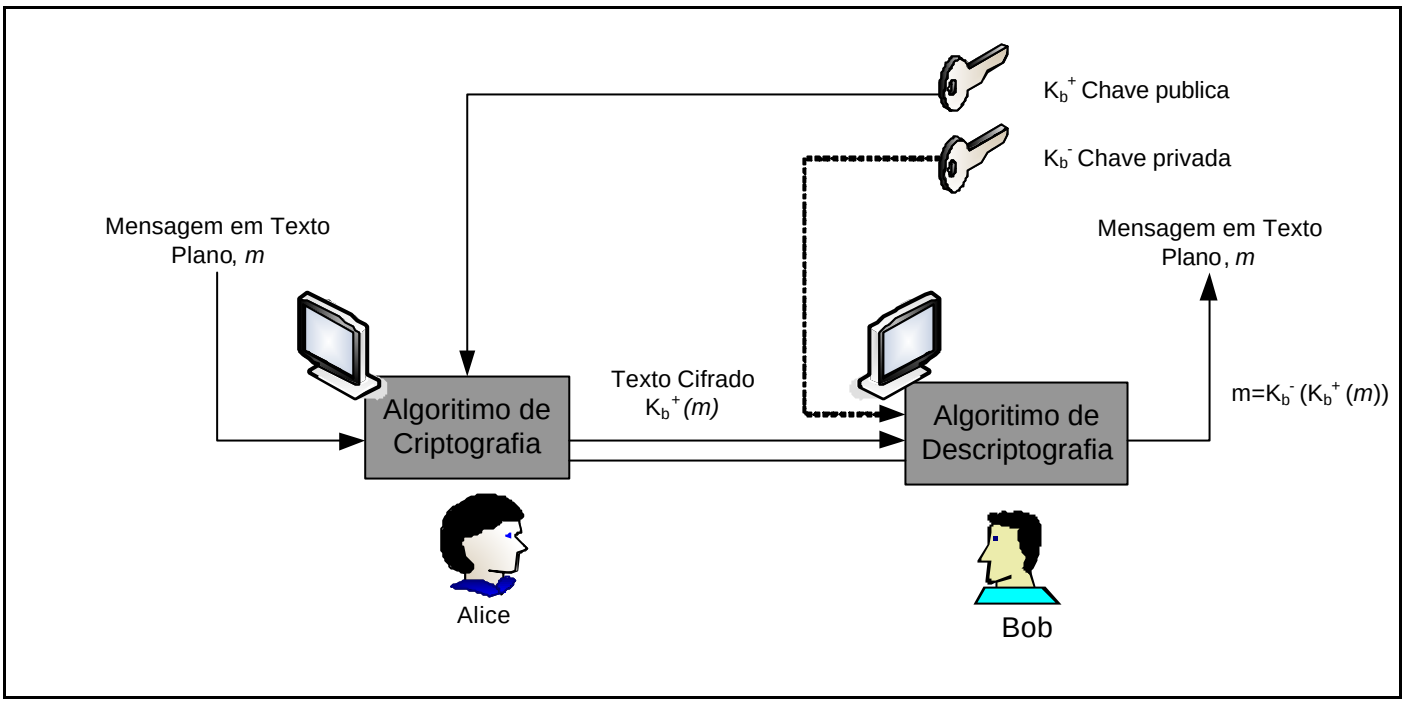

Figura 2. Sistema de criptografia assimétrica ou de chave pública
Fonte: Adaptado de Kurose (2003) pagina 615

Na figura 3 podemos ver hierarquia dos algoritmos simétricos

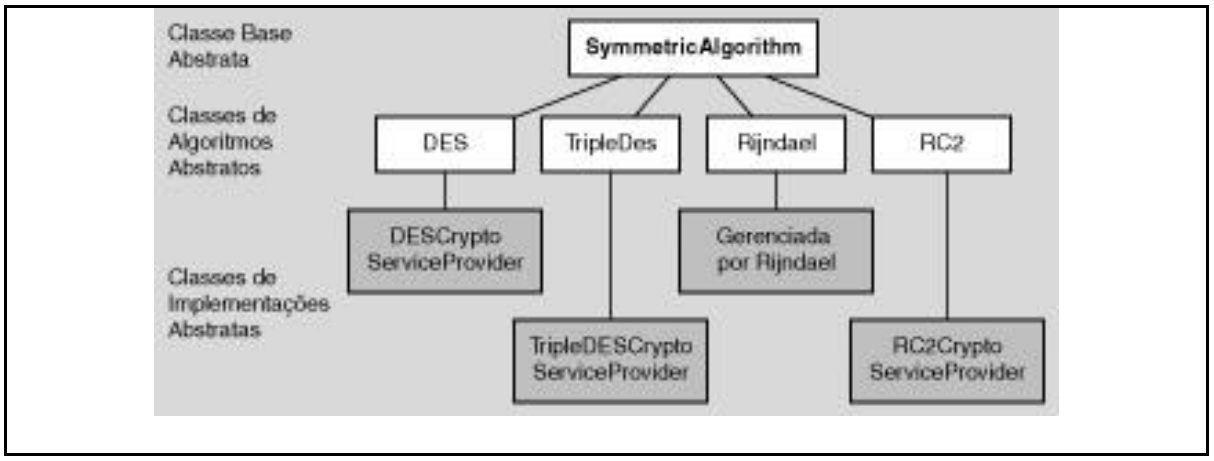

Figura 3. Hierarquia de algoritmos criptografia Simétrica
Fonte: Microsoft® , Centro de orientação de segurança (2004).

2.4. ASSINATURAS DIGITAIS

Um benefício da criptografia com chave pública são as assinaturas digitais. Elas permitem garantir a autenticidade de quem envia a mensagem, associada à integridade do seu conteúdo. O

mecanismo fundamental para o adequado emprego da assinatura digital é chamado de função *Hashing*. Os principais algoritmos usados nesta função são MD5, SHA-1, MD2 e MD4.

Na figura 4 podemos ver hierarquia dos algoritmos de *hashing*.

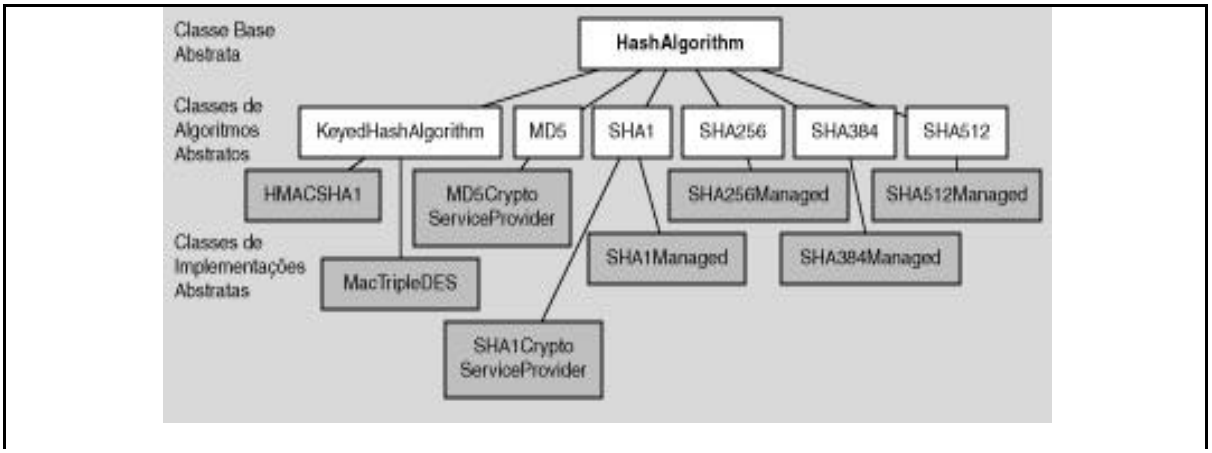

Figura 4. Hierarquia de algoritmos *Hashing*
Fonte: Microsoft® , Centro de orientação de segurança (2004).

2.5. CERTIFICADO DIGITAL

Um certificado digital é uma informação incluída com a chave pública de uma pessoa que ajuda outros verificarem que uma chave é genuína ou válida. Certificados digitais são usados para contrariar tentativas de substituição da chave de uma pessoa por outra. Hoje os padrões de certificados mais usados são o PGP e o X.509.

3. PROJETO

O projeto pretende apresentar uma visão sobre a criptografia e segurança computacional.

3.1. CRIPTOGRAFIA

Criptografia é o ato de codificar dados em informações aparentemente sem sentido, para que pessoas não consigam ter acesso às informações que foram cifradas. Há vários usos para a criptografia em nosso dia-a-dia: proteger documentos secretos, transmitir informações confidenciais pela Internet ou por uma rede local, etc.

O método de criptografia mais difundido utiliza a técnica de chave pública / chave privada. A fórmula matemática gera duas chaves, uma pública e outra privada (ou secreta). As chaves públicas, que qualquer pessoa pode saber, são usadas para criptografar os dados. Já a chave privada, que só o destinatário dos dados conhece, é usada para descriptografar os dados, ou seja, "abrir" os dados que ficaram aparentemente sem sentido. O interessante dessa técnica é que a partir da chave pública é impossível descriptografar os dados nem tampouco deduzir qual é a chave privada.

O sistema de criptografia usado atualmente é bastante seguro. Especialistas estimam que para alguém conseguir quebrar uma criptografia usando chaves de 64 bits na base da tentativa e erro, levaria cerca de 100.000 anos usando um PC comum. Em setembro de 2002, um site chamado Distributed.net[2] conseguiu vencer um concurso promovido pela RSA Security [3], que existe desde a data da fundação desta empresa, pagando US$ 10.000 para o primeiro que conseguisse quebrar sua criptografia de 64 bits. Só um detalhe: o Distributed.net só conseguiu quebrar essa senha porque ele pedia para as pessoas que quisessem colaborar com esse desafio rodassem em seu micro parte do processo de tentativa-e-erro, baixando um pequeno programa existente no site deles. No total, foram 300.000 pessoas colaborando com esse projeto ao longo de cinco anos. Levando-se em conta que a criptografia de 128 bits já é uma realidade e os especialistas estão cada vez mais empenhados em criar sistemas de criptografia ainda mais seguros, podemos afirmar com certeza que a criptografia usada no PC hoje é muito segura (em outras palavras, mesmo que um hacker intercepte o número de um cartão de crédito pela Internet em uma transação segura, ele estará criptografado e, a não ser que

[2] http://www.distributed.net, visitado em 20/11/2004
[3] http://www.rsasecurity.com, visitado em 20/11/2004

o hacker arrume 300.000 computadores e 5 anos da vida dele sobrando, ele não terá acesso às infomações).

> O sucesso é uma questão de não desistir. Fracasso é uma questão de não desistir cedo demais. Todo trabalho tem dignidade. Qualquer tarefa, grande ou pequena, é importante o bastante para ser realizada, completada, apreciada. Orgulhe-se do seu trabalho e de você mesmo, por realizar o que realiza.(Autor: Desconhecido)

3.1.1. História da Criptografia

A Criptografia, palavra oriunda do grego: "kryptós" (oculto) e "grápheim" (escrever), pode ser conceituada como a ciência que estuda os princípios e técnicas que visam proporcionar às informações ou dados, armazenados ou em trânsito, os serviços de segurança da Confidencialidade, Integridade e Autenticidade.A meta da Criptografia é alcançada pela aplicação de sistemas, chamados criptográficos, projetados e construídos de maneira adequada.

A Criptoanálise, do grego: "kryptós" (oculto) e "analyein" (desfazer), é a ciência (e /ou arte) que estuda os princípios, processos e métodos para desvendar os segredos dos sistemas criptográficos existentes (no jargão técnico da área, "quebrar" os sistemas), objetivando ganhar acesso ou mesmo capacidade de alterar as informações ou dados pretensamente protegidos.

Juntas, a Criptografia e a Criptoanálise, compõem a Criptologia ("kryptós" + "lógos" – palavra).Apesar de antagônicas em seus desígnios, os dois campos da criptologia, com que obedecendo ao princípio físico de atração de pólos opostos, têm se apoiado mutuamente em seus desenvolvimentos e progressos.

De fato, no desenho de um sistema criptográfico, os especialistas buscam provê-lo de um especificado nível requerido de robustez, tendo em vista os possíveis tipos de ataques (tentativas de "quebra") a que poderão ser submetidos. Por outro lado, a evolução tecnológica dos processos de criptoanálise realimenta a criptografia, atuando como verdadeiro controle gerador da melhoria de qualidade.

Inicialmente, nos seus primórdios, que remontam às origens da escrita, a criptografia era, quase que exclusivamente, voltada para os setores militar e diplomático. Nos tempos atuais, graças à inexorável marcha de difusão de conhecimentos e à crescente necessidade de segurança de dados, os benefícios da sua utilização vêm sendo ampliados à toda sociedade.

A busca de mecanismos adequados, seguros e os mais "amigáveis" possíveis, para proteção das informações mais sensíveis, cresce na proporção da crescente disponibilidade de sistemas de telecomunicações cada vez mais sofisticados e abrangentes, ressaltando, na sua esteira, a importância dos sistemas criptográficos.

Segundo o autor Bruce Schneier (SCHNPA), "...a Criptografia ajuda a imputar responsabilidade, promover a justiça, prover certeza e privacidade. Pode prevenir fraudes em comércio eletrônico e garantir a validade de transações financeiras. Se usada apropriadamente, protege a anonimato e fornece provas de identidade de pessoas. Pode, ainda, impedir vândalos de alterarem sua página na Internet e competidores industriais de lerem seus documentos confidenciais. Com o comércio seguindo sua marcha pelas redes de computadores, a Criptografia se tornará cada vez mais vital".

Porém, no mesmo artigo, destaca Schneier que é enganoso acreditar-se inocentemente no "marketing" apregoado pelos vendedores ou mesmo projetistas de sistemas criptográficos. Ressalta ele a dificuldade de se obter um sistema com um alto grau de segurança, e que, na verdade um bom sistema criptográfico atinge o equilíbrio entre o que é possível e o que é aceitável.

3.1.2. A necessidade de um padrão

Coube ao Departamento de Comércio Norte Americano, mais especificamente, ao antigo NBS, hoje NIST, a tarefa de iniciar o desenvolvimento do padrão de criptografia que se buscava.

Logo de início o NBS levantou o problema da aquisição de dispositivos criptográficos de fornecedores comerciais. O NBS era contra este tipo de aquisição, pois desconfiava da existência da real capacidade, entre aqueles fornecedores, em desenvolver equipamentos com um adequado grau de segurança. Um outro aspecto interessante, de cunho comercial, foi levantado. Como a maioria dos equipamentos até então em uso eram de fabricantes estrangeiros, a adoção de um padrão não afetaria a indústria americana. Por outro lado, se o padrão fosse aceito mundialmente os fabricantes de equipamentos até então existentes teriam de tomar uma decisão: ou mudariam para o padrão, satisfazendo os interesses americanos, ou então se arriscariam na permanência no mercado com seus antigos produtos.

Em 1972, efetivamente o NBS iniciou a busca de um algoritmo criptográfico adequado, que pudesse se constituir na base de um Padrão de Processamento de Informação Federal ("FIPS"). Segundo o NBS, o algoritmo a ser adotado como padrão deveria obedecer aos seguintes requisitos:

1. Apresentar um alto nível de segurança.

2. Estar completamente especificado e ser de fácil entendimento.

3. A segurança fornecida pelo algoritmo não deveria se basear no segredo sobre o mesmo.

4. Deveria estar disponível para todos os usuários e fornecedores.

5. Deveria ser facilmente adaptável para ser usado em diversas aplicações.

6. Deveria ser implementado de maneira econômica em dispositivos eletrônicos, bem como ser de utilização eficiente.

7. Deveria ser de fácil validação (legalização).

8. Deveria possuir atributos que o possibilitasse ser exportado.

Em agosto de 1974. vários candidatos se apresentaram, uns muito especializados, outros não tão seguros. Apenas um deles se destacou, na visão do NBS, como forte candidato: o algoritmo desenvolvido nos laboratórios da IBM ("International Business Machines"), desde o início da década de 70, que era o cifrador conhecido como "Lucifer" (FEIS73).

3.1.3. Estabelecimento do Padrão DES

Durante alguns meses após a sua apresentação como candidato ao padrão, o algoritmo "Lucifer" foi submetido a uma rigorosa e exaustiva análise pelo NBS, juntamente com a NSA, visando obviamente levantar suas boas características, bem com suas possíveis fraquezas, com o objetivo, pelo menos oficialmente, de adequá-lo às exigências e torná-lo mais seguro.

Tal trabalho deve ter sido facilitado pelo fato de o "Lucifer", naquela ocasião, já ser conhecido publicamente, e mesmo já ter sido alvo de vários trabalhos de análise. Na ocasião, o NBS e a IBM entraram em acordo a respeito da propriedade intelectual e permissão de fabricação, implementação e venda do algoritmo nos Estados Unidos, para outras entidades interessadas.

Em 23 de Novembro de 1976, o algoritmo foi adotado como padrão federal, e publicado, em 15 de janeiro de 1977, com o nome de "*Data Encryption Standard*" (DES), na FIPS PUB 46 (FIP461).

Em 15 de julho de 1977, tornou-se obrigatório o seu uso pela administração federal americana, para proteção, como já mencionado anteriormente, de informações não-classificadas. Mais tarde seu uso se espalhou por outras organizações, como bancos e empresas particulares, tanto no território americano, como em âmbito mundial.

Assim, o DES foi adotado como algoritmo padrão para criptografia americana, e assim continuaria até 1998, quando, segundo as previsões, a criação de máquinas capazes de quebrar o algoritmo seria algo viável.

Dessa forma, em janeiro de 1997, o NIST (National Institute of Standards and Technology) anunciou um concurso para o substituto do DES. Os candidatos, que deveriam ser algoritmos de chave simétrica, capazes de suportar blocos de 128 bits e chaves de 128, 192 e 256 bits, deveriam ter direitos autorais livres, pois seriam divulgados publicamente. Esse concurso foi chamado de AES (Advanced Encryption Standard), que acabou dando nome também ao algoritmo vencedor (antes chamado de Rijndael), no ano de 2000.

As principais especificações colocadas aos candidatos a AES foram:

1. Algoritmo criptográfico simétrico cifrador de bloco;
2. Será usado por organizações governamentais ou comerciais (de modo voluntário), para proteção de informações "não-classificadas", justamente como ocorrera com o DES;
3. O algoritmo deve ser completamente aberto ao conhecimento público e possuir facilidades para execução de testes, para permitir, de modo irrestrito, análises e avaliações;
4. Deve ter a possibilidade de se tornar disponível mundialmente e livre de "*royalties*";
5. Deve ser mais eficiente e mais seguro que o 3-DES, possibilitando tamanhos de chave de 128, 192 e 256 bits;

6. Deve também poder cifrar.

3.1.4. Criptografia Simétrica

O ciframento de uma mensagem baseia-se em dois componentes: um algoritmo e uma chave. Um algoritmo é uma transformação matemática. Ele converte uma mensagem em claro em uma mensagem cifrada e vice-versa. Quando *Alice* (origem) cifra uma mensagem, ela utiliza um algoritmo de ciframento para transformar o conteúdo em claro da mensagem em texto cifrado. Quando *Bob* (destinatário) decifra uma mensagem, ele utiliza o algoritmo de deciframento correspondente para converter o texto cifrado de novo em uma mensagem clara (FIG. 01).

Antigamente, a segurança do ciframento estava baseada somente no sigilo do algoritmo criptográfico. Se *Trudy* (um intruso) conhecesse o algoritmo sem chave, poderia decifrar uma mensagem cifrada tão facilmente quanto *Bob*. Pode-se contornar o problema apresentado utilizando o segundo componente básico da criptografia de mensagens: a chave. 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.

Embora existam algoritmos que dispensem o uso de chaves, sua utilização oferece duas importantes vantagens. A primeira é permitir a utilização do mesmo algoritmo criptográfico para a comunicação com diferentes receptores, apenas trocando a chave. A segunda vantagem é permitir trocar facilmente a chave no caso de uma violação, mantendo o mesmo algoritmo.

O número de chaves possíveis depende do tamanho (número de bits) da chave. Por exemplo, uma chave de oito bits permite uma combinação de no máximo 256 chaves (2^8). Quanto maior o tamanho da chave, mais difícil quebrá-la, pois estamos aumentando o número de combinações.

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 precisaríamos de algo da ordem de n^2 chaves, quantidade esta que dificulta 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).

3.1.5. Criptografia Assimétrica

A maneira de contornar os problemas da criptografia simétrica é a utilização da criptografia assimétrica ou de chave pública. A criptografia assimétrica está baseada no conceito de par de chaves: uma chave privada e uma chave pública. Qualquer uma das chaves é utilizada para cifrar uma mensagem e a outra para decifrá-la. As mensagens cifradas com uma das chaves do par só podem ser decifradas com a outra chave correspondente. A chave privada deve ser mantida secreta, enquanto a chave pública disponível livremente para qualquer interessado.

De uma forma simplificada, o sistema funciona assim: *Bob* e todos os que desejam comunicar-se de modo seguro geram uma chave de ciframento e sua correspondente chave de deciframento. Ele mantém secreta a chave de deciframento; esta é chamada de sua chave privada. Ele torna pública a chave de ciframento: esta é chamada de sua chave pública.

A chave pública realmente condiz com seu nome. Qualquer pessoa pode obter uma cópia dela. Bob inclusive encoraja isto, enviando-a para seus amigos ou publicando-a em boletins. Assim, *Trudy* não tem nenhuma dificuldade em obtê-la. Quando *Alice* deseja enviar uma mensagem a *Bob*, precisa primeiro encontrar a chave pública dele. Feito isto, ela cifra sua mensagem utilizando a chave pública de *Bob*, despachando-a em seguida. Quando *Bob* recebe a mensagem, ele a decifra facilmente com sua chave privada. *Trudy*, que interceptou a mensagem em trânsito, não conhece a chave privada de *Bob*, embora conheça sua chave pública. Mas este conhecimento não o ajuda a decifrar a mensagem. Mesmo *Alice*, que foi quem cifrou a mensagem com a chave pública de *Bob*, não pode decifrá-la agora. (FIG.02)

A grande vantagem deste sistema é permitir que qualquer um possa enviar uma mensagem secreta, apenas utilizando 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 à chave privada terá acesso às mensagens.

3.1.6. Exemplos de algoritmos criptografia simétrica

1. IDEA (*International Data Encryption Algorithm*): O método se baseia na utilização de uma chave de 128 bits, onde blocos de texto da mensagem de entrada são alterados em uma seqüência de interações, produzindo blocos de saída. É à base do programa PGP (*Pretty Good Privacy*) usado em criptografia de correio eletrônico.
2. DES (*Data Encryption Standard*) : é um algoritmo de bloco simétrico desenvolvido pela IBM. Atualmente é o algoritmo mais divulgado e utilizado em sistemas criptográficos no mercado mundial. Ele possui uma chave de 56 bits e seu algoritmo tem 19 estágios.
3. DESX: é uma modificação simples do algoritmo DES em que se estabelece uma dupla criptografia.
4. *Triple*-DES: é uma outra modificação em que se aplica três vezes o algoritmo DES com três chaves diferentes. Vem sendo usado atualmente por instituições financeiras. No caso do DES, várias tentativas de quebra (criptoanálise) já foram feitas e publicadas. O DES pode ser quebrado pelo método da "força bruta" tentando-se todas as combinações possíveis para a chave. Como a chave é de 56 bits tem-se um total de 2^56 chaves possíveis, ou aproximadamente 10^17 chaves possíveis. Existe uma técnica de melhorar a segurança do algoritmo DES utilizando uma criptografia tripla que é conhecida como DES triplo. Nesta técnica cada mensagem passa por três processos criptográficos que irão reduzir a possibilidade da segurança ser quebrada. Isto equivale a, no mínimo, dobrar o tamanho da chave DES para 112 bits. (FIG. 03)
5. *Advanced Encryption Standard (AES)* Uma iniciativa do governo federal para selecionar um algoritmo de criptografia padrão capaz de proteger material confidencial do governo. O algoritmo substituiu o DES como padrão para criptografia de cifras simétricas em blocos e incluiu uma chave de 128, 192 ou 256. O AES foi desenvolvido para criptografar dados com maior rapidez e eficiência do que o DES ou o DES Triplo, especialmente em implementações de software.

3.1.7. Exemplos de algoritmos de criptografia assimétrica

1. *Diffie-Hellman*: Também baseado no problema do logaritmo discreto, e o criptosistema de chave pública mais antigo ainda em uso. O conceito de chave pública, aliás, foi introduzido pelos autores deste criptosistema em 1976. Contudo, ele não permite nem ciframento nem assinatura digital. O sistema foi projetado para

permitir a dois indivíduos entrarem em um acordo ao compartilharem um segredo tal como uma chave, muito embora eles somente troquem mensagens em público.

2. ElGamal: É outro algoritmo de chave pública utilizado para gerenciamento de chaves. Sua matemática difere da utilizada no RSA, mas também é um sistema comutativo. O algoritmo envolve a manipulação matemática de grandes quantidades numéricas. Sua segurança advém de algo denominado problema do logaritmo discreto. Assim, o ElGamal obtém sua segurança da dificuldade de se calcular logaritmos discretos em um corpo finito, o que lembra bastante o problema da fatoração.

3. DSS (*Digital Signature Standard*): é usado para realização de assinatura digital, mas pode ser usado para criptografia. Atualmente usa chaves entre 512 a 1024 bits.

4. RSA (*Rivest, Shamir, Adleman*): é o mais popular algoritmo de chave pública. Usa duas chaves criptográficas, uma chave pública e uma privada. A segurança desse algoritmo está baseada na dificuldade de fatorar grandes números: as chaves são calculadas matematicamente combinando dois números primos de grande tamanho. Mesmo se conhecendo o produto desses números primos (que faz parte da chave pública divulgada), a segurança do algoritmo é garantida pela complexidade de fatorar esse produto e se obter os valores secretos.

3.1.8. Assinaturas digitais

Outro benefício da criptografia com chave pública é a assinatura digital. Ela permite garantir a autenticidade de quem envia a mensagem, associada à integridade do seu conteúdo. Por exemplo, suponha que *Alice* (origem) queira comunicar o nascimento de sua filha para todos os seus amigos (destinatários = *Bob*), mas queira garantir aos mesmos que a mensagem foi enviada realmente por ela. Embora não se importe com o sigilo da mensagem, deseja que a mesma chegue integra aos destinatários, sem alterações como, por exemplo, do sexo da criança.

Alice então cifra a mensagem com sua chave privada e a envia, em um processo denominado de assinatura digital. Cada um que receber a mensagem deverá decifrá-la, ou seja, verificar a validade da assinatura digital, utilizando para isso a chave pública de *Alice*. Como a chave pública de *Alice* apenas decifra (ou seja, verifica a validade de) mensagens cifradas com sua chave privada,

fica garantida assim a autenticidade, integridade e não-repudiação da mensagem. Pois se alguém modificar um bit do conteúdo da mensagem ou se outra pessoa assiná-la ao invés de *Alice*, o sistema de verificação não irá reconhecer a assinatura digital de *Alice* como sendo válida. É importante perceber que a assinatura digital, como descrita no exemplo anterior, não garante a confidencialidade da mensagem. Qualquer um poderá acessá-la e verificá-la, mesmo um intruso (*Trudy*), apenas utilizando a chave pública de *Alice*. Para obter confidencialidade com assinatura digital basta combinar os dois métodos. Alice primeiro assina a mensagem, utilizando sua chave privada. Em seguida, ela criptografia a mensagem novamente, junto com sua assinatura, utilizando a chave pública de *Bob*. Este, ao receber a mensagem, deve, primeiramente, decifrá-la com sua chave privada, o que garante sua privacidade. Em seguida, "decifrá-la" novamente, ou seja, verificar sua assinatura utilizando a chave pública de Alice, garantindo assim sua autenticidade.

3.1.9. Algoritmos utilizados para assinatura digital

RSA: O RSA também é comutativo e pode ser utilizado para a geração de assinatura digital. A matemática é a mesma: há uma chave pública e uma chave privada, e a segurança do sistema baseia-se na dificuldade da fatoração de números grandes.

ElGamal: Como o RSA, o ElGamal também é comutativo, podendo ser utilizado tanto para assinatura digital quanto para gerenciamento de chaves; assim, ele obtém sua segurança da dificuldade do cálculo de logaritmos discretos em um corpo finito.

DAS: O *Digital Signature Algorithm*, unicamente destinado a assinaturas digitais, foi proposto pelo NIST em agosto de 1991, para utilização no seu padrão DSS (Digital *Signature Standard*). Adotado como padrão final em dezembro de 1994, trata-se de uma variação dos algoritmos de assinatura ElGamal e Schnorr. Foi inventado pela NSA e patenteado pelo governo americano.

3.1.10. Função de *Hashing*

A assinatura digital obtida através do uso da criptografia assimétrica ou de chave pública infelizmente não pode ser empregada, na prática, de forma isolada. Está faltando, portanto, descrever um mecanismo fundamental para o adequado emprego da assinatura digital. Este mecanismo é a função *Hashing*. Sua utilização como componente de assinaturas digitais se faz

necessário devido à lentidão dos algoritmos assimétricos, em geral cerca de 1.000 vezes mais lentos do que os simétricos.

Assim, na prática é inviável utilizar puramente algoritmos de chave pública para assinaturas digitais, principalmente quando se deseja assinar grandes mensagens, que podem levar preciosos minutos ou mesmo horas para serem integralmente "cifradas" com a chave privada de alguém. Ao invés disso, é empregada uma função *Hashing*, que gera um valor pequeno, de tamanho fixo, derivado da mensagem que se pretende assinar, de qualquer tamanho. Assim, a função *Hashing* oferece agilidade nas assinaturas digitais, além de integridade confiável, conforme descrito a seguir.

Também denominada *Message Digest*, *One-Way Hash Function*, "Função de Condensação" ou "Função de Espalhamento Unidirecional", a função *Hashing* funciona como uma impressão digital de uma mensagem gerando, a partir de uma entrada de tamanho variável, um valor fixo pequeno: o *digest* ou valor *hash*.

Este valor está para o conteúdo da mensagem assim como o dígito verificador de uma conta-corrente está para o número da conta ou o check sum está para os valores que valida. Serve, portanto, para garantir a integridade do conteúdo da mensagem que representa. Assim, após o valor *hash* de uma mensagem ter sido calculado através do emprego de uma função *hashing*, qualquer modificação em seu conteúdo – mesmo em apenas um bit da mensagem – será detectada, pois um novo cálculo do valor *hash* sobre o conteúdo modificado resultará em um valor *hash* bastante distinto.

3.1.10.1. Funcionamento da assinatura digital

Como funciona a assinatura digital (baseada na criptografia assimétrica) de um texto ou mensagem eletrônica? Aplica-se sobre o documento editado ou confeccionado um algoritmo de autenticação conhecido como *hash* . A aplicação do algoritmo *hash* gera um resumo do conteúdo do documento conhecido como *message digest*, com tamanho em torno de 128 bits. Aplica-se, então, ao *message digest*, as chaves privadas do usuário, obtendo-se um message digest criptografado ou codificado. O passo seguinte consiste um anexar ao documento em questão a chave pública do autor, presente no arquivo chamado certificado digital. Podemos dizer que assinatura digital de um documento eletrônico consiste nestes três passos:

a) geração do *message digest* pelo algoritmo *hash*;

b) aplicação da chave privada ao *message digest*, obtendo-se um *message digest* criptografado e;

c) anexação do certificado digital do autor (contendo sua chave pública).

Destacamos, neste passo, um aspecto crucial. As assinaturas digitais, de um mesmo usuário, utilizando a mesma chave privada, serão diferentes de documento para documento. Isto ocorre porque o código *hash* gerado varia em função do conteúdo de cada documento.

Como o destinatário do texto ou mensagem assinada digitalmente terá ciência da integridade (não alteração/violação) e autenticidade (autoria) do mesmo? Ao chegar ao seu destino, o documento ou mensagem será acompanhado, como vimos, do *message digest* criptografado e do certificado digital do autor (com a chave pública nele inserida). Se o aplicativo utilizado pelo destinatário suportar documentos assinados digitalmente ele adotará as seguintes providências:

a) aplicará o mesmo algoritmo *hash* no conteúdo recebido, obtendo um *message digest* do documento;

b) aplicará a chave pública (presente no certificado digital) no *message digest* recebido, obtendo o *message digest* decodificado e;

c) fará a comparação entre o message digest gerado e aquele recebido e decodificado.

A coincidência indica que a mensagem não foi alterada, portanto mantém-se íntegra. A discrepância indica a alteração / violação do documento depois de assinado digitalmente.

3.1.10.2. Funções Hashing empregadas em produtos e protocolos criptográficos

MD5: É uma função de espalhamento unidirecional inventada por Ron Rivest, do MIT, que também trabalha para a RSA Data Security. A sigla MD significa *Message Digest*. Este algoritmo produz um valor *hash* de 128 bits, para uma mensagem de entrada de tamanho arbitrário. Foi inicialmente proposto em 1991, após alguns ataques de criptoanálise terem sidos descobertos contra a função *Hashing* prévia de Rivest: a MD4. O algoritmo foi projetado para ser rápido, simples e seguro. Seus detalhes são públicos, e têm sido analisados pela comunidade de criptografia. Foi descoberta uma fraqueza em parte do MD5, mas até agora ela não afetou a segurança global do algoritmo. Entretanto, o fato dele produzir um valor *hash* de somente 128 bits é o que causa maior preocupação; é preferível uma função Hashing que produza um valor maior.

SHA-1: O *Secure Hash Algorithm*, uma função de espalhamento unidirecional inventada pela NSA, gera um valor hash de 160 bits, a partir de um tamanho arbitrário de mensagem. O funcionamento interno do SHA-1 é muito parecido com o observado no MD4, indicando que os estudiosos da NSA basearam-se no MD4 e fizeram melhorias em sua segurança. De fato, a fraqueza existente em parte do MD5, citada anteriormente, descoberta após o SHA-1 ter sido proposto, não ocorre no SHA-1. Atualmente, não há nenhum ataque de criptoanálise conhecido contra o SHA-1. Mesmo o ataque da força bruta torna-se impraticável, devido ao seu valor hash de 160 bits. Porém, não há provas de que, no futuro, alguém não possa descobrir como quebrar o SHA-1.

MD2 e MD4: O MD4 é o precursor do MD5, tendo sido inventado por Ron Rivest. Após terem sido descobertas algumas fraquezas no MD4, Rivest escreveu o MD5. O MD4 não é mais utilizado. O MD2 é uma função de espalhamento unidirecional simplificada, e produz um hash de 128 bits. A segurança do MD2 é dependente de uma permutação aleatória de bytes. Não é recomendável sua utilização, pois, em geral, é mais lento do que as outras funções hash citadas e acredita-se que seja menos seguro. (FIG. 04)

3.1.11. Criptografia Simétrica x Assimétrica: Protocolos Criptográficos

Qual o modelo de criptografia que devemos utilizar? Simétrico ou assimétrico? A resposta é simples: devemos utilizar os dois, em um modelo denominado híbrido. O algoritmo simétrico, por ser muito mais rápido, é utilizado no ciframento da mensagem em si. Enquanto o assimétrico, embora lento, permite implementar a distribuição de chaves e a assinatura digital. Além disso, deve-se utilizar também o mecanismo de *Hashing* para complemento da assinatura digital.

Em resumo, os algoritmos criptográficos podem ser combinados para a implementação dos três mecanismos criptográficos básicos: o ciframento, a assinatura e o *Hashing*. Estes mecanismos são componentes dos protocolos criptográficos, embutidos na arquitetura de segurança dos produtos destinados ao comércio eletrônico. Estes protocolos criptográficos, portanto, provêm os serviços associados à criptografia que viabilizam o comércio eletrônico: disponibilidade, sigilo, controle de acesso, autenticidade, integridade e não-repúdio. Seguem exemplos de protocolos que empregam sistemas criptográficos híbridos:

IPSec: Padrão de protocolos criptográficos desenvolvidos para o IPv6. Realiza também o tunelamento de IP sobre IP. É composto de três mecanismos criptográficos: *Authentication Header* (define a função *Hashing* para assinatura digital), *Encapsulation Security Payload* (define o

algoritmo simétrico para ciframento) e ISAKMP (define o algoritmo assimétrico para Gerência e troca de chaves de criptografia). Criptografia e tunelamento são independentes. Permite *Virtual Private Network* fim-a-fim.

SSL e TLS: Oferecem suporte de segurança criptográfica para os protocolos HTTP, SMTP e Telnet. Permitem utilizar diferentes algoritmos simétricos, *message digest (hashing)* e métodos de autenticação e gerência de chaves (assimétricos).

3.1.12. Padrões

S/MIME: O S/MIME (*Secure Multipurpose Internet Mail Extensions*) consiste em um esforço de um consórcio de empresas, liderado pela RSADSI e pela Microsoft, para adicionar segurança a mensagens eletrônicas no formato MIME. Apesar do S/MIME e PGP serem ambos padrões Internet, o S/MIME (RFC 3114), deverá se estabelecer no mercado corporativo, enquanto o PGP no mundo do e-mail pessoal.

SET: O SET é um conjunto de padrões e protocolos, para realizar transações financeira seguras, como as realizadas com cartão de crédito na Internet. Oferece um canal de comunicação seguro entre todos os envolvidos na transação. Garante autenticidade X.509v3 e privacidade entre as partes.

X.509: Recomendação ITU-T, a especificação X.509 define o relacionamento entre as autoridades de certificação. Faz parte das séries X.500 de recomendações para uma estrutura de diretórios globais, baseados em nomes distintos para localização. Utilizado pelo S/MIME, IPSec, SSL / TLS e SET. Baseado em criptografia com chave pública (RSA) e assinatura digital (com *hashing*).

3.1.13. Certificado Digital

Uma regra em sistemas de criptografia com chave pública é que os usuários devem ser constantemente vigilantes para assegurar que eles estão encriptando para a chave correta da pessoa. Em um ambiente onde é seguro a troca livre de chaves por servidores públicos, ataques de uma pessoa do meio (*"man in the middle")* é uma ameaça potencial. Neste tipo de ataque, alguém põe no correio uma falsa chave com o nome e ID do usuário do recipiente. Dados encriptados e interceptados pelo verdadeiro dono desta falsa chave está agora em mãos erradas.

Em um ambiente de chave pública, é vital que se esteja seguro que a chave pública para a qual se está encriptando dados é de fato a chave pública do recipiente no qual se queira mandar a mensagem e não uma falsificação. Suponhamos que se necessite trocar informação com pessoas que nunca nos encontramos; como poderíamos confiar que teríamos a chave correta?

Certificados digitais simplificam a tarefa de estabelecer se uma chave pública realmente pertence ao dono pretendido.

Um certificado é uma forma de credencial. Exemplos no mundo real poderiam uma licença de motorista, cartão de CPF, ou uma certidão de nascimento. Cada um destes tem um pouco de informação que podem nos identificar. Essas autorizações também declaram que outras pessoas confirmaram nossa identidade. Alguns certificados, como passaportes, são bastante importantes para a confirmação da identidade no mundo real. Ninguém gostaria de perder um documento destes, visto que outra pessoa poderia fazer-se passar pelo dono do documento.

Um certificado digital é uma informação incluída com a chave pública de uma pessoa que ajuda outros verificarem que uma chave é genuína ou válida. Certificados digitais são usados para contrariar tentativas de substituição da chave de uma pessoa por outra.

Um certificado digital consiste em três coisas:

- Uma chave pública.
- Certificado de informação. (Informação de "identidade" sobre o usuário, como nome, ID do usuário, e assim por diante.).
- Uma ou mais assinaturas digitais

O propósito da assinatura digital em um certificado é declarar que a informação do certificado foi atestada por alguma outra pessoa ou entidade. A assinatura digital não atesta à autenticidade do certificado como um todo; só atesta que a informação da identidade assinada vai junto com, ou é ligada, a chave pública. Assim, um certificado é basicamente uma chave pública com uma ou duas formas de ID anexado, mais um selo cordial de aprovação de alguma outra pessoa confiável.

3.1.14. Distribuição do certificado

São utilizados certificados quando é necessário trocar chaves públicas com outra pessoa. Para grupos pequenos de pessoas para que se deseje comunicar com segurança, é fácil trocar disquetes manualmente ou e-mail contendo a chave pública de cada dono. Esta é uma distribuição manual da chave pública, e isto é prático até um certo ponto. Além deste ponto, é necessário optar por sistemas em lugares que possam prover a segurança necessária, armazenamento e mecanismos de troca. Estes poderiam entrar na forma de armazenamento em repositórios chamado Servidores de certificado, ou sistemas mais estruturados que provêem administração adicional de chaves e é chamada de Infra-estruturas de chaves públicas (PKI).

3.1.15. Servidores de certificados

Um servidor de certificado, também chamado de servidor de chave, é um banco de dados que permite para os usuários submeter e recuperar certificados digitais. Um servidor de certificados normalmente provê algumas características administrativas que permite uma companhia a manter suas políticas de segurança — por exemplo, permitindo que só as chaves que satisfazem certas exigências serem armazenadas.

3.1.16. PKI - Infra-estrutura de chaves públicas (“*Public Key Infrastructure*”)

Um PKI contém as facilidades de armazenamento de certificados de um servidor de certificado, mas também provê facilidades de administração de certificado (a habilidade para emitir, revogar, armazenar, recobrir, e certificados de confiança). A principal característica de um PKI é a introdução do que é conhecido como uma autoridade de certificação, ou CA (“*certification authority*”), na qual é uma entidade humana—uma pessoa, um grupo, departamento, companhia, ou outra associação—que uma organização autorizou a emitir certificados para seus usuários de computador. O papel de CA é análogo ao escritório de passaporte do governo de um país. Uma CA cria certificados e digitalmente os assina usando a chave privada de CA. Por causa de seu papel de criar certificado, o CA é o componente central de um PKI. Usando a chave pública do CA, qualquer um que quer verificar a autenticidade de um certificado verifica a emissão da assinatura digital do CA, e conseqüentemente, a integridade dos conteúdos do certificado (e mais ainda, a chave pública e a identidade do possuidor do certificado).

3.1.17. PGP

É um poderoso programa de encriptação (ou criptografia) que se tornou bastante popular na Internet. Ele permite encriptar mensagens (e-mails, por exemplo) de forma a "ocultar" seu conteúdo. Assim, apenas quem possuir a chave de desencriptação poderá ter acesso ao conteúdo da mensagem. Isso é útil, sobretudo, na transmissão de documentos confidenciais pela Internet, cujos canais de comunicação não são seguros e podem ser monitorados com facilidade. O PGP pode, também, autenticar uma mensagem, atribuindo-lhe uma assinatura digital. O receptor da mensagem pode, então, verificar a autenticidade da mensagem e do autor. Caso a mensagem seja adulterada antes de chegar ao destinatário, o PGP acusará essa violação da integridade da mensagem. O PGP foi criado por Philip Zimmeramann em 1991. Zimmermann colocou o PGP disponível na Internet, desafiando as leis norte-americanas, que proíbem a exportação de sistemas de criptografia. Foi processado pelo governo, mas depois de alguns anos o processo foi arquivado. Hoje o PGP é distribuído como freeware para usuários domésticos, havendo também uma versão comercial para empresas vendida pela Network Associates.

Quão segura é uma mensagem encriptada por PGP? Podemos afirmar que é muito segura. O PGP utiliza um algoritmo chamado RSA de encriptação por chave pública. Um ataque de força bruta ao RSA é impensável. Estima-se que uma rede de um milhão de computadores Pentium 133MHz levaria cerca de 25.000 anos para quebrar uma única chave de 1024 bits.

Inventado por Phil Zimmermman em 1991, é um programa criptográfico famoso e bastante difundido na Internet, destinado a criptografia de e-mail pessoal. Algoritmos suportados: hashing: MD5, SHA-1, simétricos: CAST-128, IDEA e 3DES, assimétricos: RSA, Diffie-Hellman/DSS. Versão mais recente: 8.1[4]

3.1.18. Formatos de certificados PGP e X.509

Um certificado digital é basicamente uma coleção de informações de identificação junto com uma chave pública e assinado por uma terceira parte confiável para provar sua autenticidade. PGP reconhece dois formatos de certificado diferentes: certificados PGP e certificados X.509

[4] http://www.pgp.com/downloads/freeware/index.html, visitado em 20/11/2004

O número da versão do PGP, isto identifica qual versão de PGP foi usado para criar a chave associada com o certificado. A chave pública do possuidor do certificado, ou seja, porção pública do seu par de chaves, junto com o algoritmo da chave: RSA, DH, (Diffie-Hellman), ou DSA (*Digital Signature Algorithm*). A informação do possuidor do certificado, isto consiste na informação da identidade sobre o usuário, como o nome dele ou dela, ID do usuário, fotografia, e assim por diante.

A assinatura digital do dono do certificado é a assinatura que usa a correspondente chave privada da chave pública associada com o certificado.O período de validez do certificado a data de começo do certificado / tempo e data de vencimento / tempo; indica quando o certificado vai expirar.

O algoritmo preferido de encriptação simétrica para a chave indica o algoritmo de encriptação para qual o dono do certificado prefere ter a informação encriptada. Poderíamos pensar que um certificado de PGP fosse como uma chave pública com um ou mais etiquetas amarradas a isto. Nestas etiquetas encontraremos a informação que identifica o dono da chave e uma assinatura do dono de chave, significando que a chave e a identificação vão junto (esta assinatura particular é chamada de *self-signature*; todo certificado de PGP contém uma *self-signature*.).

Um aspecto de formato que só tem no PGP é que um único certificado pode conter assinaturas múltiplas. Várias ou muitas pessoas podem assinar o par chave/identificação para atestar a própria garantia deles/delas que a chave pública definitivamente pertence ao dono especificado. Se olharmos em um servidor de certificado público, podemos notar que certos certificados, como aquele do criador do PGP, Phil Zimmermann, contem muitas assinaturas. Alguns certificados de PGP consistem em uma chave pública com várias etiquetas, cada dos quais contém diferentes meios de identificar o dono chave (por exemplo, o nome do dono e conta de e-mail, o apelido do dono e conta de e-mail de casa, uma fotografia do dono—tudo em um certificado). A lista de assinaturas de cada uma dessas identidades podem diferir; aí as assinaturas atestam à autenticidade de que um das etiquetas pertencem à chave pública, não que todas as etiquetas na chave são autênticos.

O formato X.509 é um outro formato de certificado muito comum. Todos os certificados X.509 obedecem ao padrão internacional ITU-T X.509; assim (teoricamente) certificados X.509 foram criados para uma aplicação podem ser usados por qualquer aplicação que obedece X.509. Na prática, porém, companhias diferentes criaram as próprias extensões para certificados X.509, nos quais não trabalham junto. Um certificado exige alguém para validar que uma chave pública e o

nome do dono da chave vão juntos. Com certificados de PGP, qualquer um pode representar o papel de validador. Com certificados X.509, o validador é sempre uma *Autoridade de Certificação* ou alguém designado por uma CA (tenha em mente que certificados PGP também suportam completamente uma estrutura hierárquica que usam uma CA para validar certificados.)

Um certificado X.509 é uma coleção de um conjunto padrão de campos contendo informações sobre um usuário ou dispositivo e sua correspondente chave pública. O padrão X.509 define qual informação vai no certificado, e descreve como codificar isto (o formato dos dados). Todos certificados X.509 têm os seguintes dados:

?? O número da versão do X.509, isto identifica qual padrão é aplicado na versão do X.509 para este certificado, o que afeta qual informação pode ser especificada neste.

?? A chave pública do possuidor do certificado—a chave pública do possuidor do certificado, junto com um algoritmo de identificação que especifica qual sistema de criptografia pertence a chave e qualquer parâmetros associados.

?? O número de série do certificado, a entidade (aplicação ou pessoa) que criou o certificado é responsável por neste um número de série para distinguir este de outros certificados que ele emite. Esta informação é usada de várias maneiras; por exemplo, quando um certificado é revogado, seu número serial é colocado em uma Lista de Revogação de Certificado ou CRL.

?? A identificação única do possuidor de certificado ou (DN-nome distinguido). Este nome tem que ser único pela Internet. Um DN consiste em múltiplas subseções e pode parecer com algo do tipo: *CN=Bob Allen, OU=Total Network Security Division, O=Network Associates, Inc., C=US*. O nome único do emissor do certificado, o nome único da entidade que assinou o certificado. Esta entidade normalmente é uma CA. Usar o certificado implica em confiar na entidade que assinou este certificado. (Note que em alguns casos, como raiz do certificado ou top-level da CA, o emissor assina seu próprio certificado.).

?? A assinatura digital do emissor, assinatura que usa a chave privada da entidade que emitiu o certificado. A identificação do algoritmo de assinatura identifica o algoritmo usado pela CA para assinar o certificado.

Para obter um certificado X.509, temos que pedir para uma CA que nos emita um certificado. Providenciaríamos uma chave pública, provaríamos que possuímos a chave privada correspondente e alguma outra informação específica sobre o requisitante. Seria necessário então, assinar digitalmente a informação e enviar no pacote todo o pedido de certificado—para a CA. A CA então executa alguma devida diligência verificando que a informação provida está correta, e nesse caso, gera o certificado e o retorna.

Poderíamos pensar em um certificado X.509 como parecendo um certificado de papel padrão com uma chave pública gravada nele. Isto contém o nome e um pouco de informação sobre o solicitante, mais a assinatura da entidade que emitiu o certificado.

4. CONSIDERAÇÕES FINAIS

A criptografia, talvez tão antiga quanto à própria escrita, hoje é um dos métodos mais eficientes de se transferir informações. Em relação à computação é muito importante para que se possa garantir segurança em todo o ambiente computacional que necessite de confidencialidade em relação às informações que são manipuladas.

Para isso existem os algoritmos criptográficos ou criptosistemas que podem ser tanto simétricos como assimétricos. Num algoritmo simétrico a encriptação e a decriptação são feitas com uma única chave. E em um algoritmo assimétrico duas chaves são empregadas, uma chave pública e uma privada onde a pública é divulgada e a privada é deixada em segredo. Os algoritmos são utilizados para efetuar o ocultamento das informações, que é a chamada criptografia.

A forma mais eficiente de utilizar os recursos criptográficos existentes é combinndo-os. Isto é:

1. Usar os certificados digitais para garantir a identidade do emissor;

2. Assinar digitalmente as mensagens garantindo assim que elas não foram alteradas;

3. Usar a criptografia assimétrica para a distribuição da chave simétrica e, por fim;

4. Usar a criptografia simétrica para cifrar os dados transmitidos, já que a criptografia simétrica mostra-se, dependendo do algorítmo utilizado, cem vezes mais rápida do que a assimétrica.

REFERÊNCIAS BIBLIOGRÁFICAS

DOMIGUES L. F. F. F. Cryptanalist. Universidade Atlântica, ano letivo 1999/2000

DUARTE M.; FONTE, S. Como produzir uma assinatura eletrônica segura? Disponível em: <http://mcduarte.planetaclix.pt/assinaturadigital.html >, visitado em 20/11/2004

JORMALAINEN S.; LAIN, J. Security in the WTLS. Disponível em: <http://www.hut.fi/~jtlaine2/wtls/>. MOREIRA, A. visitado em 20/11/2004

PFLEEGER C. P. Security in Computing. 2. ed. Upper Saddle River : Prentice Hall, 1997. Disponível em: http://media.wiley.com/assets/152/08/information.pdf., visitado em 20/11/2004

POLÍTICA de segurança da ICP - Brasil: parte III. Disponível em: <http://www.planalto.gov.br/ccivil_03/consulta_publica/PDF/PoliticadeSeguranca.pdf >, visitado em 20/11/2004

REZENDE, P. A. D. Certificados digitais, chaves púbicas e assinaturas: o que são, como funcionam e como não funcionam. Brasília: UnB. Disponível em: <http://www.cic.unb.br/docentes/pedro/segdadtop.htm >, visitado em 20/11/2004

[SCHNPA] B. SCHNEIER, "Why Cryptography is Harder than it looks". http://www.counterpane.com/whycrypto.html, visitado em 20/11/2004

DAVID KOSIUR, Security and Electronic Commerce, Microsoft Press, 1997. http://mspress.microsoft.com/prod/books/sampchap/1252.htm, visitado em 20/11/2004

Tese De Mestrado: "Introdução De Mecanismos De Segurança Em Sistemas De Correio Eletrônico" Paulo Sergio Pagliusi, Orientada Por Cláudio Leonardo Lucchesi E Luiz Eduardo Buzato, Unicamp, 1998.

DIERKS, T.; ALLEN, C. The Tls Protocol: Version 1.0. Disponível em: http://www.ietf.org/rfc/rfc2246.txt, visitado em 20/11/2004

KUROSE JAMES, F.; ROSS KEITH W., Computer Networking: A Top-Down Approach Featuring the Internet, 2nd ed. Addison-wesley Publi, 2003

GLOSSÁRIO

Algoritmo	Conjunto de operações elementares que devem ser efetuadas para se obter um resultado desejado. Por exemplo, uma receita de bolo é um algoritmo.
Assinatura Digital	Um código utilizado para verificar a integridade de um texto ou mensagem. Também pode ser utilizado para verificar se o remetente de uma mensagem é mesmo quem diz ser
Autoridade Certificadora	É a entidade responsável pela emissão do Certificado do CLIENTE, no caso, a VeriSign, Inc., que é a responsável pela emissão do Certificado Digital do CLIENTE
Certificado Digital	É um conjunto de dados de computador, gerados em observância à Recomendação Internacional ITU-T X.509, que se destina a registrar, de forma única, exclusiva e intransferível, a relação existente entre uma chave de criptografia, o CLIENTE e a Autoridade Certificadora. O Certificado Digital é instalado no computador do CLIENTE.
Cifra	Conjunto de procedimentos e conjunto de símbolos (letras, nomes, sinais, etc) usados para substituir as letras de uma mensagem para encriptá-la. É geralmente classificada como cifra de transposição e cifra de substituição.
Cifragem	Cifrar ou cifragem. Procedimento pelo qual se torna impossível a compreensão de um documento a qualquer pessoa que não possua a chave da cifra.
Chave de Criptografia	É o valor numérico ou código que uma vez aplicado a um determinado dado de computador o torna ininteligível para terceiros.
Chave Privativa de Criptografia	Espécie de chave de criptografia de uso exclusivo, secreto e intransferível do CLIENTE, que é detentor de um Certificado Digital.
Chave Pública de Criptografia	Espécie de chave de criptografia de uso geral e irrestrito, acessível por qualquer pessoa, que tenha interesse em se comunicar com o CLIENTE, detentor de um Certificado Digital.
CriptoAnálise	Criptoanálise ou criptanálise. Métodos de analisar mensagens cifradas com o objetivo de decifrá-las.
Criptosistema	Cifra
Criptografia	É a ciência e arte de escrever mensagens em forma cifrada ou em código. É parte de um campo de estudos que trata das comunicações secretas. É usada, dentre outras finalidades, para: autenticar a identidade de usuários;

	autenticar transações bancárias; proteger a integridade de transferências eletrônicas de fundos, e proteger o sigilo de comunicações pessoais e comerciais
Criptologia	Criptoanálise ou criptanálise. Métodos de analisar mensagens cifradas com o objetivo de decifrá-las.
Desencriptar	Restaurar documentos cifrados, restaurando-os ao estado original, sem dispor das chaves teoricamente necessárias.
Hash	Uma função hash é uma equação matemática que utiliza texto (tal como uma mensagem de e-mail) para criar um código chamado message digest (resumo de mensagem). Alguns exemplos conhecidos de funções hash: MD4 (MD significa message digest), MD5 e SHA. Uma função hash utilizada para autenticação digital deve ter certas propriedades que a tornem segura para uso criptográfico. Especificamente, deve ser impraticável encontrar: - Texto que dá um hash a um dado valor. Ou seja, mesmo que você conheça o message digest, não conseguirá decifrar a mensagem.
Site	Local na Internet identificado por um nome de domínio, constituído por uma ou mais páginas de hipertexto, que podem conter textos, gráficos e informações multimídia
VPN	Do Inglês Virtual Private Network. Termo usado para se referir à construção de uma rede privada utilizando redes públicas, como a Internet, como infra-estrutura. Estes sistemas utilizam criptografia e outros mecanismos de segurança para garantir que somente usuários autorizados possam ter acesso à rede privada e que nenhum dado será interceptado enquanto estiver passando pela rede pública.

ANEXO I – O HEROI DA CRIPTOGRAFIA MODERNA

Por que você precisa do PGP?

Por Phil Zimmermann

fonte: http://www.pgpi.org/doc/whypgp/br/, visitado em 20/11/2004

É pessoal. É particular. E não é da conta de mais ninguém a não ser você. Você pode estar planejando uma campanha política, discutindo seus impostos, ou tendo um caso ilícito. Ou você pode estar fazendo algo que você sente que não deveria ser ilegal, mas é. Qualquer que seja o caso, você não quer que seu correio eletrônico particular (E-mail) ou documentos confidenciais sejam lidos por mais ninguém. Não há nada errado em assegurar sua privacidade. Privacidade é algo tão natural e respeitável quanto a Constituição.

Talvez você pense que seu E-mail seja tão lícito que criptografia não seja justificável. Se você realmente é um cidadão que respeita as leis estritamente com nada a esconder, então por que você não envia sempre sua correspondência em cartões-postais? Por que não se submeter a testes de drogas sempre que requisitado? Por que exigir um mandado para revistas da polícia em sua casa? Você está tentando esconder algo? Você deve ser um subversivo ou um traficante de drogas se você esconde sua correspondência em envelopes. Ou talvez um maluco paranóico. Cidadãos que respeitam as leis têm alguma necessidade de criptografar seus E-mails?

E se todos acreditassem que cidadãos que respeitam as leis devessem usar cartões-postais para sua correspondência? Se alguma brava alma tentasse assegurar sua privacidade ao usar um envelope para sua correspondência, isso levantaria suspeita. Talvez as autoridades abririam sua correspondência para ver o que ela estaria escondendo. Felizmente, nós não vivemos naquele tipo de mundo, porque todos protegem a maior parte de sua correspondência com envelopes. Então, ninguém levanta suspeita ao assegurar sua privacidade com um envelope. Não há segurança em números. Analogamente, seria bom se todos rotineiramente usassem criptografia para todo seu E-mail, inocente ou não, de modo que ninguém levantaria suspeita ao assegurar sua privacidade no E-mail com criptografia. Pense nisso como uma forma de solidariedade.

Hoje, se o Governo quer violar a privacidade de cidadãos comuns, ele tem que gastar uma certa quantidade de recursos e esforço para interceptar e abrir dissimuladamente correspondência em papel, e ouvir e possivelmente transcrever conversas faladas ao telefone. Este tipo de monitoramento trabalhoso e intensivo não é prático em grande escala. Isto é feito somente em casos importantes quando parece valer a pena.

Mais e mais das nossas comunicações particulares estão sendo roteadas por canais eletrônicos. Correio eletrônico está gradualmente substituindo a correspondência convencional em papel. Mensagens por E-mail são simplesmente fáceis demais de se interceptar e de se vasculhar em busca de palavras interessantes. Isto pode ser feito facilmente, rotineiramente, automaticamente, e indetectavelmente em grande escala. Cabogramas internacionais já são vasculhados deste modo em grande escala pela NSA.

Nós estamos nos dirigindo a um futuro no qual a nação será entrecortada por redes de dados de fibra ótica de alta capacidade, ligando todos os nossos cada vez mais onipresentes computadores. O

E-mail será a norma para todos, não a novidade que é hoje. O Governo protegerá nosso E-mail com protocolos de criptografia projetados pelo Governo. Provavelmente a maioria das pessoas se sujeitará a isso. Mas talvez algumas pessoas preferirão suas próprias medidas de proteção.

O Projeto de Lei 266 do Senado (dos EUA), um extenso projeto anti-crime de 1991, tinha uma perturbadora medida encravado nele. Se esta resolução sem restrições tivesse se tornado uma lei real, ela teria forçado fabricantes de equipamentos de comunicação segura a inserir alçapões nos seus produtos, de modo que o Governo pudesse ler as mensagens criptografadas de qualquer um. Eis seu conteúdo:

"É o julgamento do Congresso que fornecedores de serviços de comunicações eletrônicas e fabricantes de equipamentos de serviços de comunicações eletrônicas devem assegurar que sistemas de comunicações permitam ao Governo obter o conteúdo integral de voz, dados e outras comunicações quando apropriadamente autorizado pela lei."

Esta medida foi derrotada após rigorosos protestos de civis libertários e de grupos industriais.

Em 1992, a proposta de grampo da Telefonia Digital do FBI foi apresentada ao Congresso. Ela requereria que todos os fabricantes de equipamentos de comunicações embutissem portas especiais de grampo remoto as quais capacitariam o FBI a grampear remotamente todas as formas de comunicação eletrônica a partir de escritórios do FBI. Apesar de nunca ter atraído qualquer apoio no Congresso por causa da oposição popular, ela foi reapresentada em 1994.

O mais alarmante de tudo é a nova iniciativa de política de criptografia da Casa Branca, em desenvolvimento na NSA desde o início da administração Bush, e revelada em 16 de abril de 1993. A peça central desta iniciativa é um dispositivo de criptografia construído pelo Governo, chamado de chip Limitador, contendo um novo algoritmo confidencial NSA. O Governo está encorajando a indústria privada a projetá-lo em todos seus produtos de comunicação segura, como telefones seguros, fax seguro, etc. A AT&T agora está colocando o Limitador nos seus produtos de voz seguros. A trama: No momento de sua fabricação, cada chip Limitador será carregado com sua própria chave única, e o Governo fica com uma cópia, a qual é arquivada. Nada com o que se preocupar, entretanto -- o Governo promete que eles usarão estas chaves para ler seu tráfego somente quando apropriadamente autorizados pela lei. É claro, para fazer o Limitador completamente efetivo, o próximo passo lógico seria tornar ilegais outras formas de criptografia.

Se a privacidade se tornar ilegal, somente criminosos terão privacidade. Agências de inteligência têm acesso a uma ótima tecnologia criptográfica. Assim como os grandes traficantes de armas e drogas. Assim como fornecedores de sistemas militares, empresas petrolíferas, e outros gigantes empresariais. Mas pessoas comuns e organizações políticas populares majoritariamente não têm tido acesso à tecnologia criptográfica de chave-pública de nível militar. Até agora.

O PGP dá o poder às pessoas para tomar sua privacidade em suas próprias mãos. Há uma crescente necessidade social para ele. É por isso que eu o escrevi.

ANEXO II – O ALGORITMO RSA

Fonte: www.dei.isep.ipp.pt/~andre/documentos/criptografia.html ,visitado em 20/11/2004

Este algoritmo é devido a Ron Rivest, Adi Shamir e Len Adleman (RSA), baseia-se no seguinte: é simples arranjar dois números primos grandes, mas é muito complicado fatorar o seu produto. O RSA tem agüentado todas as investidas dos cripto - analistas, contudo temos que atender ao fato de ser um problema matemático, existe sempre o risco de descoberta de uma técnica para resolver o problema de forma eficiente.

Geração das chaves: escolhem-se dois número primos grandes **a** e **b**.

- Calcula-se **n = a x b**.
- Calcula-se **ø(n) = (a-1) x (b-1)**.
- Escolhe-se um número pequeno **p** que seja primo relativo de **ø(n)** e **< ø(n)**, calcula-se **s** tal que **(p x s) mod ø(n) = 1**, onde **mod** é o operador "resto da divisão inteira" (aritmética modulo ø(n)).
- (Dois números são primos relativos se o maior divisor comum é 1)
- O par **(n,p)** constitui a chave pública, **d** é a chave secreta.

Aplicação:

- Cifragem: $\mathbf{C = M^p \bmod n}$
- Decifragem: $\mathbf{M = C^s \bmod n}$

, onde **M** e **C** são respectivamente a mensagem original e mensagem cifrada, ambas com valores possiveis de zero a **n-1**.

Uma propriedade interessante do RSA é a possibilidade de inversão das chaves, pode-se cifrar uma mensagem com a chave s, para decifrar será agora necessária a chave pública: utilizável para autenticação e assinatura digital.

Por exemplo, tomem-se os números primos a=7 e b=17:
n = a x b = 119
ø(n) = (a-1) x (b-1) = 96
como primo relativo de ø(n) podemos escolher p=5
então para obter p x s mod 96 = 1, podemos usar s = 77
pois 5 x 77 = 385, 385 mod 96 = 1
A chave pública é (5;119) e a chave secreta é 77

Exemplos de aplicação:
Para evitar perdas de dados torna-se necessário aplicar a seguinte propriedade da aritmética modular:

(a x b) mod n = ((a mod n) x (b mod n)) mod n

Para cifrar o número **2** com a chave pública temos C = 2^5 mod 119 = 32 mod 119 = **32**
Para decifrar utiliza-se M = $\mathbf{32}^{77}$ mod 119,

Para usar uma calculadora, sem perder dados podemos usar 11 parcelas:
$((32^7 \bmod 119) \times ... \times (32^7 \bmod 119)) \bmod 119$,
Obtemos então $25^{11} \bmod 119$, podemos agora aplicar $((5^{11} \bmod 119) \times (5^{11} \bmod 119)) \bmod 119$,
Obtemos agora $45^2 \bmod 119 =$ **2**

Também se pode cifrar o número **2** com a chave secreta temos $C = 2^{77} \bmod 119$
para usar uma calculadora, sem perder dados podemos usar 11 parcelas:
$((2^7 \bmod 119) \times ... \times (2^7 \bmod 119)) \bmod 119 = 9^{11} \bmod 119 =$ **32**
Para decifrar utiliza-se $M =$ **32**$^5 \bmod 119 =$ **2**

Para cifrar o número **3** com a chave pública temos $C = 3^5 \bmod 119 = 243 \bmod 119 =$ **5**
Para decifrar utiliza-se $M =$ **5**$^{77} \bmod 119$,
Para usar uma calculadora, sem perder dados podemos usar 7 parcelas:
$((5^{11} \bmod 119) \times ... \times (5^{11} \bmod 119)) \bmod 119$
obtemos então $45^7 \bmod 119 =$ **3**

Como podemos verificar as operações a realizar na decifragem não são simples, especialmente se atendermos a que os números **a** e **b** devem ser grandes.

Para os valores 0 e 1 a mensagem e o resultado da cifragem coincidem, contudo isto não é muito grave, os valores usados para n são muito elevados (na ordem de 10^{200}), o tamanho mais comum para as mensagens a cifrar (M) é de 512 bits (que representa números até mais de 10^{154}), para este número de bits não são vulgares os valores 0 e 1, de qualquer modo isto pode ser resolvido pela adição de duas unidades a M antes de entrar no algoritmo de cifragem e subtração de duas unidades depois de sair do algoritmo de decifragem.

Gerar chaves RSA não é uma operação simples, o primeiro problema é arranjar dois números primos **a** e **b** com uma ordem de grandeza de 10^{100}, usar os algoritmos tradicionais de geração de números primos é impossível, a solução é usar testes eliminatórios, estes testes permitem saber se um número não é primo, ou qual a probabilidade de ser primo, se um dado número depois de testado intensivamente não é eliminado será adotado. A segunda questão prende-se com a determinação de um primo relativo de ø(n), p ou s e de seguida é necessário determinar outro número para verificar a relação $(p \times s) \bmod ø(n) = 1$.

O algoritmo RSA serve de base a muitos sistemas de segurança atuais, tais como o PGP ("Pretty Good Privacy"), usado em correio eletrônico . A seu grande inconveniente é a lentidão já que terá de ter como suporte sistemas capazes de lidar com números muito grandes. Na maioria dos casos o RSA é usado para distribuir uma chave de criptografia simétrica, apenas no inicio de sessão, durante a sessão os dados são cifrados com essa chave, usando criptografia convencional, muito mais rápida.

Sob o ponto de vista de criptoanálise e devido ao número de bits das chaves a aplicação de força bruta (tentar todas as chaves secretas possíveis) está totalmente excluída , o algoritmo é lento e os tamanhos de chave RSA mais usados são 512 bits e 1024 bits. A abordagem é tentar obter os dois fatores primos de n. Contudo tal é extremamente complexo para a ordem de grandeza usada para n, o tempo necessário cresce exponencialmente com o valor de n. Novamente interessa referir que o RSA se baseia num problema Matemático, estando por isso sujeito a alguma solução brilhante de algum gênio .

ANEXO III – ADVANCED ENCRYPTION STANDARD (AES)

Fonte http://www.inf.furb.br/~pericas/orientacoes/Esteganografia2003.pdf ,visitado em 20/11/2004

Em outubro de 2000, o *National Institute of Standards and Technology* (NIST) anunciou um novo padrão de uma chave secreta de cifragem, escolhido entre 15 padrões candidatos. Este novo padrão pretendia substituir o velho algoritmo DES, cujo tamanho das chaves estava se tornando muito pequeno. O Rijndael - nome originário dos seus inventores Rijmen e Daemen - foi escolhido para se tornar o novo padrão, que se chamou *Advanced Encryption Standard* (AES).

Este sistema de encriptação é dito ser um "bloco" de cifragem à medida que as mensagens são encriptadas em blocos inteiros, com unidades de 128 bits. Existem múltiplas idéias que propõem a utilização de chaves com 128, 192, 256 bits. Fazendo uma comparação, o DES encripta blocos de 64 bits com uma chave de 56 bits, e o DES triplo, normalmente, encripta blocos de 64 bits com uma chave de 112 bits.

O algoritmo Rijndael diferencia-se da maioria dos outros algoritmos usados atualmente, pois não usa uma estrutura do tipo *Feistel* na sua fase de rotação. Numa estrutura *Feistel* os bits de estado intermediário são transpostos em uma outra posição sem serem alterados. No Rijndael, a fase é composta de transformações uniformes inversíveis distintas chamadas de *layers*.

O Rijndael é um cifrador de substituição e permutação. Ele cifra um texto claro de 128 bits em um texto cifrado de 128 bits, usando para isso *n* fases, onde cada resultado intermediário entre as transformações é chamado de “estado”. O número de fases *n* definido para a cifra depende do tamanho de bloco e do tamanho de chave que estão sendo utilizados. O menor números de fases é 10 (correspondendo ao tamanho de bloco de 128 bits e tamanho de chave de 128 bits), sendo este limite válido para todos os tamanhos de blocos e de chaves.

Cada fase consiste na aplicação seguida das transformações de: substituição (*ByteSub*), deslocamento de linhas (*ShiftRow*), mesclagem de colunas (*MixColumn*) e adição de chaves (AddRoundKey).

ANEXO IV – BRUCE SCHNEIER

Fonte http://www.schneier.com/, visitado em 20/11/2004

Schneier é o autor de oito livros, incluindo seu livro atual, B*eyond Fear: Thinking Sensibly About Security in an Uncertain World*, no qual fala dos problemas da segurança do pequeno ao grande: segurança pessoal, crime, segurança incorporada, segurança nacional.

Secrets & Lies: Digital Security in a Networked World que foi publicado em outubro 2000, tem vendido já 80.000 cópias. Um de seus livros, A*pplied Cryptography* agora em sua segunda edição, 150.000 cópias e foi traduzido em cinco línguas.

Escreve o C*rypto-gram, newsletter free* na qual tem sobre 70.000 leitores. Apresentou papéis em muitas conferências internacionais, e é um escritor freqüente, editor contribuindo, e lecionando sobre os tópicos: criptografia segurança do computacional , e privacidade.

Schneier projetou o algoritmo B*lowfish Encryption Algorithm* e também o algoritmo T*wofish* que foi um dos finalistas para o padrão A*dvanced Encryption Standard (AES)*. Também foi membro do grupo de diretores do I*nternational Association for Cryptologic Research*, e é membro consultor do E*lectronic Privacy Information Center*.

Schneier é formado em Ciência da Computação pela A*merican University* e também é físico pela U*niversity of Rochester*.

ANEXO V – IMPLEMETAÇÃO DO ALGORITMO DES

fonte: https://www.redes.unb.br/security/criptografia/des/fontedes.c , visitado em 20/11/2004

Aqui podemos ver um exemplo em linguagem C de um algoritmo DES. Este aquivo fonte e o programa excutável podem ser encontrados no link acima.

```
/*****************************************************************/
/*                                                               */
/*                      ARQUIVO : FONTEDES.c                     */
/*                                                               */
/*****************************************************************/
/*          ------------------   D . E . S .  ----------------------  */
            #include <stdio.h>
            #include <stdlib.h>
            #include <string.h>

/*  ------------------ des.h ------------------------------  */

/*  ARQUIVO CABEÇALHO PARA OS ALGORITMOS DO D.E.S            */

/*  DIVISÃO DE UM BLOCO DE 64 BITS EM  02 SUB-BLOCOS         */
            struct LR
             {
              long L;
              long R;
              };

/*          PERMUTAÇÃO DA CHAVE (48 BITS)                      */
            struct ks
             {
              char ki[6];
              };

            int opcao;
            char chave[8];

            char nome1[32];
            char nome2[32];

/*          MACROS PARA A DEFINIÇÃO DE UMA TABELA DE PERMUTAÇÃO   */
            #define ps(n)                    ((unsigned char) (0x80 >> (n-1)))
            #define b(n,r)      ((n>r||n<r-7)?0:ps(n-(r-8)))
            #define p(n)                     b(n, 8),b(n,16),b(n,24),b(n,32),\
                                                          b(n,40),b(n,48),b(n,56),b(n,64)

/*          DEFINIÇÕES DE FUNÇÕES                              */
            void inverse_permute(long *op, long *ip, long *tbl, int n);
            void permute(long *op, long *ip, long *tbl, int n);
            long f(long blk, struct ks ky);
            struct ks KS(int n);
            void encrypt(void);
        void decrypt(void);

            static void rotate(unsigned char *c, int n);
            static long S(struct ks ip);
            static int fourbits(struct ks, int s);
            static int sixbits(struct ks, int s);


/*          TABELAS                                            */
            extern unsigned char Pmask[];
            extern unsigned char IPtbl[];
            extern unsigned char Etbl[];
            extern unsigned char Ptbl[];
            extern unsigned char stbl[8][4][16];
            extern unsigned char PC1tbl[];
            extern unsigned char PC2tbl[];
            extern unsigned char ex6[8][2][4];

/*****************************************************************/
/*****************************************************************/

            void main()
             {
              int i;
```

```
              printf("\n\n");
              printf("                        DATA ENCRYPTION STANDARD - D.E.S.    \n");
              printf("             Opções possíveis:\n");
              printf("             (1) - CIFRAR;\n");
              printf("             (2) - DECIFRAR:\n");
              printf("                        Entre opção desejada: ");
              scanf("%d",&opcao);

              switch(opcao)  {
               case 1:
                              printf("             D.E.S.  -    CIFRAR...\n");
                              printf("  Entre com a chave (8 caracteres):");
                              scanf("%s",&chave[0]);
                              printf("\n           Entre nome do arquivo contendo o claro: ");
                              scanf("%s",&nome1[0]);
                              printf("\n           Entre nome do arquivo p/ conter o cifrado: ");
                              scanf("%s",&nome2[0]);
                              encrypt();
                              break;
               case 2:
                              printf("             D.E.S.  -    DECIFRAR...\n");
                              printf("  Entre com a chave (8 caracteres): ");
                              scanf("%s",&chave);
                              printf("\n           Entre nome do arquivo contendo o cifrado: ");
                              scanf("%s",&nome1[0]);
                              printf("\n           Entre nome do arquivo p/ conter o decifrado: ");
                              scanf("%s",&nome2[0]);
                              decrypt();
                              break;
               default:
                               exit(0);
                               break;
                                              }
              }

/*          ----------------------  encrypt.c ----------------------  */

/*          CIFRAÇÃO COM D.E.S.                                  */

            void encrypt()
             {
               int i,j;
               struct LR op, ip;
               struct ks keys[16];
               FILE *arqin, *arqout;

              for (i = 0; i < 16; i++)
               {
                         keys[i] = KS (i);
                        }
                           if ((arqin = fopen(nome1, "rb")) != NULL)     {
                                       if  ((arqout = fopen(nome2, "wb")) != NULL)    {
                                                   printf("\n                        D.E.S.  -    CIFRANDO...\n");
                                                   while  (fread(&ip, 1,
                                                                                      sizeof(struct LR), arqin) != 0)
{
                                                                int n;
                                                                /*    PERMUTAÇÃO INICIAL            */
                                                                permute(&op.L, &ip.L, (long *)IPtbl, 64);
                                                                /*    "SWAP" E ITERAC,O~ES DA CHAVE    */
                                                                for (n = 0; n < 16; n++)    {
                                                                           ip.L = op.R;
                                                                           ip.R = op.L ^ f(op.R, keys[n]);

                                                                           op.R = ip.R;
                                                                           op.L = ip.L;
                                                                }
                                                                /*        PERMUTAÇÃO INICIAL INVERSA       */
                                                                inverse_permute(&op.L, &ip.L,
                                                                           (long *)IPtbl, 64);
                                                                fwrite(&op, 1, sizeof(struct LR), arqout);
                                                                /*    FINALIZAR O ÚLTIMO BLOCO        */
                                                                ip.L = ip.R = 0;
                                                   }
                                                   fclose(arqout);
                                       }
                                       fclose(arqin);
                                 printf("\n         ... CIFRAÇÃO CONCLUÍDA !!!\n");
                           }
             }

/*          ------------------------ decrypt.c--------------------  */

/*          DECIFRAÇÃO COM O D.E.S.                              */
```

```
            void decrypt()
             {
               int i;
               struct LR op, ip;
               struct ks keys[16];
               FILE *arqin, *arqout;

              for (i =0; i < 16; i++)
                          keys[i] = KS (i);
                           if ((arqin = fopen(nome1, "rb")) != NULL)   {
                                      if ((arqout = fopen(nome2, "wb")) != NULL)   {
                                                 printf("\n                         D.E.S.  -    DECIFRANDO...\n");
                                                 while  (fread(&ip, 1,
                                                                        sizeof(struct LR), arqin) != 0)      {
                                                            int n;
                                                            /*  PERMUTAÇÃO INICIAL                */
                                                            permute(&op.L, &ip.L, (long *) IPtbl, 64);
                                                            /*  "SWAP" E ITERAÇÕES DA CHAVE        */
                                                            for  (n = 15; n >= 0; --n)    {
                                                                       ip.R = op.L;
                                                                       ip.L = op.R ^ f(op.L, keys[n]);
                                                                       op.R = ip.R;
                                                                       op.L = ip.L;

                                                            }
                                                            /*  PERMUTAÇÃO INICIAL INVERSA         */
                                                            inverse_permute(&op.L, &ip.L,
                                                                       (long *)IPtbl, 64);
                                                            fwrite(&op, 1, sizeof(struct LR), arqout);
                                                            /*   FINALIZAR O ÚLTIMO BLOCO           */
                                                            ip.L = ip.R = 0;

                                                 }
                                                 fclose(arqout);
                                      }
                                      fclose(arqin);
                                      printf("\n                ... DECIFRAÇÃO CONCLUÍDA !!!\n");
                           }
              }
/*          -------------------------  des.c  ----------------------  */

/*           FUNÇÕES E TABELAS PARA A CIFRAÇÃO E DECIFRAÇÃO           */
/*       COM O ALGORITMO DO D.E.S.                                 */

/*          REALIZAR A PERMUTAÇÃO INVERSA DE UMA STRING DE 64 BITS  */
            void inverse_permute(long *op, long *ip, long *tbl, int n)
             {
              int i;
              long *pt = (long *)Pmask;

              *op = *(op+1) = 0;
              for (i = 0; i < n; i++)    {
                          if ((*ip & *pt) || (*(ip+1) & *(pt+1)))    {
                                      *op |= *tbl;
                                      *(op+1) |= *(tbl+1);
                          }
                          tbl += 2;
                           pt += 2;
               }
             }

/*          PERMUTAR UMA STRING DE 64 BITS                          */
            void permute(long *op, long *ip, long *tbl, int n)
             {
              int i;
              long *pt = (long *)Pmask;

              *op = *(op+1) = 0;
              for (i = 0; i < n; i++)    {
                          if ((*ip & *tbl) || (*(ip+1) & *(tbl+1)))   {
                                      *op |= *pt;
                                      *(op+1) |= *(pt+1);
                          }
                          tbl += 2;
                           pt += 2;
               }
            }

/*          COMPUTAÇÃO DA FUNÇÃO  f(R,K) - DEPENDENTE DA CHAVE    */
            long f(long blk, struct ks key)
             {
              struct LR ir = {0,0};
              struct LR or;

              union   {
                          struct LR f;
```

```
                        struct ks kn;
                } tr = {0,0}, kr = {0,0};
                ir.L = blk;
                kr.kn = key;
                permute(&tr.f.L, &ir.L, (long *)Etbl, 48);

                tr.f.L ^= kr.f.L;
                tr.f.R ^= kr.f.R;

                ir.L = S(tr.kn);

                permute(&or.L, &ir.L, (long *)Ptbl, 32);
                return or.L;
              }

/*            CONVERSÃO DE UM BLOCO/CHAVE DE 48 BITS PARA 32 BITS      */
              static long S(struct ks ip)
               {
                int i;
                long op = 0;
                for (i = 8; i > 0; --i)            {
                        long four = fourbits(ip,i);
                        op |= four << ((i-1) * 4);
                }
                return op;
              }

/*            EXTRAÇÃO DE 4 BITS DO BLOCO/CHAVE                        */
              static int fourbits(struct ks k, int s)
               {
                int i = sixbits(k, s);
                int row, col;
                row = ((i >> 4) & 2) | (i & 1);
                col = (i >> 1) & 0xf;
                return stbl[8-s][row][col];
              }

/*            EXTRAÇÃO DE 6 BITS DE UMA POSIÇÃO "S" DO BLOCO/CHAVE  */
              static int sixbits(struct ks k, int s)
               {
                int op = 0;
                int n = (8-s);
                int i;
                for (i = 0; i < 2; i++)    {
                        int off = ex6 [n] [i] [0];
                        unsigned char c = k.ki[off];
                        c >>= ex6[n][i][1];
                        c <<= ex6[n][i][2];
                        c &= ex6[n][i][3];
                        op |= c;
                 }
                 return op;
              }

/*  FUNÇÃO "KS" - "KEY SCHEDULE" - DO D.E.S.                */
/*            struct ks KS(int n, char *key)  */
              struct ks KS(int n)
               {
                static unsigned char cd[8];
                static int its[] = {1,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1};
                union   {
                        struct ks kn;
                        struct LR filler;
                } result;

                if (n == 0)
                        permute((long *)cd, (long *) chave, (long *)PC1tbl,64);

                rotate(cd, its[n]);
                rotate(cd+4, its[n]);

                permute(&result.filler.L, (long *)cd, (long *)PC2tbl, 48);
                return result.kn;
              }

/*            ROTAÇÃO DE UMA STRING DE 4 BYTES, "n" POSIÇÕES P/ ESQ */
              static void rotate(unsigned char *c, int n)
               {
                int i;
                unsigned j,k;
                k = *c >> (8 - n);
                for (i = 3; i >= 0; --i)    {
                        j = (*(c+1) << n) + k;
                        k = j >> 8;
                        *(c+1) = j;
                }
```

```
        }
/*      -------------------------  tables.c -------------------  */

/*      TABELAS PARA O ALGORÍTMO DO D.E.S.                       */

/*      MÁSCARAS DE PERMUTAÇÃO                                   */
        unsigned char Pmask[] = {
                    p( 1),p( 2),p( 3),p( 4),p( 5),p( 6),p( 7),p( 8),
                    p( 9),p(10),p(11),p(12),p(13),p(14),p(15),p(16),
                    p(17),p(18),p(19),p(20),p(21),p(22),p(23),p(24),
                    p(25),p(26),p(27),p(28),p(29),p(30),p(31),p(32),
                    p(33),p(34),p(35),p(36),p(37),p(38),p(39),p(40),
                    p(41),p(42),p(43),p(44),p(45),p(46),p(47),p(48),
                    p(49),p(50),p(51),p(52),p(53),p(54),p(55),p(56),
                    p(57),p(58),p(59),p(60),p(61),p(62),p(63),p(64)
        };

/*      TABELA DAS PERMUTAÇÕES INICIAL E INVERSA DA INICIAL     */
        unsigned char IPtbl[] = {
                    p(58),p(50),p(42),p(34),p(26),p(18),p(10),p( 2),
                    p(60),p(52),p(44),p(36),p(28),p(20),p(12),p( 4),
                    p(62),p(54),p(46),p(38),p(30),p(22),p(14),p( 6),
                    p(64),p(56),p(48),p(40),p(32),p(24),p(16),p( 8), /*p(18),*/
                    p(57),p(49),p(41),p(33),p(25),p(17),p( 9),p( 1),
                    p(59),p(51),p(43),p(35),p(27),p(19),p(11),p( 3),
                    p(61),p(53),p(45),p(37),p(29),p(21),p(13),p( 5),
                    p(63),p(55),p(47),p(39),p(31),p(23),p(15),p( 7)
        };

/*      TABELA DA PERMUTAÇÃO "E", PARA A FUNÇÃO "f"          */
        unsigned char Etbl[] =  {
                    p(32),p( 1),p( 2),p( 3),p( 4),p( 5),
                    p( 4),p( 5),p( 6),p( 7),p( 8),p( 9),
                    p( 8),p( 9),p(10),p(11),p(12),p(13),
                    p(12),p(13),p(14),p(15),p(16),p(17),
                    p(16),p(17),p(18),p(19),p(20),p(21),
                    p(20),p(21),p(22),p(23),p(24),p(25),
                    p(24),p(25),p(26),p(27),p(28),p(29),
                    p(28),p(29),p(30),p(31),p(32),p( 1)
        };

/*      TABELA DA PERMUTAÇÃO "P", PARA A FUNÇÃO "f"          */
        unsigned char Ptbl[] =  {
                    p(16),p( 7),p(20),p(21),p(29),p(12),p(28),p(17),
                    p( 1),p(15),p(23),p(26),p( 5),p(18),p(31),p(10),
                    p( 2),p( 8),p(24),p(14),p(32),p(27),p( 3),p( 9),
                    p(19),p(13),p(30),p( 6),p(22),p(11),p( 4),p(25)
        };

/*      TABELA PARA CONVERSÃO DE UM "STREAM" DE 6 BITS P/ 4 BITS */
        unsigned char stbl[8][4][16] =  {
                    /* -------------- s1 -------------- */
                    14,4,13,1,2,15,11,8,3,10,6,12,5,9,0,7,
                    0,15,7,4,14,2,13,1,10,6,12,11,9,5,3,8,
                    4,1,14,8,13,6,2,11,15,12,9,7,3,10,5,0,
                    15,12,8,2,4,9,1,7,5,11,3,14,10,0,6,13,
                    /* -------------- s2 -------------- */
                    15,1,8,14,6,11,3,4,9,7,2,13,12,0,5,10,
                    3,13,4,7,15,2,8,14,12,0,1,10,6,9,11,5,
                    0,14,7,11,10,4,13,1,5,8,12,6,9,3,2,15,
                    13,8,10,1,3,15,4,2,11,6,7,12,0,5,14,9,
                    /* -------------- s3 -------------- */
                    10,0,9,14,6,3,15,5,1,13,12,7,11,4,2,8,
                    13,7,0,9,3,4,6,10,2,8,5,14,12,11,15,1,
                    13,6,4,9,8,15,3,0,11,1,2,12,5,10,14,7,
                    1,10,13,0,6,9,8,7,4,15,14,3,11,5,2,12,
                    /* -------------- s4 -------------- */
                    7,13,14,3,0,6,9,10,1,2,8,5,11,12,4,15,
                    13,8,11,5,6,15,0,3,4,7,2,12,1,10,14,9,
                    10,6,9,0,12,11,7,13,15,1,3,14,5,2,8,4,
                    3,15,0,6,10,1,13,8,9,4,5,11,12,7,2,14,
                    /* -------------- s5 -------------- */
                    2,12,4,1,7,10,11,6,8,5,3,15,13,0,14,9,
                    14,11,2,12,4,7,13,1,5,0,15,10,3,9,8,6,
                    4,2,1,11,10,13,7,8,15,9,12,5,6,3,0,14,
                    11,8,12,7,1,14,2,13,6,15,0,9,10,4,5,3,
                    /* -------------- s6 -------------- */
                    12,1,10,15,9,2,6,8,0,13,3,4,14,7,5,11,
                    10,15,4,2,7,12,9,5,6,1,13,14,0,11,3,8,
                    9,14,15,5,2,8,12,3,7,0,4,10,1,13,11,6,
                    4,3,2,12,9,5,15,10,11,14,1,7,6,0,8,13,
                    /* -------------- s7 -------------- */
                    4,11,2,14,15,0,8,13,3,12,9,7,5,10,6,1,
                    13,0,11,7,4,9,1,10,14,3,5,12,2,15,8,6,
                    1,4,11,13,12,3,7,14,10,15,6,8,0,5,9,2,
                    6,11,13,8,1,4,10,7,9,5,0,15,14,2,3,12,
```

```
                /* -------------- s8 -------------- */
                13,2,8,4,6,15,11,1,10,9,3,14,5,0,12,7,
                1,15,13,8,10,3,7,4,12,5,6,11,0,14,9,2,
                7,11,4,1,9,12,14,2,0,6,10,13,15,3,5,8,
                2,1,14,7,4,10,8,13,15,12,9,0,3,5,6,11
        };

/*      ESCOLHA PERMUTADA NR. 1 P/ CÁLCULO DA "KEY SCHEDULE"     */
        unsigned char PC1tbl[] =  {
                p(57),p(49),p(41),p(33),p(25),p(17),p( 9),p( 0),
                p( 1),p(58),p(50),p(42),p(34),p(26),p(18),p( 0),
                p(10),p( 2),p(59),p(51),p(43),p(35),p(27),p( 0),
                p(19),p(11),p( 3),p(60),p(52),p(44),p(36),p( 0),
                p(63),p(55),p(47),p(39),p(31),p(23),p(15),p( 0),
                p( 7),p(62),p(54),p(46),p(38),p(30),p(22),p( 0),
                p(14),p( 6),p(61),p(53),p(45),p(37),p(29),p( 0),
                p(21),p(13),p( 5),p(28),p(20),p(12),p( 4),p( 0)
        };

/*      ESCOLHA PERMUTADA NR. 2 P/ CÁLCULO DA "KEY SCHEDULE"     */
        unsigned char PC2tbl[] =  {
                p(14),p(17),p(11),p(24),p( 1),p( 5),p( 3),p(28),
                p(15),p( 6),p(21),p(10),p(23),p(19),p(12),p( 4),
                p(26),p( 8),p(16),p( 7),p(27),p(20),p(13),p( 2),
                p(41),p(52),p(31),p(37),p(47),p(55),p(30),p(40),
                p(51),p(45),p(33),p(48),p(44),p(49),p(39),p(56),
                p(34),p(53),p(46),p(42),p(50),p(36),p(29),p(32)
        };

/*      PARA EXTRAÇÃO DE STRs. DE 6 BITS DE UMA STR. DE 64 BIT  */
        unsigned char ex6[8][2][4]   =   {
                /* byte, >>, <<, & */
                /* ---- s = 8 ----*/
                0,2,0,0x3f,
                0,2,0,0x3f,
                /* ---- s = 7 ----*/
                0,0,4,0x30,
                1,4,0,0x0f,
                /* ---- s = 6 ----*/
                1,0,2,0x3c,
                2,6,0,0x03,
                /* ---- s = 5 ----*/
                2,0,0,0x3f,
                2,0,0,0x3f,
                /* ---- s = 4 ----*/
                3,2,0,0x3f,
                3,2,0,0x3f,
                /* ---- s = 3 ----*/
                3,0,4,0x30,
                4,4,0,0x0f,
                /* ---- s = 2 ----*/
                4,0,2,0x3c,
                5,6,0,0x03,
                /* ---- s = 1 ----*/
                5,0,0,0x3f,
                5,0,0,0x3f
        };
```