Universidade Federal da Paraíba
Centro de Ciências e Tecnologia
Coordenação da Pós-Graduação em Informática

Um Serviço de Correio Eletrônico Internet Seguro e Transparente ao Usuário

Edjozane Campos Cavalcanti

Campina Grande - PB
Julho - 1996

Edjozane Campos Cavalcanti

Um Serviço de Correio Eletrônico Internet Seguro e Transparente ao Usuário

Dissertação apresentada ao Curso de MESTRADO EM INFORMÁTICA da Universidade Federal da Paraíba, em cumprimento às exigências para a obtenção do grau de Mestre.

Área de Concentração: **Redes e Sistemas Distribuídos**

Walfredo da Costa Cirne Filho
(Orientador)

Francisco Vilar Brasileiro
(Co-Orientador)

Campina Grande - PB
Julho - 1996

C376s Cavalcanti, Edjozane Campos.
Um servico de correio eletronico internet seguro e transparente ao usuario/ Edjozane Campos Cavalcanti. - Campina Grande, 1996.
87 f.

Dissertacao (Mestrado em Informatica) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Correio Eletronico. 2. Internet. 3. Dissertacao - Informatica. I. Cirne Filho, Walfredo da Costa. II. Brasileiro, Francisco Vilar. III. Universidade Federal da Paraiba - Campina Grande (PB). IV. Título

CDU 004.773.3(043)

UM SERVIÇO DE CORREIO ELETRÔNICO INTERNET SEGURO E TRANSPARENTE AO USUÁRIO

EDJOZANE CAMPOS CAVALCANTI

DISSERTAÇÃO APROVADA EM 19.07.96

Walfredo da Costa Cirne Filho

PROF. WALFREDO DA COSTA CIRNE FILHO, M.S.c
Presidente

Francisco Vilar Brasileiro

PROF. FRANCISCO VILAR BRASILEIRO, Ph.D
Examinador

PROF. MARCELO ALVES DE BARROS, Dr.
Examinador

PROF. ROGÉRIO DRUMMOND B. P. DE M. FILHO, Dr.
Examinador

CAMPINA GRANDE - PB

À minha filha, Poema.

"Os grandes eventos não são
nossas horas mais ruidosas,
mas nossos
instantes mais
silenciosos".

Nietzsche

Agradecimentos

Este trabalho foi possível graças ao apoio, fundamental, dos meus orientadores, da minha família, do meu namorado e dos meus amigos. É difícil citar nomes, sem incorrer no risco de esquecer algumas das pessoas maravilhosas que fizeram parte da minha vida durante este período. Então, gostaria de manifestar todo o meu apreço, gratidão e amor, a vocês, que estiveram bem próximos a mim (mesmo a centena de quilômetros), que estiveram dispostos a compartilhar seus conhecimentos, que me fizeram amadurecer, que me ouviram em momentos difíceis e que partilharam das minhas alegrias; mesmo que não os cite nominalmente.

Agradeço em especial ao meu orientador, Walfredo Cirne Filho, por sua orientação e por partilhar comigo sua experiência e conhecimentos técnicos.

Agradeço ao professor Francisco Brasileiro e Jacques Phillipe Sauvé pelos comentários sobre o meu trabalho e orientação.

E por fim, agradeço a meus pais, minhas irmãs, aos funcionários da COPIN (especialmente à Aninha) e do DSC, aos meus amigos distantes e sempre presentes (Beatriz, Paulo Henrique e Ruth), aos amigos que conheci durante essa fase e dos quais levarei muitas recordações (especialmente Ernandes e Vera), e ao meu namorado ("o Estrangeiro") que foi, sem dúvida, determinante para que esse trabalho viesse a ser realizado.

Lista de Figuras

Lista de Tabelas

Sumário

Resumo

O Simple Mail Transfer Protocol (SMTP) é o padrão TCP/IP que especifica como o sistema de correio eletrônico transmite uma mensagem na Internet. Ele não trata aspectos como a possibilidade de interceptação da mensagem e falsificação do remetente. Atualmente, para impedir o acesso ou manipulação da correspondência eletrônica por usuários não autorizados, é necessário utilizar ferramentas externas ao sistema de e-mail ou depender de uma complexa infra-estrutura de distribuição de chaves de criptografia. Este trabalho apresenta a especificação de um protocolo com recursos de segurança, o Secure Simple Mail Transfer Protocol (SSMTP), que pode facilmente ser integrado na aplicação de e-mail e que não depende de nenhum serviço de suporte. Além do SSMTP, que possibilita a transmissão segura de mails sobre uma rede TCP/IP, definimos o Local Mail Security Procedures (LMSP), que se constitue de um conjunto de procedimentos para a garantia da privacidade e segurança locais.

Abstract

The Simple Mail Transfer Protocol (SMTP) is the TCP/IP standard for the electronic mail transfer from one computer to another. It does not address security aspects, like catching the message or faking the sender. Today, to avoid non-authorized e-mail access or modification, one needs to use tools external to the e-mail software or to depend on a complex infrastructure for cryptography key distribution. This work specifies the Secure Simple Mail Transfer Protocol (SSMTP), which is an SMTP secure extension. It was designed to be easily integrated within the e-mail software and to run independently of any distribution key supporting service. Besides SSMTP, this work also defines the Local Mail Security Procedures (LMSP) to assure e-mail servers local security.

1

Introdução

O Correio Eletrônico ou simplesmente e-mail (Eletronic Mail) é, atualmente, o sistema digital de maior alcance para troca de mensagens, tendo se tornado um grande fator de motivação para interconexão de redes. O e-mail tem quebrado as barreiras geográficas. Não se trata apenas da troca de mensagens eletrônicas dentro das empresas: hoje a troca de mensagens eletrônicas entre instituições é corriqueira e intensa. Mais de cem milhões de e-mails atravessam as redes de computadores todos os dias [Garfinkel 95].

As vantagens da utilização do e-mail são evidenciadas por diversos aspectos. A rapidez, que permite que a sua mensagem atravesse ruas, cidades, países e seja entregue ao destino quase imediatamente. O fato do destinatário não precisar estar conectado nem precisar interromper suas atividades para receber mensagens (o que representa uma enorme vantagem sobre o telefone, que exige a disponibilidade imediata do destinatário). Além da possibilidade do receptor manipular digitalmente a mensagem recebida da forma que desejar. Existe, ainda, uma outra característica vantajosa: a brevidade. Quando escrevemos, costumamos organizar melhor as idéias, resultando numa forma de comunicação mais concisa e produtiva.

As mensagens eletrônicas podem ser compostas de texto, som, imagem, animação e até código executável, apesar do uso majoritário do e-mail ser na troca de mensagens do tipo texto. Para as mensagens do tipo texto, existe uma linguagem composta por um conjunto de símbolos (*emoticons* ou *smiles*), específica dos e-mails, que tem o objetivo de suprir a impossibilidade na comunicação escrita da inflexão da voz ou da linguagem corporal. Por exemplo, :-) é interpretado como um sorriso (para fazer sentido, deve-se olhar com a margem esquerda da folha para cima). O uso de correio eletrônico tem

se disseminado de tal forma que há até regras de decoro para a escrita das mensagens [Crispen 95].

Com todas essas características e vantagens, os sistemas de e-mail são o meio de comunicação mais prático da atualidade. O correio eletrônico tem expandido seu alcance: estamos falando de troca de mensagens entre clientes e fornecedores, parceiros de negócios, amigos. Falamos de estar conectado por e-mail, mesmo que estejamos em viagem, no trânsito, no escritório e independentemente do tipo de sistema de correio eletrônico utilizado.

Mas, os e-mails trafegam de uma rede a outra até o destino, através de canais nem sempre seguros. O maior serviço de e-mail existente, o da Internet, não oferece aos seus usuários recursos de segurança necessários à privacidade e autenticação das mensagens que encaminha, deixando a implementação de mecanismos de proteção para o próprio usuário.

Assim, em nosso trabalho, tratamos o aspecto da segurança na troca de e-mails na Internet, oferecendo uma solução à nível do protocolo Simple Mail Transfer Protocol (SMTP), utilizado no transporte de e-mails em redes TCP/IP. Além do transporte seguro, especificamos procedimentos que garantem a segurança local em uma máquina que implemente o sistema de e-mail Internet.

1.1. Objetivo

Por razões de privacidade e segurança, os usuários do correio eletrônico Internet procuram impedir o acesso ou manipulação dos seus e-mails por usuários não autorizados, através do uso de software de criptografia externo ao seu pacote de correio eletrônico. Para obter, então, a segurança desejada, além do software de correio eletrônico o usuário precisa aprender a manipular o software de criptografia; demandando tempo e a aprendizagem de comandos nem sempre amigáveis. Além do mais, o uso de um software externo ao ambiente de correio eletrônico leva à redução da produtividade.

O objetivo deste trabalho é a especificação de um ambiente de correio eletrônico Internet com recursos de segurança, que tenha um bom custo/benefício para o usuário e que seja o mais transparente possível.

1.2. Importância

O problema de segurança em sistema de computação é um tópico bastante abrangente da Informática, envolvendo desde aspectos como o acesso ou manipulação, intencional ou não, de informações confidenciais por usuários não autorizados, até a utilização não autorizada de um computador ou de seus dispositivos periféricos [Soares 95].

Atualmente, com a integração crescente de redes, que antes eram isoladas, à Internet, usuários do mundo inteiro podem ter acesso a qualquer rede interconectada, tornando os problemas de segurança mais evidentes. Os problemas mais comuns são acessos indevidos a máquinas Unix e a captura dos pacotes que trafegam pela rede. Assim, o e-mail Internet é um alvo fácil de ataque, por não implementar mecanismos de proteção às mensagens, que ficam gravadas em formato texto legível na conta do usuário e são encaminhadas sem quaisquer recursos de segurança.

Uma das formas de garantir a segurança na troca de e-mails é ter um protocolo de transferência de mensagens que as criptografe. Aqui propomos o Secure Simple Mail Transfer Protocol (SSMTP), que utiliza criptografia para tal fim. Dessa forma, temos um canal de comunicação seguro entre os agentes de troca de mensagens eletrônicas. Além do SSMTP, especificamos o Local Mail Security Procedures (LMSP), que se constituem de procedimentos locais para serem seguidos quando da confecção de software seguro de e-mail.

Acreditamos que a especificação do protocolo SSMTP e da recomendação LMSP contribuirão com esse aspecto relevante que é a segurança na troca de e-mails em redes TCP/IP, em particular na Internet.

1.3. Organização do Trabalho

Neste capítulo apresentamos características de e-mail e o aspecto da falta de segurança na troca de e-mails Internet, bem como o nosso objetivo e a importância do nosso trabalho.

Em **Correio Eletrônico**, capítulo 2, apresentamos a arquitetura geral de um sistema de e-mail e seus principais componentes e falamos, de forma geral, dos principais sistemas de e-mail utilizados, atualmente.

No capítulo 3, **Correio Eletrônico Internet**, falamos de forma específica de correio eletrônico Internet, mostrando o protocolo SMTP, especificações do formato das mensagens do RFC 822, o protocolo MIME, o protocolo POP3 e o sendmail.

No capítulo 4, **Segurança Digital**, apresentamos os principais aspectos relativos à segurança em ambientes computacionais, em especial àqueles relacionados à comunicação de dados.

Em **Segurança em E-Mail Internet**, capítulo 5, abordamos a questão da segurança em e-mail com ênfase em correio eletrônico Internet.

No capítulo 6, **SSMTP e LMSP**, apresentamos a nossa solução para segurança em e-mail Internet, mostrando a especificação de um protocolo para transferência segura de e-mail, o SSMTP, bem como procedimentos LMSP para seguridade local.

Em **ssmtpd** e **spine**, capítulo 7, fazemos algumas considerações sobre a implementação de software seguro de e-mail usando SSMTP e LMSP. E, finalmente, no capítulo 8, fazemos uma conclusão à cerca da nossa dissertação, abordando a sua contribuição e trabalhos futuros.

2

Correio Eletrônico

Um sistema de correio eletrônico é constituído de componentes que interagem para a troca de mensagens entre usuários em uma mesma máquina ou entre usuários em máquinas diferentes, ligadas em rede. Os principais componentes, a inter-relação entre eles e os principais protocolos utilizados em sistemas de e-mail estão descritos neste capítulo.

2.1. Arquitetura de um Sistema de E-Mail

As mensagens eletrônicas são estruturadas. Além do conteúdo (corpo), existem campos que constituem o cabeçalho e o envelope da mensagem. Os parâmetros necessários para o transporte do e-mail, como endereços eletrônicos do remetente e dos destinatários, são referenciados como envelope. Os parâmetros necessários à interpretação da mensagem formam o cabeçalho: são campos para identificar nome e endereço do remetente, nome e endereço do destinatário, data e hora de envio, lista de pessoas que devem receber cópias, tipo de conteúdo, forma de codificação etc.

Para possibilitar a recepção de mensagens, mesmo quando o destinatário está desconectado, existe o conceito de **caixa postal**. Caixa Postal é uma área de armazenamento, na qual as mensagens permanecem até que o usuário execute uma ação de eliminação ou de transferência para outra área. É a versão eletrônica da caixa postal do sistema de correios tradicional. Para possibilitar a localização do destinatário, o correio eletrônico utiliza o conceito de endereço. Um endereço de correio eletrônico, assim como um endereço postal, contém todas as informações necessárias para enviar uma mensagem para um destinatário.

Os sistemas de e-mail são constituídos de duas partes distintas. Uma

parte provê a interface com o ser humano, denominada Agente Usuário (UA)[1] e outra provê o envio da mensagem para o destinatário, o Agente de Transferência de Mensagem (MTA)[1]. Ao conjunto de MTAs chamamos de Sistema de Transferência de Mensagens (STM) (veja **Fig. 2.1**).

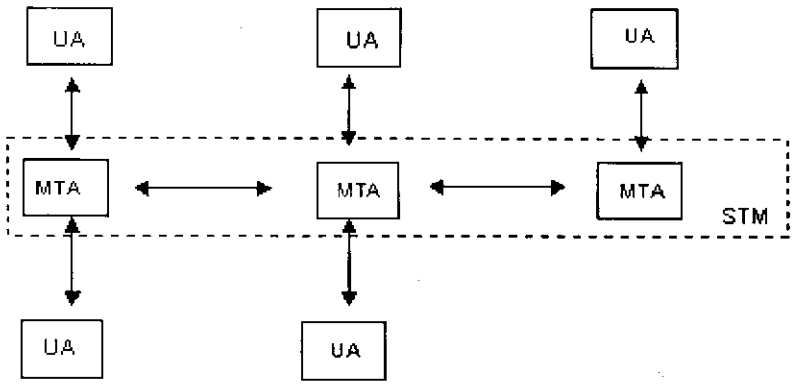


Figura 2.1. Modelo Geral de um Sistema de Correio Eletrônico

O UA permite ao usuário confeccionar, enviar e receber mensagens, bem como manipular a sua caixa postal. Existe uma grande variedade de programas disponíveis, com os mais diversos recursos. Como não há uma padronização para os UAs, os projetistas de software implementam diferentes recursos em seus produtos: interfaces amigáveis, regras para o gerenciamento das mensagens recebidas, agenda de endereços, corretor ortográfico, entre outros. Note que a comunicação entre UA e o MTA é um aspecto local, onde cabe ao UA traduzir os desejos do usuário para a linguagem do MTA. O UA interage com o sistema de transferência de mensagens, para receber e entregar os e-mails.

O MTA é o programa que encaminha as mensagens localmente ou entre máquinas; neste último caso, usando um protocolo de rede. Quando uma mensagem chega ao MTA é verificada a validade de sua sintaxe, se a mensagem for inválida é devolvida ao usuário com uma explicação acrescentada à mensagem. Por outro lado, se a mensagem é válida, é verificado se o destinatário é local e se for, a mensagem é colocada em sua

[1] Não traduzimos as siglas UA (User Agent) e MTA (Message Transfer Agent) para evitarmos confusões entre MTA e ATM (Assynchronous Transfer Mode), que normalmente não é traduzida.

caixa postal, caso contrário é encaminhada a um outro MTA. O MTA é o equivalente eletrônico de agência de correios no sistema tradicional.

Normalmente, os protocolos de transferência de mensagens são padronizados, permitindo que MTAs diferentes troquem mensagens. Dentre os protocolos de transferência de mensagens mais utilizados estão o X.400, o UUCP e o SMTP, que trataremos nas seções seguintes.

2.2. Protocolos utilizados na troca de E-Mails

2.2.1. X.400

O X.400 consiste nas recomendações do ITU-T[2], que definem o Message Handling System (MHS). O MHS especifica um sistema de tratamento de mensagens e pode ser usado para troca de mensagens interpessoais[3] ou documentos comerciais[4] [Soares 95].

O objetivo do padrão X.400 é, através da identificação dos principais elementos de um sistema de correio eletrônico, bem como de suas funcionalidades, fazer com que a interconexão de sistemas de e-mail seja tão transparente quanto as interligações das centrais telefônicas.

Para isso foram especificadas recomendações, que cobrem os vários aspectos de um sistema de tratamento de mensagens [Medeiros Neto 94], como: a especificação do modelo para sistemas de tratamento de mensagens e os seus elementos de serviços; especificação dos serviços e sua classificação em básicos, essenciais e opcionais; especificação das regras de conversão para codificação das mensagens; especificação da sintaxe de transferência das mensagens através de uma notação padrão; especificação das operações remotas e dos serviços de transferência confiáveis; especificação dos protocolos P1, P2 e P3, que são, respectivamente, os protocolos de Serviço Básico de Transferência de Mensagens, Serviço de

---

[2] International Telecommunications Union - Telecommunications

[3] Serviço de Mensagens Interpessoal (IP Service - Inter-Personal Service)

[4] Serviço Básico de Transferência de Mensagens (MT Service - Message Transfer Service)

Transferência de Mensagens Interpessoais e de Entrega e Submissão de Mensagens Interpessoais.

Nas recomendações X.400 estão também especificados recursos de segurança para o tratamento de mensagens, como: confidencialidade, autenticação, integridade e reconhecimento. Falaremos destes serviços no capítulo 5, seção 5.1.

2.2.1.1. O MHS

No MHS, as mensagens eletrônicas são constituídas de envelope e conteúdo. Se a mensagem for interpessoal, o conteúdo compreende a informação a ser transmitida (corpo) e o cabeçalho. A informação pode ser constituída de várias partes e pode conter dados de formatos diferentes[5] e o cabeçalho é constituído de campos com dados sobre o remetente, o destinatário, o assunto etc. Na troca de documentos comerciais, o conteúdo é uma unidade de dados em um sistema EDI[6] [Soares 95].

O MHS é composto de um conjunto de MTAs e vários UAs, onde cada UA é conectado a um único MTA [Soares 95] (veja **Fig. 2.2**). Ao contrário de outros sistemas de e-mail, quem armazena as mensagens recebidas é o UA, não o MTA. Isto gerava uma limitação para execução de UAs remotos, que ficam desligados em determinados momentos. Assim, em 1988, a entidade funcional Message Store (MS) foi adicionada às recomendações X.400, para garantir que computadores pessoais, que não ficam ligados durante todo o tempo, possam ter a funcionalidade de um UA remoto.

Os UAs são organizados de acordo com o tipo de dado que podem tratar: documentos comerciais (EDI), documentos interpessoais, arquivos binários etc. e são agrupados em classes diferentes[7]. UAs que pertencem a uma mesma classe são denominados cooperativos. Dessa forma, UAs utilizados para confecção de documentos interpessoais são diferentes de UAs usados

---

[5] O X.400 manipula texto, banco de dados, gráficos, fotografias digitalizadas, transações financeiras e voz.

[6] Eletronic Data Interchange, corresponde à transferência, entre computadores, de transações de negócios formatadas.

[7] Isso não impede que um UA esteja associado a mais de uma classe.

para receber documentos EDI, pois estes últimos recebem dados codificados que serão tratados por uma aplicação. A identificação da classe a qual os UAs pertencem é obtida através dos MTAs.

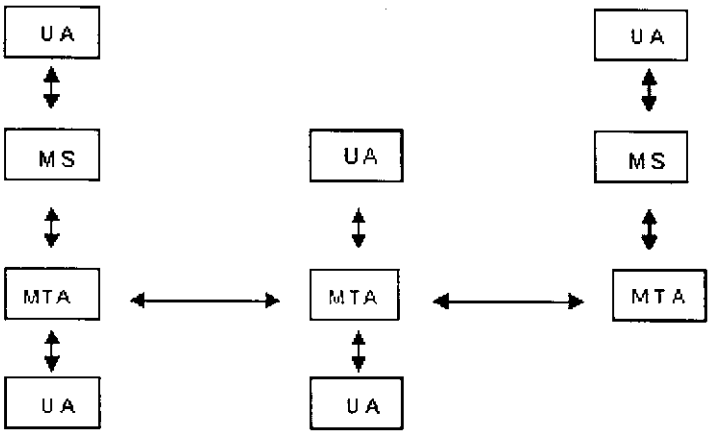


Figura 2.2. Modelo Funcional de um MHS

Os MTAs encarregam-se de prover a transferência das mensagens para os UAs aos quais estão diretamente conectados ou para outros MTAs, nos casos em que não há conexão direta com o UA destinatário. Os endereços dos usuários, que fazem parte do envelope da mensagem, são utilizados para o encaminhamento feito pelos MTAs.

Uma forma de endereçamento hierárquico é utilizada para identificar os usuários. O usuário é denominado Originador/Recipiente (O/R) e pode ser um usuário individual ou uma lista de distribuição [Malamud 92] [RFC 1685]. A cada nome é associado uma lista de atributos [Medeiros Neto 94]:

- **Atributos Geográficos**: Nome do país, nome do domínio.
- **Atributos Complementares**: Nome, sobrenome, organização, unidade organizacional (divisão, seção, setor) etc.
- **Endereço de Rede**: Especifica um terminal conectado a uma rede pública.

No esquema de interconexão internacional do MHS, os MTAs são interconectados e organizados em domínios. O conjunto de um ou mais MTAs, zero ou mais MSs, e zero ou mais UAs gerenciados por uma organização constituem um domínio. Existem dois níveis de domínios: o Administrative Management Domain (ADMD), geralmente administrado por organizações que

fornecem serviços de transmissão ao público e o Private Management Domain (PRMD), gerenciado por corporações usuárias [Soares 95].

2.2.2. UUCP

O Unix-to-Unix CoPy foi projetado e implementado na Bell Laboratories, em 1976. Ele surgiu da necessidade de um meio para se transferir arquivos de um sistema Unix para outro.

O UUCP é um conjunto de programas para transferência de arquivos entre sistemas Unix (**uucp**), execução de comandos em um sistema remoto (**uux**) e recepção de e-mails de um sistema remoto (**rmail**). Um sistema que utiliza UUCP suporta pelo menos essas três funcionalidades, mas diferentes implementações podem ter programas adicionais [O'Reilly 92].

O UUCP trabalha sobre ligações ponto a ponto, tais como linhas telefônicas ou cabos, ou sobre redes, como Ethernet, no modo *store-and-forward*. O crescimento das redes com o UUCP como protocolo básico ocorreu muito rapidamente, porque praticamente quase todas as universidades ocidentais possuíam um sistema Unix e modems, únicos itens necessários para conexão UUCP [Tanenbaum 94].

O correio eletrônico é enviado via UUCP como demonstrado na **Fig. 2.3**, abaixo:

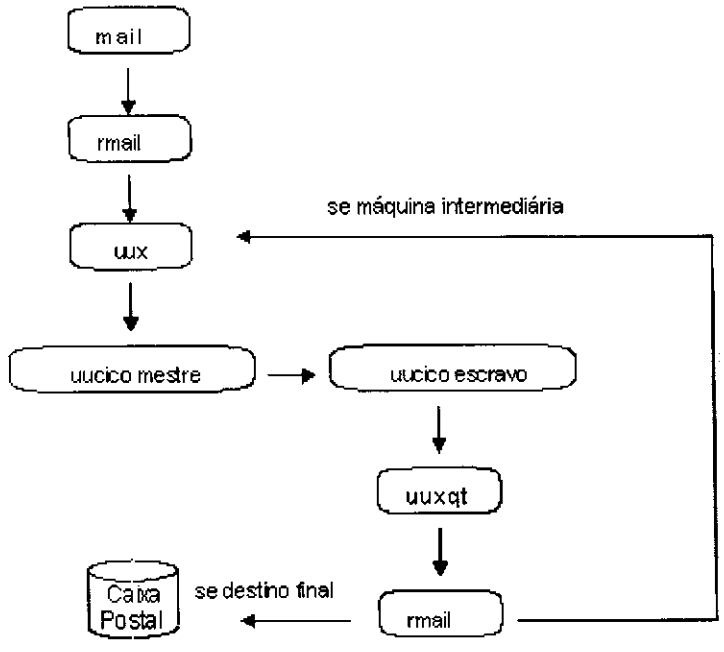


Figura 2.3. Forma como o e-mail é enviado via UUCP

De acordo com [Costales 95]: primeiro utiliza-se o **mail**[8] para compor e enviar a mensagem. Em seguida, um programa despachador vai examinar a linha "To:" do cabeçalho da mensagem. O despachador nativo do UUCP é o programa **rmail**[9]. Assim, o **mail** passa a mensagem para o **rmail**. O **rmail** checa a linha "To:"[10] e chama o **uux**. O **uux** enfileira a requisição a ser encaminhada para o sistema remoto. Para fazer isso, o **rmail** roda o comando **uux**, da seguinte forma:

```
uux - maquina1!rmail (maquina2!usuario)
```

A informação `maquina1!rmail` indica que o **rmail** será executado na máquina do sistema vizinho e o conteúdo entre parênteses é o argumento que especifica qual o endereço eletrônico do destino, do ponto de vista da `maquina1`. O hífen indica que é para o **uux** ler a mensagem da entrada padrão e enviar para o **rmail** na máquina remota. A entrada padrão contém a mensagem.

Após ler a mensagem o **uux** a coloca na fila de *spool* do UUCP, criando três arquivos de trabalho no diretório de *spool* do sistema. O primeiro arquivo contém o corpo da mensagem a ser transmitida. O segundo arquivo, contém a descrição da ação a ser tomada com o primeiro arquivo pelo programa **uuxqt**, na máquina remota. O terceiro arquivo contém instruções de como os dois primeiros arquivos devem ser transmitidos à máquina vizinha. Periodicamente, o *daemon* **uucico** será ativado para fazer a transferência dos arquivos que estiverem na fila da área de *spool*. Ele entra em contato com o sistema remoto e procura pelo arquivo que especifica qual dispositivo utilizar, número para discar etc., e estabelece a chamada. O sistema remoto responde e inicializa o processo escravo **uucico**. O **uucico** aceita os arquivos sendo enviados e armazena no diretório de *spool* do UUCP, após o programa **uucico** mestre encerrar suas transmissões, o **uucico** escravo procura pelos arquivos executáveis (o terceiro arquivo criado pelo **uux**). Para cada arquivo executável encontrado, ele roda o programa **uuxqt** para processar as informações.

---

[8] O mail é um UA do Unix.

[9] Entretanto, o despachador mais comumente utilizado em ambientes Unix é o sendmail, que chama o rmail quando necessário. Trataremos do sendmail na seção 3.5.

[10] A linha To: especifica a máquina e o usuário destinatário da mensagem.

O programa **uuxqt** executa as requisições do **uux**. Ele move os arquivos para o diretório /usr/spool/uucp/.Xqtdir e roda o **rmail**, como especificado pelo **uux**, com o argumento passado entre parênteses (que indica o destino da mensagem) e com o conteúdo da mensagem. Se esta ainda não for a máquina destino, o **rmail** executará o mesmo processo descrito até então. Caso contrário, o e-mail chegou ao destino. Neste caso, o **rmail** coloca a mensagem na caixa postal do destinatário, especificado no segundo arquivo gerado pelo **uux**.

O endereçamento no UUCP é sistema!usuário, e se uma mensagem tem que passar por vários sistemas, eles fazem parte do endereço [RFC 976]. Por exemplo:

```
% mail anjinho!maria          ← destino
Este é um mail de teste.      ← texto a ser enviado
^D                            ← enviando
```

Envia uma mensagem para `maria` na máquina `anjinho` que está diretamente conectada à máquina do remetente. O corpo da mensagem é composto de uma linha de texto e o ^D envia a mensagem (para cancelar seria ^C). Se, entretanto, `maria` está em outro sistema, acessado via `anjinho`, teremos algo do tipo:

```
% mail anjinho!monica!maria
```

Neste caso, o e-mail é enviado da máquina do remetente para máquina `anjinho`, daí para `monica`, onde é finalmente o e-mail é entregue ao destinatário.

Mas, ao longo do tempo, com o aumento das conexões, era possível encontrar usuários digitando endereços como:

```
% mail anjinho!monica!magali!cebolinha!pelezinho!zane
```

A partir das explicações acima, podemos detectar duas desvantagens no endereçamento UUCP: a forma de endereçar a mensagem pode ser muito longa e o endereço do destinatário é dependente da máquina em que o usuário remetente está enviando o e-mail.

O UUCP é, após o Simple Mail Transfer Protocol (SMTP), o protocolo mais comumente utilizado para o transporte de e-mail [Avolio 94], mas devido

ao crescimento da Internet, está caindo em desuso.

2.2.3. SMTP

O SMTP é o padrão que especifica como o sistema de correio eletrônico transmite uma mensagem entre máquinas ligadas via TCP/IP (**Fig. 2.4**).

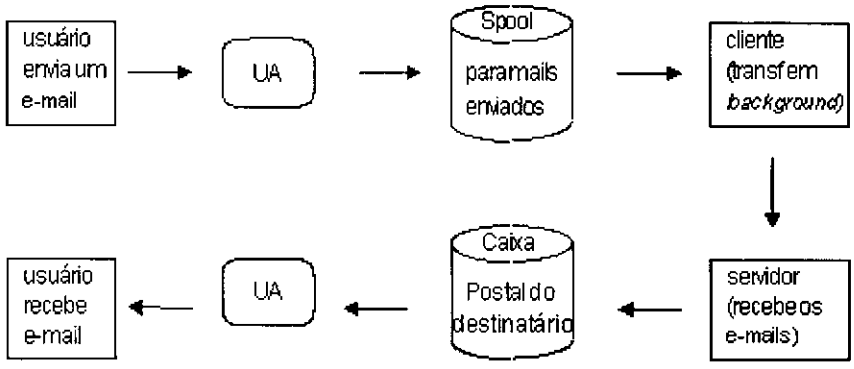


Figura 2.4. Arquitetura do Sistema de E-Mail SMTP

Um usuário utiliza um UA para confeccionar sua mensagem e solicita ao sistema de correio eletrônico que encaminhe o e-mail ao destinatário. O processo de transferência de mensagem é executado em *background*, estabelecendo uma conexão de transporte com o servidor de correio eletrônico da máquina destino. Estabelecida a comunicação, a transação para a transmissão dos e-mails é iniciada e uma vez que o destinatário é identificado, o e-mail é armazenado em sua caixa postal. O usuário destinatário também utilizará um UA para manipular seus e-mails.

O SMTP será explicado em detalhes na seção 3.1.

2.3. Resumo

Neste capítulo, fizemos uma introdução aos sistemas de correio eletrônico, abordando a arquitetura de um sistema de e-mail e seus componentes: o agente usuário e o agente de transferência de mensagens. Apresentamos também os protocolos mais comumente utilizados na troca de mensagens eletrônicas, fazendo uma introdução ao X.400, UUCP e SMTP.

3

Correio Eletrônico Internet

A Internet é notoriamente conhecida como a maior rede de computadores do mundo [Comer 95]. Ela fornece uma variada gama de serviços a seus usuários, sendo que o Correio Eletrônico é um dos mais antigos e também dos mais populares. Na verdade, a Internet é o maior sistema de troca eletrônica de mensagens do mundo [Burns 95] [Garfinkel 95].

A Internet está baseada no conjunto de protocolos TCP/IP. No TCP/IP, a transferência de correio eletrônico entre máquinas distintas se dá através do protocolo Simple Mail Transfer Protocol (SMTP). Os MTAs SMTP armazenam as mensagens recebidas diretamente na caixa postal do destinatário, não envolvendo o UA nesta atividade. Obviamente nem todas as máquinas conectadas à Internet rodam um MTA SMTP, pois para tal é necessário permanecer ligado ininterruptamente, além de possuir capacidade de processamento e armazenamento adequadas a esta tarefa. As máquinas que rodam os MTAs SMTP são coloquialmente conhecidas como servidoras de e-mail.

Qualquer máquina na Internet pode executar um UA e, desta forma, possibilitar que seus usuários recebam, manipulem e enviem mensagens. Note que todo UA precisa de um MTA para enviar e receber e-mails. Caso o UA possa ter acesso aos mesmos arquivos que o MTA (por exemplo, ambos rodam na mesma máquina), a comunicação entre eles se dá pelo acesso direto aos arquivos de e-mail (no caso, a Caixa Postal e a Área de Enfileiramento de Mensagens para Transmissão, discutidos mais adiante). Entretanto, se o UA não compartilha arquivos com o MTA, a comunicação entre os dois é feita através de um protocolo, sendo o Post Office Protocol versão 3 (POP3) o mais comumente utilizado para este fim.

A grande maioria dos servidores de e-mail Internet são máquinas Unix [Avolio 94]. Elas utilizam uma série de programas para obter a função de MTA. A configuração mais popular consiste de utilizar o programa **popper** para prover a comunicação POP3 com os UAs remotos e o **sendmail** para troca de mensagens com outros servidores de e-mail (via SMTP).

3.1. SMTP

O SMTP é o protocolo utilizado pelo TCP/IP para troca de e-mails. Ele define como o sistema de transferência de mensagem envia um e-mail entre máquinas [Comer 95].

A estrutura de um *host* que implementa o SMTP inclui as caixas postais dos usuários, uma ou mais áreas para enfileiramento de mensagens em trânsito e um ou mais processos rodando em *background* (*daemons*) para entrega e recebimento dos e-mails [RFC 1123]:

- Caixa Postal - É uma área de armazenamento, na qual as mensagens permanecem até que o usuário execute uma ação, sobre a mensagem, de eliminação ou transferência para outra área. É uma versão eletrônica da caixa postal do sistema de correios tradicional.
- Área de Enfileiramento das Mensagens - É uma área do sistema de arquivos, onde as mensagens são armazenadas para futura transmissão.
- Agente Usuário - O UA é o software de correio eletrônico que permite ao usuário confeccionar, enviar, receber e ler mensagens, bem como manipular a caixa postal.
- Agente de Transferência de Mensagens - O MTA é o programa que coloca a mensagem diretamente na caixa postal, quando o UA de destino está ligado a ele, ou a encaminha para a máquina de destino, usando o SMTP. Durante o diálogo entre duas máquinas que implementam SMTP, a máquina de origem age como cliente SMTP, e a máquina de destino age como servidor SMTP aceitando as mensagens e as colocando na caixa postal do destinatário.

O MTA SMTP encarrega-se de extrair do envelope do e-mail as

informações necessárias para transferência da mensagem para a máquina remota e de abrir uma conexão TCP em uma porta bem-conhecida (no caso, a porta 25) [Malamud 92]. Neste ponto temos um diálogo entre o cliente e o servidor. No diálogo é encaminhada a mensagem, que consiste de cabeçalho e conteúdo, e os endereços de origem e destino, referenciados como envelope [RFC 1123].

A **Fig. 3.1** mostra a comunicação entre dois sistemas de e-mail utilizando SMTP. As linhas iniciadas por S são geradas pelo servidor e as começadas por C, pelo cliente.

```
Solicitação de conexão TCP
S: 220 fenix.cgsoft.softex.br Simple Mail Transfer Service Ready
C: HELO dsc.ufpb.br
S: 250 fenix.cgsoft.softex.br
C: MAIL From: <maria@dsc.ufpb.br>
S: 250 ok
C: RCPT To: <joao@cgsoft.softex.br>
S: 250 ok
C: DATA
S: 354 Enter mail, end with .
C: Date: Sun, 28 Jan 1996 23:05:00 -0200 (EDT)
C: From: Maria Silva <maria@dsc.ufpb.br>
C: To: joao@cgsoft.softex.br
C: Subject: Reuniao
C:
C: Nossa reuniao esta confirmada para hoje (28/01), as 15:00.
C: Maria
C: .
S: 250 ok
C: QUIT
S: 221 fenix.cgsoft.softex.br Service closing transmission channel
Liberação da conexão TCP
```

Figura 3.1. Exemplo de diálogo cliente/servidor SMTP

3.1.1. Procedimentos SMTP

Os procedimentos para transação de e-mails, comandos e respostas, são definidos em [RFC 821] e [RFC 1123], que especificam o SMTP. Como demonstrado no exemplo acima, a comunicação entre o cliente e o servidor é um diálogo, controlado pelo cliente. O cliente envia um comando e recebe uma resposta.

O comando HELO é usado para que o cliente SMTP identifique-se para o servidor. O argumento contém o nome do host cliente. Este comando e a resposta Ok confirmam que cliente e servidor estão no estágio inicial do diálogo, mas ainda não há uma transação em progresso [RFC 821]. Os endereços do envelope são derivados das informações no cabeçalho da mensagem; a parte que constitui o envelope é enviada com os comandos MAIL

e RCPT. O comando MAIL é usado para iniciar a transação e o comando RCPT para identificar o destinatário do e-mail, como na **Fig. 3.1**; serão enviados tantos RCPTs quantos forem os destinatários [RFC 1123].

As linhas seguintes ao comando DATA são tratadas como o conteúdo da mensagem e o final da mensagem é identificado por uma linha contendo apenas um ponto, que é uma seqüência de caracteres <CRLF>.<CRLF>. Ao aceitar a mensagem o receptor SMTP inclui uma linha Received, com data e a hora no início da mensagem, esta linha contém o endereço do host cliente e do host servidor. O comando QUIT especifica que o servidor deve enviar uma resposta Ok e fechar a conexão [RFC 821].

Cada comando SMTP gera uma resposta. As respostas consistem de uma cadeia de caracteres. A cadeia de caracteres é composta de um número de três dígitos, seguido por uma mensagem. Os números determinam o estado da transação e as mensagens são comentários.

As respostas garantem a sincronização de solicitações e ações na transação de transferência do e-mail [Medeiros Neto 94]. O primeiro caractere da cadeia de resposta indica se a resposta é boa, ruim ou incompleta. Existem cinco valores possíveis:

- **1** indica que o comando foi aceito, mas requer a confirmação da informação contida na resposta.
- **2** indica que a solicitação foi aceita e completada com sucesso.
- **3** indica que a solicitação foi aceita e que aguarda mais informações para executar a ação requisitada.
- **4** indica que ocorreu uma situação de erro e que a ação não será executada, mas que pode ser requerida outra vez.
- **5** indica que ocorreu um erro e que não se deve requerer novamente a mesma ação.

O segundo caractere do string de resposta codifica a informação em categorias específicas, existem 4 valores possíveis:

- **0** indica erro de sintaxe, de comandos não especificados ou de comandos supérfluos.
- **1** é uma resposta que contém informações de *status* ou *help*.

- **2** é uma resposta referente à conexão.
- **5** indica o status de uma transferência ou ação requisitada.

O terceiro caractere é dependente do segundo, sendo um refinamento do significado de cada categoria.

A **Tab. 3.1**, que segue abaixo, mostra as respostas definidas pelo SMTP.

221	*status* do sistema ou *help*
214	mensagem de *help*
220	serviço pronto
221	serviço fechando o canal de transmissão
250	requisição de ação bem sucedida
251	usuário não é local, encaminhe para <forward-path>
354	inicie a entrada do conteúdo do e-mail, termine com ponto (.)
421	serviço não disponível
450	caixa postal não disponível (ex: área de caixa postal cheia)
451	requisição de ação abortada
452	memória insuficiente, requisição não completada
500	erro de sintaxe, comando não reconhecido
501	erro de sintaxe nos parâmetros ou argumentos
502	comando não implementado
503	seqüência incorreta de comandos
504	parâmetro de comando não implementado
550	caixa postal não existe
551	usuário não é local, tente <forward-path>
552	requisição de ação abortada, excedeu alocação de memória
553	requisição de ação não completada, nome de caixa postal incorreto
554	transação falhou

Tabela 3.1. Códigos de respostas SMTP e respectivas mensagens

Além dos comandos já citados, o SMTP especifica também os seguintes comandos: VRFY, EXPN, SEND, SOML, SAML, TURN, RSET, NOOP e HELP. Os comandos SMTP HELO, MAIL, RCPT, DATA, RSET, VRFY, NOOP e QUIT são mandatórios. Os comandos SEND, SOML, SAML, EXPN, HELP e TURN

especiais (aspas, parênteses, sinais maior que e menor que, colchetes e ponto-e-vírgula) são considerados símbolos delimitadores e palavras (átomos) são delimitadas por estes símbolos ou por espaço em branco.

Por exemplo, no preenchimento do campo de um endereço, onde encontramos usuário@dsc.ufpb.br: usuário, dsc, ufpb e br são átomos. @ e ponto (.) são caracteres especiais. Não há espaços entre um átomo e o ponto ou entre um átomo e o @. Um comentário pode ser colocado no campo de endereço, entre parênteses. Mas ele é ignorado, servindo apenas para facilitar o entendimento humano. Uma outra forma de colocar um comentário é escrever o endereço entre os sinais < e >, delimitando-o. Neste caso, tudo que ficar fora do < e > é considerado comentário.

O endereçamento de mensagens segue o padrão Internet, definido no [RFC 822] e tem duas partes separadas pelo sinal @, onde a sintaxe é: usuário@destino. A parte usuário do endereço é resolvida localmente. O nome destino, que identifica o domínio ao qual a máquina receptora pertence ou é a própria máquina, é identificado através do Domain Name System (DNS). O DNS é o resolvedor de nomes de máquinas numa rede TCP/IP. Uma de suas funções é especificar quais as máquinas receptoras de e-mail para um domínio. Isto é feito através da criação do registro Mail Exchanger (MX) para o domínio. O programa despachador de e-mails, procura pelo registro MX para identificar para onde o e-mail deve ser encaminhado. Se o registro MX não estiver definido, o DNS procura por o registro Address (A) do destino. O registro A contém o endereço de uma máquina, i.e., o destino é uma máquina e este registro contém um dos seus endereços.

Os nomes DNS formam um conjunto de nomes registrados, que obedecem a um esquema de endereçamento hierárquico, onde o topo é o nome mais à direita. Então, em usuário@registro.organização, organização é uma unidade lógica, que é o topo de uma hierarquia, e o DNS define um mecanismo universal para encontrar organização, que sabe também como encontrar registro.

3.1.3. Problemas do SMTP

O SMTP não apresenta suporte a mensagens que não sejam compostas

exclusivamente de caracteres ASCII imprimíveis de 7 bits. O [RFC 822], comentado na seção 3.1.2, especifica detalhes sobre os campos do cabeçalho da mensagem, mas descreve o conteúdo da mensagem apenas como linhas de texto. A solução hoje existente para a falta de suporte para mensagens não-texto é o Multipurpose Internet Mail Extensions (MIME), definido em [RFC 1521] (veja seção 3.2).

Um outro aspecto que não é tratado pelo SMTP é a possibilidade de interceptação da mensagem e a falsificação do remetente. Uma das soluções para a privacidade da comunicação e certificação do remetente é o uso de softwares de criptografia, externos à aplicação de correio eletrônico do usuário. O software atualmente mais utilizado para este objetivo é o Pretty Good Privacy (PGP). Detalharemos o funcionamento do PGP na seção 5.3.1.

3.2. MIME

Como já foi dito, o SMTP não suporta o transporte de dados que não estejam representados em caracteres ASCII imprimíveis de 7 bits, dessa forma, para o transporte de outros tipos de dados via e-mail foi definido o MIME.

O MIME é um mecanismo para especificar e descrever o formato dos conteúdos de mensagens na Internet [RFC 1521], permitindo que informações que não estão representadas em caracteres ASCII imprimíveis, como sons, imagem, vídeo, possam fazer parte do corpo das mensagens. Ou seja, o MIME tem o objetivo de resolver a limitação dos caracteres do corpo das mensagens SMTP terem apenas caracteres ASCII imprimíveis de 7 bits[1].

Para manter a compatibilidade, é necessário que os softwares MIME convivam com os não MIME. Assim, não foi simplesmente especificado que o corpo das mensagens passariam a ser compostos de caracteres de 8 bits. O MIME define 5 formas de codificação para transmissão de dados, através do campo de cabeçalho "Content-Transfer-Encoding:". Estes métodos são o

[1] Mensagens com corpo formado por caracteres de 7 bits estão limitadas a texto americano. Nada de caracteres acentuados, muito menos mídias sofisticadas (como vídeo, áudio, imagem), nem tampouco binários.

base64, quoted-printable, 7bit, 8bit e binary, que descrevemos a seguir.

O método 7bit é a forma original do SMTP. Apenas caracteres ASCII imprimíveis de 7 bits são aceitos. E é a forma de codificação para transmissão *default*.

No método base64 só caracteres de 7 bits são transmitidos. Todavia, uma convenção é usada para que quaisquer dados sejam transformados em dados de 7 bits. Na verdade, cada 3 bytes são convertidos em 4 caracteres americanos (i. e., com o código ASCII entre 32 e 127). Vejamos: no processo de representação com 24 bits de entrada, nós temos 3 grupos de 8 bits, que serão transformados em 4 grupos de 6 bits. Cada grupo de 6 bits é utilizado como um índice para um caractere imprimível, da tabela de codificação, chamada de alfabeto base64 (**Tab. 3.2**). O caractere referenciado para a representação dos 6 bits é o caractere de saída. Os algoritmos de codificação base 64 são bastante simples [RFC 1521], mas há um aumento de 33% em relação ao tamanho original.

Valor	Codificação	Valor	Codificação	Valor	Codificação	Valor	Codificação
000000	A	010001	R	100010	i	110011	z
000001	B	010010	S	100011	j	110100	0
000010	C	010011	T	100100	k	110101	1
000011	D	010100	U	100101	l	110110	2
000100	E	010101	V	100110	m	110111	3
000101	F	010110	W	100111	n	111000	4
000110	G	010111	X	101000	o	111001	5
000111	H	011000	Y	101001	p	111010	6
001000	I	011001	Z	101010	q	111011	7
001001	J	011010	a	101011	r	111100	8
001010	K	011011	b	101100	s	111101	9
001011	L	011100	c	101101	t	111110	+
001100	M	011101	d	101110	u	111111	/
001101	N	011110	e	101111	v	(pad)	=
001110	O	011111	f	110000	w	-	
001111	P	100000	g	110001	x	-	
010000	Q	100001	h	110010	y	-	

Tabela 3.2. Alfabeto base64

No método quoted-printable, só caracteres imprimíveis de 7 bits são transmitidos. Entretanto, é definida uma convenção para codificar os caracteres que normalmente não poderiam ser transmitidos. Qualquer caractere não-transmissível, é substituído por igual (=) seguido o código hexadecimal (i. e., dois dígitos hexadecimais) do caractere em questão. Para evitar ambigüidade, o caractere igual também será transmitido segundo a convenção quoted-printable. Assim sendo, se o caractere igual ocorrer no texto, será transmitido "=3D", onde 3D é código ASCII do igual. Desta forma, o igual acaba funcionando como um caractere de escape, nunca sendo interpretado, sinalizando apenas que uma caractere não-tramissível foi codificado. Note que cada caractere codificado usa três caracteres para ser representado, o que representa um grande *overhead*. Por isso, o uso desta forma de transmissão só é vantajoso se os caracteres que não-transmissíveis

representarem uma pequena fração da mensagem. Isso explica porque o quoted-printable é usado para transferir texto acentuado, nunca para binários.

No 8bit, caracteres de 8 bits são permitidos, entretanto as linhas não podem ultrapassar 1000 caracteres. Há a necessidade de que todos os softwares que manipulam a mensagem da origem até o destino (sejam eles UAs, MTAs, servidores de lista ou *getways* de e-mail) sejam MIME-compatíveis.

No binary, não há qualquer restrição ao corpo do e-mail. Caracteres de 8 bits são permitidos e não há limite de tamanho da linha. Assim como 8bit, todos os softwares que manipulam a mensagem da origem até o destino precisam ser MIME-compatíveis.

Assim, podemos perceber, que o campo "Content-Transfer-Encoding:" indica qual o tipo de transformação que foi efetuada na representação dos dados que fazem parte do corpo da mensagem.

As informações MIME são acrescentadas ao cabeçalho definido no [RFC 822]. Veja abaixo, na **Fig. 3.3**, o exemplo de um cabeçalho MIME:

```
Date: Sun, 28 Jan 1996 23:05:00 -0200 (EDT)
From: Edjozane Campos Cavalcanti <zane@dsc.ufpb.br>
To: todos@paqtc.rpp.br
MIME.Version: 1.0
Content-Type: image/gif
Content-Transfer-Enconding: base64
```

Figura 3.3. Exemplo de um cabeçalho com informações MIME

Estas informações especificam a versão do MIME, o tipo de dado sendo enviado e a codificação utilizada. Elas são geradas pelo UA do remetente, informando como foi codificada a mensagem. O UA do destinatário, que deve implementar a mesma especificação, utiliza estas mesmas informações para decodificação e apresentação da mensagem.

A presença do campo "MIME.Version:" no cabeçalho de uma mensagem, indica que ela está de acordo com as especificações MIME. Atualmente, este campo é preenchido com o valor "1.0".

No campo "Content-Type:" é especificado o tipo do conteúdo da mensagem para que o UA possa selecionar o mecanismo adequado à sua exibição ao usuário. Para isso, ele fornece indicadores de tipo e sub-tipo. O tipo declara o formato geral do dado e o sub-tipo o formato específico para um

tipo declarado. No [RFC 1521] estão definidos sete tipos com seus sub-tipos válidos (veja na **Tab. 3.3**, os tipos descritos no [RFC 1521]).

Text	Texto, nenhum software específico é necessário à sua exibição
Image	Fotografia ou imagem gerada por computador. Necessita de um dispositivo de exibição adequado
Audio	Áudio e voz. Necessita de um mecanismo para reprodução de som
Video	Vídeos. Precisa de um mecanismo capaz de exibir dados em movimento
Application	Tipicamente dados binários
Multipart	Mensagem composta por vários tipos independentes de dados
Message	Uma mensagem padrão RFC 822

Tabela 3.3. Tipos de dados que podem ser declarados no campo Content-Type

O MIME, através do tipo Multipart, permite ainda que mensagens sejam compostas de várias partes, introduzindo o conceito de anexo. O MIME permite dizer quais são os componentes e seus tipos.

Um ponto importante a falar de como o MIME codifica as mensagens é que não está definido pelo protocolo qual das alternativas usar em quais casos. Assim sendo, se for utilizado os tipos binary ou 8 bits e algum software entre a origem e o destino não os suportar, a mensagem será corrompida.

Com o protocolo MIME é possível incluir de forma padronizada tipos arbitrários de dados em mensagens compatíveis com o formato definido pelo [RFC 822], uma vez que, usando as codificações quoted-printable ou base64, nenhuma restrição imposta pelos [RFC 821] e [RFC 822] é transgredida.

3.3. POP3

Em certos tipos de nós da Internet é impraticável manter um MTA continuamente rodando ou residente, seja por falta de recursos computacionais suficientes ou pelo fato do nó não permanecer continuamente ligado. Quando um destes nós roda um UA, é necessário acessar um MTA remoto.

Em particular, máquinas que fazem acesso discado, via uma conexão

SLIP ou PPP, não estão conectadas ininterruptamente à rede e, portanto, dependem obrigatoriamente do acesso remoto a um servidor de e-mail, para que possam rodar um UA e recuperar a sua correspondência.

É também normal que os usuários corporativos não utilizem sempre as mesmas máquinas, mas sim as diversas máquinas de um departamento (ou divisão etc.), para executarem os UAs, existindo assim a necessidade de que as mensagens eletrônicas estejam armazenados em um servidor de e-mail, rodando um protocolo gerenciador de caixas postais, acessível a todos.

O POP3 é um dos principais protocolos gerenciadores de caixas postais utilizados na Internet. Ele resolve a questão de comunicação dos MTAs com UAs remotos.

O POP3 está especificado no [RFC 1725]: quando o host cliente ativa o UA, estabelece uma conexão TCP com o servidor POP3. O cliente e o servidor trocam comandos e respostas até que a conexão seja encerrada ou abortada. Os comandos consistem de uma palavra-chave seguida por um argumento e as respostas consistem de indicadores de sucesso/insucesso e uma palavra-chave seguida por informações adicionais.

Uma sessão POP3 passa por diferentes estados. Uma vez que a conexão TCP é estabelecida, o servidor POP3 envia uma mensagem de identificação e a sessão entra no estado de AUTORIZAÇÃO. Neste estado o cliente deve identificar-se, se ocorrer tudo bem no processo de identificação, o servidor obtém recursos associados à caixa postal do cliente e entra no estado de TRANSAÇÃO, no qual o cliente requisita ações ao servidor POP3, manipulando a sua caixa postal. Quando o cliente envia o comando QUIT, a sessão entra no estado ATUALIZAÇÃO. Neste estado o servidor POP3 libera os recursos adquiridos durante o estado de TRANSAÇÃO e envia uma mensagem de encerramento. A conexão TCP é então fechada [RFC 1725].

3.4. Gateways de E-Mail

Como vimos no capítulo 2, existem diversos protocolos para troca de e-mails. Cada um especifica de forma diferente como deve ser o endereço de um usuário, qual o formato de mensagens que suporta, como a mensagem é encaminhada ao destino etc. A princípio, estas diferenças impedem a troca de

e-mails entre usuários em sistemas que utilizam protocolos distintos.

Para viabilizar que usuários de sistemas de e-mail que utilizam diferentes protocolos pudessem se comunicar foram criados os *gateways* de e-mail. Os *gateways* de e-mail são máquinas que implementam mais de um protocolo de transferência de e-mail. O usuário, então, envia o e-mail para essa máquina e ela tem a funcionalidade de realizar o mapeamento de endereços e mensagens entre os diversos protocolos que implementa, transmitindo a mensagem ao destinatário final.

Quando um MTA executa uma troca de mensagem, o formato do endereço e da mensagem devem ser conhecidos por ambos os agentes de transferência de mensagens ou devem ser convertidos para serem aceitos pelo agente destinatário. Os gateways convertem endereços de correio eletrônico, mensagens e arquivos anexos, de modo que os usuários em um sistema de correio eletrônico possam trocar e-mails com usuários em outros sistemas diferentes.

O uso de *gateways* tem a grande vantagem de prover a conexão entre sistemas de e-mail distintos, resolvendo o problema da interoperabilidade.

3.5. Sendmail

O sendmail roda em máquinas Unix e é capaz de receber e enviar e-mails via SMTP, sendo inclusive o programa para este fim mais utilizado na Internet [Hunt 92] [RNP 95] [Avolio 94]. O sendmail implementa a maior parte das funções do MTA Internet. Falta-lhe apenas o suporte a POP3 para possibilitar que UAs remotos acessem as caixas postais armazenadas no servidor de e-mail. Esta tarefa normalmente é executada pelo programa popper.

Muitas vezes, um determinado servidor de e-mail está ligado a várias redes e tem, portanto, a necessidade de utilizar diversos protocolos de transferência de mensagens. A variedade de protocolos e os correspondentes MTAs utilizados em correios eletrônicos complicam a sua configuração e suporte: e-mails em redes TCP/IP geralmente são encaminhados através do sendmail, o **uux** e o **rmail** encaminham e-mails entre redes UUCP, um outro programa encaminha e-mails localmente, entre usuários do mesmo sistema, e assim por diante. Ou seja, programas diferentes são usados para diferentes

formas de transmissão de e-mails e fazê-los funcionar em conjunto de maneira harmônica é, muitas vezes, um grande desafio de suporte.

O sendmail oferece uma solução ao problema da coexistência de vários MTAs em um servidor de e-mails, pois ele não se restringe a ser um MTA SMTP. O sendmail é capaz de agir como um despachador, escolhendo qual MTA é adequado para o envio de uma determinada mensagem (veja a **Fig. 3.4**). Esse aspecto de ser aberto (i. e., admitir que outros MTAs sejam utilizados) é o principal diferencial do sendmail.

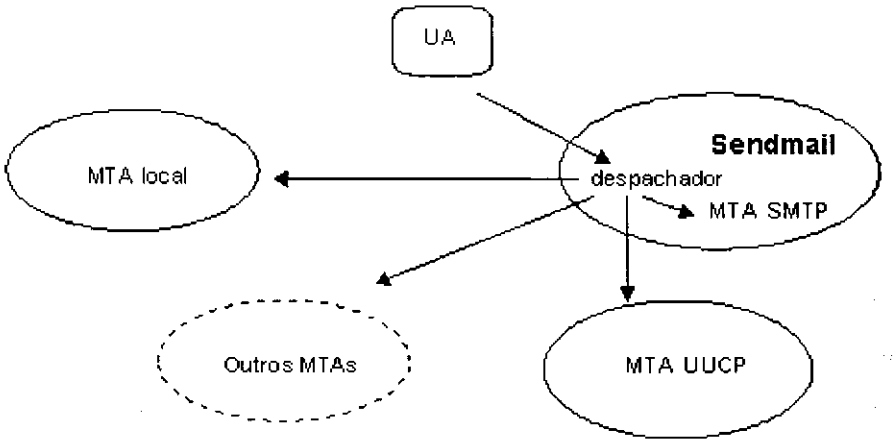


Figura 3.4. Roteamento de e-mail através do sendmail

Para escolher o MTA adequado para entrega da mensagem, o sendmail utiliza o endereço do destinatário, pois os formatos de endereços eletrônicos são diferentes para os diversos MTAs. Essa diferença cria também a necessidade de que os endereços do destinatário e do remetente sejam escritos nos formatos requeridos pelos MTAs.

Para isto, o sendmail oferece o recurso de reescrita de endereços. As regras de reescrita comparam os endereços eletrônicos com padrões, e, de acordo com o casamento desses padrões, colocam os endereços no formato adequado ao MTA a ser utilizado.

O sendmail possui ainda outras características adicionais: enfileira os e-mails não entregues para posterior retransmissão, pode tratar automaticamente mensagens recebidas e provê um serviço de pseudônimos.

Se ao tentar entregar um e-mail ao destino, ocorrer algum problema recuperável, o sendmail guarda o e-mail para posterior retransmissão. Ele

tentará encaminhar o e-mail dentro de um intervalo de tempo estabelecido na configuração e, caso não seja bem sucedido, devolverá o e-mail, acrescido de uma mensagem de erro, para o usuário remetente.

No tratamento automático de mensagens recebidas, o sendmail oferece recursos de redirecionamento de mensagens e envio automático de mensagens-respostas, para situações em que o usuário não esteja consultando seu e-mail (como férias). Este recurso é configurado através do arquivo .forward, localizado no diretório *home* do usuário.

Para ativar o redirecionamento, o conteúdo do .forward deve ser um endereço eletrônico. Para ativar a resposta automática, o usuário deverá executar o comando vacation (veja exemplo na **Fig. 3.5**) que editará um arquivo (o .vacation.msg), onde o usuário digitará a mensagem resposta desejada, informando, por exemplo, que está fora ou quando retornará.

Como pode ser percebido do exemplo, o vacation também utiliza o arquivo .forward, acrescentando à linha do endereço do usuário (caso exista) a chamada à execução do comando vacation através de um *pipe* Unix.

```
[anjinho] maria:/home/maria% vacation
This program can be used to answer your mail automatically
when you go away on vacation.
You have a message file in /home/maria/.vacation.msg.
Would you like to see it? y
From: maria (mensagem de férias)
Subject: Férias

Lerei seu mail "$SUBJECT" ao retornar, em 02/01.

Atenciosamente,
Maria
&:-)
Would you like to edit it? n
To enable the vacation feature a ".forward" file is created.
Would you like to enable the vacation feature? y
Vacation feature ENABLED. Please remember to turn it off when you
  get back from vacation. Bon voyage.
```

Figura 3.5. Exemplo do comando vacation

Um importante recurso também oferecido pelo sendmail é o serviço de pseudônimos (*aliases*). Como o nome sugere, o serviço de pseudônimos permite criar endereços que têm as mensagens a eles destinadas automaticamente enviadas a um endereço "real". Através dos pseudônimos do sendmail pode-se redirecionar, por exemplo, toda a correspondência do root e do postmaster para uma única caixa postal. O endereço "real" para o qual as mensagens são automaticamente redirecionadas pode até ser um endereço de

um usuário remoto (i. e., que recebe e-mails em outro servidor). Mais ainda, um pseudônimo pode representar vários usuários "reais", não necessariamente apenas um. Dessa forma, pode-se criar listas de distribuição através do sendmail. Os pseudônimos são definidos no arquivo /etc/aliases. Veja exemplo na **Fig. 3.6**:

```
# special names
postmaster: maria
root: maria
# a mailing list
admin: ph@astronauta, bia@cebolinha, fubica@anjinho
```

Figura 3.6. Exemplo do arquivo /etc/aliases

Os dois primeiros pseudônimos são nomes especiais: e-mails endereçados para o `postmaster` e para o `root` são entregues ao usuário `maria`. O pseudônimo `admin` define uma lista de distribuição, na qual todos os usuários são remotos. Modificações efetuadas no arquivo /etc/aliases só se tornam válidas após a execução do comando `newaliases`.

Toda a configuração do sendmail, com exceção do serviço de pseudônimos, que utiliza o arquivo /etc/aliases, é feita através do arquivo sendmail.cf. O arquivo sendmail.cf contém as informações requeridas para o encaminhamento de e-mails, incluindo as regras de reescrita de endereços de acordo com o MTA a ser utilizado na transmissão da mensagem.

O sendmail.cf define o ambiente sendmail, configurando características específicas do servidor de e-mails. As regras de reescrita de endereços são consideradas o "coração" do sendmail.cf [Hunt 92]. Através delas, é possível mapear os endereços utilizados pelo usuário para as formas requeridas pelos diferentes MTAs existentes em um servidor de e-mails.

Construir um arquivo sendmail.cf é uma tarefa complicada (veja na **Fig. 3.7**, um pequeno exemplo). A flexibilidade que o sendmail oferece para atender as necessidades dos diferentes MTAs usados num sistema de e-mail, tornam a sua configuração uma tarefa complexa. É preciso especificar como o sendmail deve rotear e-mails e reescrever os endereços de acordo com cada MTA utilizado.

Além da complexidade intrínseca da configuração do sendmail, os comandos usados são muito concisos. Comandos e variáveis são formados por apenas um caractere e não há espaços entre o comando e o valor da

variável. Pior ainda: caracteres de tabulação têm um significado diferente de espaço (sic!).

Logo, a tarefa de configurá-lo para refletir as diversas características e opções do sistema não é trivial. Ler e entender o arquivo sendmail.cf é realmente obtuso e difícil [Avolio 94].

```
# Simple sendmail.cf - Sendmail configuration file
#
#Your local domain
DDdco.frobozz.com
#Your full hostname
Dj$w
#Dj$w.$D
DRrelayhost.dco.frobozz.com
DVsimple
DnMailer-daemon
DlFrom $g  $d remote from $U
Do@.%
Dq$?x$x <$g>$|$g$.
Odbackground
Om
OF0600
Og1
OH/etc/sendmail.hf
OL6
Oo
OQ/var/spool/mqueue
Or1h
OS/etc/sendmail.st
Os
OT3d
Ou1
Ox8
OX12
Pfirst-class=0
Pspecial-delivery=100
Pjunk=-100
T root daemon uucp
H?F?From: $q
H?D?Date: $a
H?M?Message-Id: <$p.$t@$j>
HSubject:
S0
R$*@$j $#local$:$1       optional
R$*@$w $#local$:$1       optional
R$-          $#local$:$1       optional
R$*          $#remote$@$R$:$1
S1
S2
S3
R$*<$+>$*    $2
S4
Mremote, P=[IPC], F=nsmFDMuXC, S=10, R=10, A=IPC $h
Mlocal, P=/bin/mail, F=lsDFrmn, S=10, R=10, A=mail -r $f -d $u
Mprog,  P=/bin/echo,   F=lsDFmn,  S=10, R=10, A=mail $u
S10
```

Figura 3.7. Exemplo do arquivo sendmail.cf [Avolio 94]

3.6. Resumo

Neste capítulo, apresentamos o SMTP, especificando seus comandos e respostas e enumeramos os seus problemas. Descrevemos também o MIME,

que soluciona um dos problemas do SMTP: a limitação do conteúdo da mensagem ter apenas texto sem acentuação. Abordamos o POP3, um gerenciador de caixas postais que possibilita o uso de UAs remotos. Também descrevemos *gateways* de e-mail, que possibilitam a interoperabilidade de diferentes sistemas de e-mail, permitindo que usuários de sistemas que implementam protocolos distintos possam se comunicar. Depois disso, mostramos o sendmail, o MTA mais comumente utilizado em máquinas Unix e que oferece uma solução para a coexistência de diferentes protocolos de troca de e-mails em uma mesma máquina.

4

Segurança Digital

As informações estão, cada vez mais, sendo trocadas eletronicamente: o e-mail tem substituído o sistema de correios tradicional; o FAX digitaliza informações que estavam em papel e as transmite; transações financeiras são realizadas eletronicamente; os jornais e revistas podem ser lidas via Internet etc.

Assim, a informação tem sido largamente disponibilizada, gerando a questão de como fazer isso com segurança. Antes do advento da idade eletrônica, documentos importantes e secretos eram colocados num cofre, fechado com um segredo (chave) e era necessário um acesso físico para recuperá-los. Hoje, importantes transações comerciais e financeiras são realizadas através de redes de computadores, "num vasto mundo de zeros e uns".

Atualmente, com a integração crescente de redes antes isoladas à Internet, os problemas de segurança tornam-se mais evidentes, pois usuários do mundo inteiro podem ter acesso a qualquer rede interconectada. Os problemas mais comuns são acessos indevidos a uma máquina, devido a escolha de senhas óbvias e a captura de pacotes, através de programas que ficam monitorando os dados que trafegam pela rede.

Os serviços básicos fornecidos pelos sistemas de segurança são autenticação, autorização, integridade e privacidade [Russel 92]. A autenticação permite a verificação da identidade do usuário, garantindo que ele é realmente quem diz ser. A autorização define quais as operações que o usuário pode executar. A integridade garante que o dado não foi modificado por alguém não autorizado. E a privacidade visa garantir que os dados não serão visualizados sem que se tenha autorização para tanto.

É bem mais simples fornecer os quatro serviços de segurança acima mencionados em um ambiente centralizado. Em tal situação, a autenticação pode ser feita através de senhas, a autorização através de algum esquema de controle de acesso e a integridade como também a privacidade podem ser garantidas pelo próprio sistema operacional, que em última instância é o responsável pela implementação da política de controle de acesso.

Já em ambiente distribuído, não há, *a priori*, como um computador ter certeza que as informações originadas em outras máquinas são verdadeiras. Assim, torna-se necessário utilizar outras soluções para que seja possível oferecer os serviços básicos de segurança. Neste ambiente, a criptografia é o recurso utilizado para garantir a privacidade e a integridade dos dados que trafegam na rede, bem como para se fazer a autenticação de usuários. A criptografia fornece privacidade e integridade de forma direta, pois seu objetivo é justamente tornar informações ilegíveis a todos, menos a quem detém a chave de decriptografação. Em relação a autenticação, a criptografia pode ser usada como base de um sistema de assinatura digital ou possibilitar a construção de um serviço seguro de autenticação, como, por exemplo, o Kerberos.

Por outro lado, fazer a autorização em ambiente distribuído, uma vez que os demais aspectos já foram resolvidos, é tão fácil quanto em um sistema centralizado. Ou seja, é utilizado algum mecanismo de controle de acesso para determinar quais operações um determinado usuário pode realizar. O método de autorização mais popular é a Acess Control List (ACL) [Russel 92]. As ACLs são listas onde os recursos, como arquivos, diretórios, serviços, impressoras, discos, estão associados a usuários, processos e máquinas que podem utilizá-los.

Um outro recurso muito utilizado para a segurança na Internet é o *firewall*. *Firewall* é uma coleção de componentes colocados entre duas redes com o objetivo de controlar o tráfego de entrada e saída, certificando-se de que apenas o tráfego autorizado poderá ser encaminhado [Chewisk 94].

4.1. Criptografia

A criptografia é a ciência de escrever a informação em cifras, de forma

que somente as pessoas autorizadas possam compreendê-la. O seu objetivo é a construção de cifras invioláveis que garantam a privacidade e a autenticação das mensagens transmitidas [Weber 95].

O texto simples, após a criptografia, é chamado de **texto cifrado** ou **criptograma.** O processo de decompor o texto cifrado sem a utilização da chave utilizada para criptografá-lo é denominado **criptoanálise** (comumente chama-se a tentativa de criptoanálise de ataque).

Existem duas abordagens básicas para a criptografia: a simétrica e a assimétrica. Na criptografia simétrica (ou criptografia de chave única), a mesma chave é utilizada para cifrar e decifrar a informação. Na criptografia assimétrica (ou criptografia de chave pública), são geradas duas chaves, uma pública e outra privada, em que a chave privada decriptografa o que a chave pública criptografou.

Como já foi dito anteriormente, na criptografia simétrica a mesma chave é utilizada para criptografar e decriptografar a informação. Isso significa que o destinatário da mensagem precisa utilizar a mesma chave que o remetente utilizou, levando a dois grandes problemas: cada par de usuários que estabelece comunicação precisa escolher uma chave. Assim, se um dado usuário comunica-se com outros **n** usuários, ele precisa conhecer **n** chaves distintas Em um ambiente onde **n** usuários comunicam-se dois a dois, há necessidade de $(n^2 - n)/2$ chaves. Além da quantidade de chaves, existe a questão de como distribuir a chave de maneira segura.

Tradicionalmente, inventavam-se pares de chaves idênticas e que eram distribuídas aos seus destinos por um mensageiro [Tanenbaum 94]. Dependendo da periodicidade de alteração dessas chaves e do tamanho da organização, esse método torna-se inviável.

Até a publicação do artigo de Diffie e Hellman [Diffie 76] sobre criptografia assimétrica aceitava-se como fato natural que tanto a chave de criptografia quanto a chave de decriptografia deveriam ser mantidas em segredo. A proposta de Diffie e Hellman baseia-se na utilização de duas chaves distintas: uma para criptografia e outra para decriptografia, escolhidas de forma que a derivação de uma a partir da outra seja muito difícil de ser realizada. Uma vez respeitada essa condição, não há razão para não tornar

uma das chaves pública, mantendo somente uma secreta. Isto simplifica bastante a tarefa de gerenciamento das chaves.

Na criptografia assimétrica, só há a necessidade da existência de duas chaves por usuário: a chave privada, que não é enviada a nenhum outro participante do sistema, e a chave pública, que é distribuída sem restrições. O usuário utiliza a chave pública do destinatário para criptografar a informação a ser transmitida e, ao receber a informação cifrada, o destinatário utiliza a sua chave privada para decifrá-la. Note que somente o destinatário pode decifrar a mensagem, pois ele é o único que conhece a chave privada correspondente a sua chave pública.

Assim, está resolvido o problema da privacidade. Para a autenticação da informação existe a assinatura digital. A assinatura digital é a forma de se garantir a autenticidade de um documento que trafega pela rede. Normalmente, uma assinatura digital é criada pela utilização de funções conhecidas como *message digest*, combinadas com criptografia assimétrica.

As *message digest* são funções *hash* que, a partir de um texto de qualquer tamanho, produzem um número. As funções usadas como *message digest* são escolhidas de tal forma que não se consegue adulterar um texto e manter o mesmo resultado, nem mesmo através da inclusão de um *dumb string*.

Para gerar a assinatura digital, aplica-se a *message digest* ao texto e a criptografia assimétrica, com a chave privada, ao resultado obtido. A informação obtida da criptografia assimétrica é a assinatura. A chave pública, que todos os usuários conhecem, é utilizada para decriptografá-la, permitindo comprovar a origem, pois, como já citamos antes, só o próprio usuário conhece sua chave privada, e portanto, somente ele poderia ter criptografado. Uma vez obtido o valor da *message digest* calculado pelo enviador, computa-se novamente o valor *message digest* do texto e em seguida compara-se ambos. Caso sejam iguais, fica provado que não houve nenhuma alteração na informação.

Note que a assinatura digital só é possível se o algoritmo de criptografia assimétrica também funcionar com as chaves invertidas (i. e., a chave privada sendo usada para cifragem e a chave pública para decifragem).

A seguir, mostramos um algoritmo para criptografia simétrica, o IDEA, um para a criptografia assimétrica, o RSA, e uma função *message digest*, a MD5. Pois, os utilizamos em nosso trabalho.

4.1.1. IDEA

O International Data Encryption Algorithm (IDEA) foi desenvolvido na Suíça, por Xuejia Lai e James Massey [Schneier 93]. É um algoritmo de cifragem de chave única considerado muito seguro. Ele usa chaves de 128 bits e utiliza operações de ou-exclusivo (XOR), soma em módulo de 2^{16} e multiplicação em módulo de $2^{16} + 1$ (a soma e a multiplicação ignoram qualquer *overflow*).

O IDEA opera sobre blocos de 64 bits: 64 bits de texto normal produz 64 bits de texto cifrado. O mesmo algoritmo é utilizado para criptografar e decriptografar. Todas as operações são efetuadas sobre sub-blocos de 16 bits, o que o torna eficiente também em processadores de 16 bits.

O algoritmo é executado em 8 passos. A chave de 128 bits é dividida em 8 sub-chaves de 16 bits. Assim são geradas as 8 primeiras sub-chaves para a execução do algoritmo. Dessas, 6 são utilizadas no primeiro passo e as outras 2 são as utilizadas como a primeira e segunda sub-chaves do segundo passo. Para gerar as próximas sub-chaves é executada uma rotação para esquerda de 25 bits em cima da chave. Assim são geradas mais 8 sub-chaves, as quatro primeiras serão utilizadas no segundo passo e as outras no terceiro passo. Então, repete-se o processo até o final do algoritmo.

Um bloco de dados de 64 bits é dividido em 4 sub-blocos de 16 bits (X1, X2, X3 e X4) e as seguintes operações são realizadas:

1. Multiplica-se X1 pela primeira sub-chave.
2. Adiciona-se X2 à segunda sub-chave.
3. Adiciona-se X3 à terceira sub-chave.
4. Multiplica-se X4 pela quarta sub-chave.
5. Efetua-se um ou-exclusivo dos resultados dos passos 1 e 3.
6. Efetua-se um ou-exclusivo dos resultados dos passos 2 e 4.
7. Multiplica-se o resultado do passo 5 pela quinta chave.

8. Adiciona-se o resultado do passo 6 ao resultado do passo 7.
9. Multiplica-se o resultado do passo 8 pela sexta chave.
10. Adiciona-se o resultado do passo 7 ao resultado do passo 9.
11. Efetua-se um ou-exclusivo dos resultados dos passos 1 e 9.
12. Efetua-se um ou-exclusivo dos resultados dos passos 3 e 9.
13. Efetua-se um ou-exclusivo dos resultados dos passos 2 e 10.
14. Efetua-se um ou-exclusivo dos resultados dos passos 4 e 10.

A saída da execução são os blocos resultantes dos passos 11, 12, 13 e 14, passando a equivaler a X1, X3, X2 e X4, respectivamente.

Assim, o algoritmo é executado mais 7 vezes. Após o oitavo passo, há uma transformação final:

1. Multiplica-se X1 pela primeira sub-chave.
2. Adiciona-se X2 à segunda sub-chave.
3. Adiciona-se X3 à terceira sub-chave.
4. Multiplica-se X4 pela quarta sub-chave.

Os quatros blocos são reagrupados produzindo um bloco de texto cifrado.

Na **Fig. 4.1**, podemos ver graficamente o fluxo do algoritmo:

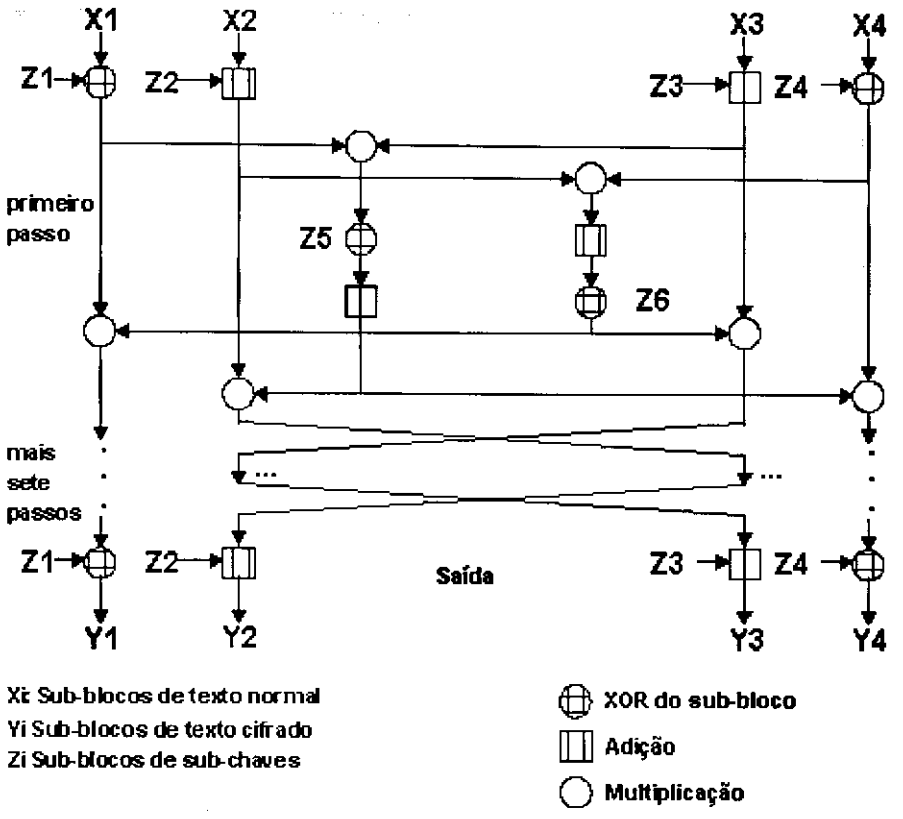


Figura 4.1. Diagrama do algoritmo IDEA [Schneier 93]

Desenvolvido em 1990, não existe ainda um método de ataque efetivo contra o IDEA, pois até o momento tem resistido a métodos aplicados com sucesso contra outros algoritmos [Weber 95].

4.1.2. RSA

O RSA [Rivest 78] é um algoritmo de criptografia assimétrica usado para sigilo e autenticação. Os autores do algoritmo o nomearam com as primeiras letras de seus sobrenomes: Rivest, Shamir e Adleman. É, atualmente, o algoritmo de chave pública mais largamente utilizado e é uma das mais poderosas formas de criptografia de chave pública conhecida até o momento [Frosen 95]. Nossa descrição do RSA foi baseada em [Soares 95].

O RSA obtém a segurança da dificuldade de se fatorar números muito grandes. Para se gerar as chaves pública e privada, escolhe-se dois números primos **p** e **q** (que devem permanecer secretos e terem centenas de bits de comprimento). Em seguida calcula-se **n**, onde **n** é o produto de **p** e **q**. Então é escolhido randomicamente um número **e**, que é primo de **(p-1)×(q -1)** (i.e., não têm fatores comuns). A seguir, calcula-se um número **d**, tal que **e×d** módulo

(p-1)×(q -1) seja igual a 1, ou seja, **e×d = 1 mod [(p-1)×(q-1)]**.

A chave pública consiste dos números **n** e **e**, e a chave privada, dos números **n** e **d**.

Para cifrar uma mensagem **P**, realiza-se a aplicação da operação: $C \leftarrow P^e \pmod n$. E para decifrar a informação $P \leftarrow C^d \pmod n$.

Um outro fator também determinante para o largo uso do RSA, é que ele permite inverter a ordem de uso das chaves, podendo ser usado para autenticação. Pois, ao se inverter a ordem de uso das chaves, tem-se uma mensagem que só pode ter sido cifrada por uma pessoa, pois somente o dono da chave conhece a sua chave privada. Melhor ainda, este fato é facilmente verificável pois, em princípio, todos conhecem a chave pública do remetente.

4.1.3. MD5

MD5 [RFC 1321] é uma função *message digest*, que a partir de um texto de entrada de qualquer tamanho produz um número de 128 bits. MD5 é utilizado em assinatura digital, onde um texto de qualquer tamanho é comprimido de forma segura, antes de ser criptografado com a chave privada de um algoritmo de chave pública, como o RSA.

Conforme descrito no [RFC 1321], para encontrar o resultado MD5 de uma mensagem de tamanho **b**, onde **b** é um número inteiro não-negativo, são executados cinco passos:

1. A mensagem é estendida até que seu tamanho fique congruente a 448, módulo 512. Isto é, ficam faltando apenas 64 bits para que seja um múltiplo de 512. A extensão é executada da seguinte forma: um bit 1 é acrescentado à mensagem e após isso são acrescentados tantos bits 0, quantos sejam necessários.
2. O tamanho de **b** representado em 64 bits (que é o tamanho da mensagem antes dos bits de extensão serem adicionados à mensagem), é acrescentada ao resultado do passo 1. No improvável evento de **b** ser maior do que 2^{64}, então apenas os 64 bits de mais baixa ordem de **b** serão utilizados. Neste ponto, o resultado será exatamente um tamanho que é múltiplo de 512 bits.

3. Um *buffer* de quatro palavras (A, B, C e D) é usado para computar a *message digest*. Aqui, cada uma dessa palavras é um registro de 32 bits. Esses registros são inicializados com os seguintes valores em hexadecimal, com os bytes de baixa ordem primeiro:

   A - 01 23 45 67

   B - 89 ab cd ef

   C - fe dc ba 98

   D - 76 54 32 10

4. Primeiro são definidas quatro funções auxiliares que recebem como entrada três palavras de 32 bits e produzem como saída uma palavra de 32 bits.

   F(X, Y, Z) = XY v not(X) Z

   G(X, Y, Z) = XZ v Y not(Z)

   H(X, Y, Z) = X xor Y xor Z

   I(X, Y, Z) = Y xor (X v not(Z))

   Cada posição de um bit em F age como uma condição: se X então Y senão Z. Note que, se os bits de X, Y e Z são independentes, cada bit de F(X, Y, Z) será independente. As funções G, H e I são similares à função F.

   Este passo usa uma tabela de 64 elementos: T[1..64]. Faz-se T[i] designar o i-ésimo elemento da tabela, que é igual a parte inteira de 4294967296 multiplicado por abs(sen(i)), onde i está em radiano. Veja o processo abaixo:

```
/* Processa cada bloco de 16 bits */
For i = 0 to N/16-1 do
     /* Copia o bloco i em X. */
     For j = 0 to 15 do
          X[j] = M[i*16+j].
     end
     /* Salva A como AA, B como BB, C como CC e D como DD. */
     AA = A
     BB = B
     CC = C
```

```
    DD = D
    /* Primeiro Passo */
    /* Seja [abcd k s i] a abreviação da seguinte atribuição:
          a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */
    /* Faça as 16 operações seguintes: */
    [ABCD 0 7 1][DABC 1 12 2][CDAB 2 17 3][BCDA 3 22 4]
    [ABCD 4 7 5][DABC 5 12 6][CDAB 6 17 7][BCDA 7 22 8]
    [ABCD 8 7 9][DABC  9 12 10][CDAB 10 17 11][BCDA 11 22 12]
    [ABCD 12 7 13][DABC 13 12 14][CDAB 14 17 15][BCDA 15 22 16]
    /* Segundo Passo. */
    /* Faça [abcd k s i] denotar a operação
          a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */
    /* Faça as 16 operações seguintes: */
    [ABCD 1 5 17][DABC 6 9 18][CDAB 11 14 19][BCDA  0 20 20]
    [ABCD 5 5 21][DABC 10 9 22][CDAB 15 14 23][BCDA 4 20 24]
    [ABCD 9 5 25][DABC 14 9 26][CDAB 3 14 27][BCDA 8 20 28]
    [ABCD 13 5 29][DABC 2 9 30][CDAB 7 14 31][BCDA 12 20 32]
    /* Terceiro Passo. */
    /* Faça [abcd k s t] denotar a operação
          a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */
    /* Faça as 16 operações seguintes: */
    [ABCD 5 4 33][DABC 8 11 34][CDAB 11 16 35][BCDA 14 23 36]
    [ABCD 1 4 37][DABC 4 11 38][CDAB 7 16 39][BCDA 10 23 40]
    [ABCD 13 4 41][DABC 0 11 42][CDAB  3 16 43][BCDA  6 23 44]
    [ABCD 9 4 45][DABC 12 11 46][CDAB 15 16 47][BCDA 2 23 48]
    /* Quarto Passo. */
    /* Faça [abcd k s t] denotar a operação
          a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */
    /* Faça as 16 operações seguintes: */
    [ABCD 0 6 49][DABC 7 10 50][CDAB 14 15 51 ][BCDA 5 21 52]
    [ABCD 12 6 53][DABC 3 10 54][CDAB 10 15 55][BCDA 1 21 56]
    [ABCD 8 6 57][DABC 15 10 58][CDAB 6 15 59][BCDA 13 21 60]
    [ABCD 4 6 61][DABC 11 10 62][CDAB 2 15 63][BCDA 9 21 64]
    /* Então execute as seguintes adições. (Que é incrementar
          cada um dos quatro registros com o valor anterior à
                execução este bloco)
    A = A + AA
    B = B + BB
    C = C + CC
    D = D + DD
end /* fim do laço i */
```

5. A message digest produz como saída A, B, C e D.

Acredita-se que é computacionalmente impraticável produzir duas mensagens tendo o mesmo resultado MD5 ou produzir uma mensagem a partir de um resultado MD5 [RFC 1321].

4.2. Kerberos

O Kerberos[1] [Cheswick 94] é um sistema de autenticação para redes baseado em criptografia de chave única. Desenvolvido como parte do Athena, no Massachusetts Institute of Technology, o Kerberos fornece credenciais que autenticam usuários ou serviços. O Kerberos serve a dois propósitos: autenticação e distribuição de chaves.

Um sistema Kerberos é constituído de um autenticador, o Kerberos Distribution Center (KDC) e de um servidor de *tickets*, o Ticket-Granting Server (TGS). Cada usuário ou serviço compartilha uma chave única com o KDC. Essas chaves únicas permitem atestar a identidade dos usuários ou serviços [Soares 95] [Woo 92].

Quando o usuário inicia sua sessão de trabalho, é solicitada a sua identificação de *login*. Antes de solicitar a senha, o processo de *login* envia uma mensagem ao KDC com a identificação do usuário, solicitando o *ticket* TGS. Essa mensagem é respondida criptografada com a chave única do usuário, contendo o *ticket* para o TGS e a chave que será usada para codificar as mensagens trocadas entre o cliente e o TGS. O processo de *login* recebe a mensagem e solicita, então, a senha do usuário. Veja que a máquina cliente só será capaz de decodificar essa mensagem, que dá acesso ao serviço do TGS, se possuir a senha de *login* do cliente, que não é enviada pela rede em momento algum [Soares 95].

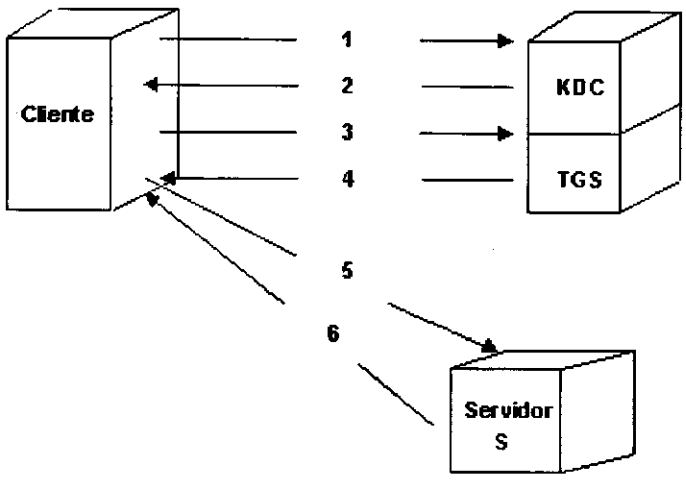
Quando um cliente deseja utilizar os serviços de um servidor registrado no Kerberos, deve, em primeiro lugar e obrigatoriamente, obter um *ticket* do servidor TGS. O *ticket* contém informações que identificam o serviço, o cliente, o momento de emissão do *ticket* (*timestamp*) e a validade (*lifetime*), além da chave (denominada de chave de sessão) que será utilizada para criptografar as mensagens que serão trocadas na transação entre o cliente e o servidor. O

[1] Este é o nome do cão de três cabeças, guardião do inferno mitológico grego.

ticket é criptografado com a chave do servidor de forma que só ele pode decriptografar o seu conteúdo. Como só o Kerberos e o serviço sendo usado conhecem a chave do servidor, a autenticação do *ticket* pode ser verificada pelo servidor quando esta lhe for entregue pelo cliente.

Quando envia o *ticket* para um serviço, o cliente envia também um autenticador, cujo objetivo é evitar que uma entidade armazene o *ticket* e o utilize posteriormente em ataques do tipo *replay*. O autenticador é criptografado com a chave de sessão e existe para garantir que a segurança Kerberos não será quebrada através da monitoração do tráfego da rede. O *timestamp* é testado no servidor, portanto os relógios dos usuários Kerberos precisam estar sincronizados, dentro de uma devida margem de erro conhecida, para garantir que o pedido foi emitido recentemente e está dentro do intervalo de validade do *ticket*.

Veja esquematicamente como ocorre a autenticação Kerberos na **Fig. 4.2**, que segue abaixo.



1 - O Cliente solicita acesso ao TGS, fornecendo sua identificação
2 - O KDC devolve o *ticket* TGS e chave de sessão Cliente/TGS, ambos criptografados com a senha do Cliente
3 - O Cliente solicita acesso ao serviço, enviando o *ticket* TGS e um autenticador, criptografado com a chave de sessão Cliente/TGS
4 - O TGS devolve o *ticket* do serviço e chave de sessão Cliente/Servidor, ambos criptografados com a chave de sessão Cliente/TGS
5 - O Cliente acessa o serviço, enviando o *ticket* de serviço e um autenticador, criptografado com a chave de sessão Cliente/Servidor, que será usada subseqüentemente para cifrar toda comunicação entre o Cliente e o Servidor
6 - O Servidor prova que não é um impostor devolvendo o *timestamp* usado pelo cliente acrescido de 1, criptografado com chave de sessão Cliente/Servidor

Figura 4.2. Sistema de Autenticação Kerberos

4.3. Firewalls

De maneira geral, o termo *firewall* implica em proteção de perigo através de um barreira. Em um sistema de computador significa a proteção de uma rede das outras redes, podendo ser utilizado para isolar uma rede conectada à Internet [RFC 1636]. Um *firewall* é definido em [Chewisk 94] como um conjunto de componentes colocados entre duas redes que tem as propriedades seguintes:

- Todo o tráfego de dentro para fora da rede e vice-versa passa pelo *firewall.*
- Só o tráfego autorizado passa pelo *firewall.*
- O *firewall* é imune a violações.

Um *firewall* pode ser implementado em diferentes camadas do protocolo, sendo mais comumente implementado na camada de aplicação ou na camada de rede (IP, por exemplo).

Na camada de aplicação, o *firewall* atua como um *gateways* de aplicação. Ao invés de solicitar o serviço diretamente a um servidor, uma requisição de serviço do cliente abre uma conexão de transporte com o *firewall*. O cliente identifica-se para o *firewall* e especifica sua intenção. O *firewall* decide se o cliente está ou não autorizado a executar o serviço solicitado. Se o cliente estiver autorizado será aberta uma conexão de transporte com o servidor para início da operação. Assim, o *firewall* passará as requisições e respostas entre o cliente e o servidor. Veja na **Fig. 4.3**, como pode ser a representação de um diagrama de um *firewall* à nível de aplicação.

Figura 4.3. Diagrama de *firewall* da camada de aplicação [RFC 1636]

Na camada de rede, o *firewall* atua filtrando pacotes, isto é, aplicando um conjunto de regras sobre cada pacote para determinar se o encaminha ou não. As informações dos cabeçalhos dos pacotes (como endereço das máquinas origem e destino, número do protocolo, número de portas) são utilizadas para essa decisão. Por exemplo, as regras podem especificar que todo o tráfego de pacote SMTP, de entrada e saída, é permitido e que o tráfego de pacotes telnet só é permitido para saída.

Entretanto, encontramos algumas desvantagens na implementação de *firewalls*. Quando o *firewall* funciona à nível de aplicação é necessário criar um código específico para cada aplicação, o que dificulta a implantação de novas tecnologias. Outro ponto negativo é que se o *firewall* falhar, a aplicação também irá falhar. Nos *firewalls* de rede, existe a vulnerabilidade de intrusos forjarem os endereços de origem e destino, comprometendo a filtragem dos pacotes. Além disso, utilizar *firewalls* torna a comunicação com o mundo externo mais complicada para o usuário. De qualquer forma, *firewalls* são utilizados na prática, pois oferecem um maior nível de segurança do que seria possível obter sem eles.

Firewalls não são uma solução para o problema de segurança, são uma reação para um problema básico da administração de sistemas: configurar um grande número de máquinas com nível muito bom de segurança. Assim, através de *firewalls*, é possível concentrar os esforços administrativos e garantir segurança de toda uma rede apenas pela configuração cuidadosa de poucas máquinas. Se esse problema puder ser resolvido, os *firewalls* são desnecessários [RFC 1636].

4.4. Resumo

Neste capítulo fizemos algumas considerações sobre o que é desejado à nível de segurança digital: sigilo, integridade, autenticação e autorização. Descrevemos a criptografia, que é o recurso utilizado para se obter confidencialidade e autenticação de informações. Descrevemos o IDEA, que é um algoritmo de criptografia de chave única; descrevemos o RSA que é, atualmente, o algoritmo de chave pública mais utilizado; e descrevemos, também, a MD5, um tipo especial de função, que a partir de uma entrada de qualquer tamanho produz um número de 128 bits. Mostramos como o Kerberos trabalha com a autenticação e abordamos, resumidamente, *firewalls*, um recurso bastante utilizado para isolar uma rede de outras redes.

5

Segurança em E-Mail

Com a crescente utilização de redes de computadores, a segurança da comunicação assume uma importância cada vez maior. As mensagens que trafegam numa rede precisam ser protegidas, pois podem ser interceptados por intrusos que estejam monitorando a rede.

Conforme já explicitamos antes, o serviço mais utilizado em redes é o e-mail. O fato de se ter um custo relativamente baixo para instalar e manter um sistema de correio eletrônico é um dos aspectos que o torna o bastante atraente, mas o correio eletrônico tem riscos de segurança que envolvem a possibilidade de se forjar um remetente e a possibilidade de interceptação de mensagem por um intruso. Assim, recursos como a confidencialidade e autenticação de mensagens devem ser considerados em um ambiente de e-mail.

5.1. Segurança em X.400

O padrão X.400, na versão de 88, adicionou recursos de segurança ao X.400. Entretanto a especificação é apenas uma estrutura geral, com provisões para privacidade, integridade, reconhecimento e autenticação de mensagens, mas que não define os algoritmos de criptografia a serem utilizados, como deve ser o gerenciamento de chaves e nem mesmo o tamanho das chaves [Alvestrand 96].

A recomendação X.402 define um modelo de segurança, a recomendação X.411 define um protocolo de segurança e a recomendação X.400 define os aspectos de segurança para o MHS e os elementos de serviços necessários para essa segurança [Heikkinen 96].

Os serviços são divididos em transferência e armazenamento das

mensagens [Heikkinen 96], incluindo os aspectos de:

- Autenticação da origem das mensagens
- Provas de entrega e submissão
- Integridade na conexão
- Integridade do conteúdo
- Não repudiação do encaminhamento, origem e submissão
- Confidencialidade da conexão
- Confidencialidade do conteúdo
- Contexto de Segurança
- Rótulo de Segurança

A autenticação é efetuada entre o originador e o receptor, para se comprovar da origem da mensagem. Os serviços de integridade fornecem a garantia, ao receptor, de que a mensagem recebida é a mesma que foi enviada, protegendo a mensagem de alterações não autorizadas. Os serviços de confidencialidade, fornecem a proteção do conteúdo da mensagem, do início ao fim da transferência, contra divulgação não autorizada.

A não repudiação corresponde ao problema do transmissor negar a autoria de uma mensagem ou de um receptor negar que a recebeu. Os serviços de não repudiação, nas recomendações MHS, lidam com o reconhecimento da origem, da submissão e da entrega. Através do reconhecimento dos serviços de origem, os receptores podem provar individualmente que a origem é a especificada. Com o reconhecimento de submissão, depois que o transmissor iniciar uma transferência, recebe a confirmação de que ela foi submetida ao sistema. Com o reconhecimento de entrega, o transmissor tem a garantia de que uma mensagem foi recebida. Esta prova é gerada pelo receptor e transmitida de volta ao transmissor.

O contexto de segurança limita a passagem da mensagem entre entidades MHS, de acordo com o rótulo de segurança associado à mensagem. O Rótulo de Segurança classifica uma mensagem em restrita, confidencial e secreta. Entretanto, este conceito foi apenas definido. O esquema para identificação do rótulo não foi especificado [Alvestrand 96].

5.2. Segurança em E-Mail UUCP

No UUCP, a segurança está relacionada com a comunicação na transferência da informação e também localmente, na máquina Unix.

A capacidade de ler e escrever arquivos em outras máquinas Unix é usualmente bastante restrita, ficando sob responsabilidade do administrador do sistema definir e implementar o grau de permissão para acesso aos diversos arquivos. Num sistema que implementa o UUCP, o administrador pode especificar no arquivo de permissão, `/usr/lib/uucp/Permissions`, o nível de restrição de acesso por outras máquinas. Veja, na **Fig. 5.1**, o exemplo de um arquivo `Permissions`, abaixo:

```
# Exemplo de arquivo Permissions
LOGNAME=usuario \
REQUEST=NO \
READ=/usr/spool/uucpPublic \
WRITE=/usr/spool/uucpPublic \
COMMANDS=rmail
```

Figura 5.1. Exemplo de um arquivo Permissions do UUCP

Esta configuração é bastante restritiva. Diz que a máquina vizinha tem acesso com o *login* `usuario`. Que o único diretório que pode ser acessado para leitura e escrita é o diretório público do UUCP e que o único comando permitido para execução é o **rmail**.

Entretanto, não existem quaisquer procedimentos no sentido de garantir o sigilo das mensagens durante a transmissão.

5.3. Segurança em E-Mail Internet

No ambiente de correio eletrônico Internet não há nenhum tratamento para segurança da mensagem, nem na transmissão, pois o SMTP não trata a possibilidade de interceptação da mensagem e falsificação do remetente, nem na máquina servidora de e-mail, pois as mensagens ficam armazenadas em formato texto legível nas caixas-postais dos usuários e na fila de transmissão.

O [RFC 1421] define procedimentos para privacidade e autenticação de e-mails, chamados de serviços *Privacy Enhanced Mail* (PEM), utilizando criptografia. O [RFC 1422] especifica suporte a mecanismos de gerenciamento de chaves baseado em criptografia de chave pública. O [RFC 1423] provê os

algoritmos de criptografia usados em [RFC 1421] e [RFC 1422] e o [RFC 1424] detalha procedimentos para o gerenciamento de chaves e infra-estrutura para o suporte a esses serviços [RFC1421]. O PEM é uma especificação, não um programa.

No PEM, as chaves públicas devem ser assinadas por uma Autoridade Central e apenas as chaves oficialmente assinadas podem ser usadas. O processo de assinatura de chaves é denominado certificação. Um certificado é uma estrutura de dados que contém o nome do usuário, sua chave pública e o nome da entidade organizacional a qual o usuário pertence.

A Autoridade Central agindo como a raiz da hierarquia de certificados para a comunidade Internet, com a necessidade de pagamento de uma taxa para o seu funcionamento e necessidade de que todas as chaves sejam certificadas, demanda a necessidade de um suporte e custo adicional. A complexidade imposta pela certificação de chaves, sem dúvidas, torna a possibilidade de acesso ao sistema por usuários não autorizados mais difícil. Entretanto, tal abordagem, é também mais difícil de implementar e de utilizar.

Assim, a criptografia de e-mails Internet é hoje feita quase que exclusivamente através do Pretty Good Privacy (PGP), software de criptografia disponível gratuitamente. O PGP não especifica uma política para o estabelecimento da confiabilidade das chaves: os usuário assinam as chaves uns dos outros e criam uma comunidade interconectada de usuários de chaves públicas [Stallings 95], diferentemente do enfoque do PEM, o qual depende de uma Autoridade Central para certificação das chaves.

Além do usuário ter que, de alguma forma, descobrir a chave pública do destinatário, uso do PGP para garantir a segurança na troca de e-mails apresenta outro grande inconveniente: o PGP é um software totalmente independente do ambiente de correio eletrônico. Isto significa que o usuário precisa escrever a mensagem fora do ambiente de e-mail, criptografá-la com o PGP, importar o arquivo cifrado para o ambiente de correio eletrônico e, finalmente, enviar a mensagem. De maneira análoga, na recepção de uma mensagem cifrada pelo PGP, é necessário exportá-la e decriptografá-la com o PGP, antes de lê-la, já fora do ambiente de e-mail. O fato do PGP ser uma solução externa ao software de correio eletrônico implica em uma grande

perda de produtividade no seu uso. Além do mais, demanda a aprendizagem de novos comandos que, muitas vezes, não são amigáveis para o usuário final.

Então, no PGP, para obter a segurança desejada, além do software de correio eletrônico o usuário deverá aprender a manipular um software de criptografia.

5.3.1. PGP

O PGP é um software de segurança que usa os principais algoritmos de criptografia para garantir a privacidade na troca de e-mails e para proteger arquivos armazenados em memória não-volátil. Com o PGP podemos proteger arquivos, criar e gerenciar chaves, criptografar e-mails, usar assinatura digital, certificar, distribuir e anular chaves, além de desabilitá-las ou repassá-las [Garfinkel 95].

Como já foi mencionado, o PGP é um pacote externo à aplicação de e-mail. É utilizado em criptografia de arquivos para garantia de sigilo e para assiná-los digitalmente.

No PGP, as chaves ficam armazenas em arquivos chaveiros (*key rings*): **pubring.pgp**, que contém as chaves públicas e **secring.pgp**, que contém as chaves privadas criptografadas. As chaves privadas dos usuários são criptografadas com uma senha chamada de *pass phrase*. A *pass phrase* aceita como senha uma cadeia de caracteres de qualquer tamanho, contendo espaços, pontos, letras maiúsculas e minúsculas etc. Através de MD5, a *pass phrase* é transformada numa chave de 128 bits, que é, de fato, a chave usada para criptografia da chave privada.

Para criptografia do e-mail é usado o algoritmo IDEA. A chave IDEA usada na criptografia do e-mail é criptografada utilizando algoritmo RSA de criptografia de chave pública e é enviada com o e-mail para o destinatário. Utiliza-se a criptografia simétrica para criptografar o conteúdo do e-mail e a criptografia de chave pública para criptografar a chave, pois na criptografia assimétrica os algoritmos são lentos. Assim criptografa-se a chave IDEA, que contém bem menos informações.

5.3.1.1. Protegendo arquivos

A maneira mais simples de utilizar o PGP é na criptografia de arquivos visando protegê-los do acesso por outros usuários. O comando que permite a criptografia é: `pgp -c <nome-do-arquivo>`, o PGP solicitará então a *pass phrase* e efetuará a criptografia (**Fig. 5.2**):

```
% pgp -c meu-arquivo
Pretty Good Privacy (tm) 2.6 - Public-key encryption for the
masses.
(c) 1990-1994 Phillip Zimmermann, Phil's Pretty Good Software. 23
May 94
Distributed by Massachusetts Institute of Technology. Uses RSAREF.
Export of this software may be restricted by U.S. government.
Current time: 1994/06/26 21:47 GMT
You need a pass phrase to encrypt the file.
Enter pass phrase: Nobody knows my name.
Enter same pass phrase again: Nobody knows my name. Just a
moment...
Ciphertext file: meu-arquivo.pgp
%
```

Figura 5.2. Criptografia de arquivo usando PGP [Garfinkel 95]

5.3.1.2. Gerenciando chaves no PGP

Gerenciamento de chaves é um termo genérico para qualquer operação executada em uma chave [Garfinkel 95]. O PGP fornece opções para manipulação dos arquivos de chaves (**Tab. 5.1**):

pgp -kv	para examinar chaves públicas
pgp -kv secring.pgp	para examinar chaves no chaveiro secreto
pgp -kv nomearquivo.pgp	para examinar chaves em outros arquivos
pgp -ke	para alterar uma chave (editar), para mudar a pass phrase, para acrescentar um endereço de e-mail ao arquivo público
pgp -ka	para acrescentar um outro endereço de e-mail a seu arquivo público
pgp -kx	para extrair sua chave pública do arquivo e passá-la adiante

Tabela 5.1. Comandos PGP de gerenciamento de chaves

5.3.1.3. Criptografando/Decriptografando E-Mail

Para enviar um e-mail criptografado com PGP deve-se criar o e-mail, obter a chave pública do destinatário, criptografar a mensagem usando esta chave pública, e enviar normalmente para o destino. O comando a ser utilizado

é o `pgp -eat`. A opção **e** diz ao pgp para criptografar a mensagem, a opção **a** que o resultado é ASCII imprimível de 7 bits (o que é suportado pelo sistema de e-mail Internet), e a opção **t** que é um arquivo do tipo texto. O resultado será colocado em um arquivo com a extensão asc (como no exemplo da **Fig. 5.3**).

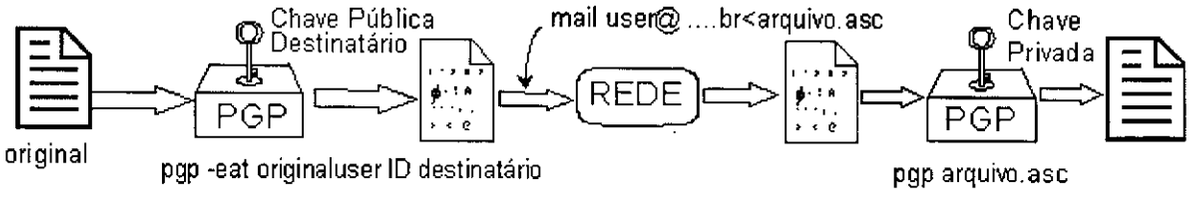


Figura 5.3: Exemplo do envio de e-mail usando o PGP

Esta mensagem só será decriptografada com a chave secreta do destinatário. O conhecimento da chave pública é insuficiente. Veja na **Fig. 5.4**, uma tentativa de acesso não autorizada a um arquivo criptografado com a chave pública do usuário Phil:

```
% pgp arquivo.asc
Pretty Good Privacy (tm) 2.6 - Public-key encryption for the
masses.
(c) 1990-1994 Phillip Zimmermann, Phil's Pretty Good Software. 23
May 94
Distributed by Massachusetts Institute of Technology. Uses RSAREF.
Export of this software may be restricted by U.S. government.
Current time: 1994/06/26 21:47 GMT

File is encrypted. Secret key is required to read it.
This message can only be read by:
   Phil's Pretty Good  <phil@pgp.com>
You do not have a secret key needed to decrypt this file.
For a usage summary, type: pgp -h
For more detailed help, consult the PGP User's Guide.
%
```

Figura 5.4. Exemplo de acesso indevido a um arquivo [Garfinkel 95]

O uso do comando **pgp -eat** mais de uma vez em um mesmo arquivo produz resultados completamente diferentes, pois o PGP escolhe aleatoriamente a chave IDEA usada para criptografar o e-mail.

Ao receber um e-mail criptografado com o PGP, deve-se salvá-lo em um arquivo e em seguida usar o comando `pgp <arquivo>`. Desejando-se gerar um arquivo com um nome diferente na saída, é só utilizar a opção **-o**: `pgp arquivo.asc -o novo_nome`.

5.3.1.4. Usando Assinatura Digital

A assinatura digital no PGP tem duas funções importantes para a segurança da informação: integridade e autenticação. A integridade indica se uma mensagem foi ou não modificada e a autenticação torna possível que se verifique a origem da mensagem. O PGP utiliza a *message digest* MD5, que é distribuída pelo RSA Data Security [Garfinkel 95].

Dessa forma, desejando-se saber se um texto que trafegou na Internet foi alterado, calcula-se a função MD5 do texto e em seguida compara-se com o resultado MD5 do texto original.

A função MD5 é apenas uma parte da solução para se criar assinaturas digitais confiáveis. Como já dissemos antes, o algoritmo RSA, de criptografia assimétrica, também funciona de forma inversa, ou seja, é possível criptografar mensagens com a sua chave secreta e decriptografar com a sua chave pública. Cada chave pública tem apenas uma chave secreta associada, se uma chave pública pode decriptograr uma mensagem, então pode-se ter certeza que a chave secreta associada àquela chave pública foi usada para a criptografia. Assim, o PGP calcula a assinatura digital ao se aplicar a chave secreta do remetente sobre o resultado da função MD5 da mensagem.

Para adicionar a assinatura digital a uma mensagem, utiliza-se o comando: `pgp -sta <arquivo>`. Para verificar a assinatura, basta executar o PGP sobre o arquivo assinado.

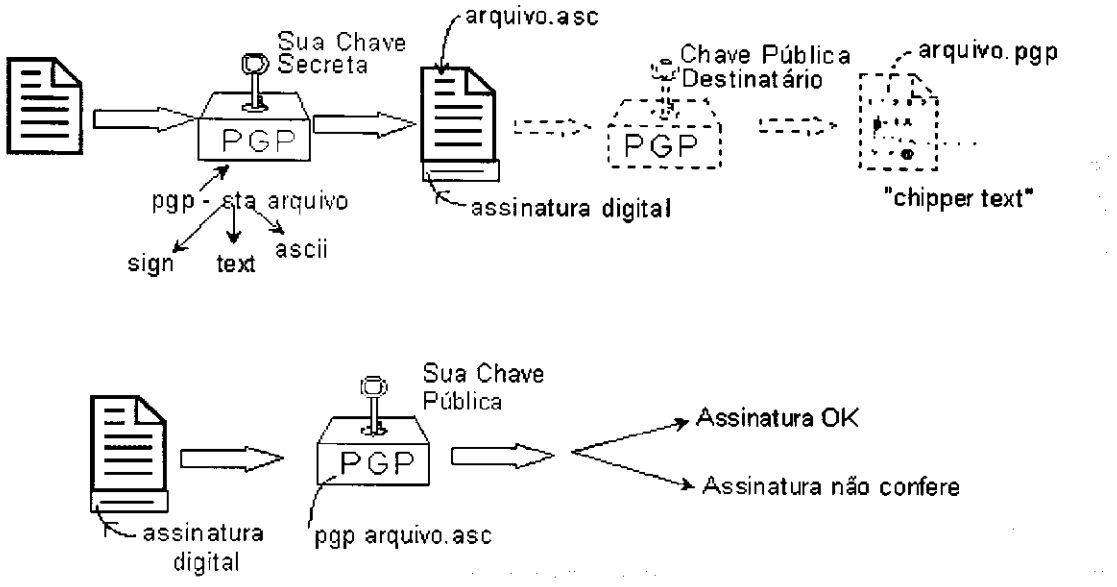


Figura 5.5. Exemplo de assinatura digital usando PGP

Freqüentemente, além de assinar a mensagem para provar sua autenticidade, existe a necessidade de privacidade. Com o PGP pode-se também assinar e criptografar uma mensagem utilizando a opção **-se**. Assina-se a mensagem com a chave secreta do remetente e criptografa-se com a chave pública do destinatário (**Fig. 5.5.**).

5.3.1.5. Certificando e distribuindo chaves

A chave pública, a identificação do usuário (incluindo nome e endereço eletrônico do proprietário da chave) e uma ou mais assinaturas digitais para a chave pública e para identificação do usuário, são itens essenciais para o certificado da chave pública, que é a ferramenta que habilita o amplo uso do PGP [Stallings 95].

No PGP os usuário assinam as chaves uns dos outros e criam uma comunidade interconectada de usuários de chaves públicas [Stallings 95], diferentemente do enfoque do PEM, o qual depende de uma Autoridade Central (AC) de certificação das chaves, onde a AC é responsável pelo registro de todas as chaves e uma vez que todos confiam na AC, uma chave assinada pela autoridade é automaticamente aceita como válida.

Vejamos como funciona a certificação no PGP: um usuário A passa

fisicamente sua chave pública para um usuário B que a assina. O usuário B guarda uma cópia da chave assinada e devolve uma cópia para A. Quando A que se comunicar com um usuário C, envia para C a sua chave pública e a chave assinada por B. C que também tem a chave pública de B e que confia em B para certificar chaves, verifica a assinatura de B na chave de A para finalmente aceitá-la como válida [Stallings 95]. O usuário A pode também confiar em um usuário E e utilizar o mesmo procedimento.

Para assinar uma chave utiliza-se a opção **-ks**, para remover a assinatura de uma chave a opção **-krs**. Pode-se também assinar sua própria chave usando o comando: `pgp -ks <seu endereço de e-mail> -u <seu endereço de e-mail>`.

5.3.1.6. Comandos PGP

Podemos perceber pelos exemplos dos comandos PGP de como executar operações de criptografia, assinatura, gerenciamento etc. que existem diversas opções diferentes e que podem ser combinadas, para realizar a tarefa desejada pelo usuário. A quantidade de opções leva à necessidade do usuário aprender a manipular corretamente o software, pois a utilização indevida pode acarretar danos como a perda de acesso a arquivos importantes (devido a perda da chave) ou a não privacidade de dados que precisam permanecer secretos.

5.3.2. PEM

O PEM é o padrão Internet para prover segurança na troca de e-mails. Os RFCs 1421-1424 especificam procedimentos, para prover a autenticidade, privacidade e gerenciamento de chaves, que utilizam a criptografia simétrica e assimétrica.

O PEM utiliza dois tipos de chaves. As Data Encryption Keys (DEKs) e as Interchange Keys (IKs). As DEKs são chaves únicas utilizadas para cifrar o texto e as IKs criptografam as chaves DEKs, utilizando criptografia de chave pública.

Conforme foi dito anteriormente, no PEM as chaves públicas (IKs) devem ser assinadas por uma Autoridade Central e somente chaves oficialmente

assinadas podem ser usadas. Assim, o PEM define um suporte para arquitetura de gerenciamento de chaves baseado na certificação de chaves públicas [RFC 1422]:

A infra-estrutura especificada estabelece uma raiz única para toda certificação dentro da Internet que é o Internet Policy Registration Authority (IPRA). Ou seja, o IPRA é Autoridade Central, anteriormente citada. O IPRA estabelece políticas globais, aplicadas a toda certificação abaixo desta hierarquia. O IPRA age como a raiz da hierarquia de certificados para a comunidade Internet. A chave pública do IPRA é usada para validação de todos os certificados dentro da hierarquia.

Abaixo do IPRA existem Policy Certification Authorities (PCAs), cada uma das quais estabelece e publica suas políticas para registro de usuários e organizações. Cada PCA é certificado pela IPRA. Abaixo dos PCAs estão as Certifications Authorities (CAs). As CAs são estabelecidas para certificar usuários e entidades de uma organização (departamentos, escritórios etc.). Cada PCA deve armazenar com o IPRA a descrição da política de segurança proposta. Uma cópia do documento assinado pelo IPRA fica disponível para acesso via e-mail. Como parte do registro, cada PCA necessitará concordar com a política do IPRA e pagar uma taxa para os custos de operação do IPRA.

O conceito de certificado de chave pública está definido no padrão X.509 [RFC 1422]. Essa estrutura é assinada criptograficamente através da chave privada da entidade organizacional. Uma vez assinado, os certificados podem ser armazenados em servidores e distribuídos através de canais não-seguros.

Para criptografar uma mensagem usando PEM, o remetente deve ter o certificado de cada destinatário e validar estes certificados. A validação é executada checando a assinatura digital no certificado, usando a chave pública da entidade organizacional que o assinou, com sua chave privada.

O tempo de validade de cada certificado e as listas de revogação de certificados (Certificate Revocation Lists - CRLs) são verificados para garantir que nenhum certificado tenha sido revogado ou o prazo de validade tenha expirado.

Um requerimento fundamental para o esquema de certificação é que certificados tenham um nome único. O IPRA não certifica dois PCAs com o

mesmo nome nem um PCA certifica dois CAs com o mesmo nome.

Uma vez que o certificado para um destinatário tenha sido validado, a chave pública do certificado é extraída e usada para criptografar a chave de criptografia de dados (DEK), que por sua vez é utilizada para criptografar a mensagem. O resultado da chave DEK criptografada é colocado no campo de cabeçalho "Key-Info:". O destinatário utilizará então a chave privada para decriptograr a chave DEK e a chave DEK para decriptografar a mensagem.

Para autenticação e integridade da mensagem o remetente gera um código de integridade da mensagem (MIC), e o criptografa, o resultado é colocado no campo de cabeçalho "MIC-Info:".

A arquitetura descrita no [RFC 1422] descreve procedimentos para registro de certificação para autoridades e usuários, para a geração e distribuição de certificados e para a certificação e distribuição de CRLs (listas de revogação de certificados).

5.3.2.1. Usuários e Agentes Usuários

Um UA que suporta PEM deve proteger a chave privada da entidade associada (usuário ou lista). A chave privada deve ser armazenada criptografada, protegida através de criptografia simétrica ou pode ser mantida em disquete, que só é utilizado quando o usuário recebe mensagens criptografadas.

Apenas o próprio usuário deve ter o acesso a sua chave privada. Uma vez que o usuário armazena as mensagens criptografadas, se a chave for perdida ou trocada as mensagens se tornam inacessíveis. Para resolver esse problema recomenda-se que as mensagens sejam armazenas no formato MIC-ONLY ou MIC-CLEAR. O formato MIC-ONLY especifica que todos os recursos utilizados para criptografia da mensagem, com exceção da confidencialidade, foram aplicados à mensagem (i.e. a mensagem foi apenas assinada). No formato MIC-ONLY a mensagem passa por o processo de codificação *printable encoding*. O processo de codificação *printable encoding* é o mesmo processo de codificação base64, já descrito no capítulo 3; existe uma tabela de codificação definida, que utiliza um subconjunto de 64 caracteres do International Alphabet IA5, onde 6 bits são utilizados para representar um

caractere (os 64 caracteres são representados da mesma forma em IA5 e ASCII). Uma mensagem assinada apenas, não precisa da chave privada para decriptografia. O MIC-CLEAR não passa pelo processo de codificação *printable encoding.*

5.3.2.2. Gerenciamento de CRL

Entre os procedimentos articulados por cada PCA estão as regras para manutenção e distribuição de CRLs pelo PCA e pelos CAs subordinados.

A freqüência de distribuição das CRLs deve variar de acordo com a política do PCA. Mas todo PCA e CA deve distribuir CRLs para prover a base para procedimentos de validação de certificados uniforme através da hierarquia Internet. O IPRA manterá uma CRL para todos os PCAs que certifica e que será atualizado mensalmente. Cada PCA irá manter uma CRL para todos os CAs que certifica.

5.3.3. Comparação entre PEM e PGP

A primeira diferença fundamental está no gerenciamento de chaves no PEM e PGP. Conforme descrevemos nas duas seções anteriores (3.4 e 3.5), o PGP tem um mecanismo informal de certificação de chaves que se baseia numa "teia de confiança" entre os seus usuários, que assinam as chaves uns dos outros, as chaves dos usuários podem ser distribuídas por e-mail, podem ser disponibilizadas como informações finger ou armazenadas em servidores de chaves PGP. O PEM especifica um hierarquia de certificação de chaves e só as chaves certificadas por uma autoridade podem ser utilizadas.

O PEM tem ainda mais uma restrição: apenas e-mails assinados podem ser enviados. O que no PGP é uma opção do usuário. Mas, no PGP, se uma mensagem está assinada, o usuário precisa primeiro decriptografá-la para poder verificar sua autenticidade. No PEM a assinatura está no envelope da mensagem, o que perimite que se verifique a sua integridade, sem necessitar lê-la.

Além disso, PGP é um produto e PEM é um padrão. Produtos que foram desenvolvidos seguindo as especificações PEM, como o RIPEM, não implementam a totalidade de suas especificações. O RIPEM, por exemplo, não

utiliza a certificação (descrita na seção 5.3.2).

5.4. Resumo

Neste capítulo abordamos a segurança na troca de e-mails. Abordamos, rapidamente, como são tratados os aspectos de segurança no X.400 e UUCP. E abordamos mais especificamente a segurança na troca de e-mails Internet. Apresentamos o PGP e o PEM. O PGP é um software de criptografia, utilizado para cifrar mensagens que trafegam pela rede e para criptografia de arquivos na máquina do usuário. O PEM é um padrão para segurança na troca de mensagens na Internet, especificado nos RFCs 1421-1424. Mostramos também uma comparação entre o PGP e o PEM, onde abordamos suas principais diferenças.

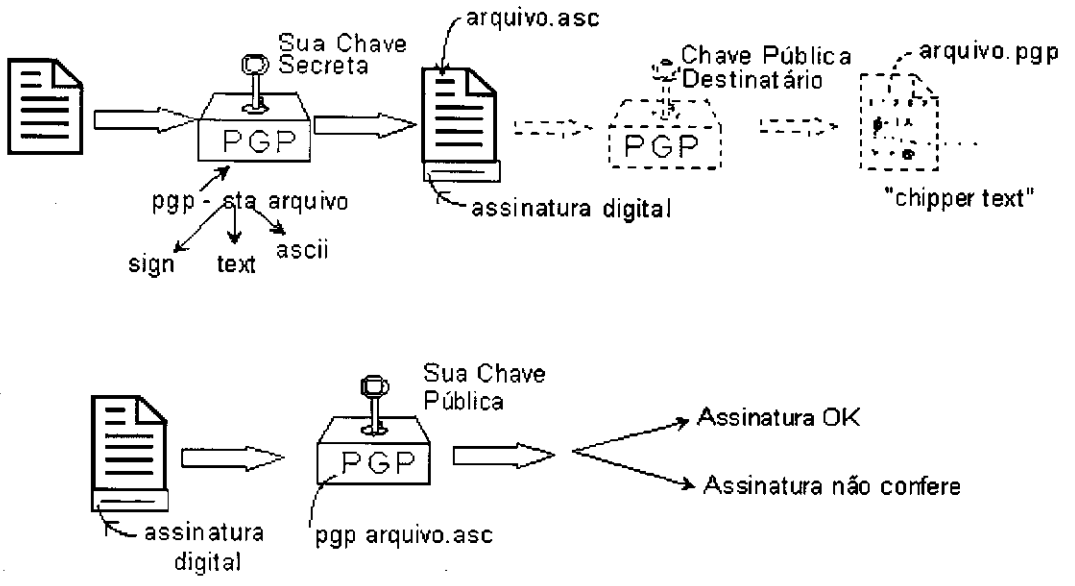


Figura 5.5. Exemplo de assinatura digital usando PGP

Freqüentemente, além de assinar a mensagem para provar sua autenticidade, existe a necessidade de privacidade. Com o PGP pode-se também assinar e criptografar uma mensagem utilizando a opção **-se**. Assina-se a mensagem com a chave secreta do remetente e criptografa-se com a chave pública do destinatário (**Fig. 5.5.**).

5.3.1.5. Certificando e distribuindo chaves

A chave pública, a identificação do usuário (incluindo nome e endereço eletrônico do proprietário da chave) e uma ou mais assinaturas digitais para a chave pública e para identificação do usuário, são itens essenciais para o certificado da chave pública, que é a ferramenta que habilita o amplo uso do PGP [Stallings 95].

No PGP os usuário assinam as chaves uns dos outros e criam uma comunidade interconectada de usuários de chaves públicas [Stallings 95], diferentemente do enfoque do PEM, o qual depende de uma Autoridade Central (AC) de certificação das chaves, onde a AC é responsável pelo registro de todas as chaves e uma vez que todos confiam na AC, uma chave assinada pela autoridade é automaticamente aceita como válida.

Vejamos como funciona a certificação no PGP: um usuário A passa

6

SSMTP e LMSP

Em nossa solução para um ambiente com recursos de segurança na troca de e-mails Internet, procuramos deixar a segurança o mais transparente possível para o usuário. Assim, ao invés de termos o software de correio eletrônico e um software de criptografia, optamos por incluir os recursos de criptografia do PGP no MTA e UA. O PGP, explicado no capítulo anterior, é reconhecidamente um ótimo programa para proteger a privacidade e autenticação da correspondência eletrônica Internet, mas faz isso externamente ao programa de e-mail. Isso demanda do usuário a necessidade de aprender a usar o software corretamente, pois seu uso inapropriado pode levar à perda de informações ou não utilização dos recursos de sigilo e integridade necessários a uma determinada informação.

Assim, a nossa solução está baseada na extensão do SMTP para o Secure Simple Mail Transfer Protocol (SSMTP). O SSMTP possibilita a transmissão segura de mensagens eletrônicas em uma rede TCP/IP. Uma importante característica do SSMTP é sua não dependência de um serviço hierárquica de distribuição de chaves, como o requerido pelo PEM. Esta foi uma decisão de projeto, pois a experiência mostra que tal serviço é extremamente difícil de ser implementado [RFC 1636].

Entretanto, garantir apenas a transmissão segura não basta. É necessário ter certeza que a segurança não seja violada localmente. Dessa forma, a nossa proposta inclui ainda a especificação de procedimentos que garantem o sigilo local da informação. Ao conjunto desses procedimentos denominamos Local Mail Security Procedures (LMSP). O LMSP é composto de regras a serem utilizadas em um ambiente para e-mail Internet, na implementação de MTAs e UAs seguros.

Para incluir o sigilo e autenticação, o SSMTP adiciona comandos e respostas ao SMTP. Ele incorpora recursos de segurança através do uso de criptografia (simétrica e assimétrica) e para isso utiliza-se de um MTA seguro (Secure MTA - SMTA), implementado de acordo com o SSMTP e os procedimentos LMSP. Caso não tivéssemos o LMSP, um intruso ou mesmo o administrador do sistema poderia ler as caixas postais de outros usuários. Uma máquina que implementa o SSMTP é também servidora de chaves públicas.

O UA seguro (Secure UA - SUA), também implementado seguindo procedimentos LMSP, permite ao usuário utilizar ou não os recursos de segurança. É através do SUA que o usuário poderá ter o acesso aos e-mails criptografados, armazenados na caixa postal.

6.1. O Protocolo SSMTP

O protocolo SMTP é constituído de um conjunto de itens básicos para a troca de e-mails e tem provado ser notavelmente robusto. Todavia, a necessidade de extensões ao protocolo tem se tornado evidente [RFC 1869].

O [RFC 1869] provê uma estrutura na qual extensões podem ser especificadas de forma consistente, definindo meios através dos quais uma extensão ao diálogo SMTP pode ser reconhecida e possibilitando a um servidor informar quais as extensões que suporta. O SSMTP foi definido com base no [RFC 1869], sendo uma extensão que acrescenta ao SMTP recursos de segurança. Um e-mail criptografado, em ambiente que implementa SSMTP, possui o campo de cabeçalho "Encrypted:", preenchido com o valor "SSMTP 1.0".

O Objetivo do SSMTP é transferir e-mails entre máquinas na Internet de forma segura, através do uso de criptografia assimétrica e simétrica. Utiliza-se a criptografia simétrica para cifrar a mensagem, pois na criptografia assimétrica os algoritmos são lentos[1], o que torna muito vagarosa a cifragem e decifragem de uma grande quantidade de dados. Uma vez cifrada a mensagem, utiliza-se a criptografia assimétrica para cifrar a chave única (que

[1] A criptografia assimétrica é aproximadamente mil vezes mais lenta que a criptografia simétrica.

representa muito menos informação que a mensagem propriamente dita) usada na criptografia da mensagem, garantindo a privacidade, no seu envio pela rede. Esse é o mesmo método utilizado pelo PGP, no qual nos baseamos.

Para cifragem do e-mail será utilizado o algoritmo IDEA. A chave IDEA usada na criptografia do e-mail é chamada de chave de sessão. Ela será cifrada utilizando algoritmo RSA de criptografia de chave pública e será enviada com o e-mail para o destinatário, como valor de preenchimento do campo de cabeçalho "Key-Info:" (definido no [RFC 1421]).

A criptografia do e-mail torna a mensagem secreta. Para torná-la também autenticada (assinada) é calculada a função MD5 da mensagem e o resultado é criptografado com a chave privada do remetente. O UA do destinatário poderá, então, calcular a função sobre a mensagem recebida e comparar os resultados, detectando se houve ou não adulteração da informação. Note que o destinatário pode decriptografar a assinatura com a chave pública do remetente.

Com a assinatura é possível comprovar a origem de uma mensagem, pois somente o usuário remetente conhece a sua chave privada e portanto somente ele poderia tê-la cifrado. Um e-mail assinado possui o campo "MIC-Info:" [RFC 1421] preenchido com o valor produzido pelo cálculo da função MD5 do e-mail, criptografado com a chave privada do remetente.

A chave privada do usuário precisa permanecer secreta. Porém, a chave privada é um número muito grande (da ordem de dezenas de dígitos) e o usuário não pode escolhê-lo. Isto torna a memorização da chave praticamente impossível. Dessa forma, optamos, baseados no PGP, por armazená-la criptografada com uma senha de acesso[2]. A senha de acesso pode ser uma frase contendo espaços, pontos, letras maiúsculas e minúsculas, caracteres especiais e pode ter qualquer tamanho. Como é escolhida pelo usuário, é muito mais fácil de ser decorada. Usamos a senha de acesso para obter, através da MD5, a chave IDEA que, de fato, criptografa (e decriptografa) a chave privada.

[2] *Pass phrase*, no PGP.

6.1.1. Procedimentos do SSMTP

O SSMTP utiliza o mesmo modelo do SMTP (**Fig. 6.1**), definido em [RFC 821] e [RFC 1123], onde o UA e MTA foram substituídos pelo SUA e SMTA, respectivamente. O usuário utiliza um SUA para confeccionar a mensagem e solicita ao SMTA que encaminhe o seu e-mail ao destinatário. O processo de transferência de mensagem é executado em *background*.

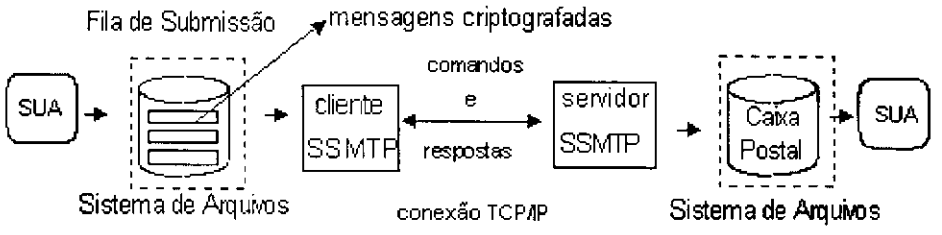


Figura 6.1. Modelo SSMTP

Estabelecida a conexão de transporte com o servidor de correio eletrônico da máquina destino, o cliente espera por uma mensagem 220 (Ready for mail). Dando prosseguimento à inicialização da transferência, o cliente envia um comando EHLO[3] e deve receber as extensões que o servidor suporta. Em nosso caso, a extensão SSMTP indica que o servidor está apto a receber e-mails criptografados.

Caso o servidor não reconheça o EHLO ou não suporte SSMTP, o MTA continuará ou não a transferência da mensagem usando o SMTP, dependendo da escolha do usuário no momento do envio do e-mail (veja seção 6.2). Caso o usuário tenha solicitado o envio obrigatoriamente com recursos de segurança, a transferência será abortada. Caso contrário, continuará sem os recursos de segurança.

O campo de cabeçalho "Security-Level:" denota o nível de segurança requerido para transmissão de um determinado e-mail. Os valores possíveis para este campo são *none*, *optional* e *mandatory*. É importante ressaltar que o valor *none* implica no fato de que o e-mail será enviado via SMTP.

[3] Conforme especificado no [RFC 1869], um cliente que suporta extensões SMTP, inicia uma sessão usando o comando EHLO.

Caso o servidor suporte SSMTP, a comunicação será estabelecida, a transação para a transmissão dos e-mails é iniciada com o comando MAIL. O comando MAIL envia a identificação do usuário remetente para a máquina destino. O receptor prepara-se para receber o novo e-mail e responde com uma mensagem 250 (Ok).

Após o a identificação do usuário, o enviador informa qual o destinatário do e-mail. Neste momento, há duas alternativas possíveis para o funcionamento do SSMTP. Caso o enviador já conheça a chave pública do destinatário, ele utiliza o comando RCPT. Caso contrário, envia um comando RCPT com o parâmetro PKEY (o parâmetro PKEY[4] indica a solicitação da chave pública do usuário destinatário). Em ambos os casos, o destinatário esta sendo especificado, embora no segundo, também esteja sendo solicitada sua chave pública.

O enviador já conhece a chave pública quando está configurado para usar uma infra-estrutura segura de distribuição de chaves. Neste caso, antes de iniciar a conexão de transferência de e-mail, o enviador faz uma consulta ao servidor de chaves solicitando a chave pública do destinatário.

Quando o enviador solicita a chave pública do destinatário durante a transferência do e-mail, o receptor age também como o servidor de chaves. Um comando RCPT com parâmetro PKEY deve ser respondido com a chave pública do destinatário, com uma mensagem de usuário não tem chave pública ou com a mensagem de usuário inválido. Embora este procedimento não seja tão seguro quanto a obtenção da chave de um serviço especialmente projetado para este fim (veja seção 6.3), ele garante a independência de tais serviços. Este é um ponto fundamental no SSMTP, pois a dependência de um serviço externo para obtenção de chaves tem inviabilizado o uso de outras soluções (tal como o PEM), exatamente porque estes serviços implicam na criação de uma hirarquia segura de certificação de chaves que abranja toda a Internet, o que representa um enorme esforço administrativo e de suporte.

[4] Foi previsto em [RFC 1869] que extensões ao SMTP poderiam utilizar parâmetros adicionais associados aos comandos MAIL e RCPT.

Após um comando RCPT bem sucedido, o cliente envia o comando DATA, que informa ao servidor que o conteúdo de um e-mail será transferido. O comando DATA negocia com a máquina destino o inicio do envio do e-mail criptografado. Se aceito, transmitir-se-á o e-mail. O servidor deverá responder com uma mensagem 250 (Ok).

Para encerrar, é enviado um comando QUIT. O comando QUIT deve receber uma resposta 221 concordando com o encerramento da conexão, ou uma resposta 500 se houver algum erro. Após isso, a conexão TCP/IP é encerrada. Veja na **Fig. 6.2**, o exemplo de uma transação SSMTP.

```
Solicitação de conexão TCP
S: 220 fenix.cgsoft.softex.br Ready
C: EHLO anjinho.dsc.ufpb.br
S: 250-fenix.cgsoft.softex.br
S: 250-SSMTP
S: 250 8BITMIME
C: MAIL From: <maria@dsc.ufpb.br>
S: 250 ok
C: RCPT To: <joao@cgsoft.softex.br> PKEY
S: 250-ok
S: 250
I3rRIGXUGWAF8js5wCzRTkdhO34PTHdRZY9Tuvm03M+NM7fx6qc5udixps2Lng0+wGrtiUm/ovtK
S: dinz6ZQ/aQ==
C: DATA
S: 354 Enter mail, end with .
C: Date: Sun, 28 Jan 1996 23:05:00 -0200 (EDT)
C: From: Maria Silva <maria@dsc.ufpb.br>
C: To: joao@cgsoft.softex.br
C: Encrypted: SSMTP 1.0
C: Key-Info: I3rRIGXUGWAF8js5wCzRTkdhO34PTHdRZY9Tuvm03M+NM7fx6qc5udixps2Lng0+
wGrti
C: Um/ovtKdinz6ZQ/aQ==w78xodj
C: MIC-Info:
UdFJR8u/TIGhfH65ieewe2lOW4tooa3vZCvVNGBZirf/7nrgzWDABz8w9NsXSexvAjRFbH
C: oNPzBuxwmOAFeA0HJszL4yBvhG
C: Subject: Informacoes
C:
C: LLrHB0eJzyhP+/fSStdW8okeEnv47jxe7SJ/iN72ohNcUk2jHEUSoH1nvNSIWL9M
C: 8tEjmF/zxB+bATMtPjCUWbz8Lr9wloXIkjHUlBLpvXR0UrUzYbkNpk0agV2IzUpk
C: J6UiRRGcDSvzrsoK+oNvqu6z7Xs5Xfz5rDqUcMlK1Z6720dcBWGGsDLpTpSCnpot
C: dXd/H5LMDWnonNvPCwQUHt==
C: .
S: 250 ok
C: QUIT
S: 221 fenix.cgsoft.softex.br Service closing transmission channel
Liberação da conexão TCP
```

Figura 6.2. Exemplo de diálogo cliente/servidor SSMTP

Além dos procedimentos de uma transação para o envio de um e-mail, temos um comando adicional, o PKEY (note que existe também o parâmetro PKEY do comando RCPT). O PKEY é usado em uma nova conexão SSMTP, estabelecida apenas com o propósito de pegar a chave pública do usuário remetente, necessária para que se proceda a verificação de assinatura de um e-mail. Com uma nova conexão pretendemos obter mais segurança, dificultando a falsificação do remetente.

6.1.2. Comandos SSMTP

Assim como os comandos SMTP, especificados no [RFC 821], os comandos SSMTP são cadeias de caracteres terminadas por <CRLF>. Os códigos dos comandos são caracteres alfabéticos terminados por espaço, se houver algum parâmetro, ou por <CRLF>, se não houver parâmetros.

Além dos comandos SMTP mandatórios (capítulo 3, seção 3.1.1), o SSMTP suporta comandos adicionais, definidos como extensões ao SMTP:

EHLO

- Este comando é utilizado para identificar as máquinas que implementam SSMTP. O argumento do comando identifica o nome da máquina cliente.

RCPT To: <usuario@maquina> PKEY

- Este comando é utilizado para informar à máquina servidora qual o usuário que recebe o e-mail e solicitar a chave pública de criptografia deste usuário. O argumento é a identificação do usuário destinatário e o parâmetro PKEY indica a solicitação de chave pública de criptografia do usuário.

PKEY From: <usuario@maquina>

- Este comando é usada em uma conexão SSMTP para solicitação da chave pública do usuário, para decriptografia de e-mails assinados. O argumento é a identificação do usuário remetente.

6.1.3. Sintaxe dos comandos SSMTP

Os comandos consistem de códigos seguidos por um argumento. Os códigos são caracteres alfabéticos. Não há distinção entre caracteres em maiúscula ou minúsculas. O argumento consiste de uma seqüência de strings terminadas por <CRLF>.

6.1.3.1. O comando EHLO

A sintaxe para o comando EHLO [RFC 1869], usando a notação ABNF de [RFC 822] é:

```
ehlo-cmd      ::= "EHLO" SP domínio CR LF
```

E a resposta positiva ao comando EHLO é [RFC 1869]:

```
ehlo-ok-rsp   ::= "250" domain [ SP greeting ] CR LF
                  ("250-" domain [ SP greeting ] CR LF
                  *("250-" ehlo-line CR LF)
                  "250" SP ehlo-line CR LF)
                  ; the usual HELO chit-chat
greeting      ::= 1*<any character other than CR or LF>
ehlo-line     ::= ehlo-keyword *( SP ehlo-param )
ehlo-keyword  ::= (ALPHA / DIGIT) *(ALPHA / DIGIT / "-")
ehlo-param    ::= 1*<any CHAR excluding SP and all control
          characters              (US ASCII 0-31 inclusive)>
ALPHA         ::= <any one of the 52 alphabetic characters (A
          through        Z in upper case, and, a through z in
          lower case)>
DIGIT         ::= <any one of the 10 numeric characters (0 - 9)>
CR            ::= <the carriage-return character (ASCII code 13)>
LF            ::= <the line-feed character (ASCII code 10)>
SP            ::= <the space character (ASCII code 32)>
```

6.1.3.2. Respostas ao comando EHLO

Se o servidor com o qual o cliente está se comunicando também suporta a extensão, o cliente irá obter uma resposta de código 250, caso contrário receberá um código de falha diferente dos gerados pelo SMTP (500, 501, 502, 504 ou 421).

Caso seja recebida uma mensagem de falha, código de mensagem 554, o cliente não implementa o serviço de extensão SMTP. Abaixo seguem as repostas e os significados:

```
250       ok
554       servidor, por alguma razão, não lista as extensões que
          suporta
501       argumentos incorretos
502       servidor reconhece o ESMTP, mas não implementa o EHLO
421       servidor fora de operação
500       servidor não suporta ESMTP
```

6.1.3.3. O comando RCPT To: <usuario@maquina> PKEY

A sintaxe para o comando RCPT To PKEY, usando a notação ABNF do [RFC822] é:

```
rcpt-cmd      ::= "RCPT TO:" recipient [ SP PKEY ] CR LF
recipient     ::= "<" keyword "@" domain ">"
domain        ::= keyword / (keyword "." domain)
keyword       ::= (ALPHA / DIGIT) *(ALPHA / DIGIT / "-")
```

6.1.3.4. Respostas ao comando RCPT TO: <usuario@maquina> PKEY

O comando RCPT do SMTP permite verificar a existência de uma caixa postal para o usuário destinatário na máquina destino. Além disto, o parâmetro adicional PKEY permite verificar qual a chave pública do destinatário. Os possíveis erros, definidos no [RFC 1869] são:

```
555 servidor não reconhece/implementa parâmetros adicionais
455 servidor, por alguma razão temporária, não aceita o parâmetro
```

6.1.3.5. Comando PKEY From: <usuario@maquina>

A sintaxe para o comando PKEY, usando a notação ABNF do [RFC822] é:

```
pkey-cmd      ::= "PKEY FROM:" recipient CR LF
```

6.1.3.6. Respostas ao comando PKEY

As possíveis respostas ao comando PKEY são:

```
250       chave pública do usuario@maquina
500       comando não reconhecido
501       erro de sintaxe nos parâmetros ou argumentos
```

6.2. O LMSP

O LMSP constitue-se de um conjunto de regras que visam garantir a segurança local em uma máquina que implementa o SSMTP.

A regra número um é referente ao grau de segurança que o usuário estabelece para transmissão. Um e-mail pode ser enviado com três graus de segurança: obrigatória, opcional e nenhuma. A opção segurança obrigatória significa que o usuário estabeleceu que o e-mail só deve ser transmitido se a máquina destino também implementa recursos de segurança, i.e., utiliza SSMTP. A opção de segurança opcional indica que o e-mail deve ser enviado preferencialmente de forma segura, mas que o usuário permite que o e-mail

seja enviado caso a máquina destino não implemente recursos de segurança, ou seja, use apenas SMTP. Obviamente, quando nehuma segurança é especificada para um determinado e-mail, ele é transferido usando apenas SMTP.

Note que, de acordo com o SSMTP, a definição do nível de segurança deve constar no campo "Security-Level:" do cabeçalho da mensagem (veja 6.1). Quando a segurança for obrigatória, este campo é preenchido com o valor *mandatory*. Para segurança opcional, o valor usado é *optional* e, por fim, um e-mail sem segurança tem o valor *none* neste campo.

A regra número dois, diz respeito a segurança do conteúdo dos e-mails. Os conteúdos dos e-mails recebidos devem ser armazenados criptografados. Se os e-mails forem recebidos já criptografados, são simplesmente armazenados pelo SMTA. Senão, são criptografados[5] e armazenados pelo SMTA.

A regra número três, estabelece que o SUA deve arquivar os e-mails nos *folders* devidamente criptografados.

A quarta regra especifica que os e-mails devem ser armazenados na área de enfileiramento de mensagens já cifrados pela chave de sessão. A chave de sessão é uma chave IDEA, utilizada para criptografia do e-mail. Ela é criptografada com a chave pública do usuário destinatário e enviada com o e-mail.

Se o usuário tiver determinado segurança *mandatory* (de acordo com a regra número um), a chave de sessão fica armazenada apenas na memória do SMTA, até a obtenção da chave pública do destinatário e seu encaminhamento. Isto implica que se o SMTA ou a máquina caírem, os e-mails enfileirados que solicitaram segurança total serão devolvidos ao remetente. Caso tenha sido solicitado segurança parcial, a chave de sessão é também guardada em um arquivo, o que possibilita a recuperação de uma falha do servidor.

A quinta regra diz respeito à segurança do SUA e SMTA: como forma de

[5] Neste caso, o e-mail foi enviado sem segurança, mas a máquina destino implementa SSMTP e dispõe da chave pública do usuário destinatário.

garantir que o SUA e o SMTA não sejam modificados sem que essas alterações possam ser detectadas, devem existir disponíveis, em um servidor ftp, os programas fontes que testam a integridade de ambos, podendo o próprio usuário compilá-los em caso de suspeita de adulteração.

Esses programas devem utilizar MD5 (Message Digest 5). A MD5, a partir de um texto, produz um número de 128 bits. Há uma conjectura de que é computacionalmente impossível produzir um texto a partir de um resultado MD5 [RFC 1321]. Assim, um eventual intruso não consegue adulterar o SUA (ou o SMTA) mantendo o mesmo resultado MD5, nem mesmo através da inclusão de um *dumb string*.

Dessa forma, objetivamos proteger as informações do usuário também localmente e não apenas durante a comunicação com outras máquinas.

6.3. Análise da Segurança

A nossa solução oferece um forma mais segura para troca de e-mails na Internet. Mas, na alternativa em que utilizamos a máquina servidora SSMTP também como servidora de chaves públicas, há, ainda, um problema de segurança nas transmissões: um impostor, num roteador, pode se fingir de destino, via a alteração do transporte TCP/IP deste roteador. Ao se fingir de destino, o impostor pode fornecer uma chave pública falsa para o destinatário. Isso possibilita o acesso à mensagem cifrada, uma vez que ele possui a chave privada do par pública/privada que forjou.

Todavia, grande parte dos problemas é resolvida, mesmo porque a maior parte deles é local, provocados por acessos indevidos ao sistema. Assim, a implementação dos procedimentos descritos no LMSP torna a quebra da segurança do sistema de e-mail muito mais difícil.

Além do mais, interceptar mensagens num roteador é difícil, pois há roteamento dinâmico[6] no *backbone* Internet, ou seja, existe mais de uma rota possível para um mesmo destino, e as mensagens podem ser divididas em

[6] No roteamento dinâmico as tabelas de rotas são construídas automaticamente por protocolos projetados para este fim. Os protocolos ajustam dinamicamente as rotas, refletindo mudanças nas condições da rede.

pacotes que seguem caminhos diferentes. É também muito difícil comprometer a segurança de roteadores dedicados e, mais ainda, alterar a implementação TCP/IP destes roteadores.

Para se ter mais segurança exclusivamente através da rede, há um alto preço a pagar: é necessário mais suporte técnico, mais *overhead* e, claramente, isso pode demandar recursos vultuosos[7]. A opção por um canal seguro para transmissão da chave, i.e., fora da rede, também traz problemas: sempre que se imagine que esta chave esteja comprometida ou, ainda, se os requisitos de segurança determinarem a alteração periódica das chaves, uma nova chave terá que ser gerada e, novamente, transportada. O pior é que não há como fazer com que duas pessoas estabeleçam uma chave de forma completamente segura, exceto se elas se encontrarem pessoalmente.

Assim, a nossa solução oferece um esquema alternativo, onde ao se utilizar o servidor de e-mail como servidor de chaves públicas, não temos um sistema completamente seguro. Por outro lado, oferecemos um custo/benefício muito bom, onde se tem o melhor possível em segurança para troca de e-mails Internet, sem demandar suporte especial para a distribuição de chaves. Esta abordagem atende grande parte dos usuários, pois a maioria das mensagens que trafegam na Internet não justificam o esforço de quebrar o esquema de segurança que propomos, e se o custo requerido para quebrar um sistema ou decifrar uma mensagem é maior que o valor da informação que poderá ser obtida, então, para todos os fins práticos, o sistema é seguro [Weber 95].

Para os casos em que é necessário mais segurança, existe a flexibilidade de se trabalhar com servidores de chaves, onde as chaves armazenadas são certificadas.

6.4. Resumo

Neste capítulo apresentamos a nossa proposta de segurança em e-mail Internet, mostrando a especificação do protocolo SSMTP e das regras LMSP. O SSMTP é a nossa extensão do SMTP, a qual acrescentamos recursos para

[7] Pode-se optar por utilizar uma hierarquia de certificação como a do PEM, que demanda de custos adicionais.

a transmissão de e-mails criptografados. O LMSP constitue-se de um conjunto de regras para garantia da segurança local em máquinas que implementam um sistema de e-mail Internet. Fazemos também uma análise de custo/benefício da nossa proposta de segurança na troca de e-mails Internet.

7

ssmtpd e spine

Embora o SSMTP defina de forma clara quais mecanismos garantem a segurança do envio de correio eletrônico em redes TCP/IP e o LMSP assegure que a privacidade dos e-mails não será violada localmente, ainda há algumas decisões que podem ser tomadas no momento da implementação destas especificações. Apresentamos aqui o SMTA ssmtpd e o SUA spine, construídos em conformidade com o SSMTP e o LMSP. O objetivo é mostrar como alguns detalhes de implementação podem ser bem resolvidos no momento de tirar do papel as idéias até aqui apresentadas e colocá-las para funcionar.

7.1. ssmtpd

O ssmtpd é um MTA de configuração muito mais simples que o sendmail, substituindo-o sempre que toda a troca de correio eletrônico for feita através de redes TCP/IP. Caso seja necessário utilizar outros protocolos para troca de e-mail (como o X.400 e UUCP), utilizar-se-á o sendmail para escolher entre os diversos MTAs instalados no servidor. Ressalvamos, entretanto, que a grande maioria dos servidores de e-mail conectados à Internet utilizam apenas o TCP/IP como mecanismo de troca de correio. Assim sendo, o ssmtpd resolverá o conhecido problema de configuração do sendmail na maior parte dos casos.

O ssmtpd é capaz de receber e enviar e-mails via SSMTP, tendo também suporte ao SMTP tradicional. Além disso, ele entrega e-mails locais. O ssmtpd roda em *background* e, para processar os e-mails que estão chegando, fica "ouvindo" na porta TCP 25, a mesma utilizada eplo SMTP.

O ssmtpd tem suporte a transferência de mensagens no formato 8bit

MIME[1]. Ou seja, não estabelece restrições aos caracteres que compõem a mensagem. Essa é uma característica importante, pois também representa uma extensão ao SMTP, que suporta apenas o transporte de dados que estejam representados em caracteres ASCII imprimíveis de 7 bits. Por isso, quando o SMTP "puro" é usado para o transporte de outros tipos de dados via e-mail, é necessário compatibilizar o formato dos dados através das codificações MIME base64 ou quoted-printable.

No ssmtpd encontramos as seguintes operações básicas:

- Criptografia do e-mail usando o algoritmo IDEA e criptografia da chave de sessão usando RSA.
- Assinatura do e-mail, usando RSA para criptografar, com a chave privada do remetente, o resultado MD5 da mensagem.
- Envio e recepção do e-mail sobre um suporte TCP/IP.
- Envio de e-mails locais.
- Fornecimento de chaves públicas.

No momento da transação SSMTP, ao receber a chave pública do usuário destinatário, o ssmtpd criptografa a chave de sessão gerada para o e-mail, que é cifrado com o algoritmo IDEA. A chave de sessão é criptografada com o algoritmo RSA. Como já foi explicado antes, os algoritmos de criptografia de chave pública são muito lentos, por isso opta-se por criptografar o contéudo da mensagem com algoritmos de chave única.

A assinatura do e-mail é feita aplicando-se a MD5 sobre o texto do e-mail. Ao aplicar MD5 é produzido um número de 128 bits que é criptografado com a chave privada (do par pública/privada) do usuário remetente.

O arquivo de chaves públicas e privadas, o public.key e o secret.key, respectivamente, contendo as chaves de todos os usuários do servidor ficam disponíveis em área do sistema. O public.key será consultado para fornecer as chaves públicas no momento de uma transação SSMTP de envio de e-mails e o secret.key é consultado para assinatura de e-mails e para decriptografia das

[1] O MIME foi abordado no capítulo 3.

mensagens, no momento que o usuário tem acesso à sua caixa postal.

É importante ressaltar que cada chave contida no secret.key é criptografada através do IDEA com o resultado MD5 da senha de acesso de seu dono. Além da chave, é criptografado também uma constante arbitrária (*magic number*) de 128 bits. Esta constante permite ao spine descobrir se a senha de acesso fornecida por um suposto usuário é a correta.

O ssmtpd provê a chave pública do usuário remetente, para o caso em que os e-mails são também assinados. Para isso, é verificado o preenchimento ou não do campo de cabeçalho "MIC-Info:". Caso esteja preenchido, uma nova conexão é estabelecida pelo SMTA para solicitar a chave pública do usuário remetente, após o recebimento da mensagem. Como discutido anteriormente, o objetivo de utilizarmos uma nova conexão é o de obtermos mais segurança.

O ssmtpd só reconhece endereçamento padrão Internet (usuário@domínio) ou local (usuário). Uma máquina que tenha mais de um MTA para transferência de e-mails terá que, obrigatoriamente, usar o sendmail, que age como despachador para os diversos MTAs que compõem um sistema de correio eletrônico. Nesta situação, o sendmail deve ser configurado para passar o processamento para o ssmtpd todas as vezes que o endereço do destinatário for Internet ou local.

No ssmtpd existem duas formas básicas de como o e-mail será encaminhado, uma localmente (**Fig. 7.1a**) e a outra remotamente, via SSMTP (**Fig. 7.1b**).

Figura 7.1a. Mensagem local

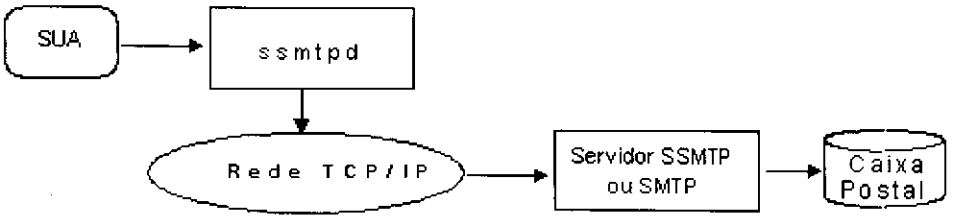


Figura 7.1b. Mensagem SSMTP

O ssmtpd, por só encaminhar e-mails locais e SSMTP, é muito mais fácil de configurar que o sendmail (citado na seção 3.5), como o MTA mais utilizado em máquinas Unix). A maioria das máquinas utilizam apenas estas formas de encaminhar e-mails, não tendo suporte a vários MTAs em um único servidor. Dessa forma, estaremos facilitando a configuração para a maior parte dos casos. Pois, a flexibilidade que o sendmail oferece para atender as necessidades dos diferentes MTAs usados num sistema de e-mail, tornam a sua configuração uma tarefa complicada.

7.2. spine - Secure Pine

O Program for Internet News and Email (pine) é um ambiente de manipulação de e-mails (i.e. um UA) desenvolvido pela Universidade de Washington, em Seattle, para usuários finais. O pine utiliza protocolos padrões de mensagens na Internet, como o SMTP, MIME, IMAP e NNTP e está disponível para Unix, DOS/Windows e Macintosh.

O pine permite a manipulação da caixa postal de correio eletrônico padrão e também de *folders* criados pelo usuário. A edição de mensagens é em "tela cheia". É um agente usuário bastante poderoso e de fácil utilização, com menus disponíveis em todos os seus estados[2]. A interface com o usuário é bastante amigável, com comandos de apenas um caractere, menus de

[2] No pine os estados são as opções do menu principal, ou seja, as diversas funcinalidades que o pine oferece.

comandos sempre presentes e alta tolerância aos erros dos usuários. Devido a esses fatores, escolhemos o pine como referência para o desenvolvimento do nosso agente usuário seguro (SUA), o spine.

O spine solicita a senha de acesso do usuário ao ser executado. Uma vez que a senha de acesso tenha sido fornecida corretamente (o que pode ser verificado graças a constante arbitrária criptografada junto com a chave privada), os e-mails do usuário que estiverem com o campo de cabeçalho "Encrypted:" preenchido com o valor "SSMTP 1.0" poderão ser decriptografados para leitura no momento de sua exibição.

Assim como no pine, o spine apresenta um menu principal ao usuário, que é identificado por MAIN MENU e pela versão do spine, na parte superior da tela. No menu principal estão exibidas claramente as opções do spine. O arquivo de mensagens recebidas é denominado INBOX e o arquivo de mensagens enviadas sent-mail. A diferença em relação ao pine é que estes arquivos guardam as mensagens criptografadas. O comando **C** do menu principal e o comando **R** no estado de índice do *folder* (resumo do cabeçalho das mensagens do *folder*, ativado com a comando **I**) ativa o modo de confecção do e-mail.

O spine tem uma configuração *default*, armazenada no arquivo .spinerc, onde estão ativadas todas as opções de segurança, quais sejam: o nível de segurança é colocado em obrigatório e os e-mails são assinados. Essas opções pré-configuradas podem ser alteradas pelo usuário, via o comando **S** (Setup) do menu principal do spine, que permite o acesso ao sub-comando **S** (Security).

O usuário pode modificar as opções de segurança apenas para o e-mail que está confeccionando através do comando **^S** (control-S). Isto permite alterar o nível de segurança e também habilitar/desabilitar a assinatura digital somente para uma determinada mensagem.

Quanto a assinatura, há três possibilidades para os e-mails recebidos: não foram assinados, foram assinados e foram falsificados (i. e., foram assinados, mas a assinatura não confere). Quando o spine está mostrando a lista de e-mails (ou índice) de um *folder*, a primeira coluna denota o *status* de cada e-mail: **N** mostra que o e-mail não foi assinado, **S**, que foi e **F**, que foi

forjado (veja a **Fig. 7.2**). Esta informação também está disponível no momento de exibição do e-mail. Neste caso, as quatro posições mais a direita da primeira linha conterão **NSIG**, **SIGN** ou **FAKE** para denotar, respectivamente, que o e-mail não foi assinado, foi assinado ou foi falsificado.

```
SPINE 1.00   FOLDER INDEX                Folder: INBOX  Message 58 of 88

S      58  Jun 24 Jacques Philippe S  (3,049) Re: Artigo
S + A 59  Jun 24 Monica Fernandes    (1,861) Favorzinho
N      60  Jun 25 Jose E M de S Bran  (2,487) DNS fantasma?
F + A 61  Jun 25 mmelo@redebrasil.c  (1,860) Re: Noticias
N      62  Jun 25 Associacao dos Pro (10,986) [CNPQ-L] SOS UNIVERSIDADE
S + A 63  Jun 25 Walfredo Cirne Fil  (1,104) Re: Ser ou nao ser...
S      64  Jun 25 Walfredo Cirne Fil  (1,719) Re: Correio seguro
S      65  Jun 25 Walfredo Cirne Fil  (2,331) Re: Artigo
S + A 66  Jun 26 Ernandes Lopes Bez  (1,538) News
N      67  Jun 26 CERT Advisory       (14,759) CERT Advisory CA-96.12 -
F      68  Jun 26 Stephen Kent        (10,570) draft meeting minutes
S    A 69  Jun 27 Ernandes Lopes Bez  (2,542) News
N      70  Jun 27 VICENTE DE PAULO A  (1,306) CURSO DE ESPECIALIZACAO E
N      71  Jun 27 lucas_avila@easyli  (1,197)
N      72  Jun 27 Coordenacao da Bib  (4,752) Informativo Miniblio
N + A 73  Jun 28 Ismenia Mangueira     (741) UNIX
N +    74  Jul  1 Ismenia Mangueira     (582) Obrigada
S      75  Jul  1 Walfredo Cirne Fil  (1,283) Banca de Zane
F + A 76  Jul  1 Acesso aa estacao      (818) Sinais de vida!!!

? Help       M Main Menu  P PrevMsg   - PrevPage  D Delete   R Reply
O OTHER CMDS V [ViewMsg]  N NextMsg Spc NextPage  U Undelete F Forward
```

Figura 7.2. Tela do spine mostrando a lista de e-mails no INBOX

7.3. Resumo

Neste capítulo, fazemos algumas considerações sobre a implementação de um SMTA, o ssmtpd e de um SUA, o spine. O ssmtpd implementa o SSMTP e as regras LMSP. O spine está em consonância com as regras LMSP.

8

Conclusão

O correio eletrônico é uma das principais razões para o crescimento fantástico da Internet e a sua segurança tem sido um problema desde a primeira vez em que o e-mail foi utilizado. A correspondência eletrônica pode ser interceptada e copiada sem o conhecimento do remetente e do destinatário e, mais, na maioria dos UAs utilizados, o simples ato de enviar o e-mail envolve fazer uma cópia dessa mensagem. Uma cópia fica armazenada em um *folder* de e-mails enviados, podendo vir a ser lido pelo administrador do sistema, ou até, por falta de conhecimento do usuário, acessível a todo um grupo, devido às permissões estabelecidas para o arquivo. Como se não bastasse, é muito fácil enviar um e-mail falsificando o remetente.

A solução padrão para a segurança em e-mail Internet, o PEM (com os serviços essenciais de privacidade, autenticação e gerenciamento de chaves), apresenta-se muito complexa no que se refere a infra-estrutura necessária para a hierarquia de certificação de chaves, além de ser muito pesada e trazer custos adicionais. O PGP, que é a solução que de fato tem sido utilizada, tem os inconvenientes de ser externo ao software de correio eletrônico e não especificar uma política para a distribuição segura de chaves. Portanto, o desejável é uma solução que tenha o máximo de segurança possível. com a viabilidade que tem o PGP, mas sem os inconvenientes e necessidade de infra-estrutura do PEM.

Assim, o uso de um protocolo que oferece ao usuário segurança na troca de e-mail Internet, de forma intrínseca e sem depender de um complexo serviço de distribuição de chaves, e de procedimentos que garantem a segurança local representa uma solução vantajosa para o usuário, garantindo a reserva em sua comunicação via correio eletrônico.

Nos capítulos anteriores, discutimos as principais características de um sistema de correio eletrônico, bem como os principais protocolos utilizados e a questão da segurança digital. Falamos de forma específica de e-mail Internet, descrevendo o SMTP e o ambiente dos servidores de e-mails Internet. Discutimos o tema de segurança em e-mail Internet e procuramos mostrar como essa segurança pode ser incluída no protocolo utilizado para troca de e-mails, especificando o SSMTP, e reforçada pela implementação de procedimentos locais de segurança, na máquina servidora de e-mail, especificando o LMSP.

Em nossas especificações, procuramos preocupar-nos com o nível de segurança desejado pelo usuário, deixando que essa escolha pudesse ser feita em função ao custo/benefício alcançado. A possibilidade de uso de servidores de chaves torna mais seguro, mas, sem dúvida, aumenta o grau de complexidade do sistema.

Assim, a nossa solução oferece a possibilidade de usar um esquema alternativo, onde a chave do destinatário é obtida da máquina servidora de e-mail. Embora existindo a possibilidade de interceptação das mensagens por intrusos em roteadores, temos uma solução satisfatória para a maioria dos usuários Internet (devido ao valor das informações que são transmitidas), sem a complexidade imposta pela certificação de chaves e sem acrescentarmos a necessidade de nenhum suporte crítico ao sistema.

A especificação do protocolo SSMTP e do LMSP, para garantia da integridade local, bem como o desenvolvimento do SMTA ssmtpd e do SUA spine, contribuem com um aspecto relevante que é a segurança na troca de e-mails em redes TCP/IP, em particular na Internet.

8.1. Avaliação do Objetivo Proposto

O objetivo do nosso trabalho, especificado na seção 1.1, é a especificação de um ambiente de correio eletrônico Internet com recursos de segurança, que tenha um bom custo/benefício para o usuário e que seja o mais transparente possível.

Acreditamos que a nossa especificação de protocolo SSMTP e das regras LMSP, se usadas na implementação de um SMTA e um SUA, garantem

ao usuário um ambiente de e-mail Internet, com a melhor segurança possível e ainda viabiliza a manutenção da transparência dos recursos para o usuário, oferecendo uma boa relação de custo/benefício.

8.2. Trabalhos Futuros

Acreditando que o nosso objetivo foi atingindo e que as especificações para implementação de SMTA e SUA poderão vir a ser bastante utilizados. Algumas sugestões, que podem contribuir para maior segurança do ambiente de e-mail Internet proposto, são:

- Incluir a autorização, verificando se o usuário pode executar o serviço de e-mail.
- Incluir recursos de auditoria, para se permitir a monitoração de conexões à porta 25 do TCP.
- Fazer um estudo sobre os servidores de chaves e certificações, para definir qual a melhor forma de realizar esse serviço e incorporá-lo às especificações SSMTP e LMSP.
- Finalizar a implementação do spine e do ssmtpd.
- Incluir interoperabilidade com o PEM e o PGP.
- Definir como esse serviço pode ser implementado em um *firewall*.

Referências Bibliográficas

[Anderson 95] ANDERSON, Bart et. al. *Unix Communications and the Internet,* SAMS Publishing , Indiana, 1995.

[Albuquerque 95] ALBUQUERQUE, Fernando. *Uso Seguro da Net. Connections*, Novembro de 1995.

[Alvestrand 96] ALVESTRAND, Harald.T. *X400*, May,1996. http://domen.uninett.no/~hta/x400/debate/security.html

[Avolio 94] AVOLIO, Frederick M. & VIXIE, Paul A. *Sendmail Theory and Practice*, Digital Press, California, 1994.

[Burns 95] BURNS, Nina. *O Correio Eletrônico Vai Além da Rede Local*, PC Magazine Brasil, Julho de 1995.

[Chewisk 94] CHESWIK, William R. & BELLOVIN, Steven M. *Firewalls and Internet Security*. Addison-Wesley Professional Computing Series, 1994.

[Comer 95] COMER, Douglas E. *Internetworking with TCP/IP*, Prentice Hall, New jersey, 1995.

[Costales 95] COSTALES, Bart A. H. & HANDERSON, Harry. *Unix Communication and The Internet*, Sanes Publishing, 1995.

[Crispen 95] Patrick Douglas Crispen, MAP07: *Netiquette*, The University of Alabama, 1995.

[Diffie 76] DIFFIE, W. & HELLMAN, M. E. *Multi-User Cryptographic Techniques*. National Computer Conference, 1976.

[Frosen 95] FROSEN, Janne. *Practical Cryptosystems and their Strength*, Tik-110.501 Seminar on Network Security, Helsinki University of Technology. Novembro de 1995

[Garfinkel 95] GARFINKEL, Simson. *PGP Pretty Good Privacy*, O'Reilly & Associates Inc., 1995.

[Heikkinen 96] HEIKKINEN, Jyrki. *X400* ICL Client-Server SystemsSystems Division / Groupware Development, 1996. http://www.nixu.fi/~pnr/netsec-lopulliset/9-0-secure-email.html#x400

[Hunt 92] HUNT, Craig. *TCP/IP Network Administration*, O'Reilly & Associates Inc., 1992.

[Malamud 92] MALAMUD, Carl. *Analyzing SUN Networks*, VNR Computer Library, New York, 1992.

[Medeiros Neto 94] MEDEIROS NETO, Benedito et. al. *Arquitetura de Redes de Computadores* - Correio Eletrônico, Makron Books, 1994.

[O'Reilly 92] O'REILLY, Tim & TODINO, Grace. *Managing UUCP and USENET*, O'Reilly & Associates Inc, 1992.

[RFC821] POSTEL, Jonathan B. *Simple Mail Transfer Protocol*, ISI, August, University of Southern California,1982.

[RFC822] CROKER, David H. *Standard for the Format of ARPA Internet Text Messages*, University of Delaware (UDEL), August,1982.

[RFC976] HORTON, Mark R. *UUCP Mail Interchange Format Standard*, Bell Laboratories, February ,1986.

[RFC1123] BRADEN, Robert. *Requirements for Internet Hosts - Application and Suport*, Internet Engineering Task Force, October, 1989.

[RFC 1321] RIVEST, R. *The MD5 Message Digest Algorithm*, MIT Laboratory for Computer Science and RSA Data Security, Inc., April 1992.

[RFC1421] LINN, J. *Privacy Enhancement for Internet Eletronic Mail: Part I: Message Encryption and Authentication Procedures*, IAB Privacy Task Force, February, 1993.

[RFC1422] KENT, J. *Privacy Enhancement for Internet Eletronic Mail: Part II: Certificate-Based Key Management*, IAB Privacy Task Force, February, 1993.

[RFC 1423] BALENSON, David. *Privacy Enhancement for Internet Electronic Mail: Part III: Algorithms, Modes, and Identifiers*, IAB Privacy Task Force, February, 1993

[RFC 1424] KALISKI, Burton S. *Privacy Enhancement for Internet Electronic Mail: Part IV: Certification and Related Services*, RSA Laboratories, February, 1993

[RFC1521] BORESTEIN, N. & FREED, N. *MIME - Multipurpose Internet Mail Extensions*, Bellcore & Innsoft, September, 1993.

[RFC 1636] BRANDEN, R. et. al. *Report of IAB Wokshop on Security in the Internet Archicteture*, February, ISI, June, 1994.

[RFC1685] ALVESTRAND, H. *Writing X.400 O/R Names*, UNINETT, August, 1994.

[RFC1725] MYERS, J. & MELLON, Carnegie. *Post Office Protocol - Version 3*, Carnegie Mellon & Dover Beach Consulting, Inc., November, 1994.

[RFC 1869] KLESIN et. al. *SMTP Service Extensions*, United Nations University, November 1995.

[Rivest 78] RIVEST, R. L., SHAMIR, A., ADELMAN, L. *A Method for Obtaining Digital Signatures and Public Key Cryptography*, Communications of the ACM, vol 21, no. 2, February 1978.

[RNP 95] *Guia de Operações Internet Brasil. Documento RNP/RPU/0015D*, Junho de 1995.

[Russel 92] RUSSEL, Deborah & SR. GANGEMIN, G. T. *Computer Security Basics*, O'Reilly & Associates Inc., 1992.

[Shang, 94] SHANG, David J. & MOON, Sylvia. *Segredos de Segurança em Rede*, Berkeley Brasil Editora, Rio de Janeiro, 1994.

[Schneier 93] SCHNEIER, Bruce. *The IDEA Encryption Algorithm*. Dr. Dobb's Journal. Dezembro de 1993.

[Soares, 95] SOARES, Luiz Fernando Gomes, LEMOS, Guido e COLCHER, Sérgio. *Redes de Computadores, Das LANs, MANs e WANs às Redes ATM*, Editora Campus, Rio de Janeiro, 1995.

[Stallings, 95] STALLINGS, William. *A Teia de Confiança do PGP*, Byte Brasil. Março de 1995.

[Tanenbaum, 94] TANENBAUM, Andrew S. *Redes de Computadores*, Editora Campus, Rio de Janeiro, 1994.

[Weber 95] WEBER, Raul Fernando. *Criptografia Contemporânea*, VI Simpósio de Computadores Tolerantes a Falhas, 1995.

[Woo 92] WOO, Thomas & LAM, Simson. *Authentication for Distributed Systems*, IEEE Computer, January, 1992.

Glossário de Siglas

ACL	Access Control List. Listas de autorização que estabelecem quais os recursos que processos, máquinas e usuários podem utilizar.
CRL	Certificate Revogation Lists. Listas, especificadas no PEM, com informações de certificados de chaves que foram revogados.
DEK	Data Encryption Key. Chaves únicas utilizadas para criptografia das mensagens, no PEM.
DNS	Domain Name System. É o resolvedor de nomes de máquinas numa rede TCP/IP.
EDI	Eletronic Data Interchange.
IDEA	International Data Encryption Algorithm. Algoritmo de cifragem de chave única, que utiliza chave de 128 bits e operações de adição em módulo de 2^{16}, ou-exclusivo e multiplicação em módulo de $2^{16} + 1$.
IK	Interchange Key. Chaves utilizadas para criptografia das chaves DEKs, utilizando criptografia assimétrica, especificadas no PEM.
IPRA	Internet Policy Registration Authority. Autoridade Central para certificação de chaves na hierarquia definida pelo PEM.
KDC	Kerberos Distribution Center. Autenticador de usuários no sistema de autenticação Kerberos.
LMSP	Local Mail Security Procedures. Regras para segurança local na máquina servidora de correio eletrônico.
MIC	Message Integrity Code. Código gerado para verificação da integridade e autenticação de mensagens, no PEM.
MD5]	Message Digest 5. Função que a partir de um texto de qualquer tamanho produz um número de 128 bits.
MHS	Message Handling System. O MHS é uma das recomendações X400, e especifica um sistema de tratamento de mensagens.

MIME	Multipurpose Internet Mail Extension. Mecanismo para especificar e descrever o formato de conteúdos de mensagens na Internet.
MTA	Message Transfer Agent (Agente de Transferência de Mensagens). Programa que encaminha as mensagens para o destinatário, localmente ou entre máquinas.
MX	Mail Exchanger. Registro do DNS que identifica a máquina servidora de correio eletrônico para um domínio.
PEM	Privacy Enhanced Mail. Conjunto de procedimentos especificados nos RFCs 1241-1424 para segurança em correio eletrônico Internet.
PGP	Pretty Good Privacy. Software de criptografia, que provê recursos de sigilo e autenticação para mensagens.
PINE	Program for Internet News and Email. Ambiente para manipulação de mensagens de correio eletrônico.
POP3	Post Office Protocol versão 3. Protocolo gerenciador de caixas postais.
RSA	Algoritmo de criptografia assimétrica mais largamente utilizado. R, S e A, são as primeiras letras dos sobrenomes dos autores deste algoritmo.
SMTA	Secure Message Transfer Agent (Agente de Transferência de Mensagens Seguro). Programa seguro para encaminhamento de mensagens.
SMTP	Simple Mail Transfer Protocol. Protocol. Protocolo, padrão TCP/IP, que especifica como o sistema de correio eletrônico transmite uma mensagem entre máquinas.
SPINE	Secure Program for Internet News and Email. Agente Usuário que implementa recursos de segurança para manipulação das mensagens.
SSMTP	Secure Simple Mail Transfer Protocol. Extensão ao SMTP, que acrescenta recursos de segurança para transmissão das mensagens.
ssmtpd	Daemon SSMTP. Agente de Transferência de Mensagens implementado de acordo com as especificações do SSMTP e as regras de segurança LMSP.
SUA	Secure User Agent (Agente Usuário Seguro). Agente Usuário que implementa recursos de segurança para as mensagens.

TGS	Ticket-Granting Server. Servidor de tickets para acesso a serviços em servidores, no sistema de autenticação Kerberos.
UA	User Agent (Agente Usuário). Parte do sistema de e-mail que provê a interface com o ser humano.
UUCP	Unix-to-Unix CoPy. Conjunto de programas para transferência de arquivos entre sistemas Unix.
X400	Recomendações do ITU-T, que definem o Message Handling System.