Leonardo Bispo de Oliveira

TrustMail: Um modelo de confiança entre servidores de e-mail

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática Aplicada da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Informática Aplicada.

Curitiba
2005

Leonardo Bispo de Oliveira

TrustMail: Um modelo de confiança entre servidores de e-mail

Dissertação de Mestrado apresentado ao Programa de Pós-Graduação em Informática Aplicada da Pontifícia Universidade Católica do Paraná como requisito parcial para obtenção do título de Mestre em Informática Aplicada.

Área de Concentração: Metodologia e Técnicas de Computação

Orientador: Prof. Dr. Carlos Alberto Maziero

Curitiba
2005

Oliveira, Leonardo Bispo de Oliveira
TrustMail: Um modelo
de confiança entre servidores de e-mail. Curitiba, 2005.

Dissertação de Mestrado - Pontifícia Universidade Católica do Paraná Programa de Pós-Graduação em Informática Aplicada.

1. Redes de relacionamento 2. Servidores de e-mail 3. Confiança em servidores de e-mail I.Pontifícia Universidade Católica do Paraná. Centro de Ciências Exatas e Tecnologia. Programa de Pós-Graduação em Informática Aplicada II - t

Dedico este trabalho à minha família e amigos.

Agradecimentos

Agradeço aos meus pais, por me apoiarem durante todo o tempo. Aos meus irmãos que me ajudaram em todos os aspectos.

A Daniele, que sempre teve paciência e compreensão nos momentos em que fiquei ausente e pela sua ajuda, que foi fundamental na avaliação dos testes.

Agradeço ao Carlos Maziero, pelas oportunidades que me proporcionou durante todo o desenvolvimento do trabalho.

Agradeço também aos integrantes da Banca Examinadora por suas contribuições que foram muito importantes para a finalização deste trabalho.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviações

CGS	*Confiança do Grupo no Servidor*
CLS	*Confiança Local do Servidor*
CMLS	*Contador de mensagens legítimas no servidor*
CMMS	*Contador de mensagens maliciosas no servidor*
CMS	*Contador de Mensagens*
CS	*Confiança do Servidor*
CS	*Confiança do Servidor*
CS	*Confiança Final*
DHT	*Distributed Hash Table*
EIG	*Eigenvalue Propagation*
IANA	*Internet Assigned Numbers Authority*
IMAP4	*Internet Message Access Protocol Versão 4*
IPC	*Interprocess Communication*
MAA	*Mail Access Agent*
MC	*Mensagens de Confiança*
MDA	*Mail Delivery Agent*
MIME	*Multi-Purpose Internet Mail Extensions*
MTA	*Mail Transfer Agent*

MUA	*Mail User Agent*
P2P	*Peer-To-Peer*
PGP	*Pretty Good Privacy*
PKI	*Public Key Infrastructure*
POP3	*Post Office Protocol Versão 3*
PRA	*Purpoted Responsible Address*
QM	*Quandidade de Mensagens*
QMS	*Quantidade de Mensagens Servidor*
S/MIME	*Secure/Multipurpose Internet Mail Extentions*
SIDF	*SenderID Framework*
SMTP	*Simple Mail Transfer Protocol*
SPF	*Sender Policy Framework*
TTL	*Time To Live*
UML	*User-Mode Linux*
WLC	*Weighted Linear Combination*

Resumo

O e-mail é uma ferramenta essencial na Internet. No entanto, a arquitetura dos sistemas de e-mail convencional apresenta problemas que deixam o sistema vulnerável a diversos tipos de ameaças. Alternativas vêm sendo propostas para sanar essas deficiências e prover maior escalabilidade e confiabilidade aos sistemas de e-mail. Esta dissertação apresenta um modelo de confiança que cria listas de servidores confiáveis e não confiáveis de forma dinâmica e descentralizada, através da auto-exclusão de servidores utilizados como propagadores de mensagens maliciosas. Para tal, diversas técnicas foram empregadas, como um modelo de redes de relacionamento, filtros de mensagens e um modelo de gerenciamento, armazenamento e propagação de confiança. A implementação da arquitetura proposta e os testes demonstram o modelo da implementação.

Palavras-chave: Servidores de e-mail, Redes de relacionamento, Confiança em servidores de e-mail.

Abstract

E-mail services are an essential tool in the internet, however, the standard e-mail system architecture present problems that lat this kind of system vulnerable to several types of threats. Alternatives have been proposed to solve some problems related with e-mail services, offering reliability and escalability to those systems. This work presents a reliable model to create a dynamic and descentralized trustworty server lists, through the self-exclusion of servers used as spreaders of slyness messages. Many techniques had been done as a social network model, message filters and management, storage and reliable propagation model. The proposed of the architecture implementation and the tests demosntrate the implementation model.

Keywords: E-mail servers, Social networks, E-mail servers reliable.

Capítulo 1

Introdução

Este trabalho apresenta um modelo de confiança entre servidores de e-mail que faz a gerência de confiança, armazenamento das informações sobre servidores confiáveis e não confiáveis e propagagação das informações de confiança geradas. A propagação da confiança é transmitida pela rede para um determinado grupo de relacionamento através de métodos de **redes de relacionamento**.

1.1 Motivação

Os sistemas de e-mail estão sendo comumente utilizados pelas empresas e usuários de computadores como ferramenta padrão de comunicação, por ser simples de utilizar e pelo seu baixo custo, tanto de uso quanto de implantação. O aumento no uso desses sistemas acarretou no surgimento de problemas causados por fragilidades não previstas na elaboração do protocolo original. As principais fragilidades no protocolo de transporte de e-mail residem na falta de mecanismos robustos de autenticação, mecanismos capazes de prover privacidade, confiabilidade e integridade das mensagens e mecanismos de reputação de usuários e servidores de e-mail. Estudos estão sendo propostos para criar ferramentas que implementam tais mecanismos. Os principais modelos são:

- Filtros de classificação de conteúdo de mensagens que procuram por padrões em mensagens que são recebidas pelo servidor de e-mail para classificá-las como legítimas ou maliciosas;
- Uso de chaves assimétricas nos clientes de e-mail utilizadas para assinar e cifrar as mensagens a serem trafegadas pela rede;
- Autenticação de servidores de e-mail que utilizam os servidores DNS para disponibilizar informações sobre servidores habilitados a enviar mensagens para um deter-

minado domínio;

- Ferramentas de classificação de mensagens maliciosas a partir de redes de relacionamento que classificam cada e-mail/usuário como maliciosos, legítimos ou não classificáveis.

Por mais que essas ferramentas sejam um grande avanço nos sistemas de e-mail, as mesmas continuam com problemas, pois os métodos de classificação são lentos e passíveis de erro; as chaves assimétricas nos clientes necessitam de métodos de troca de chaves de forma confiável e segura ou autoridades capazes de publicar as chaves confiáveis, tornando este sistema pouco escalável; os métodos de autenticação de servidores de e-mail não impedem que usuários maliciosos continuem enviando informações maliciosas pela rede.

1.2 Proposta

A falta de mecanismos de autenticação em serviços de e-mail e a simplicidade do protocolo de comunicação de envio de e-mail faz com que diversos problemas relacionados a autenticidade de mensagens e propagandas indesejáveis sejam explorados por usuários maliciosos, tornando o modelo padrão de tráfego de mensagens pela internet pouco confiável para seus usuários. Além disso, esses sistemas trafegam milhares de mensagens não solicitadas diariamente, ocupando espaço em disco, largura de banda da rede, processamento de máquina de provedores de e-mail e usuários da internet. Esses problemas se tornam cada vez mais crítico a medida que as pessoas começam a utilizar os sistemas de e-mail como mecanismos confiáveis de comunicação.

As redes de relacionamento vêm sendo empregadas em redes de computadores para prover maior segurança nesses sistemas. As redes *peer-to peer* utilizam algoritmos de reputação baseados em redes de relacionamento para indicar a confiança de cada indivíduo na rede; modelos de classificação de usuários maliciosos em serviços de e-mail utilizando redes de relacionamento também foram propostos, utilizando um grupo de elementos para gerar uma opinião final sobre a mensagem. As redes de relacionamento poderão ser utilizadas para gerar uma reputação de servidores de e-mail que se comunicam na rede.

A proposta deste trabalho é definir um modelo de confiança capaz de gerenciar listas descentralizadas de servidores considerados confiáveis e não confiáveis criadas dinamicamente. Isto é obtido através de técnicas de classificação de e-mail e autenticação de servidores para criar de confiança para servidores que se comunicam com freqüência em conjunto com um modelo de redes de relacionamento que fará a propagação da confiança pela rede.

Através de filtros de mensagem é possível se classificar uma mensagem como legítima ou maliciosa. Com as mensagens já classificadas, o sistema de confiança utilizará cálculos de gerência de confiança em conjunto com a confiança de um conjunto de servidores para definir o quão confiável um servidor pode ser considerado. A confiança gerada através dos cálculos de gerência de confiança implicará diretamente na quantidade de mensagens que um servidor receptor aceitará receber de um emissor em um determinado período de tempo.

1.3 Organização do trabalho

Este trabalho é composto por 8 capítulos incluindo este e está organizado da seguinte forma: o capítulo 2 apresenta um estudo sobre a arquitetura SMTP, mostrando os principais conceitos do protocolo; o capítulo 3 apresenta os principais problemas encontrados no serviços de e-mail devido às fragilidades do protocolo SMTP; o capítulo 4 apresenta as técnicas de autenticação de usuários e servidores de e-mail que utilizam chaves e informações armazenadas no servidor DNS para prover um modelo de autenticação nos sistemas de e-mail; o capítulo 5 conceitua as redes de confiança que serão utilizadas na proposta; o capítulo 6 contém a proposta da dissertação; o capítulo 7 o protótipo implementado, os ensaios realizados e a análise crítica dos resultados obtidos; por fim, é apresentada uma conclusão sobre a dissertação, apontando os benefícios alcançados e os trabalhos futuros relacionados à proposta e sua implementação.

Capítulo 2

Infra-estrutura de e-mail na internet

O sistema de e-mail é um dos meios de comunicação mais utilizados nos dias atuais. Isso se dá pelo seu baixo custo, facilidade de comunicação e por ser simples de implementar.

Este ambiente é baseado na arquitetura cliente-servidor, onde o cliente de e-mail proporciona uma interface amigável para enviar, receber e organizar as mensagens de um usuário; o servidor recebe as mensagens enviadas por clientes de e-mail e encaminha ao seu destino final, que pode ser um usuário local ou externo. Esta troca de mensagens é feita utilizando os protocolos baseado em comando SMTP/ESMTP [Pos82, Kle01a] para o envio das mensagens e IMAP [Cri96] e POP [MR96] para a entrega das mensagens ao usuário final.

A troca de mensagens pela rede é feita por agentes de transporte de e-mail, que tratam a mensagem como um arquivo ASCII composto por um cabeçalho e um corpo. Além disso, arquivos podem ser codificados e incorporados às mensagens em formato ASCII sob forma de anexo. Todo o esforço de codificação e decodificação desses arquivos anexos é feito pelos clientes de e-mail durante o envio e a recepção da mensagem.

Este capítulo apresenta a infra-estrutura de e-mail na internet e está estruturado da seguinte forma: a seção 2.1 traz uma visão geral sobre os agentes de transportes de e-mail; a seção 2.2 apresenta o protocolo de comunicação SMTP e seus principais comandos; a seção 2.3 mostra a forma com que uma mensagem é estruturada; a seção 2.4 analisa o suporte de e-mails a arquivos anexados; por último, a seção 2.5 conclui o capítulo.

2.1 Infra-estrutura de transporte de e-mail

A infra-estrutura de transporte de e-mail é constituída por quatro agentes que funcionam como emissores e/ou receptores de mensagens. Através deles um e-mail pode

ser enviado, recebido ou então repassado através de uma rede de computadores. Esses agentes podem ser classificados como:

- MTA (*Mail Transfer Agent*) ou "servidor de e-mail": o MTA é o programa responsável por receber e enviar mensagens de usuários através de redes de computadores. Sua função é criar rotas de mensagens, enviar mensagens de MUAs e repassar mensagens entre MTAs. O *sendmail* e o *Microsoft Exchange* são exemplos de MTAs.

- MUA (*Mail User Agent*) ou "cliente de e-mail": o MUA é o programa que permite um usuário acessar e gerenciar mensagens, incluindo ler, compor, imprimir, reenviar e mostrar mensagens. O MUA disponibiliza uma interface entre o usuário e o MTA. O *Eudora* e o *Outlook* são exemplos de MUAs.

- MDA (*Mail Delivery Agent*): o MDA é o agente de entrega de e-mail. MDAs são softwares responsáveis por receber uma mensagem de um MTA e fazer com que esta mensagem seja armazenada ou reencaminhada. O *procmail* é muito utilizado como um MDA em sistemas **UNIX**.

- MAA (*Mail Access Agent*): o MAA é o programa que permite que os MUAs tenham acesso aos e-mails que estão na caixa de correio do usuário. Na prática, esta função é exercida pelos servidores POP3 (*Post Office Protocol Versão 3*) e IMAP4 (*Internet Message Access Protocol Versão 4*).

Mais especificamente, os agentes de transporte são os responsáveis por todo o envio de e-mail na internet. A interação entre eles possibilita que uma mensagem seja enviada por um emissor e recuperada por um receptor de maneira simples e eficiente em qualquer ponto da rede. Essa interação pode ser vista na Figura 2.1.

Nesta Figura o **computador A** utiliza um MUA para enviar uma mensagem ao MTA do **computador B** (servidor de e-mail do cliente MUA no **computador A**) que recebe a mensagem integralmente e deposita em um diretório temporário (este procedimento é conhecido como "*store and forward*" e garante que a mensagem seja entregue ao destinatário sem risco de perdas na transmissão). Depois da mensagem ser depositada no

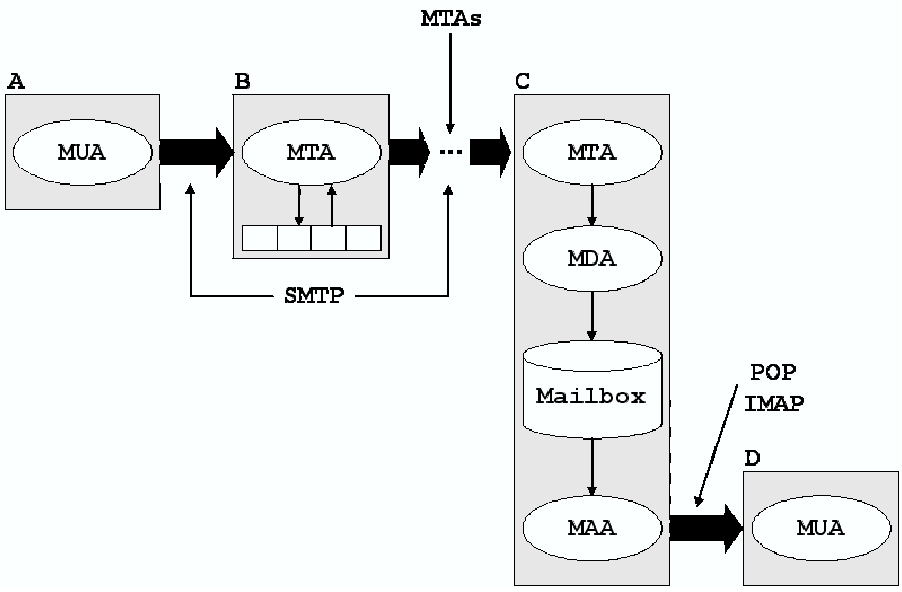

Figura 2.1: Infra-estrutura de Transporte

diretório temporário, o MTA do **computador B** comunica-se com um outro MTA. Esse processo continua até que a mensagem chegue ao MTA do destino final (no caso o **computador C**). O endereço do **computador C** deverá ser obtido através de uma consulta ao servidor de DNS que requisita o endereço *Mail eXchanger* do domínio em questão. Quando a mensagem for integralmente recebida pelo **computador C**, o MTA faz uma chamada ao MDA que deposita localmente a mensagem recebida na caixa de correio de destino. Por fim, o usuário final (**computador D**) inicia uma comunicação com o MAA e recebe as mensagens armazenadas no servidor. A comunicação de um MUA com um MTA e de um MTA com outro MTA é feita utilizando o protocolo SMTP. A comunicação do MUA com o MAA é feita utilizando o protocolo POP ou IMAP.

2.2 O protocolo SMTP

O SMTP (*Simple Mail Transfer Protocol*) é o protocolo de transferência de e-mail definido pela RFC 821 [Pos82] e revisado pela RFC 2821 [Kle01a] que atua sobre o protocolo **TCP/IP** para transportar mensagens. Este protocolo foi inicialmente projetado pela Universidade do Sul da Califórnia em 1982 com o objetivo de ser independente de um sistema de transmissão particular. Para a transmissão de uma mensagem é necessário apenas um canal de comunicação entre um determinado **emissor** (emissor SMTP) e **receptor** (receptor SMTP). O modelo genérico de uma aplicação que utiliza o protocolo SMTP pode ser visto na Figura 2.2.

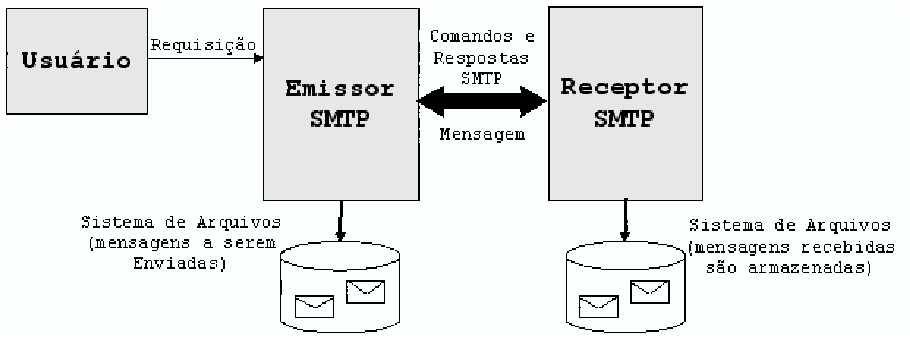

Figura 2.2: Modelo SMTP[Pos82]

Neste modelo o usuário inicia uma transação de envio de mensagem a partir do **emissor** com o **receptor**. O envio de uma mensagem consiste na troca de comandos e respostas entre aplicações SMTP. Essa troca de comandos e respostas é feita seqüencialmente. Quando um usuário faz uma requisição ao **emissor**, um canal de comunicação bidirecional é estabelecido com o **receptor** (pode ser o último destino ou um destino intermediário). Através deste canal, comandos SMTP são gerados pelo **emissor** e enviados para o **receptor**. De acordo com os comandos enviados o **receptor** deverá enviar respostas ao **emissor**.

Quando um canal de comunicação é estabelecido, o **emissor** envia o comando *MAIL* indicando o autor da mensagem. Se o comando for aceito pelo **receptor**, será gerada uma resposta de *OK* para o **emissor**, do contrário, será gerada uma resposta rejeitando o conteúdo (esta resposta não implica no encerramento do canal de transmissão). Caso o comando *MAIL* tenha sido aceito, o **emissor** envia o comando *RCPT* mais o destinatário da mensagem (este comando poderá ser utilizado várias vezes). Se o comando *RCPT* for aceito, será gerada uma resposta *OK*, senão uma resposta de erro será enviada ao **emissor**.

Depois de terem sido enviados todos os destinatários da mensagem, o **emissor** deverá enviar um comando *DATA* que indica o inicio da transmissão do conteúdo do e-mail. O **receptor** tratará todas as informações enviadas após o comando *DATA* como sendo conteúdo da mensagem até que seja enviada uma linha que contenha apenas um ponto final (o ponto indica o final do conteúdo). Este ponto final não deverá implicar no encerramento do canal de transmissão estabelecido. Utilizando o canal, **emissor** e **receptor** poderão negociar o envio de diversas mensagens. Quando não existirem mais mensagens na fila de e-mail do emissor, o comando *QUIT* é enviado ao **receptor** e a conexão é fechada.

É bom lembrar que toda mensagem a ser enviada deve estar armazenada no sistema

de arquivos do **emissor** e toda mensagem recebida deverá ser armazenada em um sistema de arquivos do **receptor**.

2.2.1 Comandos e respostas SMTP

Os comandos SMTP são constituídos por uma linha de texto iniciada por uma "palavra especial" de quatro letras, que poderá ser seguida de argumentos. Na maioria dos casos as respostas a comandos são constituídas de uma única linha, contudo, múltiplas linhas de resposta também são possíveis.

A Tabela 2.1 apresenta os comandos utilizados pelo SMTP. Todos os comandos até o *QUIT* devem ser implementados no protocolo. Os demais são opcionais e podem ser ignorados pelo receptor.

Tabela 2.1: Comandos SMTP

Nome	Comando	Descrição
HELO	HELO <SP> <domain> <CRLF>	Identificador do emissor
MAIL	MAIL <SP> FROM:<reverse-path> <CRLF>	Identifica o criador do e-mail
RCPT	RCPT <SP> TO: <forward-path> <CRLF>	Identifica o destinatário do e-mail
DATA	DATA <CRLF>	Transferência de mensagem texto
RSET	RSET <CRLF>	Aborta a transação de e-mail atual
NOOP	NOOP <CRLF>	Sem operação
QUIT	QUIT <CRLF>	Fecha a conexão TCP
SEND	SEND <SP> FROM:<reverse-path> <CRLF>	Envia e-mail para terminal
SOML	SOML <SP> FROM:<reverse-path> <CRLF>	Envia e-mail para terminal, quando possível
SAML	SAML <SP> FROM:<reverse-path> <CRLF>	Envia e-mail para terminal e mailbox
VRFY	VRFY <SP> <string> <CRLF>	Confirma nome do usuário
EXPN	EXPN <SP> <string> <CRLF>	Retorna o membro da lista de e-mail
HELP	HELP [<SP> <string>] <CRLF>	Envia documentação específica do sistema
TURN	TURN <CRLF>	Reverte regra do destinatário e do receptor

As respostas SMTP são formadas de três dígitos seguidos por uma informação adicional, onde o primeiro digito indica a categoria da resposta; os demais dígitos indicam o erro em si. Essas respostas são classificadas como:

- **Resposta de conclusão positiva:** ações de requisição foram completadas com sucesso. Uma nova requisição deve ser iniciada.
  - **211:** status do sistema ou resposta de ajuda do sistema
  - **214:** mensagem de ajuda

  - **220:** <domain> serviço pronto
  - **221:** <domain> fechando serviço de canal de transmissão
  - **250:** requisição de ação de e-mail ***OK***, completada
  - **251:** usuário não faz parte do domínio; a mensagem será reencaminhada para <forward-path>

- **Resposta intermediária positiva:** o comando foi aceito, mas estão faltando informações para a ação requisitada. O **emissor** deve enviar um outro comando especificando esta informação. Esta resposta é utilizada quando é necessária uma seqüência de comandos.

  - **301:** inicio do envio do e-mail; finalizar com <CRLF>.<CRLF>

- **Resposta de conclusão temporariamente negativa:** o comando não foi aceito e a ação requisitada não ocorreu. Contudo, a condição de erro é temporária e pode ser requisitada novamente.

  - **421:** <domain> serviço indisponível: perdendo canal de transmissão
  - **450:** requisição negada: caixa de correio indisponível
  - **451:** requisição abortada: erro local no processo
  - **452:** requisição negada: espaço insuficiente

- **Resposta de conclusão permanentemente negativa:** o comando não foi aceito e a requisição não poderá ser feita novamente.

  - **500:** erro de sintaxe, comando não reconhecido
  - **501:** erro de sintaxe nos parâmetros ou argumentos
  - **502:** comando não implementado
  - **503:** seqüência errada de comandos
  - **504:** parâmetro não implementado
  - **550:** requisição negada: caixa de correio indisponível
  - **551:** usuário não faz parte do domínio; tente <forward-path>
  - **552:** requisição de e-mail abortada: excedido espaço de alocação
  - **553:** requisição negada: caixa de correio indisponível
  - **554:** erro na transação

Em posse dessas informações, comandos e respostas são geradas e trocadas entre servidores e clientes de e-mail até que a mensagem seja entregue ao seu destino final. O código a seguir apresenta um exemplo de comunicação entre servidores utilizando o protocolo SMTP (as linha iniciadas por um **C** correspondem ao comandos emitidos pelo cliente; as linhas iniciadas por um **S** correspondem às respostas do servidor SMTP):

```
1  S: 220 ppgia.pucpr.br Simple Mail Transfer Service Ready
2  C: HELO acme.com.br
3  S: 250 ppgia.pucpr.br greets acme.com.br
4  C: MAIL FROM:<joao@acme.com.br>
5  S: 250 OK
6  C: RCPT TO:<maria@ppgia.pucpr.br>
7  S: 250 OK
8  C: RCPT TO:<jose@ppgia.pucpr.br>
9  S: 550 No such user here
10 C: RCPT TO:<antonio@ppgia.pucpr.br>
11 S: 250 OK
12 C: DATA
13 S: 354 Start mail input; end with <CRLF>.<CRLF>
14 C: Bom Dia,
15 C:  []Joao
16 C: .
17 S: 250 OK
18 C: QUIT
19 S: 221 ppgia.pucpr.br Service closing transmission channel
```

Neste exemplo, `joao` (usuário do domínio `acme.com.br`) envia um e-mail para maria (usuário do domínio `ppgia.pucpr.br`), `antonio` (usuário do domínio `ppgia.pucpr.br`) e `jose`. O e-mail enviado para `antonio` e `maria` são aceitos por `ppgia.pucpr.br`; `jose` não é um usuário válido, então, uma mensagem de erro é gerada por `ppgia.pucpr.br` para `acme.com.br`.

2.2.2 Modelo de SMTP Estendido

Em um esforço iniciado em 1990, aproximadamente uma década após a RFC 821, o protocolo SMTP foi modificado para suportar o modelo de serviço estendido. Este modelo permite que clientes e servidores compartilhem funcionalidades diferentes das que foram definidas no protocolo SMTP original, dando uma maior flexibilidade ao sistema de

transporte, devido ao fato de ser possível inserir novos comandos e respostas ao protocolo sem que seja necessário alterar servidores e clientes já implementados.

Os mecanismos estendidos do protocolo ESMTP definem como um cliente e um servidor devem reconhecer um ao outro. Além disso, o servidor pode informar ao cliente os serviços estendidos que são suportados [Kle01a]. O *framework* de serviços estendidos consiste em:

- O comando SMTP *EHLO* substituindo o *HELO*;
- Um registro de serviços estendidos SMTP;
- Parâmetros adicionais para os comandos *MAIL* e *RCPT*;
- Substituição opcional para os comandos definidos na RFC 821, tal como o comando *DATA* ser capaz de transportar informações que não estejam em formato texto (não ASCII).

Segundo [Kle01a], as implementações SMTP precisam ter suporte básico aos mecanismos estendidos. Servidores precisam suportar o comando *EHLO* mesmo que não implementem qualquer tipo de serviço estendido. Os clientes devem preferencialmente utilizar o *EHLO* ao invés do *HELO*, contudo, para haver uma compatibilidade com implementações antigas, clientes e servidores SMTP precisam suportar o mecanismo de *HELO*. Além disso, deverá existir um registro de cada serviço estendido associado a um comando na IANA (*Internet Assigned Numbers Authority*). O IANA é o órgão responsável em manter todos os registros de serviço de estendido SMTP. Este registro deverá possuir:

- Nome do serviço SMTP estendido;
- Chave associada com a extensão;
- Sintaxe e possíveis valores dos parâmetros associados à chave;
- Qualquer comando SMTP adicional associado com a extensão;
- Qualquer novo parâmetro de extensões associadas aos comandos *MAIL* e *RCPT*;
- Um descritor de como suportar diferenças no comportamento de clientes e servidores SMTP causados pela extensão.

Se um campo estendido não estiver registrado no IANA, a chave do comando deverá ser iniciada por um “X”. O “X” indica que um campo é de uso estritamente local.

2.3 Estrutura de um e-mail

O padrão SMTP utiliza a RFC 822 [Cro82] como formato de construção da mensagem que será transmitida por ele. A RFC 822 trata o e-mail como um arquivo em formato ASCII composto por cabeçalho (*header*) e corpo (*body*) separados por uma linha em branco (vazia). Todas as informações de um e-mail encontram-se no cabeçalho e são denominadas "*header tags*". Os "*header tags*" são representados por linhas na forma **Nome: valor**. Os "header tags" mais comuns são `From`, `To`, `Subject` e `Date`, além do `Message-ID` que contém um identificador único associado à mensagem. Essa estrutura pode ser vista no exemplo a seguir:

```
1  From joao@acme.com.br  Wed Feb 18 07:35:47 2004
2  Return-Path: <joao@acme.com.br>
3  Received: from joao@acme.com.br (acme.com.br [200.200.200.200])
4          by ppgia.pucpr.br (8.12.5/8.12.5) with ESMTP id i1IAZjXR005568
5          for <maria@ppgia.pucpr.br>; Wed, 18 Feb 2004 07:35:45 -0300
6  Message-ID: <1077104322.40334ec2d39a3>
7  Date: Wed, 18 Feb 2004 08:38:42 -0300
8  From: joao@acme.com.br
9  To: maria@ppgia.pucpr.br
10 Subject: Email Plano
11 MIME-Version: 1.0
12 Content-Type: text/plain; charset=iso-8859-1
13 Content-Transfer-Encoding: 8bit
14 User-Agent: Internet Messaging Program (IMP) 3.2.2
15 X-Originating-IP: 200.192.113.56
16 X-ACME-LOCAL: local_scan v1.0
17 X-ACME-SPAM: 2.4
18 X-AntiVirus: scanned for viruses by AMaViS 0.2.1 (http://amavis.org/)
19
20 Este e um exemplo de Email
21
22 []Joao
```

Este exemplo apresenta uma mensagem enviada por `joao@acme.com.br` para `maria@ppgia.pucpr.br`. Conforme visto na seção 2.2.2 todos os "*header tags*" iniciados por um "X" são locais ao servidor que recebeu a mensagem. A vantagem dos e-mails estarem no formato texto é a facilidade de enviar, receber e armazenar as mensagens.

2.4 MIME - Multi-purpose internet mail extensions

Na concepção da arquitetura do protocolo SMTP não foram previstos outros tipos de codificações de mensagens e uso de arquivos anexados (imagens, sons, programas, etc.) à mensagem. Com o aumento do uso do correio eletrônico, pesquisadores da área sentiram a necessidade de incorporar tais características ao sistema. Vários estudos foram realizados, até que em 1993, foi proposto o padrão MIME (*Multi-Purpose Internet Mail Extensions*) definido em [FB96a, FB96b, Moo96, FKP96, FB96c]. O padrão MIME codifica os arquivos de anexo para o formato texto. Em seguida este texto é inserido na mensagem. Com essa abordagem não foram necessárias alterações no protocolo SMTP original. A especificação MIME inclui os seguintes elementos:

1. Cinco novos "header tags" foram definidos, podendo ser incluídos no cabeçalho de mensagens conforme padrão RFC 822. Estes campos possuem informações sobre o corpo da mensagem. São eles:

   - **MIME-Version:** Declara a versão MIME e informa que a mensagem está neste formato, permitindo que os processadores de e-mail distinguam uma mensagem deste tipo de outras mais antigas;
   - **Content-Type:** Utilizado para especificar o tipo de mídia e sub-tipo de dados utilizado no corpo da mensagem;
   - **Content-Transfer-Encoding:** Especifica as transformações de codificação que foram aplicadas ao corpo da mensagem (normalmente para portá-la para 7 bits ASCII);
   - **Content-ID:** Usado para identificar o conteúdo do corpo da mensagem;
   - **Content-Description:** Usado para descrever o conteúdo do corpo da mensagem. Deve ser atribuído quando o objeto não pode ser lido (ex: áudio).

2. São definidas transferências codificadas que habilitam a conversão de qualquer formato do conteúdo em uma forma que é protegida de alterações pelo sistema de e-mail. Podem ser:

   - **7bit:** Contém texto codificado em 7 bits (conjunto de caracteres US-ASCII);
   - **8bit:** Contém texto com alguns caracteres que necessitam de 8 bits para serem codificados (conjunto de caracteres não US-ASCII). Caso uma mensagem deste tipo passe por uma "zona" da rede que permite transportar somente caracteres

de 7 bits, todos os caracteres que necessitam de 8 bits chegarão ao destino com erro;

- **binary:** Contém caracteres não US-ASCII e linhas mais longas do que o permitido na RFC 821. Caso uma mensagem deste tipo passe por uma "zona" da rede que permite transportar somente caracteres de 7 bits, todos os caracteres que necessitam de 8 bits chegarão ao destino com erro;
- **quoted-printable:** Transforma caracteres não US-ASCII em combinações de caracteres US-ASCII. Programas que suportam codificação "`quoted-printable`" traduzem a combinação de caracteres para o caractere original. Exemplo: "`São Paulo`" se torna "`S=E3o Paulo`" quando transmitido no modo "`quoted-printable`";
- **base64:** Contém binários codificados. Todos os caracteres são codificados como grupos de caracteres de 7 bits, de tal modo que o binário não será alterado ao trafegar pela rede.
- **x-token:** O x-token possibilita a inclusão de um mecanismo de codificação diferente dos apresentados anteriormente. Para utilizar o x-token, o cliente deve suportar o tipo de codificação especificada. O x-token pode ser definido por qualquer string de caracteres e deve ser precedido por um "x-" ou "X-".

Através do MIME, um usuário pode incluir arquivos em um e-mail de diferentes formatos e/ou diferentes codificações em um e-mail. Grande parte do sucesso do padrão MIME deve-se ao fato do mesmo ser opaco aos servidores de e-mail (o corpo de cada e-mail é visto em formato ASCII, conforme a definição original da RFC 822). Todo o esforço de codificação e decodificação dos e-mails é realizado pelos softwares clientes, durante o envio e a recepção do e-mail. O exemplo a seguir mostra uma mensagem com dois arquivos anexados ao seu corpo (`anexo.txt` e `anexo.tar.gz`):

```
1  From joao@acme.com.br  Wed Feb 18 07:48:03 2004
2  Return-Path: <joao@acme.com.br>
3  Received: from acme.com.br (acme.com.br [200.200.200.200])
4          by ppgia.pucpr.br (8.12.5/8.12.5) with ESMTP id i1IAlwXR005737
5          for <maria@ppgia.pucpr.br>; Wed, 18 Feb 2004 07:47:58 -0300
6  From: Joao <joao@acme.com.br>
7  To: maria@ppgia.pucpr.br
8  Subject: Email com Anexo
9  Date: Wed, 18 Feb 2004 08:53:04 -0300
10 User-Agent: KMail/1.6
```

```
MIME-Version: 1.0
Content-Disposition: inline
Content-Type: Multipart/Mixed;
  boundary="Boundary-00=_gI1MAoB3Q75V6+b"
Message-Id: <200402180853.04435.bispo@ppgia.pucpr.br>
X-PPGIA-LOCAL: local_scan v1.0
X-PPGIA-SPAM: 5.6
X-AntiVirus: scanned for viruses by AMaViS 0.2.1 (http://amavis.org/)


--Boundary-00=_gI1MAoB3Q75V6+b
Content-Type: text/plain;
  charset="us-ascii"
Content-Transfer-Encoding: 7bit
Content-Disposition: inline

Este e um exemplo de Email com Anexo

[]Joao

--Boundary-00=_gI1MAoB3Q75V6+b
Content-Type: text/plain;
  charset="us-ascii";
  name="anexo.txt"
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment;
        filename="anexo.txt"

Exemplo de arquivo anexado
--Boundary-00=_gI1MAoB3Q75V6+b
Content-Type: application/x-tgz;
  name="anexo.tar.gz"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
        filename="anexo.tar.gz"

H4sIABRSM0AAA+3SQQrCMBCF4RxlTiCTmuYGHiTQLITqaEwlxzcguOiirooI/7
   eZxbzFY5h0zc0O
tVW3H/WqMQSnb+upehxcj/gQR42+57wPgzrRHTt9LI+aiogrZps3+Lb/
   U6eWL7fZZMqSyn05P01S
f4k02a+bAQAAAAAAAAAAAAAAAAC2vACqDu77ACgAAA==

--Boundary-00=_gI1MAoB3Q75V6+b--
```

Analisando este exemplo é possível verificar a inclusão de novos "*header tags*" no cabeçalho da mensagem e informações adicionais no corpo da mensagem que informam o tipo de codificação, o nome do arquivo anexado e outras informações pertencentes ao padrão MIME.

2.5 Conclusão

Neste capítulo foi apresentada a arquitetura geral dos sistemas de e-mail na Internet e os agentes de e-mail MTA, MUA, MDA e MAA responsáveis pelo transporte das mensagens; o protocolo de comunicação SMTP e ESMTP utilizados pelos MUAs e MTAs para envio/recebimento de mensagens; a estrutura de um e-mail (cabeçalho, corpo, etc) e o padrão MIME que especifica uma forma de enviar informações não textuais no corpo de uma mensagem.

Conhecer a infra-estrutura de transporte de e-mail será importante para a compreensão dos próximos capítulos. O próximo capítulo apresenta as principais ameaças ao serviço de e-mail nos dias atuais e algumas técnicas para minimizar essas ameaças.

Capítulo 3

Ameaças ao serviço de e-mail

O sistema de e-mail foi inicialmente projetado para ser simples, pois seu público alvo era restrito à comunidade acadêmica. O protocolo SMTP não provê mecanismos robustos para autenticação e controle de acesso e sua extensão para prover esses serviços não é trivial. Com a explosão do uso da Internet, o sistema de e-mail ganhou grande importância expondo, com isto, as fragilidades das tecnologias envolvidas e vários problemas não previstos inicialmente. Dentre esses problemas destacam-se os *spams*, *trojans*, vírus e *scams* que comprometem o desempenho, robustez, segurança e usabilidade dos sistemas de e-mail atuais.

É possível encontrar na literatura propostas que visam minimizar esses problemas e autenticar usuários de serviços de e-mail, contudo, uma solução efetiva para estes problemas está longe de ser proposta, devido às dificuldades de classificação de usuários maliciosos.

Este capítulo apresenta as ameaças vinculadas ao serviço de e-mail e está estruturado da seguinte forma: a seção 3.1 apresenta o *spam* e suas principais formas de propagação; a seção 3.2 apresenta os *scams*, um modelo de ataque utilizado por crackers para roubar senhas e outras informações de usuários utilizando técnicas de engenharia social; a seção 3.3 apresenta as principais técnicas de contenção de *spam* encontradas na literatura; a seção 3.4 aborda os cavalos de Tróia e sua forma de propagação em correios eletrônico; a seção 3.5 traz uma visão geral sobre os vírus e *worms* de e-mail; por último, a seção 3.6 conclui o capítulo.

3.1 Spam

O crescimento da popularidade e o baixo custo dos serviços de e-mail tem impulsionado comerciantes a utilizar o correio eletrônico como forma de propaganda, enviando

mensagens não solicitadas a milhares de usuários da rede. Essas mensagens são conhecidas como *spam* ou *UCE* (*Unsolicited Commercial E-mail*) e consomem recursos importantes dos sistemas de computação. Além das propagandas indesejáveis, *spammers* utilizam os sistemas de e-mail para propagar mensagens com programas maliciosos que prejudicam o uso dos correios eletrônicos [CL98]. Para um terço dos usuários da internet, cerca de 80% das mensagens recebidas são *spams* [GHR05]. No ano passado, o Brasil ocupou o 5^o lugar no ranking de países que mais enviam *spams* no mundo, representando 3,34% do total [GHR05]. Estima-se que o tráfego médio diário de *spams* seja de 17 bilhões de mensagens e que a tendência é aumentar para 23 bilhões até 2007 [IDC04].

Os *spams* ocasionam diversos inconvenientes para usuários e administradores de rede como, por exemplo, o não recebimento de e-mails válidos (pela saturação das caixas de correio), o gasto desnecessário de tempo para receber mensagens indesejadas (fator crítico em conexões discadas), a perda de produtividade (aumento no tempo necessário para a leitura de e-mails), impacto na banda de rede (o tráfego de *spams* diminui a banda útil de rede das corporações), entre outros [Lin99].

Existem dois tipos principais de *spams*. O ***Cancellable Usenet spam*** que são mensagens enviadas para 20 ou mais *newsgroups* da *Usenet* e tem como objetivo advertir usuários sobre postagem de mensagens irrelevantes e proporcionar aos administradores e donos de grupos um melhor gerenciamento de tópicos. Os ***spams*** são mensagens não solicitadas enviadas a vários usuários diretamente a suas caixas de correio. Os destinatários de *spam* são gerados através de *scaneamento* de mensagens da Usenet, roubo de listas de e-mail, pesquisas na Web a procura de endereços, entre outros. Na maioria das vezes *spammers* utilizam ferramentas especializadas para pesquisa de endereço e envio de mensagens. As principais técnicas utilizadas para o envio de *spam* são:

- **Entrega direta**: programas de entrega montam um servidor SMTP no computador *spammer* (gerador de *spam*) que enviam grandes volumes de mensagens diretamente a outros servidores de e-mail;

- ***Open relay***: um erro freqüente de configuração nas listas de controle de acesso de servidores de e-mail, conhecido como *open relay*, permite que um servidor mal-configurado seja usado como propagador involuntário de grandes volumes de *spam*;

- **Remetentes forjados**: fragilidades no protocolo SMTP permitem forjar campos dos cabeçalhos de e-mail, entre os quais o remetente; essa fragilidade é amplamente explorada por *spammers* e vírus, juntamente com as técnicas anteriores.

De acordo com [Str03], o custo total das empresas dos Estados Unidos no ano de

2003 com *spam* foi de US$ 8.9 bilhões. Esse custo foi baseado na perda de produtividade de usuários que gastam uma média de 4,4 segundos para classificar uma mensagem como spam e tomar uma ação; no aumento do tráfego da banda de rede, gasto com armazenamento e processamento de mensagens; e com problemas de responsabilidade legal de mensagens que trafegam pela rede da corporação.

3.2 Scam

O *scam* é um subconjunto de *spam* que faz uso de engenharia social em conjunto com fragilidades no protocolo SMTP, visando enganar os destinatários, convencendo-os a informar dados bancários, números de cartões de créditos e outras informações confidenciais ou instalar *spywares*, que é o objetivo mais freqüente hoje em dia.

Os *phishing scams* são os *scams* mais comuns atualmente. O objetivo deste ataque é criar mensagens falsificadas de sites e empresas que sejam capazes de enganar usuários, convencendo-os a informar senhas e informações confidenciais. Estimativas apontam que a eficácia deste tipo de ataque varia de 1% a 20% dependendo da falsificação (atacantes podem facilmente copiar imagens, links e textos de sites legítimos para que seus e-mails pareçam legítimos [Kop04]). Clientes de muitos bancos e instituições financeiras têm sido alvos deste tipo de ataque, sendo roubadas senhas e números de cartões de débito e crédito.

Organizações e companhias preocupadas com os *phishing scams* estão propondo alterações na infra-estrutura de e-mail visando diminuir os *spams* que minimizará consideravelmente os ataques de *scam*. Dentre as propostas destacam-se autenticações em transações eletrônicas, desenvolvimento de anti-spam e anti-vírus mais eficazes, desenvolvimento de softwares de privacidade, entre outros [Abd04]. Uma solução para este problema consiste em utilizar sistemas de assinatura/certificado digital ou então tecnologias que validem a autenticidade do servidor ou do remetente da mensagem. Algumas dessas técnicas serão vistas no capítulo 4.

3.3 Técnicas de contenção de Spam

Várias técnicas vêm sendo usadas para controle de *spam*. As principais são baseadas em listas de servidores confiáveis e não-confiáveis, ou na filtragem das mensagens recebidas com base em seu conteúdo. Essas técnicas estão relacionadas com a habilidade de servidores de e-mail serem capazes de distinguir *spams* de mensagens legítimas. É

possível classificar as técnicas de anti-spam da seguinte forma:

- **Listas Negras (*Black Lists*)**: servidores *RBL* (*Realtime Blackhole Lists*) distribuídos na Internet mantêm listas de endereços IP de geradores ou propagadores de *spam*. Estas listas podem ser consultadas por meio de DNS pelos servidores de e-mail para verificar a confiabilidade de um remetente [JS04]. São mantidas por meio de vários mecanismos, como contribuições de usuários, resultados de varreduras automáticas, etc.

- **Listas Brancas (*White Lists*):** cada servidor de e-mail pode manter uma lista de remetentes nos quais confia. Esta lista é normalmente mantida por meio de pedidos de confirmação de envio que remetem a um formulário web ou outra abordagem similar [Hal98]. Alguns sistemas alimentam suas listas brancas com outras técnicas, como forçar o remetente a tentar o mesmo envio várias vezes, constituindo uma estratégia denominada "*greylisting*" [Fro04].

- ***Reverse MX Lookup*:** quando um e-mail é enviado, uma conexão TCP/IP é feita entre os dois servidores. O servidor de e-mail que está recebendo o e-mail pode pegar o endereço de IP do emissor e fazer uma consulta DNS para verificar se as informações contidas na mensagem coincidem com as informações do DNS, possibilitando a verificação de emissores falsos.

- ***Honeypots*:** são servidores de e-mail com endereços falsos utilizados para enganar *spammers* e classificar técnicas de *spam*. Esses servidores armazenam os *spams* em bases de dados que serão utilizadas futuramente nos filtros anti-spam. Os *honeypots* estão sendo muito utilizados na descoberta de novos ataques a computadores efetuados por hackers e crackers.

- **Filtros antispam**: programas de filtragem que classificam os e-mails de acordo com seu conteúdo Podem ser usadas técnicas estatísticas, classificação *bayesiana*, redes neurais, etc [SDHH98]. Muitos *spammers* tentam burlar tais filtros por meio de técnicas que permitem inserir textos em imagens, usar estruturas HTML para confundir os filtros, entre outras.

3.4 Trojans

Tojans são programas criados para contornar a segurança de uma máquina. Assim, como o cavalo de Tróia da história, esses programas escondem dentro de sua estrutura

códigos maliciosos para facilitar a entrada em máquinas. Esses programas não podem executar sozinhos. É necessário a ajuda de usuários para que o código malicioso seja introduzido no computador [HKS01]. Existem três tipos de *trojans* em sistemas de computadores:

- **Programa cavalo de Tróia**: programa malicioso que se disfarça para contornar a segurança de computadores em segredo;
- **Código-fonte troiano**: uma cópia de código fonte de programas modificados para conter alguma *backdoor* ou brecha na segurança;
- **Utilitários troianos**: após uma invasão, um atacante pode substituir binários do sistema por versões que contém *backdoors* ou que ocultam suas atividades no sistema.

O programa geralmente é atraente ao usuário (jogo, cliente IRC, sistema de mensagem instantânea). Sempre que esses programas são executados, eles acabam tendo as mesmas permissões do usuário da máquina. Crackers mal intencionados utilizam grandes listas de correio eletrônico (*Spam*) para enviar códigos *Trojans*, esperando que um dos destinatários o execute. A propagação ocorre também entre amigos, que enviam o programa com "boa aparência" para suas listas de correio eletrônico. Programas *Trojans* são um meio fácil de um cracker inserir códigos maliciosos em diversos computadores.

Um exemplo é o *Trojan Remote Shell* do Linux, também conhecido como *RST.b*. Se um atacante puder enganar um usuário para baixar e executar o binário RST.b, infectará os arquivos do sistema `/bin`, se possível, e executará na porta 5503 uma *backdoor*. O atacante poderá se conectar a essa porta e emitir comandos UNIX executados como o usuário que iniciou o *Trojan RST.b* [HKS01].

3.5 Vírus e worms

Os vírus de computadores são programas capazes de infectar outros programas, escondendo uma cópia de si próprio em programas normais [Coh90]. O *worms* são programas capazes de executar e se propagar automaticamente entre diversas máquinas através de conexões na rede. Diferente dos vírus, os *worms* não alteram programas, contudo eles podem carregar códigos maliciosos tal com um vírus [SHF90]. Ambos (vírus e *worms*) são programas que têm como objetivo inutilizar sistemas, destruir arquivos, roubar informações importantes de uma máquina ou simplesmente propagar-se ao extremo. Um vírus de e-mail normalmente é constituído por um e-mail com um anexo executável (ou um

código HTML que permita carregar um arquivo executável armazenado remotamente). Esse código executável pode ser ativado pelo usuário ou até mesmo de forma automática, explorando vulnerabilidades no cliente de e-mail.

A propagação de um *worm* em sistemas de e-mail acontece por meio da exploração de fragilidades nos clientes de e-mail através do uso da lista de contatos do usuário [Kle01b]. Máquinas infectadas por vírus podem oferecer *backdooors* que permitem seu uso em diversos tipos de atividades nocivas.

Empresas de software desenvolvem anti-vírus e mantêm bases de assinaturas já conhecidas. Porém, o atraso na atualização dessas bases pode ocasionar infestações e propagação de vírus pela rede. Usuários de computadores pessoais precisam atualizar constantemente seus sistemas de anti-vírus. Com o intuito de minimizar este problema em serviços de e-mail, provedores de acesso estão implantando anti-vírus na entrada das mensagens, diminuindo a possibilidade de transporte de vírus anexados em mensagens. Todavia, estudos recentes demonstram que a abordagem baseada nas assinaturas de vírus nunca resolverá o problema, pois novos vírus se propagam mais rapidamente que as atualizações dos anti-vírus [ZGT02].

3.6 Conclusão

Neste capítulo foram apresentadas as principais ameaças em sistemas de e-mail. Usuários maliciosos são capazes de roubar informações, danificar sistemas e afetar o desempenho da rede utilizando programas que são propagados através dos sistemas de e-mail e mensagens não solicitadas enviadas a diversos usuários da rede. Para minimizar esses problemas, propostas em diversas áreas tem sido realizadas, contudo, o problema está longe de ter uma solução definitiva devido à complexidade de classificação de mensagens inválidas e autenticação de usuários na rede.

O próximo capítulo apresenta as principais propostas baseadas em cliente e em servidor que visam a autenticação de remetente em sistemas de e-mail.

Capítulo 4

Autenticação de remetentes

A autenticação de remetentes é implementada em clientes e/ou servidores de e-mail para atestar usuários de sistemas de transporte de e-mail. Autenticar os usuários desses sistemas é o primeiro passo para contra-atacar os problemas apontados no capítulo anterior, visto que os *spams*, *scams* e programas maliciosos são geralmente propagados por usuários/domínios forjados. As principais propostas de autenticação de remetentes são o *PGP*, *S/MIME*, *SPF*, *SenderID* e*DomainKeys*.

O S/MIME [Ram04] e o PGP [Zim95] são ferramentas implantadas em computadores de usuários de e-mail, sendo transparente aos servidores de e-mail, para assinar e cifrar arquivos e/ou documentos a serem enviados pela rede. Essas ferramentas garantem a privacidade, integridade e autenticidade de mensagens utilizando sistemas baseados em chaves assimétricas para cifrar e assinar mensagens. Quando uma mensagem for cifrada com a chave pública de um receptor, apenas o receptor será capaz de decifrá-la. Da mesma forma, quando uma mensagem for assinada por um emissor utilizando sua chave privada, apenas a chave pública deste emissor será capaz de decifrar a assinatura, garantindo a autenticidade do remetente.

O SPF (*Sender Policy Framework*) [Won04] cria uma infra-estrutura de verificação de autenticidade entre servidores de e-mail, sendo transparente aos clientes. O SPF utiliza os serviços de DNS para prover ao servidor destinatário informações sobre o servidor de origem de um e-mail. Essas informações são descritas em uma linguagem própria e armazenadas em um registro específico do servidor DNS do domínio de origem da mensagem. Ao receber uma mensagem, o servidor destinatário realiza uma consulta de DNS para obter as informações do domínio de origem e comprovar a autenticidade do remetente.

O *SenderID* [WML04] é uma melhoria do SPF que modifica a linguagem SPF e propõe um conjunto de novas especificações para resolver os problemas relacionados à

autenticação de mensagens encaminhadas (*forwarded messages*) encontrados no SPF.

O *DomainKeys* [DY04] utiliza uma infra-estrutura de chaves públicas para validar a autenticidade de um servidor de e-mail. A chave privada é armazenada localmente em cada servidor, sendo utilizada para gerar uma assinatura no cabeçalho de cada e-mail a ser enviado. A chave pública é armazenada em um registro de DNS do domínio local, sendo requisitada pelos servidores destinatários para validar a assinatura de cada e-mail recebido.

Este capítulo apresenta os sistemas de autenticação de remetentes de e-mail e está estruturado da seguinte forma: a seção 4.1 apresenta arquiteturas de autenticação baseadas em certificados; a seção 4.2 apresenta a especificação do SPF e a forma como ele disponibiliza informações sobre usuários do domínio; a seção 4.4 apresenta o Sender ID, uma extensão do SPF; a seção 4.5 aborda o DomainKeys, um sistema de autenticação de servidores baseado em chaves; por último, a seção 4.7 conclui o capítulo.

4.1 Autenticação baseada em certificados

Os certificados digitais foram criados para atestar a autenticidade de usuários/participantes de uma rede de computadores. Os certificados são criados a partir do algoritmo RSA de criptografia de chave pública [Sch94] (método desenvolvido em 1977 por Ron Rivest, Adi Samir e Len Adleman). Um certificado digital nada mais é que uma credencial com uma assinatura gerada a partir de uma chave privada RSA, a chave pública correspondente à chave privada e informações adicionais sobre o certificado, tais como validade, dono, tamanho da chave de criptografia, etc.

A autenticidade de um certificado digital é garantida pelas propriedades de chaves assimétricas definidas em [Sch94], onde, uma informação cifrada com uma chave poderá ser decifrada apenas utilizando a chave assimétrica correspondente a chave que cifrou a informação. Sendo assim, o certificado assinado por uma chave privada poderá ser decifrado apenas pela sua chave pública correspondente, garantindo assim a autenticidade do certificado. Os padrões de Criptografia de Chave Pública encontrados na literatura são: X.509 e PGP.

4.1.1 Padrão X.509 e S/MIME

O X.509 é um padrão para PKI (*Public Key Infrastructure*) que faz uso de autoridades certificadoras para criar e atestar certificados digitais na rede. O modelo do X.509

é baseado em uma estrutura de árvore hierárquica onde a raiz desta árvore representa o principal fornecedor de certificado chamado de autoridade certificadora raiz (CA raiz). A figura 4.1 mostra a estrutura de uma PKI.

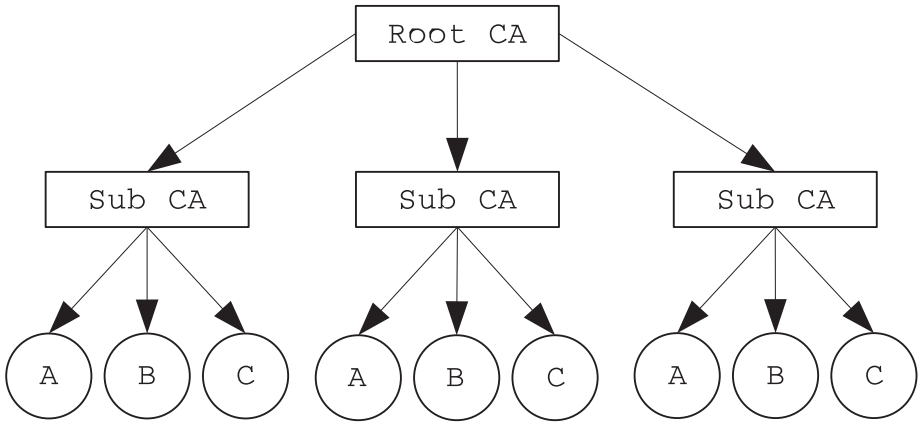

Figura 4.1: Exemplo de uma Infra-estrutura de Chave Pública X.509 [CDN05]

O padrão X.509 garante a confiança de um certificado a partir da aceitação do certificado da CA raiz como confiável. Por causa da estrutura de árvore do X.509, um usuário deverá confiar automaticamente em todos os certificados criados pela CA. Os principais navegadores WEB e clientes de e-mail implementam o padrão X.509 e possuem um conjunto de certificados digitais de CAs mais conhecidas (tais como Verisign e Trustcenter) para validar os certificados criados por tais autoridades.

O S/MIME (*Secure/Multipurpose Internet Mail Extentions*) é um modelo definido por um grupo de empresas (incluindo *RSA Security*, *Microsoft*, a *Lotus*, *Novel*, etc), que possibilita enviar mensagens através do sistema de e-mail de uma maneira segura. Clientes de e-mail que tenham suporte ao S/MIME são capazes de adicionar facilmente assinaturas digitas em mensagens e cifrá-las [Ram04].

O S/MIME foi baseado na especificação do MIME. O S/MIME disponibiliza um serviço de cifragem que possui algoritmos de criptografia especificados pelo usuário; autenticação por meio de chaves públicas X.509; e não repúdio por meio de mensagens assinadas criptograficamente. Com o S/MIME é possível:

- Cifrar uma mensagem, quando se interessa somente sigilo ou confidencialidade;
- Assinar uma mensagem, quando se interessa somente autenticidade e integridade;

- Cifrar e assinar uma mensagem, quando se interessa confidencialidade, autenticidade e integridade.

A confidencialidade das mensagens que utilizam o S/MIME é garantida pelo uso de algoritmos de criptografia simétricos. Nos Estados Unidos o algoritmo de exportação autorizado é o RC2-40 com chaves de 40 bits. Por isso os mailers utilizam algoritmos RC2-40. Fora dos Estados Unidos pode ser utilizado o DES com chave de 56 bits ou então 3DES com chaves de 168 bits. No caso da integridade e autenticidade de mensagens, o S/MIME utiliza assinaturas digitais que são certificadas por CAs a partir do padrão X.509.

A grande falha no X.509 é que se de alguma forma a raiz CA for comprometida, nenhum dos certificados criados por ela poderão ser considerados confiáveis. Além disso a estrutura de árvore não é muito flexível [Ram04].

4.1.2 PGP

O PGP (*Pretty Good Privacy*) [Zim95] é uma ferramenta de criptografia e certificação digital que utiliza criptografia assimétrica (utilizando chaves pública e privada) para cifrar/decifrar e assinar mensagens a serem trafegadas pela rede. Esta ferramenta foi desenvolvida como alternativa ao X.509 que possui falhas em sua estrutura hierárquica. Ao contrário do X.509 o PGP utiliza um modelo de confiança baseado em sistemas ponto a ponto (*peer-to-peer*) descentralizando os certificados digitais. Isto significa que cada usuário é sua própria raiz CA. A confiança é expressa através da assinatura da chave pública de um usuário com a chave assimétrica correspondente à chave pública (confiança direta) ou por inferência de confiança através de uma base de chaves local (confiança indireta).

O PGP usa chaves de sessão que são criadas a partir de um número gerado aleatoriamente a cada cifragem. As chaves de sessão são utilizadas para cifrar os dados utilizando algoritmos de criptografia simétrica. A chave de sessão é cifrada utilizando a chave pública do destinatário. Ao receber uma mensagem cifrada, o destinatário utiliza a chave privada para recuperar a chave de sessão e decifrar o texto. O motivo pelo qual os autores optaram por chaves de sessão se dá pelo fato dos algoritmos de criptografia simétrica serem muito mais rápidos (tanto na cifragem quanto na decifragem) que os algoritmos RSA. Mesmo que o dado tenha sido gerado por uma chave não segura, esta chave é cifrada pela chave pública do usuário e enviada pela rede garantindo a privacidade, velocidade de decifragem

e segurança da informação.

As assinaturas digitais são geradas de maneira similar a cifragem dos dados. A diferença é que o PGP utiliza a chave privada para gerar a assinatura digital. Para comprovar a autenticidade da assinatura, um usuário deverá utilizar a chave pública do possível emissor para decifrar a assinatura. Se a assinatura for corretamente decifrada isto indica que a chave foi gerada pelo seu par (privada - pública). Do contrário a assinatura foi gerada por um emissor malicioso.

As abordagens baseadas em ponto a ponto são mais flexíveis que a abordagem hierárquica X.509. A fragilidade desta abordagem reside na falta de propagação de valores de confiança, contudo, pesquisas recentes foram iniciadas para desenvolver algoritmos de propagação de reputação e confiança. Alguns desses estudos serão abordados no decorrer do capítulo.

4.2 Sender Policy Framework

O SPF (*Sender Policy Framework*) [Won04] é uma especificação de autenticação de servidores de e-mail que visa diminuir o tráfego de mensagens indesejáveis em servidores de e-mail (SMTP), através de políticas e filtros no recebimento das mensagens utilizando um ambiente que trabalha em conjunto com servidores DNS.

As políticas e os filtros do SPF são definidos a partir de uma linguagem simples que descreve o domínio de um determinado servidor de e-mail. A partir dessas informações, servidores serão capazes de avaliar se uma mensagem foi originada por um determinado domínio. Caso seja comprovada uma fraude de domínio, administradores de servidores de e-mail serão capazes de descartar, analisar ou então manipular o e-mail fraudulento de acordo com as políticas do servidor.

4.2.1 Fundamentos do SPF

Servidores que utilizam a especificação SPF deverão possuir atributos únicos que serão utilizados por outros servidores de e-mail para validar a autenticidade de mensagens. Esses atributos fazem parte de um registro SPF (detalhado a seguir) que será utilizado por servidores de e-mail (receptores) para classificar uma mensagem, possibilitando assim que mensagens com remetentes falsos sejam descartadas. Para facilitar a consulta de outros

servidores, os registros SPF são armazenados no servidor DNS em um campo TXT. A figura 4.2 apresenta o funcionamento básico da especificação SPF.

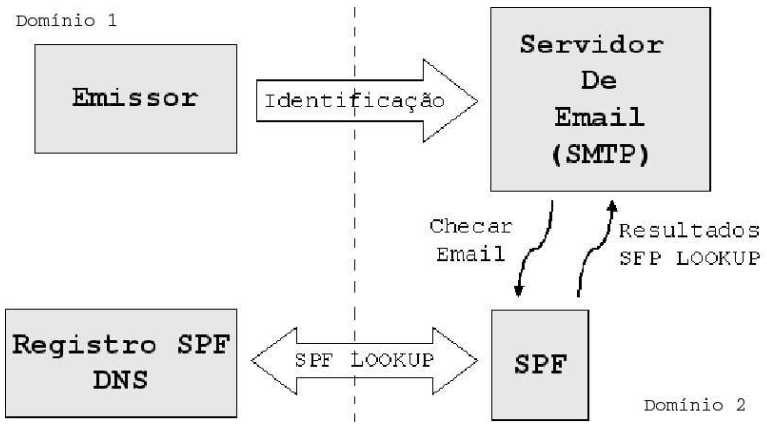

Figura 4.2: Modelo SPF[Won04]

Nesta figura o servidor emissor do **domínio 1** abriu um canal de comunicação SMTP com o servidor receptor do **domínio 2** e está enviando mensagens. Ao receber uma mensagem, o servidor receptor deverá requisitar ao SPF a validação do domínio do usuário emissor. Em seguida o SPF fará uma consulta ao servidor de DNS do **domínio 1** verificando a autenticidade do domínio. Em seguida o SPF deverá retornar ao servidor de e-mail uma das seguintes respostas:

1. **None**: não existem registros publicados pelo domínio de origem;
2. **Neutral**: deve ser tratado exatamente como o resultado *None*;
3. **Pass**: cliente está autorizada a enviar o e-mail com a identidade "Mail From". Políticas de checagem podem ser processadas com confiança baseadas na identidade "Mail From";
4. **Fail**: não está autorizado a utilizar o domínio. Se o software de checagem escolher rejeitar o e-mail durante a transação SMTP, então ele deverá retornar um código SMTP 550;
5. **SoftFail**: o domínio acredita que o host não está autorizado mas não pode definir com segurança o estado. O software não deve rejeitar a mensagem baseado no resultado, mas deverá tratar a mensagem como suspeita;

6. **TempError**: o servidor de recebimento encontra-se numa transição de erro na checagem. O software pode escolher aceitar ou rejeitar temporariamente a mensagem. Se a mensagem for rejeitada durante a transação SMTP, o software deverá responder com um código SMTP 450.

7. **PermError**: o dominío não pode interpretar a identidade de "Mail From" corretamente. O software poderá rejeitar a mensagem. Se for rejeitada durante a transação de SMTP, deverá responder com um código SMTP 550.

4.2.2 Registros SPF

Um registro SPF armazena informações sobre o domínio de e-mail na internet (tais como *hosts*, servidores MX, IP de servidores MX e outros) que indicam quais entidades possuem permissão para utilizar o nome do domínio em questão. Os registros SPF são publicados em servidores DNS de seu domínio, que deverão ser consultados toda vez que existir a necessidade de validação da autenticidade de uma mensagem. Os campos de um registro SPF são: versão SPF, mecanismos, prefixos e modificadores. Como por exemplo de registro SPF:

"`v=spf1 +mx +a +ptr include:myspf.com.br exp=spf -err +all`"

onde:

- **v=spf1**: versão do SPF;
- **mx, a, ptr, include, err e all**: mecanismos do SPF;
- **+ e -**: prefixos do SPF (devem precede os mecanismos). Se não existir um prefixo especificado o padrão será (+);
- **exp**: modificador (pode ser zero, um ou dois modificadores).

A seguir são detalhados cada um dos componentes que fazem parte do registro SPF.

Versão

A versão SPF mantém a compatibilidade de possíveis alterações no protocolo/registros SPF. Todos os registros SPF deverão ser iniciados com a versão SPF. Os clientes SPF deverão utilizar apenas registros que são da mesma versão ou então versões compatíveis ao processador SPF do cliente. Todos os registros com versão incompatível ao processador de registros SPF do cliente deverão ser ignorados.

Prefixos

Os prefixos definem como um mecanismo deverá se comportar caso um registro SPF seja encontrado. Os prefixos são: + (pass), − (fail), ∼ (softfail) e ? (neutral). Se o prefixo não estiver definido, o SPF utilizará como padrão o prefixo +. Como por exemplo:

"`v=spf1 +mx +ptr include:myspf.com.br exp=spf -err +all`"

onde, todos os mecanismos antecedidos por um prefixo serão testados e deverão retornar *pass* ou *fail* de acordo com a informação pesquisada.

Mecanismo

Os mecanismos SPF são responsáveis por identificar endereços IP autorizados a enviar mensagens a partir de um determinado domínio. Existem dois mecanismos básicos definidos no SPF relacionados às categorias de IPs (internos e externos ao domínio): **all** e **include**. Os demais mecanismos são responsáveis por autorizar e designar emissores. Os mecanismos são:

- `all`: testa se todas as informações coincidem (mecanismos após o all não serão testados);
- `include`: executa uma consulta recursiva ao SPF;
- `a`: verifica se o host de envio é um endereço IP válido;
- `mx`: verifica se o *host* de envio é um *host* MX do domínio;
- `ptr`: testa se o nome do *host* de envio encontra-se em um determinado domínio;

- `ip4/ip6`: testa se o *host* de envio está utilizando uma rede IP;
- `exists`: constrói um nome de *host* arbitrário que será utilizado numa consulta de registro DNS A. Isto possibilita um esquema complicado envolvendo partes arbitrárias de um e-mail para determinar o que é legal.

Outros mecanismos podem ser definidos (o SPF possibilita a criação de novos mecanismos para serem utilizados no futuro). Desta forma, os mecanismos podem coincidir, não coincidir ou então gerar uma exceção. Caso os mecanismos coincidam, seus valores de prefixo deverão ser retornados; caso os mecanismos não coincidam, o processamento deverá continuar; caso seja lançada uma exceção, o valor da exceção será retornado.

Alguns mecanismos possuem suporte para argumentos opcionais. Esses argumentos são responsáveis por definir nomes de domínios, conjuntos de IPs e outras informações pertinentes para a validação de *hosts*/domínios.

Modificadores

Apenas dois modificadores são definidos como padrão na especificação do SPF: "redirect" e "exp". Esses modificadores deverão ser implementados nos clientes SPF. Os modificadores disponibilizam informações adicionais ou então alterações no curso do processamento do SPF. Os modificadores disponibilizam também uma maneira fácil de estender o registro SPF. Os modificadores são:

- redirect: se todos os mecanismos falharem e um modificador redirect estiver definido, então o modificador irá alterar o domínio para ser processado;
- exp: utilizado para gerar uma mensagem SMTP para o *host* emissor caso o processamento do SPF resultar em rejeição.

4.3 Limitações da arquitetura

O SPF possui algumas limitações e problemas que não foram originalmente abordadas pelos autores. Segundo [Cun05], suas principais limitações são:

1. Servidores que possuem múltiplos domínios podem ser utilizados por *spammers* para envio de mensagens, pois usuários do domínio **dominio1.com.br** são capazes de se passar por um determinado usuário do domínio **dominio2.com.br** que encontram-se no mesmo servidor físico;

2. O *forwarding* de e-mail pode ser comprometido, pois um agente é capaz de repassar uma mensagem sem alterar o endereço em "from:". Como o SPF valida o conteúdo da mensagem, o retorno para *forwarding* será sempre **fail**, pois as informações da consulta não irão coincidir com o servidor utilizado para fazer o *forward* da mensagem;

3. Se um servidor DNS for invadido, as informações poderão ser falsificadas (DNS *poison*);

4.4 Sender ID

A *framework* SIDF (*SenderID Framework*) [WML04] foi desenvolvida por um conjunto de empresas para prover autenticação de domínio em servidores de e-mail. Esta *framework* checa cada mensagem recebida pelo servidor, validando se o **servidor de e-mail emissor** pode enviar a mensagem originada pelo domínio do **emissor da mensagem**. Esta verificação é feita de forma automática através do servidor receptor antes que a mensagem seja entregue ao usuário. O ciclo de vida de uma comunicação SMTP utilizando o SenderID pode ser visto na figura 4.3.

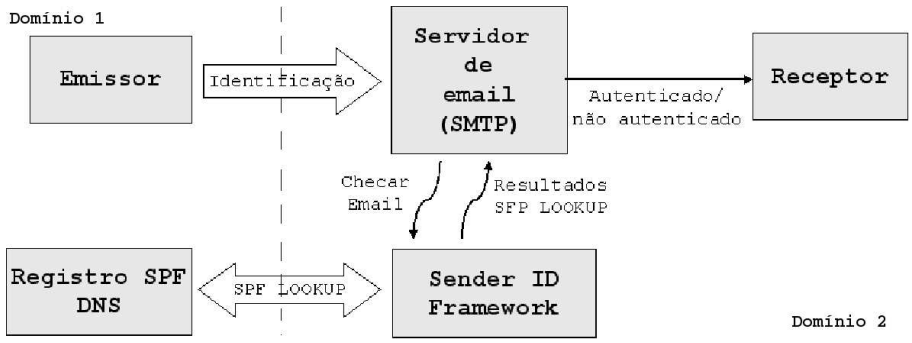

Figura 4.3: Funcionamento do Sender ID [WML04]

Nesta figura o servidor emissor do **domínio 1** abriu um canal de comunicação SMTP com o servidor receptor do **domínio 2** e está enviando mensagens. Ao receber uma mensagem, o servidor receptor deverá requisitar ao SIDF a validação do domínio do usuário emissor. Em seguida o SIDF fará uma consulta ao servidor de *DNS SPF Records* do **domínio 1** verificando a autenticidade do domínio. Em posse dessas informações o servidor de e-mail poderá receber ou rejeitar a mensagem. A *framework* SIDF é dividida

em três partes:

1. extrair o endereço do remetente do cabeçalho da mensagem;
2. extrair o domínio do remetente;
3. chamar a função `check_host` que deverá fazer uma consulta ao servidor DNS do domínio extraído no passo 2 verificando a autenticidade do usuário. Os parâmetros a serem passados para a função `check_host` são:
   (a) endereço IP;
   (b) domínio do remetente;
   (c) endereço do remetente;

   Como resultado da consulta deverá ser retornado um dos resultados apresentados na seção 4.2.1. Este resultado poderá ser utilizado tanto para negar a mensagem quanto como parâmetro adicional dos filtros de mensagem do servidor de e-mail.

O cenário a seguir apresenta um exemplo de uso do SIDF, sendo que o endereço "*From:*" do cabeçalho da mensagem é ***maria@ppgia.pucpr.br*** e existe apenas um registro SPF no DNS do domínio emissor.

Registro: "`acme.com.br v=spf1 ip4:192.0.2.0/24 -all`"

1. addr = maria@ppgia.pucpr.br
2. domain = ppgia.pucpr.br
3. result = check_host("127.0.0.1", domain, addr)

O resultado do *check_host* deverá ser fail, pois as informações coletadas pelo SIDF não estão presentes no *DNS SPF Record*.

4.4.1 Registros Sender ID e o PRA (Purpoted Responsible Address)

O SIDF utiliza o SPF [Won04] para disponibilizar as informações sobre um determinado domínio. Conforme visto na sessão 4.2.2 esses registros informam quais hosts podem enviar mensagens utilizando o domínio que armazena os registros. A seguir são apresentados alguns exemplos de **Registro Sender ID**:

- `exemplo.com.br TXT ‘‘v=spf1 ip4:192.0.2.0/24 -all’’`: especifica um conjunto de IP;
- `example.com TXT ‘‘v=spf1 -all’’`: este domínio nunca envia e-mail;
- `example.com TXT ‘‘spf2.0/pra ip4:192.0.3.0/24 -all’’`: diferente configuração de checagem PRA.

Note que o último exemplo possui uma versão 2.0 do SPF. A versão 2 foi proposta para suportar o PRA (*Purpoted Responsible Address*) [Lyo05]. O PRA foi projetado para resolver os problemas de *forwarding*, onde, em alguns casos, uma mensagem pode trafegar por mais de um MTA, não sendo possível validar o domínio/IP do servidor emissor correto. O PRA define a entidade mais recente (de acordo com o cabeçalho) que enviou a mensagem, utilizando campos adicionais no cabeçalho do e-mail. O PRA de uma mensagem é determinado pelo algoritmo a seguir [Lyo05]:

1. Selecione o primeiro cabeçalho `Resent-Sender` não vazio da mensagem. Se nenhum cabeçalho for encontrado, vá para o passo 2. Se o cabeçalho encontrado for precedido por um cabeçalho `Resent-From` não vazio e um ou mais cabeçalhos `Received` ou `Return-Path` ocorrerem depois do cabeçalho `Resent-From` e antes do cabeçalho `Resent-Sender`, vá para o passo 2. Do contrário, vá para o passo 5.
2. Selecione o primeiro cabeçalho `Resent-From` não vazio da mensagem. Se o cabeçalho `Resent-From` for encontrado, vá para o passo 5. Do contrário, vá para o passo 3.
3. Selecione todos os cabeçalhos `Sender` não vazios da mensagem. Se existir exatamente um cabeçalho, vá para o passo 5. Se existir mais de um cabeçalho, vá para o passo 6. Do contrário, vá para o passo 4.
4. Selecione todos os cabeçalhos `From` não vazios da mensagem. Se existir exatamente um cabeçalho, vá para o passo 5. Do contrário, vá para o passo 6.
5. Os passos anteriores selecionaram um único cabeçalho da mensagem. Se este cabeçalho for mal formado (aparentemente possui múltiplas caixas de correio, ou uma única caixa de correio não possui um nome de domínio, etc), vá para o passo 6. Do contrário, retorne que a caixa de correio é um PRA.
6. A mensagem é mal formada, e é impossível determinar um PRA.

4.4.2 ESMTP SUBMITTER

O SIDF propõe ainda o ESMTP `SUBMITTER` [AK04], que dá suporte a inclusão do emissor atual da mensagem no protocolo SMTP. Esta extensão deve obedecer aos seguintes padrões:

1. nome do serviço ESTMP é *Responsible Submitter*;
2. chave `EHLO` deve estar associada à extensão `SUBMITTER`;
3. chave `SUBMITTER` não possui parâmetros;
4. parâmetro opcional deverá ser incluído no comando `MAIL` que utilizará a chave `SUBMITTER` para especificar o endereço do emissor responsável pelo reenvio da mensagem.

Um exemplo do uso do ESMTP `SUBMITTER` pode ser visto a seguir [AK04]:

```
1  S: 220 company.com.example ESMTP server ready
2  C: EHLO almamater.edu.example
3  S: 250-company.com.example
4  S: 250-DSN
5  S: 250-AUTH
6  S: 250-SUBMITTER
7  S: 250 SIZE
8  C: MAIL FROM:<alice@example.com>
9  SUBMITTER=bob@almamater.edu.example
10 S: 250 <alice@example.com> sender ok
11 C: RCPT TO:<bob@company.com.example>
12 S: 250 <bob@company.com.example> recipient ok
13 C: DATA
14 S: 354 okay, send message
15 C: Resent-From: bob@almamater.edu.example
16 C: Received By: ...
17 C: (message body goes here)
18 C: .
19 S: 250 message accepted
20 C: QUIT
21 S: 221 goodbye
```

Utilizando o SUBMITTER, um servidor de e-mail será capaz de recuperar o endereço do emissor e fazer a consulta SPF no domínio do emissor através do protocolo SMTP.

4.4.3 Limitações

As principais limitações no Sender ID são: o SIDF autentica os domínios e não os usuários; valida apenas o último salto; *spammers* são capazes de registrar seus próprios domínios, fazendo com que seja necessário um esforço investigativo e criar sistema de reputação de domínios.

4.5 DomainKeys

O DomainKeys é uma tecnologia desenvolvida por um grupo de pesquisas do Yahoo [DY04] que, assim como o SPF e SIDF tem como objetivo autenticar o domínio e validar a integridade das mensagens enviadas. Além disso, o DomainKeys foi projetado para ser transparente, 100% compatível com a infra-estrutura existentes, independente de clientes e não requer uma autoridade de certificação centralizada.

4.5.1 Fundamentos do DomainKeys

O DomainKeys cria um par de chaves pública/privada para cada domínio/subdominío e armazena em locais seguros do servidor. A chave pública é armazenada em um registro do DNS (disponibiliza essas chaves para consultas de outros domínios da rede). A chave privada é armazenada em um diretório ou qualquer outra forma de armazenamento local, acessado pelo sistema de saída do servidor de e-mail para gerar uma assinatura no cabeçalho de mensagens que serão enviadas pelo sistema utilizando a chave privada. A Figura 4.4 apresenta o fluxo do DomainKeys.

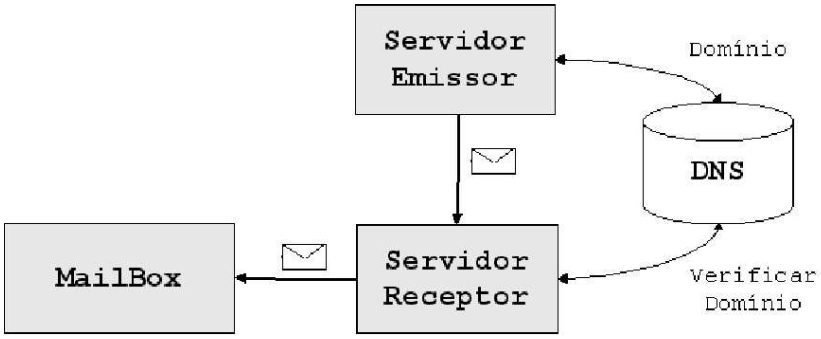

Figura 4.4: Funcionamento do DomainKeys [DY04]

Quando a mensagem for enviada por um servidor com suporte a DomainKeys, o servidor de destino deverá submeter a mensagem recebida as seguintes etapas de autenticação:

1. Extrair a assinatura cifrada pela chave privada do emissor do e-mail;
2. Solicitar a chave pública do domínio em questão;
3. Utilizar a chave pública para verificar se a assinatura foi gerada a partir da chave privada do servidor emissor. Se a chave pública for capaz de decifrar a assinatura, significa que o e-mail foi originado pelo domínio descrito no cabeçalho da mensagem. Do contrário, o emissor é falso e a mensagem poderá ser descartada.

4.5.2 Registros DomainKeys

Para evitar conflito de *namespace* no DNS, a especificação do DomainKeys propõe o uso de um *namespace* especial: "_domainkey". Este *namespace* será reservado para armazenar as chaves públicas de um domínio no formato: *_domainkey.dominio1.com.br*. O *namespace* deverá ser dividido em um ou mais "*selectors*". Os *selectors* são níveis únicos do namespace definidos por uma string. Um exemplo de *selector* pode ser visto a seguir:

"`teste._domainkey`"

onde *teste* indica o nome do *selector*.

Cada domínio deverá definir a quantidade de chaves públicas em conjunto com seus respectivos *selectors*. Por padrão os algoritmos de cifragem utilizados pelo Domainkeys

são o RSA/SHA1, contudo, novos algoritmos podem ser utilizados, tal como o PGP.

Representação de uma chave pública no DNS

A chave pública no DomainKeys é representada da seguinte forma:

“`teste._domainkey IN TXT ‘‘k=rsa; p=MEwwDQYJKoZLhvcNAQEB’’`”

onde:

- `g`: granularidade da chave;
- `k`: tipo da chave;
- `n`: notas que podem ser de interesse humano (nenhuma interpretação é feita pelos programas);
- `p`: chave pública codificada em Base64 se o valor for vazio, significa que a chave foi revogada;
- `t`: modo de teste. Indica se esta chave é apenas para teste.

Cabeçalho do e-mail

A assinatura digital gerada pelo DomainKeys para autenticar a mensagem deve ser armazenada como uma linha do cabeçalho da mensagem a ser transmitida; o campo que armazena esta assinatura é o “DomainKey-Signature:”. Um exemplo do campo pode ser visto a seguir:

“DomainKey-Signature: a=rsa-sha1 s=teste; d=dominio1.com.br; c=simple; q=dns; b=dzdVyOfAKCDLXdOc8GCQ;”, onde:

- `a`: algoritmo utilizado para gerar a assinatura;
- `b`: assinatura codificada em Base64;
- `c`: forma que o cabeçalho/conteúdo são apresentados;
- `d`: nome do domínio que gerou a assinatura;

- `q`: tipo de consulta para requisitar a chave pública (atualmente somente o método dns é suportado);
- `s`: "*selector*" que será consultado para requisição da chave pública.

Resultados de consultas

O DomainKeys disponibiliza um status para os clientes de e-mail que é armazenado no campo "DomainKey-Status". Os status podem ser:

- `good`: assinatura válida;
- `bad`: assinatura inválida;
- `no key`: chave inexistente;
- `revoked`: chave revogada;
- `bad format`: dados da chave inválidos;
- `non-participant`: não faz parte do domínio.

A grande maioria dos clientes de e-mail possuem suporte a configuração de regras de cabeçalho. Utilizando esta funcionalidade, é possível que usuários apliquem regras para manipular mensagens de acordo com o status retornado pelo DomainKeys.

4.6 Vantagens e Limitações

As principais vantagens de se utilizar o DomainKeys são:

- Um par de chaves pode ser criado para cada domínio da internet, tornando possível um sistema de autenticação de servidores de e-mail global;
- Qualquer domínio pode gerar suas chaves e disponibilizá-las no servidor DNS (não é necessário um repositório de chaves global);
- As informações que vão no cabeçalho SMTP não influenciam o e-mail, isto é, se um determinado servidor não suportar DomainKeys, a assinatura será ignorada.

As limitações do DomainKeys são:

- Precisa ser suportado em ambos os lados da conexão para funcionar;
- O desempenho do processo de recebimento de mensagens poderá cair, pois o DomainKeys deverá consultar o DNS, recuperar a chave, decifrar a assinatura e depois receber ou rejeitar a mensagem.

4.7 Conclusão

Neste capítulo foram apresentados os principais métodos de autenticação de domínios que são o PGP e SMIME (baseados em cliente), SPF, Sender ID e DomainKeys (baseados em servidor). A tabela 4.1 apresenta um comparativo entre as abordagens apresentadas neste capítulo.

Tabela 4.1: Quadro comparativo modelos de autenticação de remetentes

	Implementação no cliente/servidor	Utiliza chave	Modelo centralizado	Necessidade de Alterações no protocolo SMTP
S/MIME	Cliente	Sim	Sim	Não
PGP	Cliente	Sim	Não	Não
SPF	Servidor	Não	Não	Não
Sender ID	Servidor	Não	Não	Sim
Domain Keys	Servidor	Sim	Não	Não

Nesta tabela são apresentadas as principais diferenças entre os principais métodos de autenticação de remetentes. Os modelos de autenticação baseados em clientes não é escalável e difícil de manter as chaves. Os modelos de autenticação de remetente baseados possuem uma forma mais confiável de disponibilização das chaves. Ambos os modelos não tratam o conteúdo da mensagem e seus respectivos emissores.

A autenticação de domínios é o caminho inicial para a contenção de *spams*, contudo está longe de ser uma solução efetiva, pois *spammers* ainda são capazes de enviar e-mail utilizando domínios autenticados porque a autenticação do remetente não o impede de enviar e-mails com conteúdo não solicitado, apenas possibilita seu rastreamento.

No próximo capítulo serão abordadas as noções de confiança hierárquica, confiança em grupos e redes de relacionamento.

Capítulo 5

Redes de Confiança

As relações de confiança são fundamentais na construção e manutenção de uma sociedade. Por relação de confiança entende-se o quanto uma pessoa deposita confiança em outras pessoas que conhece, e como age em relação a desconhecidos. Relações de confiança são modelos dinâmicos onde os níveis de confiança podem aumentar ou diminuir de acordo com as atitudes de um indivíduo no meio. As regras para determinar atitudes positivas e negativas são definidas pela sociedade (grupos de indivíduos, governo, parentes, comunidades, etc.) onde o indivíduo encontra-se inserido. Essas regras definem punições para todos os indivíduos que não as cumprem.

As relações de confiança vêm sendo empregadas em redes de computadores para prover uma maior segurança nesses sistemas. Exemplos de relações de confiança podem ser encontrados em redes *peer-to-peer*, que utilizam algoritmos de reputação para indicar indivíduos mais ou menos confiáveis em uma rede. Para que possam ser analisados algoritmos eficazes e modelos aplicáveis a sistemas de e-mail faz-se necessário um estudo mais aprimorado dos modelos de confiança utilizados pelos seres humanos.

Os estudos comportamentais realizados em seres humanos para determinar como funcionam os modelos de confiança, apresentam um indicativo de que a confiança pode ser medida utilizando várias abordagens, dentre elas a binária (sim ou não) e a escalar (porcentagem) [Han98].

Utilizando a confiança escalar, é possível definir que um indivíduo **A** pode confiar em **B** de 0% a 100%. Além disso, se **A** confia em **B** e **B** confia em **C**, isto implica que **A** pode confiar em **C** de acordo com a propriedade de transitividade de **A**, **B** e **C**, ou seja, Se $A \rightarrow B$ e $B \rightarrow C$ então $A \rightarrow C$ [VM02].

A Figura 5.1 mostra que se **A** confia em **B** 80% e **B** confia em **C** 50%, conseqüentemente **A** confia em **C** 40%. Contudo, existem ainda os fatores externos (clima, humor,

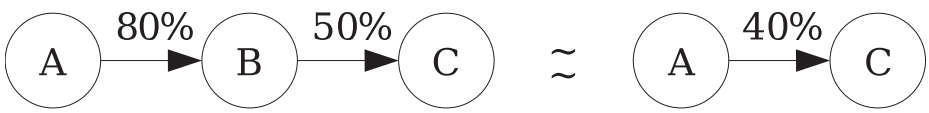

Figura 5.1: Confiança entre indivíduos distintos

local, etc) que podem influenciar na confiança final entre os indivíduos. Todos esses fatores são armazenados na memória de cada pessoa e servirão de base em suas próximas interações. A memória de um indivíduo pode ser representada na forma de uma matriz, chamada de **memória de Confiança**. Um exemplo de memória de confiança pode ser visto na tabela 5.1.

Tabela 5.1: Memória de Confiança de A

Indivíduo	**Confiança**	**Fatores externos**	**Confiança final**
B	80%	-5%	75%
C	40%	+10%	50%

Nesta tabela está sendo representada a confiança de um indivíduo **A** nos indivíduos **B** e **C**. A linha que representa a confiança de **A** em **B** mostra que a confiança é de 80%. Contudo, fatores externos fizeram com que a confiança de **A** (naquele momento) diminuísse em 5%, resultando a confiança final de **A** em **B** um total de 75%. Da mesma forma a linha que representa a confiança de **A** em **C** mostra que a confiança é de 40%. Contudo, fatores externos fizeram com que a confiança de **A** (naquele momento) aumentasse em 10%, resultando a confiança final de **A** em **C** um total de 50%.

Este capítulo traz uma visão sobre as relações de confiança entre indivíduos e seu uso em sistemas distribuídos. Este capítulo está estruturado da seguinte forma: a seção 5.1 apresenta uma visão geral sobre a confiança hierárquica de acordo com o comportamento humano; a seção 5.2 apresenta os grupos sociais; a seção 5.3 apresenta as redes de relacionamento e suas principais características; a seção 5.4 apresenta alguns algoritmos para gestão de confiança e reputação encontrados na literatura; por último, a seção 5.5 conclui o capítulo.

5.1 Confiança hierárquica

A confiança hierárquica é definida como a confiança que um indivíduo possui em seus **pais**, **parentes** e vice-versa. A confiança hierárquica é de extrema importância na formação do indivíduo, pois a personalidade de uma pessoa está relacionada diretamente com sua educação familiar, ou seja, uma pessoa é fortemente influenciada a aprender o que seus pais e parentes ensinam [Car61].

A confiança pode ser representada por uma árvore, onde os nós representam indivíduos participantes (pais/parentes) e as arestas representam o **grau de parentesco** de cada indivíduo. O grau de parentesco é a distância de um nó em relação a outro nó; quanto menor a distância a ser percorrida para chegar ao destino, menor será seu grau de parentesco. A figura 5.8 apresenta um exemplo de árvore de confiança.

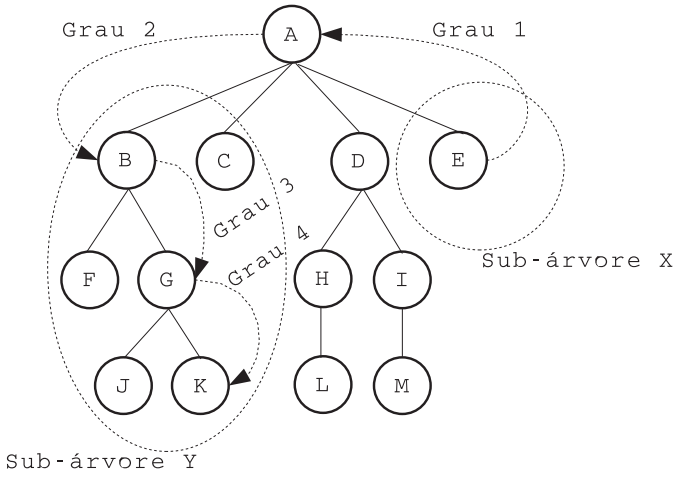

Figura 5.2: Árvore de confiança

Analisando a figura, é possível verificar que qualquer participante da árvore é capaz de chegar em um outro participante utilizando caminhamento entre os nós. Este caminhamento pode ser representado por um vetor onde os elementos são nós a serem visitados. A figura 5.3 apresenta o caminhamento do nó **E** (da figura anterior) até o nó **K**.

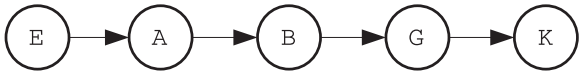

Figura 5.3: Vetor de confiança

Para o nó **E** chegar ao nó **K** este nó deverá visitar seu pai que é capaz de chegar a sub-árvore do nó destino. A medida que os nós vão sendo visitados, o grau de parentesco aumenta, diminuindo o nível de confiança, pois, mesmo que ambos os nós pertençam a mesma "família", não significa que um indivíduo da **sub-árvore X** seja confiável para um indivíduo da **sub-árvore Y**. Por este motivo, existe a necessidade de limitar o grau de parentesco confiável, tornando a árvore menor, conseqüentemente diminuindo o espaço de procura e a quantidade de parentes confiáveis.

Incluindo valores de confiança nas arestas de ligação da árvore, um nó será capaz de gerar um percentual de confiança com qualquer nó que esteja acessível e que não ultrapasse um limite de grau de parentesco pré-estabelecido. As arestas que fazem a ligação de cada nó são unidirecionais, ou seja, a confiança de **A** em **B** poderá ser diferente da confiança de **B** em **A**. A figura 5.4 apresenta a confiança entre os membros de uma árvore e as arestas que os conectam.

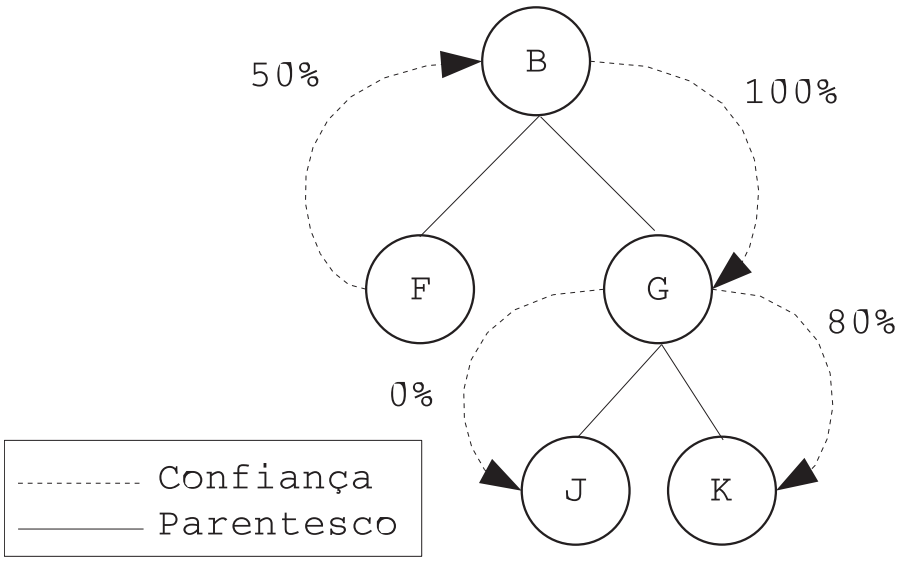

Figura 5.4: Confiança entre nós de uma árvore

Observa-se nesta figura que a confiança do nó **G** com o nó **J** é de 0%. Isto significa que qualquer nó abaixo da **sub-árvore J**, incluindo **J** estará inacessível. Os demais nós poderão gerar percentuais de confiança utilizando as propriedades de transitividade descritas em [VM02].

Outra forma de representar a confiança hierárquica é através de uma matriz **indivíduo *vs* indivíduo**, onde a intersecção dos eixos da matriz indicam o percentual de

confiança entre os indivíduos. Desta forma, é possível definir a confiança entre todos os nós da árvore anterior. A tabela 5.2 mostra a representação tabular da árvore de confiança da figura 5.4.

Tabela 5.2: Relações de confiança

	B	**F**	**G**	**J**	**K**
B	100%	-	100%	0%	80%
F	50%	100%	50%	0%	40%
G	-	-	100%	0%	80%
J	-	-	-	100%	-
K	-	-	-	-	100%

A confiança de um indivíduo nele mesmo será sempre de 100%; as intersecções com um traço (-) indicam que não existe uma relação de confiança entre os nós.

5.2 Grupos sociais

Grupo social é um conjunto de indivíduos onde suas atividades se relacionam mutuamente de forma sistemática para um determinado fim. Ou seja, um grupo deve ser concebido como um sistema cujas partes se inter-relacionam [Gra76].

Os grupos sociais são formados por indivíduos que tenham interesses comuns (computação, futebol, etc.) e estão em constante renovação (grupos dinâmicos) [Wei82]. Como por exemplo: grupo de amigos, equipe de trabalho, colegas de turma, time de futebol, entre outros. Quando a formação desses grupos for realizada de forma voluntária e préviamente planejada, o grupo deverá ser chamado de "Grupo Organizado" [Wei82].

A confiança em grupos sociais define quanto um grupo confia em um determinado indivíduo ou em outros grupos. Cada participante do grupo deverá expressar sua opinião para gerar um consenso entre todos eles. Este consenso é utilizado para aceitar ou rejeitar novos participantes, excluir participantes ativos[1], etc.

Integrantes de grupos sociais são capazes de compartilhar informações de interesse comum, fazendo com que essa informação seja propagada entre os demais integrantes. Os integrantes também utilizam a **confiança do grupo** para iniciar e manter relacionamento com outros indivíduos.

[1]Participantes ativos são aqueles que estão em constante comunicação com o grupo.

Um indivíduo pode possuir um ou mais grupos dependendo de suas necessidades/especialidades. Através deste indivíduo, integrantes de outros grupos são capazes de se comunicar e trocar informações de interesse comum, ou então unificar os grupos formando um grupo maior e mais forte.

5.2.1 Composição dos grupos sociais

A composição dos grupos sociais é definida a partir de um conjunto de fatores que influenciam diretamente na comunicação dos indivíduos. Estudos realizados em seres humanos por meio de inquéritos "sociométricos" evidenciaram a existência de laços de amizade, simpatia e antipatia que são capazes de reforçar ou destruir a coesão de um grupo. Esses laços podem ser definidos da seguinte forma [Gra76, Wei82]:

- **A** $\rightarrow$ **B**: A gosta de trabalhar com B;
- **B** $\rightarrow$ **A**: B gosta de trabalhar com A;
- **A** $\leftrightarrow$ **B**: A gosta de trabalhar com B e B gosta de trabalhar com A (Estado Ideal);
- $\leftarrow$**A B**$\rightarrow$: A e B se antipatizam;
- **A** $\rightarrow$ **B, B** $\rightarrow$ **C e C** $\rightarrow$ **A**: A gosta de B, B gosta de C e C gosta de A;
- **A** $\leftrightarrow$ **B, A** $\leftrightarrow$ **C e B** $\leftrightarrow$ **C**: Todos gostam de todos.

Nessas pesquisas foram manipuladas independentemente a semelhança e a simpatia de um estranho por outros indivíduos [Gra76, Wei82]. Os indivíduos foram induzidos a acreditar que determinadas pessoas possuíam semelhanças em comum mas que não gostavam deles e que outras pessoas não possuíam semelhanças em comum, mas que gostavam deles. Quando os indivíduos pesquisados classificaram a atração pelos desconhecidos, foi constatado que a semelhança não tinha efeito sobre as classificações. Isto significa que a semelhança pode ser uma condição desejável, mas é improvável que seja a razão de atração. A partir desses estudos gerou-se o conceito de necessidade e troca [Gra76]:

- A *teoria da necessidade* diz que somos atraídos por outros indivíduos cuja personalidade complementa a nossa, ou seja, o indivíduo submisso é atraído pelo indivíduo dominante.
- A *teoria da troca* trata o comportamento de um indivíduo como a mercadoria e o conhecimento como moeda. A partir daí, os indivíduos podem investir seus conhecimentos e receber vários comportamentos em troca.

Tomando como exemplo uma reunião: as pessoas dispendem seu tempo falando com um certo número de pessoas diferentes, contudo, elas não podem falar com todas ao mesmo tempo. Por isso é necessário uma seleção. A teoria da troca parte do princípio de que a finalidade de um indivíduo é maximizar os resultados e ter um dia produtivo. Nessa teoria são ponderadas todas as várias recompensas contra os custos. Ao final do dia a alternativa escolhida deverá ser a que produza o melhor resultado calculado [Gra76].

Os integrantes dos grupos são separados por cargos e funções específicas. Esses cargos/funções são definidas de acordo com os objetivos do grupo, o seu tamanho e as especialidades de cada indivíduo. Como o objetivo deste estudo é definir de forma genérica os grupos sociais, serão discutidos apenas dois tipos de integrantes: os líderes e os participantes.

Líderes

O líder é uma pessoa no grupo a qual foi atribuída, formal ou informalmente, uma posição de autoridade para dirigir, definir e aprovar novas regras para o grupo e tomar decisões críticas no grupo, tal como interferir em disputas de interesses entre outros participantes.

Os líderes podem ser definidos de duas formas: consenso global (por exemplo um Presidente) ou imposição (por exemplo um Imperador). Este cargo é desejado por todos os integrantes do grupo caso exista um descontentamento do grupo como um todo, ou de subgrupos, novos candidatos poderão reivindicar o cargo (através de votação ou imposição), havendo assim uma disputa entre o líder atual e um ou mais candidatos [RB56].

Participantes de um grupo

Como o próprio nome diz, participantes de um grupo são indivíduos que fazem parte de um grupo. A comunicação entre os diversos participantes, visando o interesse comum e a confiança que um indivíduo tem com o outro formam os grupos sociais.

Exclusão de integrantes

A exclusão de integrantes define o modo com que um participante de um determinado grupo social deixará de fazer parte do mesmo, sendo de extrema importância nos grupos sociais, pois mantém a comunicação dos integrantes sempre ativa e tenta diminuir as discordâncias e intrigas dentro do grupo. Existem basicamente duas formas de um indivíduo ser excluído de um grupo social:

- voluntária: onde o indivíduo deixa de se comunicar com integrantes do grupo du-

rante um determinado tempo (participante inativo);

- imposição: onde o grupo decide expulsar um integrante do grupo. Esta exclusão ocorre geralmente porque o indivíduo não se adequou ao grupo, não corresponde às expectativas ou por gerar discordância e descontentamento dos demais integrantes.

A exclusão por imposição deverá ser feita utilizando o consenso do grupo. Quando um indivíduo desconfiar de outro, ele deverá convencer a maioria dos membros para que a pessoa "não confiável" seja excluída do grupo. O problema neste modelo de exclusão é que indivíduos *mal-intencionados* são capazes de excluir indivíduos confiáveis, fazendo com que os demais participantes vejam ele *como não-confiável*.

Representação gráfica

Os grupos sociais podem ser representados a partir de um grafo (também chamado de "Átomo Social" [Wei82]), onde os nós são os integrantes e as arestas são as ligações entre os indivíduos. A figura 5.5 apresenta um exemplo de um grupo social representado por um grafo.

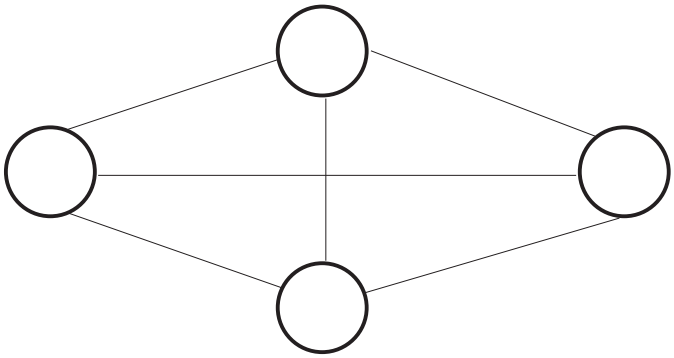

Figura 5.5: Átomo Social [Wei82]

Neste figura, todos os integrantes do grupo possuem relacionamento entre si. Isto é uma relação desejável, mas não necessária, visto que quanto maior for o grupo menor a chance de todos os integrantes do grupo se conhecerem. Contudo, utilizando o relacionamento de outros integrantes do grupo é possível chegar a qualquer outro participante do grupo (ver seção 5.3).

5.2.2 Relação entre grupos

Grupos com os mesmos interesses ou com participantes em comum podem se fundir formando um único grupo maior e mais forte. Quando um determinado grupo começa a se

expandir demais surgem descontentamentos que podem ocasionar rebeliões. Para conter esses problemas os grupos dividem-se novamente só que desta vez serão formados sub-grupos ao invés de grupos independentes, como por exemplo Países, Estados, Municípios, etc. Esses sub-grupos são comandados por líderes que devem respeitar a cadeia hierárquica (confiança hierárquica), vista na figura 5.8.

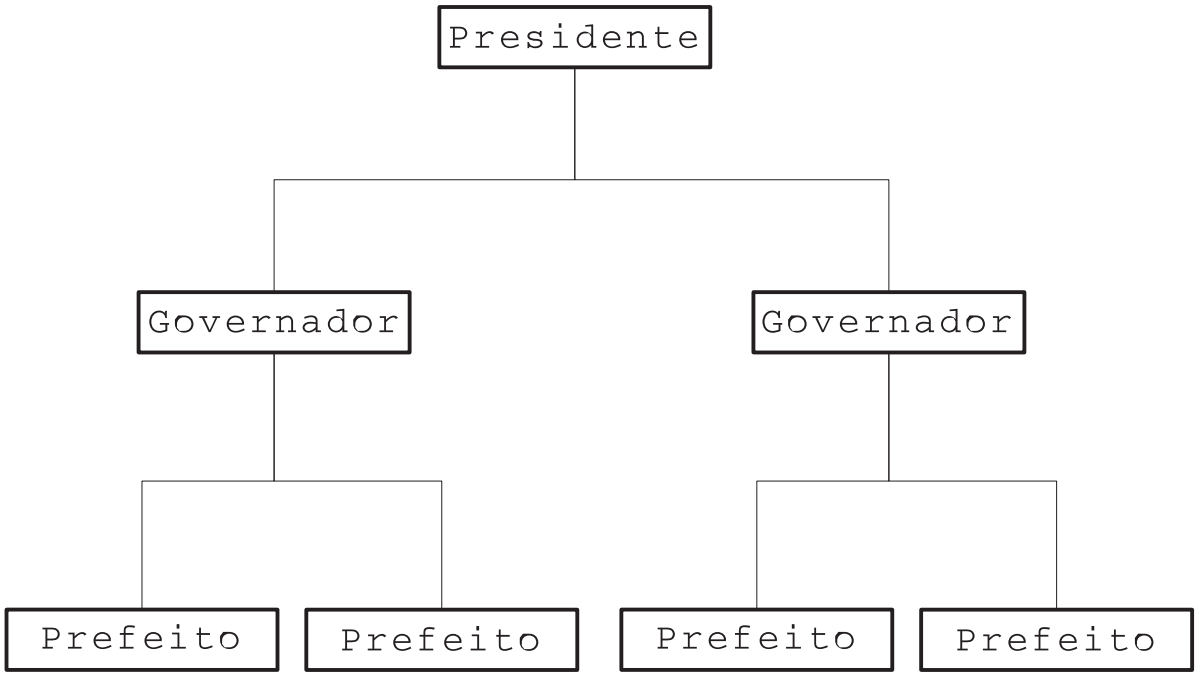

Figura 5.6: Um exemplo de árvore hierárquica

Nesta figura pode ser vista a representação hierárquica do pode executivo em uma democracia, onde o *Presidente* é o posto mais alto, seguido por *Governador* e *Prefeito*. A relação entre diversos grupos pode ser representada através de um grafo, onde o relacionamento entre os grupos se dá através de indivíduos comuns. A figura 5.7 apresenta a representação do relacionamento de grupos sociais.

Nesta figura são mostrados três grupos onde participantes possuem uma relação de confiança. O **Grupo 1** está interligado com o **Grupo 2** através de um indivíduo comum. Este indivíduo será o elo entre os dois grupos. O **Grupo 3** não possui ligações com os demais grupos, contudo, novas conexões são possíveis, basta existir interesse entre um ou mais indivíduos de grupos diferentes.

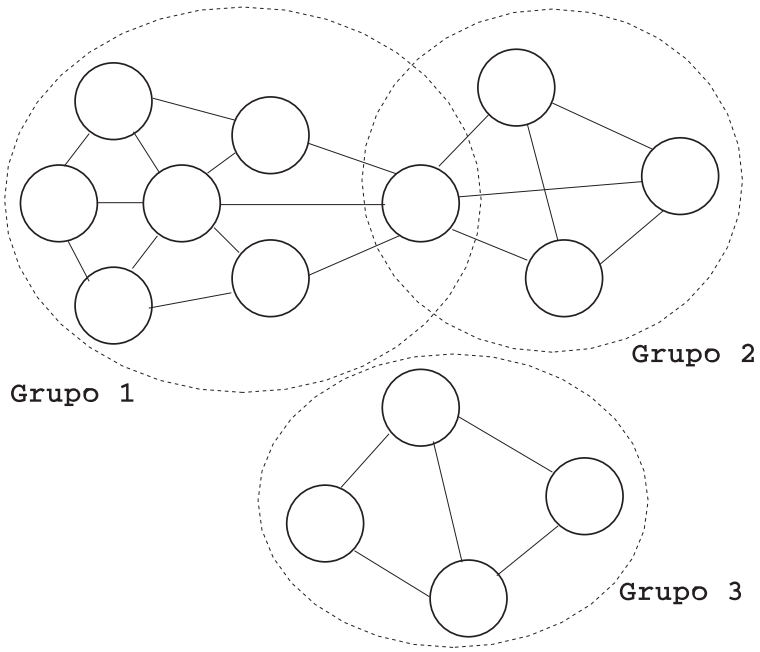

Figura 5.7: Relacionamento entre grupos sociais

5.3 Redes de relacionamento

A palavra relacionamento pode ser definida como um padrão de interação entre dois indivíduos baseado em percepções recíprocas. Já as redes de relacionamento são todos os relacionamentos "tecidos" por um indivíduo, ou seja, o convívio de um determinado indivíduo com pessoas que este conhece ou conheceu [WOB76]. Essa teia pode ser formada também por conhecidos de conhecidos e assim por diante; a partir dela é possível caminhar pelos nós conhecidos até chegar a um objetivo final.

As redes de relacionamento são utilizadas pelas pessoas para interagir com outras pessoas ou então chegar até um indivíduo qualquer. Por exemplo: Maria conhece João e não conhece Antônio; João conhece Antônio. Para Maria chegar a Antônio ela deverá utilizar a sua rede de relacionamento que é capaz de chegar até João. Utilizando João como ponte, Maria será capaz de conhecer Antônio. Assim como Maria chegou até Antônio, ela poderia chegar a qualquer pessoa que fizesse parte da sua rede de relacionamento.

Cada indivíduo possui sua própria rede de relacionamento. Essas redes estão interligadas através de nós que fazem parte de redes de relacionamento comuns. Os dados de uma rede de relacionamento são representados em uma matriz, onde a linhas da podem ser casos, assuntos ou observações. As colunas são informações que descrevem o relacionamento entre os indivíduos. Como por exemplo:

Tabela 5.3: Matriz de atributos

	Sexo	Idade	Interesse
João	Masculino	23	Esporte
Maria	Feminino	21	Computação
Antônio	Masculino	35	Esporte

Todos os dados de uma rede de relacionamento estão focados em cada indivíduo e em suas relações com outros indivíduos de sua rede. Por este motivo os indivíduos não podem ser observados independentemente. Isto significa que quando um indivíduo for escolhido para verificar suas preferências/afinidades, todos os outros indivíduos que fazem parte de sua rede de relacionamento também deverão ser analisados, fazendo com que esses métodos utilizem o consenso de uma rede ao invés de observações simples.

As redes poderão ser analisadas de uma forma "micro" ou "macro", isto é, como uma determinada rede está ligada a outras redes, é possível analisar um estado que abrange uma rede grande/global (macro) ou uma rede pequena/local (micro). A abordagem a ser escolhida depende do tipo de informação que queremos coletar. A seguir são apresentados os modelos e ligações de redes, conforme descrito em [Han98].

- **Método de redes completo:** método que procura informações em cada uma das conexões de um indivíduo de uma rede. Esse método cria um consenso entre todas as ligações de uma determinada população. Dependendo do tamanho da rede ele se torna computacionalmente inviável, pois é impossível visitar todos os indivíduos de uma rede em um curto espaço de tempo. Como por exemplo: um indivíduo **A** pertencente a um grupo de amigos composto por **A**, **B**, **C**, **D** e **E** quer saber qual é a cor de olhos predominante em seu grupo de amigos. Para isto, ele deverá perguntar a todos os integrantes do grupo e gerar a média de cada uma das cores.

- **Método *Snowball*:** inicia com um indivíduo ou um conjunto de indivíduos. A partir daí, cada indivíduo "visitado" informa os participantes que fazem parte da sua rede. Este processo continua até que não existam mais indivíduos a serem visitados ou através de uma condição de parada. O método *Snowball* não garante que serão localizados todos os indivíduos conectados a uma determinada rede. Como por exemplo: um indivíduo **A** pertencente a um grupo de amigos composto por **A**, **B**, **C**, **D** e **E** quer saber quem da família de cada grupo até a quarta geração possui olhos azuis. Todos os integrantes deverão fazer a mesma pergunta recursivamente diminuindo uma geração. Cada indivíduo deverá retornar o resultado da pergunta para o indivíduo **A**.

- **Redes Ego-Cêntricas (com conexões alternadas):** seleciona um conjunto de nós iniciais (egos) e identifica todas as conexões de cada nó selecionado. Em seguida determina quais nós estão conectados entre si. Por fim, ele visita cada nó selecionado que possuem conexões comuns e caminha entre esses nós (similar ao *Snowball*). Os dados coletados pelo método de redes Ego-Cêntricas podem ser definidos como conjuntos de micro-redes, por isso, propriedades como distância (caminho a ser percorrido para chegar em um determinado nó), centralidade (identifica se um nó é um nó que possui muitas conexões) e tipos de equivalência posicional (posição de um determinado nó é equivalente a de outro nó) não podem ser obtidos por esta abordagem. Como por exemplo: um indivíduo **A** quer determinar um conjunto de pessoas que tem amigos em comuns a ele em sua escola.

- **Redes Ego-Cêntricas (apenas ego):** está focada no indivíduo. Coleta informações nas conexões de indivíduos conectados para cada nó inicial (ego). Utilizando as redes ego-cêntricas é possível ter a visão da rede "local" e também da vizinhança de cada nó. Esta abordagem permite que seja possível entender como as redes de relacionamento afetam um nó individualmente, como também dá uma visão geral da rede como um todo. Como por exemplo: um indivíduo **A** quer identificar cada uma das conexões com laços de amizade como "parentes", "amigo de trabalho", "membro da mesma igreja", etc. **A** pode gerar então uma visão geral de posições sociais (ao invés de redes de indivíduos), no qual os indivíduos estão inseridos.

5.3.1 Escalas de medidas

As redes de relacionamento permitem que os dados coletados possam ser mensurados utilizando diferentes níveis (escalas) de medida. Essas escalas são importantes pois através delas é possível limitar um conjunto de questões a serem examinadas pelo "pesquisador". Além disso, é possível utilizar na análise dos dados coletados diferentes tipos de escala de acordo com as propriedades matemáticas dos dados (possibilidade de se utilizar algoritmos de análise diferentes). A seguir são apresentadas as principais escalas de medida para dados de redes sociais, conforme descrito em [Han98]:

- **Medidas binárias de relações**: possuem apenas dois níveis de medida (0 ou 1). Como por exemplo: Definir se um determinado indivíduo gosta de um outro indivíduo qualquer ou não. A resposta deverá ser "sim" ou "não". Muitas teorias de grafos foram desenvolvidas utilizando medidas binárias;

- **Medidas de múltipla-categoria nominal**: possuem vários níveis de medida. Como por exemplo: Descrever a relação de um determinado indivíduo de acordo com a seguinte lista: *amigo*, *namorado* ou *relacionamento de trabalho*. A resposta deverá ser uma das listadas anteriormente. Este tipo de escala é qualitativa; ao contrário das medidas binárias, a medida qualitativa permite múltiplas escolhas;

- **Medidas ordinais**: visam medir um determinado indivíduo de acordo com o grupo. Como por exemplo: Descobrir quantas pessoas gostam e quantas pessoas não gostam de um determinado indivíduo. As respostas terão a forma "0 ou mais gostam" ou "0 ou mais não gostam". Esta abordagem mede os diferentes aspectos quantitativos de uma relação;

- **Medidas ordinais por rank completo**: visam definir um ranking para um conjunto de indivíduos de acordo com o grupo. Como por exemplo: Pedir para um determinado indivíduo escrever em uma lista de nomes o número "1" ao lado da pessoa que mais gosta; o número "2" ao lado da segunda pessoa que mais gosta e assim por diante. Este tipo de escala deverá resultar um "rank completo ordenado". Tal escala reflete em diferentes graus de intensidade, mas não necessariamente em diferenças de igualdade, isto é, a diferença entre a primeira e a segunda escolha não é necessariamente a mesma diferença entre a segunda e a terceira escolha. Contudo, cada relação é única (primeiro, segundo, terceiro, etc).

- **Medidas escalares**: definem uma escala de medida. Como por exemplo: Definir quanto um indivíduo gosta de outro indivíduo. A resposta deverá ser de dada de acordo com a medida de escala convencionada pelos indivíduos.

5.4 Algoritmos de confiança/desconfiança e reputação

Confiança é a perspectiva subjetiva que um indivíduo tem sobre outros a partir dos históricos de seus encontros, **desconfiança** é a dúvida sobre a honestidade de um indivíduo e **reputação** é a percepção que os indivíduos criam uns dos outros a partir de suas ações passadas, de acordo com as intenções e normas, onde normas são heurísticas definidas a partir de pressupostos morais que os indivíduos seguem no decorrer de suas vidas [MMH02, Ost98].

Os algoritmos de confiança e reputação são algoritmos desenvolvidos para tentar "simular" a confiança de seres humanos em redes de relacionamento. Todos utilizam métodos de propagação das informações de integrantes de uma rede centralizada para

gerar um consenso global da confiança e reputação de cada um dos integrantes da rede. Nesta seção serão apresentados os algoritmos de confiança e reputação propostos por [GKRT04] e [KSGM03] respectivamente.

5.4.1 Propagação de confiança e desconfiança

O estudo de propagação de confiança e desconfiança proposto por [GKRT04], cria uma *framework* que faz uso das redes de relacionamento para predizer com um certo grau de precisão a confiança entre dois indivíduos. Segundo os autores, este trabalho foi o primeiro a incorporar tanto a confiança quanto a desconfiança de indivíduos em um sistema computacional de propagação de confiança.

Neste estudo, os autores criaram um universo de "n" indivíduos capazes de expressar níveis de confiança e desconfiança para qualquer outro indivíduo da rede. Esses valores de confiança e desconfiança foram divididos em duas matrizes: T (confiança) e D (desconfiança), onde t_{ij} representa a confiança de um usuário i em j (esses valores podem ser 0 ou 1) e d_{ij} representa a desconfiança de um usuário i em j. Além disso, foi proposta uma matriz B que representa um conjunto de crenças de um indivíduo em outros, onde b_{ij} pode ser tanto a confiança de i em j quanto a combinação das confianças de i em j. A informação é propagada utilizando métodos de propagação atômica, onde em uma única etapa é possível indicar que um indivíduo i confia em k. Por exemplo, se i confia em j e j confia em k, então j deverá informar a i a confiança dele em k.

Propagação atômica

A propagação atômica define o modelo de comunicação de dois indivíduos utilizando propriedades transitivas para definir a confiança em um terceiro indivíduo. Existem quatro tipo de propagação atômica:

- **Propagação direta - B**: se $B_{ij} = 1$ (i confia em j) e $B_{jk} = 1$ (j confia em k), então a propagação atômica deve possibilitar que i confie em k;
- **Co-citação -** $B^T B$: suponha que i_1 confia em j_1 e em j_2 e que i_2 confia em j_2. Utilizando a co-citação é possível concluir que i_2 poderá confiar em j_1. A seqüência $B^T B$ pode ser vista como uma etapa de propagação "voltando - avançando", onde i_2 confia em j_2 que é capaz de voltar para i_1 para então chegar em j_1. A figura a seguir apresenta o grafo que representa esta propagação através de co-citação;

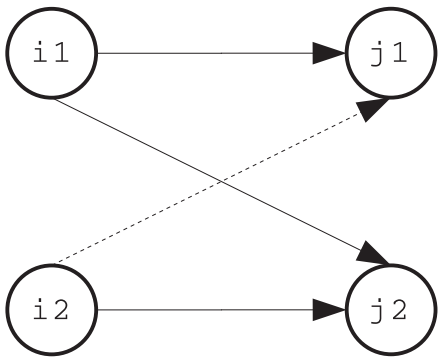

Figura 5.8: Exemplo de co-citação

- **Confiança transposta -** B^T: se i confia em j então j deverá desenvolver algum nível de confiança em i;
- **Acoplamento de confiança -** BB^T: a confiança de i em j é propagada para k, porque j e k confiam em pessoas comuns.

Supondo que $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ seja o vetor que representa os pesos das quatro combinações de propagação, então é possível definir todas as propagações atômicas em uma única matriz de crença $C_{B,\alpha}$, com um vetor de peso α. Esta matriz de crença pode ser expressa da seguinte forma:

$$C_{B,\alpha} = \alpha_1 B + \alpha_2 B^T B + \alpha_3 B^T + \alpha_4 B B^T$$

Esta matriz será utilizada na propagação de crença e descrença descrita a seguir.

Propagação de confiança e desconfiança

O objetivo da propagação de confiança e desconfiança é gerar uma matriz F que possa indicar a confiança/desconfiança entre dois nós distintos em uma rede. O resultado final será gerado a partir das seguintes etapas:

1. Gerar a matriz F seguindo as etapas a seguir:

   (a) Propagação de desconfiança: Supondo que $C_{B,\alpha}$ seja uma matriz onde a ij-ésima entrada descreva a crença de i para j através da propagação atômica; se a entrada for 0, então não é possível concluir como i enxerga j. Supondo que k seja um inteiro positivo e supondo que $P^{(k)}$ seja a matriz onde ij-ésima entrada representa a propagação de i para j após a propagação atômica de k, será possível chegar a uma nova matriz de crença B após k passos. Isto significa

que a repetição de propagação de confiança pode ser representada a partir de uma operação $P^{(k)}$. São propostos três modelos para definir B e $P^{(k)}$ para a propagação de confiança e desconfiança, a partir de uma matriz de confiança T ou desconfiança T:

Apenas confiança: ignora a desconfiança completamente. Neste caso obtém-se que $B = T$ e $P^{(k)} = C_{B,\alpha}$.

Um-passo de desconfiança: assume-se que quando um usuário desconfia de alguém, ele deverá descontar todos os julgamentos feitos por aquela pessoa. Ou seja, a desconfiança é propagada em apenas um passo, enquanto a confiança pode ser propagada repetidamente. Neste caso obtém-se que $B = T$ e $P^{(k)} = C_{B,\alpha} * (T - D)$.

Desconfiança propagada: assume-se que tanto a confiança quanto a desconfiança deverão ser propagadas juntas. Neste caso obtém-se que $B = T - D$ e $P^{(k)} = C_{B,\alpha}$.

(b) Propagação interativa: Depois de definido o algoritmo de propagação de desconfiança, é possível gerar a matriz F que representa as conclusões que qualquer indivíduo deverá ter sobre um outro, utilizando dois algoritmos:

EIG (*Eigenvalue Propagation*): sendo **K** o número de iterações, então a matriz final será gerada através de

$$F = P^{(k)}$$

WLC (*Weighted Linear Combination*): supondo que γ seja uma constante e que K seja um valor apropriado escolhido, então a matriz final será gerada através de

$$F = \sum_{k=1}^{K} \gamma^k * P^{(k)}$$

2. Coletar os resultados interpretando os valores de F para gerar a confiança e desconfiança. Esses resultados deverão informar se um indivíduo pode ou não confiar em outro. Este procedimento deverá utilizar um dos algoritmos a seguir:

(a) Arredondamento Global: tenta alinhar a taxa dos valores de confiança e desconfiança em F para uma entrada qualquer. Considere o vetor F_i, onde i deverá confiar em j se e somente se F_{ij} estiver acima da fração de uma entrada x do vetor F_i. Como por exemplo:

Sendo $F_i = \begin{pmatrix} 1 \\ 2 \\ 2 \\ 3 \\ 4 \end{pmatrix}$ sendo a i-ésima coluna da matriz $F = \begin{pmatrix} & 3 & 1 & 3 & \\ & 7 & 2 & 5 & \\ ... & 8 & 2 & 2 & ... \\ & 5 & 3 & 0 & \\ & 1 & 4 & 4 & \end{pmatrix}$

o ponto inicial requisitado **x** igual a 3 e F_{ij} sendo o segundo elemento de F_i, é possível concluir que F_{ij} está acima da fração x, conseqüentemente i confia em j.

(b) Arredondamento Local: similar ao arredondamento local. Contudo, o ponto inicial x deve ser escolhido através de julgamentos de pontos confiáveis vs desconfiáveis feitos por i.

(c) Arredondamento da Maioria: utiliza um conjunto de indivíduos para gerar a confiança final a partir do arredondamento local. Ou seja, ele utiliza a média de todas as suas confianças geradas a partir do arredondamento local para gerar a confiança final de um indivíduo.

5.4.2 EigenTrust: algoritmo de gerenciamento de reputação em redes peer-to-peer

O EigenTrust [KSGM03] é um algoritmo de geração de reputação de nós desenvolvido para diminuir o número de downloads de arquivos não autênticos em redes P2P (*Peer-To-Peer*). O EigenTrust define um método distribuído e seguro para computar valores de confiança global baseados no poder de interação. Os valores de reputação global são calculados através de valores de confiança local que cada nó da rede define para outros nós. Este algoritmo define alguns passos que devem ser seguidos para seu funcionamento.

O primeiro passo do Eigentrust é normalizar os valores locais de confiança em uma forma simples de ser interpretada probabilisticamente, do contrário, nós maliciosos poderão atribuir arbitrariamente valores de confiança alto para outros nós maliciosos e valores baixos para nós bons. Esta normalização deverá disponibilizar um valor único de confiança de i para j. A fórmula de normalização é:

$$c_{ij} = \frac{max(s_{ij}, 0)}{\sum_j max(s_{ij}, 0)}$$

O segundo passo é agregar todos os valores de confiança local normalizados de

cada nó da rede P2P. Esta agregação será realizada por um nó n_i que deseja criar uma confiança global em um nó n_j. O n_i fará requisições de confiança a outros nós perguntando qual a confiança deles em n_j. O nó n_i deverá utilizar a seguinte fórmula:

$$t_{ik} = \sum_j c_{ij} c_{jk}$$

onde t_{ik} representa a confiança que o nó n_i deposita no nó n_k baseado nas perguntas feitas a um grupo de confiança. Outra forma de se escrever esta relação é através de notação de matriz. Se C define a matriz $|c_{ij}|$ e $\overrightarrow{T_i}$ é o vetor que contém os valores de t_{ik}. Então:

$$\overrightarrow{t_i} = C^T \overrightarrow{c_i}$$

Mas esta visão ainda não considera todos os relacionamentos de outros nós da rede. Para n_i criar uma visão geral da rede, é proposta uma fórmula que faz com que n_i seja capaz de perguntar para os amigos de seus amigos sobre a confiança que eles têm em um determinado nó:

$$\overrightarrow{t_i} = (C^T)^n \overrightarrow{c_i}$$

onde n indica a quantidade de amigos de amigos a serem pesquisados. Ao final de n interações o nó n_i deverá possuir uma visão completa sobre a rede e sobre o vetor $\overrightarrow{t}$. A distribuição estacionária da cadeia de Markov definida pela matriz C de confiança local normalizada é a confiança global de um nó na rede. Esta confiança global é definida a partir do vetor $\overrightarrow{t}$.

O terceiro passo é definir como o vetor $\overrightarrow{t}$ será calculado:

EigenTrust Básico

O EigenTrust Básico ignora a natureza distribuída das redes P2P assumindo que existe um servidor central que conhece todos os valores c_{ij} e executa toda a computação. Utilizando este algoritmo deseja-se computar apenas $\overrightarrow{t} = (C^T)^n \overrightarrow{e}$, para n=grande, onde $\overrightarrow{e}$ é um vetor m representando uma probabilidade de distribuição uniforme sobre todos

os nós $e_i = 1/m$. O algoritmo pode ser visto a seguir:

$\overrightarrow{t}^{(0)} = \overrightarrow{e}$;
repeat
$\overrightarrow{t}^{(k+1)} = C^T \overrightarrow{t}^{(k)}$;
$\delta = \mid \overrightarrow{t}^{(k+1)} - \overrightarrow{t}^{(k)} \mid$;
until $\delta < \varepsilon$;

Existem três melhorias a serem realizadas neste algoritmo visando uma maior confiabilidade de cada nó.

- **nós pré-confiáveis**: alguns nós da rede devem ser naturalmente confiáveis, tais como nós iniciais de uma rede P2P;
- **nós inativos**: se um nó n_i não executa mais downloads ou se este nó atribui valores zero a todos os outros nós, ele deverá ser descartado;
- **maliciosos coletivos**: grupos de nós que disponibilizam informações não confiáveis, fazendo com que se torne difícil confiar em outros nós da rede. Estes grupos devem ser identificados e ignorados.

EigenTrust distribuído

Para prover as melhorias apontadas anteriormente, os autores propuseram o EigenTrust distribuído, onde cada nó da rede coopera para computar e armazenar o vetor de confiança global. Cada nó pode computar sua própria confiança global a partir da seguinte fórmula:

$$\overrightarrow{t}^{(k+1)} = (1-\alpha)(c_{1i}t_1^{(k)} + ... + c_{ni}t_n^{(k)}) + \alpha p_i$$

onde $p_i = 1/|P|$ se i pertence a P (conjunto de nós pré-confiáveis); $p_i = 0$ do contrário. O algoritmo a seguir apresenta o EigenTrust distribuído.

foreach *peer* i **do**
Query all peers j **end** A_i $fort_j^{(0)} = p_j$;
repeat
Compute $\overrightarrow{t}^{(k+1)} = (1-\alpha)(c_{1i}t_1^{(k)} + ... + c_{ni}t_n^{(k)}) + \alpha p_i$
Send $c_{ij}t_i^{(k+1)}$ *to all peers* $j \in B_i$;
Compute $\delta = \mid t_i^{(k+1)} - t_i^{(k)} \mid$;

Wait for all peersj $\in A_i$ *to return* $c_{ij}t_i^{(k+1)}$;
until $\delta < \varepsilon$;
end

Neste algoritmo cada nó n_i computa e reporta seu próprio valor de confiança t_i. Nós maliciosos podem facilmente reportar valores de confiança falsos, manipulando o sistema.

EigenTrust seguro

Para resolver os problemas de usuários maliciosos, os autores propõem utilizar mais de um nó para computar os valores de confiança. Esses nós são chamados de gerenciadores de escore. Cada nó da rede deve possuir um número M de gerenciadores de escore. A posição desses gerenciadores é oculta utilizando uma tabela hash distribuída DHT (*Distributed Hash Table*). Por fim, o seguinte algoritmo foi proposto:

foreach *peer* i **do**
Submit local trust values $\overrightarrow{c_i}$ *to all score managers at positions*
$h_m(pos_i), m = 1...M-1$;
Collect local trust values $\overrightarrow{c_d}$ *and*
sets of acquaintances B_i^d *of daughter peers* $d \in D_i$;
Submit daughter d's local trust values c_{dj} *to*
score managers $h_m(pos_d), m = 1...M-1, \forall\ j \in B_d^i$;
Collect acquaintances A_i^d *of daughter peers*;
foreach *daugter peer* $d \in D_i$ **do**
Query all peers $j \in A_d^i$ *for* $c_{jd}p_j$;
repeat
Compute $t_d^{(k+1)} = (1-\alpha)(c_{1d}t_i^{(k)} + c_{2d}t_2^{(k)} + ... + c_{nd}t_n^{(k)}) + \alpha p_d$;
Send $c_{dj}t_d^{(k+1)}$ *to all peers* $j \in B_d^i$;
Wait for all peers $j \in A_d^i$ *to return* $c_{jd}t_j^{(k+1)}$;
until $\mid t_d^{(k+1)} - t_d^{(k)} \mid < \varepsilon$;
end
end

Neste algoritmo são definidos gerenciadores de escore que são responsáveis por disponibilizar informações dos nós que eles gerenciam. Todos os nós que não forem gerenciadores de escore (também chamados de nós *daughter*) deverão definir um ou mais gerenciadores de escore padrão, que serão responsáveis por disponibilizar as informações deste

nó para a rede. Cada gerenciador de escore deverá armazenar informações sobre todos os nós que fazem uso dele e armazenar informações sobre os pontos que baixam arquivos dos nós *daughter*.

Utilizando esta abordagem, garante-se que apenas nós gerenciadores de escore poderão propagar a confiança pela rede, fazendo com que nós maliciosos não sejam capazes de votar.

5.5 Conclusão

Neste capítulo foram abordadas a confiança hierárquica (fundamental para o indivíduo), os grupos sociais (base de todas as sociedades) e redes de relacionamento (relacionamento tecido no decorrer da vida de um indivíduo). Foi apresentado ainda o algoritmo de propagação de confiança e desconfiança [GKRT04] e o algoritmo do EigenTrust [KSGM03] que tem como objetivo gerar um sistema de reputação para nós (*peers*) em redes P2P.

O modelo de confiança dos seres humanos pode ser muito útil em aplicações de redes, visto que o ambiente, tanto humano quanto computacional possuem características parecidas. No próximo capítulo será apresentada a proposta do trabalho, que procura utilizar alguns métodos apontados neste capítulo para gerar um sistema de confiança distribuída em sistemas de e-mail.

Capítulo 6

Um sistema de confiança entre servidores de e-mail

A principal deficiência no que tange os problemas apresentados no capítulo 3, reside na falta de um mecanismo robusto de autenticação e reputação de domínios de servidores de e-mail. *Spammers* e fraudadores utilizam técnicas de envio de e-mail em massa para propagar vírus e mensagens indesejáveis através da rede, tentando enganar usuários, persuadindo-os a fornecer informações sensíveis (senha de banco, número de cartão de crédito, etc) que possam ser utilizadas em benefício do próprio fraudador.

Técnicas de detecção de spam, detecção de vírus, *black lists*, *white lists*, entre outras, estão sendo empregadas na contenção de e-mails maliciosos, contudo elas são estritamente locais, o que dificulta a propagação de informações de *spammers* pela rede. Outra maneira de se conseguir informações sobre *spammers* são as RBLs, listas globais que armazenam informações (tais como domínio e IP) sobre servidores maliciosos. Essas listas são centralizadas em um único ponto, possibilitando ataques do tipo DoS e DDoS. Além disso, essas listas devem ser constantemente atualizadas, tornando-as difíceis de serem mantidas.

Técnicas de autenticação de servidores de e-mail e remetente estão sendo estudadas para limitar a propagação de mensagens enviadas por fraudadores (diminuindo sua atuação na rede), partindo do princípio que os *spammers* fazem uso de programas que geram remetentes de forma aleatória. Essas abordagens continuam falhas, pois mesmo que um domínio tenha sido autenticado, nada impede que seus usuários continuem enviando/propagando informações maliciosas pela rede.

Este capítulo apresenta a proposta do trabalho e está dividido da seguinte forma: a seção 6.1 apresenta uma visão geral sobre a proposta do trabalho; a seção 6.2 apresenta uma visão da arquitetura do sistema; a seção 6.3 apresenta o modelo de gerência de confiança; a seção 6.4 apresenta o modelo de armazenamento do sistema; a seção 6.5 apresenta

o modelo de propagação de confiança entre servidores de e-mail; a seção 6.6 apresenta os benefícios e limitações da proposta; a seção 6.7 apresenta os trabalhos correlatos a esta proposta; por fim, a seção 6.8 conclui o capítulo.

6.1 Proposta

A proposta deste trabalho é definir uma arquitetura que permita criar e gerenciar listas de servidores confiáveis e não confiáveis de forma dinâmica e descentralizada, utilizando técnicas consolidadas de classificação de e-mail e autenticação de servidores, em conjunto com técnicas de redes de confiança abordadas no capítulo 5 para propagação e gerência de confiança entre os servidores de e-mail.

A gerência de confiança deve ser utilizada para diminuir gradativamente a quantidade de mensagens a serem recebidas de servidores que propagam informações maliciosas, ou seja, quanto menor a confiança de um servidor receptor em um servidor emissor, menor será a quantidade de mensagens que o emissor poderá enviar ao receptor no decorrer de um determinado período de tempo.

A confiança de um servidor deve ser gerada a partir de cálculos de confiança e da comunicação de um MTA com os integrantes de seu grupo de confiança[1]. Esta confiança será verificada antes que uma mensagem seja recebida pelo MTA e será recalculada e propagada aos participantes do grupo do servidor receptor antes que a mensagem seja aceita. A partir das redes de relacionamento os servidores são capazes de trocar informações com outros servidores da rede, formando este modelo escalável, pois, qualquer servidor que faça parte de uma rede de relacionamento será indiretamente influenciado por outros integrantes desta rede. A figura 6.1 apresenta o modelo de recebimento de mensagens.

Quando o servidor **A** enviar uma mensagem ao servidor **B**, **B** utilizará a confiança que ele possui em **A**, gerada a partir de informações locais (vide seção 6.3.1) e do seu grupo de confiança para aceitar ou rejeitar a mensagem.

A confiança de cada servidor da rede deve ser propagada a outros servidores através das redes de relacionamento. A figura 6.2 apresenta o modelo de propagação de confiança.

Nesta figura os quadrados representam os servidores de e-mail, os círculos representam os grupos de confiança de cada servidor e as setas representam a propagação de confiança. Neste exemplo, **B** envia a confiança dele em outros servidores para **C** e **E**, que

[1]Conjunto de servidores considerados confiáveis pelo servidor.

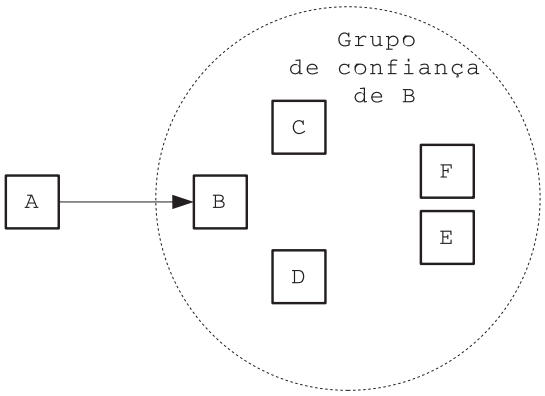

Figura 6.1: Modelo de recebimento de mensagens

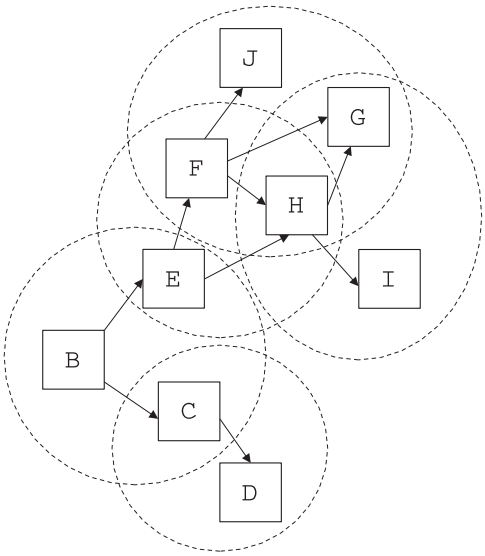

Figura 6.2: Modelo de propagação de confiança

enviam a confiança deles para o seu grupo de confiança e assim por diante. Todos os servidores que fizerem parte da rede de relacionamento de **B** serão indiretamente influenciados pela informação propagada por ele.

Esta arquitetura deverá disponibilizar um sistema de confiança que será capaz de gerar a confiança dos servidores nos receptores de e-mail, que serão capazes de aumentar ou diminuir esta confiança de acordo com as ações de cada integrante da rede de relacionamento. A partir desta confiança, os servidores de e-mail poderão limitar/aceitar uma quantidade de mensagens a serem recebidas, visando a diminuição do tráfego de mensagens indesejadas (que degradam os serviços de e-mail) pela rede.

6.2 Arquitetura do sistema

A arquitetura do sistema utiliza conceitos de redes de confiança, descritos no capítulo 5 em conjunto com ferramentas anti-spam, ferramentas anti-vírus e um modelo de autenticação de servidores de e-mail para disponibilizar um sistema de confiança entre MTAs. Nesta arquitetura são definidos um conjunto de algoritmos para cálculo, armazenamento e propagação de confiança de servidores de e-mail. O sistema de confiança funcionará de duas formas:

- deverá aprender "imitando" (através de compartilhamento de confiança) e seguindo regras de acordo com um conjunto de servidores que fazem parte do seu grupo de confiança.
- deverá aprender através de tentativa e erro, ou seja, todas as informações prejudiciais aos serviços de e-mail (spam, vírus, etc.) serão classificadas e manipuladas e servirão de base na perda ou ganho de confiança num emissor.

Para o funcionamento correto do modelo de confiança, faz-se necessária a inclusão de alguns módulos nos MTAs. A figura 6.3 ilustra a arquitetura proposta.

Nesta figura estão representados os componentes que deverão ser implementados/integrados nos MTAs que desejam prover tal modelo de confiança e o fluxo da arquitetura proposta. Os componentes da proposta são:

- **Servidor SMTP**: responsável pelo recebimento das mensagens; implementa o protocolo SMTP.
- **Autenticar emissor**: implementa um método de autenticação de domínio (ver capítulo 4).
- **Anti-spam**: filtro anti-spam responsável pela classificação das mensagens recebidas. O resultado deste filtro será utilizado pelo sistema de confiança.
- **Anti-vírus**: filtro anti-vírus de servidor que analisa se uma mensagem contém um vírus em seu corpo. O resultado deste filtro será utilizado pelo sistema de confiança.
- **Sistema de confiança**: analisa a confiança de um servidor de acordo com as ações deste servidor localmente e na rede (ver seção 6.3).

O fluxo do sistema deverá ser iniciado quando um emissor qualquer estabelecer uma comunicação SMTP com um servidor que implementa o sistema de confiança. As etapas

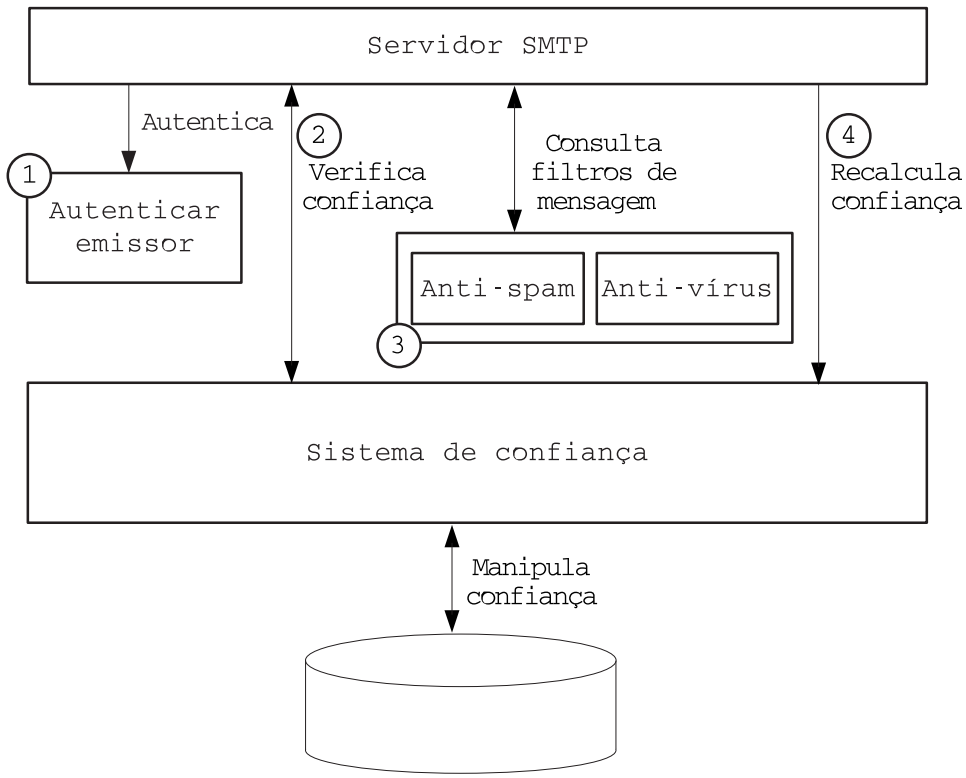

Figura 6.3: Arquitetura proposta

que serão executadas dentro do sistema de confiança serão detalhadas no decorrer deste capítulo. Quando a comunicação SMTP for estabelecida, o servidor de e-mail receptor deverá executar os seguintes passos para cada mensagem recebida:

1. Autenticar servidor. Se o servidor for corretamente autenticado, ir para o próximo passo, do contrário, não receber a mensagem;

2. Verificar se o servidor emissor pode enviar mensagem, consultando o sistema de confiança (deverá retornar sim ou não). Caso a resposta seja positiva o sistema recebe a mensagem, do contrário a conexão é encerrada;

3. Submeter e-mail recebido a diversos filtros, tais como anti-spam e anti-vírus. Esses filtros deverão retornar se o e-mail é legítimo ou malicioso;

4. Enviar resultado da consulta anterior ao sistema de confiança que atualiza a base de confiança;

5. Retornar ao passo 1 até que a conexão seja encerrada.

A arquitetura foi dividida em três módulos: gerência, armazenamento e propagação de confiança. A seguir serão apresentados cada um desses módulos.

6.3 Gerência de confiança

Os cálculos de confiança dos diversos servidores de e-mail em conjunto com regras de aceitação ou negação de e-mail formam o que será chamado de gerência de confiança. O objetivo deste módulo é definir um modelo justo de geração e manutenção de confiança em servidores de e-mail e criar uma forma de diminuir gradativamente a quantidade de mensagens a serem recebidas de servidores que propagam informações maliciosas, ou seja, quanto menor a confiança de um servidor receptor em um servidor emissor, menor será a quantidade de mensagens que o emissor poderá enviar ao receptor no decorrer de um dado período de tempo. Esta funcionalidade necessita de um modelo de medida de confiança que seja capaz de expressar o quanto um servidor confia em outro. Durante o estudo foi constatado que as medidas que mais se adequam à proposta (dentre as pesquisadas, conforme apresentado na seção 5.3.1) são:

- **binária**: um servidor confia ou não confia em outro servidor;
- **escalar**: define uma medida de escala para indicar a confiança de um servidor em outro.

Foi optado pela medida escalar pois a confiança de cada servidor emissor poderá ser tratada de 0 a 100%, possibilitando o cálculo da quantidade máxima de mensagens que cada servidor emissor poderá enviar para um servidor receptor durante determinado período. A fórmula a seguir define a quantidade máxima de mensagens que o receptor poderá receber de um determinado emissor (QMS):

$$QMS = QM * CS$$

onde, QM (*Quandidade de Mensagens*) é uma variável definida no sistema de confiança que indica a quantidade máxima de mensagens que um servidor pode receber de um emissor com 100% de confiança e CS (*Confiança do Servidor*) é a confiança que o receptor possui no emissor.

O QMS (*Quantidade de Mensagens Servidor*) é o número de mensagens que o servidor poderá receber de um emissor durante o dia. A medida que as mensagens vão sendo recebidas, um CMS (*Contador de Mensagens*), definido no servidor para cada emissor será acrescido e o *QMS* será recalculado. Quando o *CMS* for igual ao *QMS* a conexão será interrompida e o servidor emissor não poderá mais enviar mensagens naquele

dia. Todos os dias o *CMS* deverá ser reiniciado para que o servidor emissor possa receber mensagens do receptor novamente. Como por exemplo:

Um servidor receptor **A** possui a confiança em **B** igual a 30% e o QM de **A** é igual a 500. Depois do cálculo de QMS, o número máximo de mensagens diárias que o servidor **A** poderá receber de **B** é 150 mensagens.

Este modelo define uma forma justa de recebimento de mensagens, onde servidores mais confiáveis poderão enviar mais mensagens que servidores menos confiáveis. Definido o modo com que a confiança deverá implicar no envio/recebimento de mensagens, o próximo passo é definir os cálculos de confiança. Neste trabalho a medida de confiança foi dividida em três medidas: local, global e final.

- **Confiança local**: confiança gerada a partir de informações coletadas a partir de comunicações com um determinado servidor da rede;
- **Confiança global**: confiança gerada a partir de um conjunto de servidores confiáveis;
- **Confiança final**: média entre a confiança local e final.

6.3.1 Confiança local

A confiança local define o modo com que o servidor receptor **aumenta** ou **diminui** sua confiança "individual" em um determinado emissor utilizando o histórico de mensagens recebidas deste. O número de mensagens a serem recebidas (QR) que influenciarão no aumento ou na diminuição da confiança será calculado a partir da seguinte fórmula:

$$QR = \begin{cases} CS * MC & \text{se a mensagem for maliciosa} \\ (1 - CS) * MC & \text{caso contrário} \end{cases}$$

onde CS (*Confiança do Servidor*) é a confiança que o receptor possui no emissor e MC (*Mensagens de Confiança*) é uma variável definida no sistema de confiança que influenciará no resultado de QR da seguinte forma:

- quanto maior a confiança em um servidor, menor a quantidade necessária de mensagens legítimas para aumentar a confiança e maior a quantidade necessária de mensagens maliciosas para diminuir a confiança;

- quanto menor a confiança em um servidor, maior a quantidade necessária de mensagens legítimas para aumentar a confiança e menor a quantidade necessária de mensagens maliciosas para diminuir a confiança;

O *QR* será recalculado toda vez que o servidor receptor receber uma mensagem e comparado com o CMLS (*Contador de mensagens legítimas no servidor*) no caso de mensagem legítima ou com o CMMS (*Contador de mensagens maliciosas no servidor*) no caso de mensagem maliciosa. O *CMLS* é um contador definido no servidor para cada emissor, que armazena o número de mensagens legítimas recebidas e é utilizado para aumentar a confiança local (*CLS*) no servidor. O *CMMS* é um contador definido no servidor para cada emissor, que armazena o número de mensagens maliciosas recebidas e é utilizado para diminuir a confiança local no servidor. Se a mensagem for legítima, o servidor deverá incrementar o *CMLS* em 1; quando o *CMLS* for igual a *QR*, a confiança local será incrementada em 1%[2] e o contador *CMLS* será reiniciado. Da mesma forma, se a mensagem for maliciosa, o servidor deverá incrementar o *CMMS* em 1; quando *CMMS* for igual a *QR*, a confiança local será decrementada em 1% e o contador *CMMS* será reiniciado. O algoritmo a seguir apresenta o cálculo da confiança local de um servidor:

```
if Mensagem idonea then
  if QR = CMLS then
    if CLS < 100% then
      CLS = CLS + 1%
    endif
    CMLS = 0
  else
    CMLS = CMLS + 1
  endif
else
  if QR = CMMS then
    if CLS > 0% then
      CLS = CLS − 1%
    endif
    CMMS = 0
  else
    CMMS = CMMS + 1
  endif
```

[2]O valor de 1% foi escolhido de forma empírica. Trabalhos futuros deverão ser realizados para encontrar o melhor valor.

endif

A *CLS* indica a confiança local de um servidor **A** em outro servidor **B** e será utilizada no cálculo da confiança final do servidor **A**. Como por exemplo, um servidor **A** possui a variável *MC* igual a 30 e confia 70% em um servidor **B**. Supondo que o *CMMS* de **B** no servidor **A** seja 20 e o *CMLS* seja 7, então:

QR para mensagem maliciosa

$$QR = 0.7 * 30 = 21$$

QR para mensagem legítima

$$QR = (1 - 0.7) * 30 = 0.3 * 30 = 9$$

Isto significa que será necessária mais 1 mensagem maliciosa para diminuir a confiança, pois neste caso QR é igual a 21 e *CMMS* é igual a 20 e mais 2 mensagens legítimas para aumentar a confiança, pois neste caso QR é igual a 9 e *CMLS* é igual a 7.

6.3.2 Confiança global

A confiança global define a confiança de um servidor emissor em um determinado grupo de servidores confiáveis (ver seção 6.5). A confiança global é a média de todas as confianças locais de cada servidor e deverá ser propagada pela rede através do modelo de propagação de confiança. A fórmula para o cálculo de confiança do servidor **A** em **B** é apresentada a seguir:

$$CGS_{AB} = \frac{\sum_{i=0}^{k} CLS_{iB}}{k}$$

onde k é a quantidade de servidores pertencentes ao grupo de confiança de **A** e CLS_{iB} é a confiança de cada servidor do grupo de **A** (incluindo a dele) em **B**.

A CGS_{AB} indica a confiança do grupo de **A** em **B** e será utilizado no cálculo da confiança final do servidor **A**.

6.3.3 Cálculo da confiança final

A CS (*Confiança Final*) é a confiança de um servidor **A** em um servidor **B**. Esta confiança foi proposta para definir um modelo justo de cálculo de confiança que considera a confiança de um grupo de servidores e a confiança do servidor em si. Isto significa que se um servidor receptor não conhecer um emissor, ele poderá gerar uma confiança inicial influenciado pela confiança de seu grupo de servidores. Esta confiança será calculada a partir da seguinte fórmula:

$$CS = \frac{CLS + CGS}{2}$$

onde CLS (*Confiança Local do Servidor*) é a confiança local de **A** em **B** e CGS (*Confiança do Grupo no Servidor*) é a confiança do grupo de **A** em **B**.

A confiança local e a confiança em grupo foram definidas visando a propagação da confiança de um determinado emissor através da rede sem que esta confiança seja decisiva no recebimento de mensagens, visto que o grupo de confiança é capaz de mentir sobre a credibilidade de um emissor na rede.

Caso um servidor receptor não possua uma confiança definida em um servidor emissor que está tentando se comunicar, será necessária a definição de uma confiança local *CLS* inicial. Para este trabalho foi definido que a *CLS* inicial será de 80% (serão realizados trabalhos futuros para a definição da melhor *CLS* inicial). Isto faz com que a confiança inicial *CS* de um servidor para um desconhecido seja de 40%, pois ainda não foi calculada a global do emissor. Então:

$$CS = \frac{80 + 0}{2} = 40$$

tornando o critério de recebimento na primeira comunicação mais rigoroso; quanto menor a confiança, menor será o QR para diminuir a confiança.

6.4 Armazenamento de confiança

O módulo de armazenamento é responsável por armazenar, retornar e remover informações sobre servidores emissores no sistema de confiança. Esté módulo será a memória de confiança de cada servidor e deve armazenar as seguintes informações:

- **domínio**: domínio do servidor emissor;
- **nome**: nome do servidor emissor;
- **ip**: endereço IP do servidor emissor (deverá suportar múltiplos IPs por servidor);
- ***CLS***: confiança local, entre 0 e 100%;
- ***CGS***: confiança global, entre 0 e 100%;
- ***CFS***: contador de fichas que o servidor possui (utilizada na priorização de consulta, vide seção 6.5.1);
- ***CMLS***: contador de mensagens legítimas recebidas pelo servidor. Deverá ser reiniciado toda vez que atingir a quantidade de QR;
- ***CMMS***: contador de mensagens maliciosas recebidas pelo servidor. Deverá ser reiniciado toda vez que atingir a quantidade de QR;
- **TTL (*Time To Live*)**: tempo de vida desta informação.

O TTL disponibiliza um modelo de "esquecimento" de acordo com o tempo que um servidor fica sem se comunicar com o emissor. Quando um servidor é armazenado, o sistema de confiança atribui um TTL indicando a quantidade de dias que aquela informação será válida. A cada dia que se passa o TTL é decrementado em uma unidade. A cada comunicação o TTL deverá ser reiniciado. Quando o TTL chegar a zero, as informações de confiança deste servidor não serão mais válidas e serão armazenadas como histórico para futuras comunicações. O histórico de cada servidor será utilizado para recalcular a confiança inicial dos servidores que deixaram de se comunicar durante muito tempo (definido pelo TTL) com o servidor receptor. Caso um servidor emissor sem confiança em um receptor possua um histórico de comunicação, a confiança *CLS* inicial deverá ser calculada utilizando a seguinte fórmula:

$$CLS = \frac{80 + h}{2}$$

onde CLS é a média entre o histórico (h) do servidor e um grau de confiança inicial (80%).

Esta fórmula permite que um servidor com maus antecedentes recupere sua confiança na rede, e que bons servidores continuem com um confiança inicial boa. A figura 6.4 apresenta os quatro cenários de armazenamento.

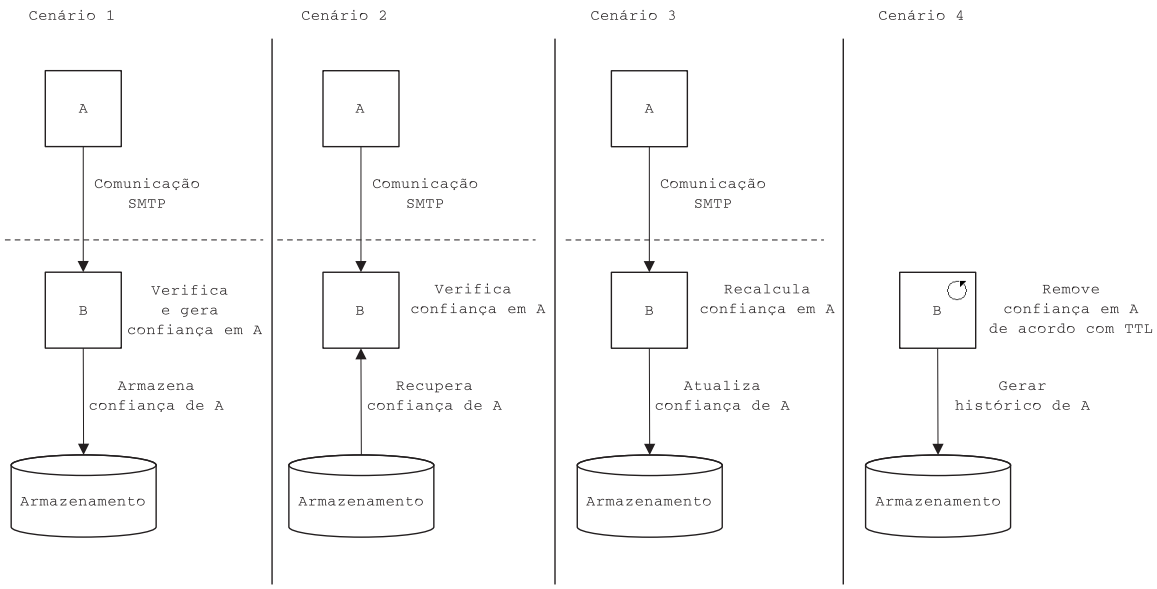

Figura 6.4: Modelo de armazenamento

No primeiro cenário o servidor de e-mail **A** inicia uma comunicação SMTP com o servidor **B**. Então **B** verifica no sistema de armazenamento se ele possui confiança em **A**. O sistema de armazenamento responde negativamente, então, **B** armazena uma confiança inicial em **A**.

No segundo cenário, o servidor de e-mail **A** inicia uma comunicação SMTP com o servidor **B**. Então **B** verifica no sistema de armazenamento se ele possui confiança em **A**. O sistema de armazenamento responde positivamente, então, **B** recupera a confiança inicial em **A**.

No terceiro cenário, **A** termina de enviar uma mensagem para **B**. Então **B** recalcula a confiança dele em **A** e atualiza o sistema de armazenamento.

Por último, no quarto cenário, **B** possui um contador de tempo que verifica o TTL de cada servidor, decrementa o TTL e se este atingir zero, gera um histórico do servidor

A e grava no sistema de armazenamento.

6.5 Propagação de confiança

A propagação de confiança é fundamental para o funcionamento da proposta, visto que o objetivo deste sistema consiste nos servidores de e-mail serem capazes de gerar listas de servidores confiáveis e não confiáveis de forma dinâmica e descentralizada através de um grupo de servidores confiáveis entre si.

Os grupos de confiança são grupos definidos pelo administrador de serviços de e-mail que indicam quais servidores deverão ser consultados para o cálculo da confiança global. Foi optado pelo método de escolha manual dos grupos de confiança pois os métodos de cálculo de reputação dos nós são difíceis de serem implementados e definidos, sendo necessário um estudo mais detalhado sobre o assunto (não faz parte do escopo deste trabalho). No decorrer do estudo foram encontradas três formas de propagação de confiança. Essas formas podem ser classificadas como:

- **Propagação síncrona**: toda vez que houver uma alteração de confiança, o servidor deverá requisitar um novo consenso da confiança global. Os servidores requisitados farão o mesmo pedido ao seu grupo de confiança e assim por diante. Este método é lento, visto que antes da confiança ser gerada, todos os participantes de uma rede de relacionamento deverão ser consultados;

- **Propagação assíncrona**: um servidor **A** pergunta ao seu grupo sobre a confiança de um servidor **B**. O resultado desta consulta será a confiança global de **A** em **B**. Este método é mais rápido do que o síncrono, contudo as pesquisas serão restritas aos servidores que **A** conhece ou que estão tentando iniciar uma comunicação com **A**.

- **Propagação híbrida**: um servidor **A** notifica, de forma síncrona, os demais integrantes de seu grupo de confiança informando que um emissor **B** perdeu ou ganhou a confiança de **A**. Os servidores utilizam essas notificações para perguntar ao seu grupo de confiança, de forma assíncrona, sobre a confiança do grupo em **B** afim de se gerar a confiança global de **B**. Este método propaga a informação de uma maneira rápida e permite com que os servidores que mais se comunicam na rede sejam propagados mais rapidamente.

Optou-se pelo uso do método de comunicação híbrida, por ser capaz de alcançar um grande número de servidores pela rede, com um desempenho razoável. A seguir serão

apresentados os métodos de notificação (síncrono) e de propagação global (assíncrono).

6.5.1 Notificações de ajuste de confiança

Esta abordagem utiliza o conceito de "fichas" para notificar o ajuste de confiança de um determinado servidor. As fichas indicam a alteração da confiança de um receptor em um emissor e será utilizada como base de priorização de consultas do sistema de propagação de confiança global, ou seja, quanto maior o número de fichas um determinado servidor possuir em outro, maior será a prioridade de consulta para definição da sua confiança global. A figura 6.5 apresenta o modelo de notificação de ajuste de confiança.

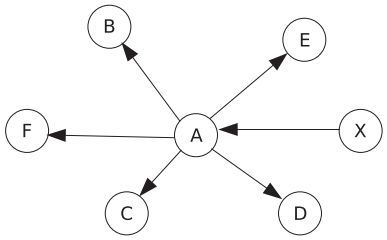

Figura 6.5: Modelo de comunicação síncrono

Nesta figura, o servidor **A** envia uma ficha para todos os servidores de seu grupo de confiança (**B**, **C**, **D**, **E** e **F**) indicando que ocorreu uma alteração de confiança no servidor **X**. Os servidores aceitam a ficha, gravam em seu sistema de armazenamento e aguardam novas notificações. O algoritmo a seguir define o modelo de envio e recebimento de fichas.

Notifica ajuste

foreach nó$_i$ pertencente ao grupo de confiança **do**
  envia ficha (domínio, ip) alterado para nó$_i$
end

Recebe ajuste

while *verdadeiro* **do**
  recebe mensagem (domínio, ip) de nó$_j$
  if nó$_j$ faz parte do grupo de confiança **then**

```
        armazena (domínio, ip, nova ficha)
        notifica sistema de propagação de confiança global
      endif
    end
```

O primeiro algoritmo notifica os demais servidores do grupo de relacionamento, que houve uma alteração de confiança em um servidor emissor. O segundo algoritmo verifica se o servidor que está notificando faz parte do grupo de confiança. Caso faça parte, recebe a mensagem e notifica o sistema de propagação global, do contrário, ignora notificação.

6.5.2 Propagação da confiança global

A propagação de confiança global faz uso das redes de relacionamento para propagar a confiança de um determinado servidor para o maior número de servidores e-mail confiáveis, ou seja, um servidor **A** deverá propagar a sua memória de confiança para o seu grupo de confiança; cada integrante deste grupo propagará a sua memória de confiança e assim por diante. Desta forma, todos os servidores que fizerem parte da rede de relacionamento de **A** deverão receber informações propagadas por **A**. A figura 6.6 apresenta o modelo de propagação de confiança global.

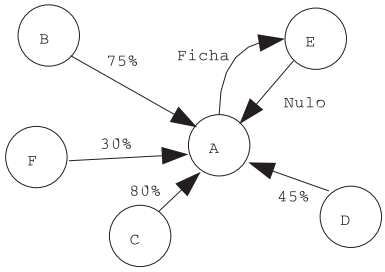

Figura 6.6: Modelo de propagação assíncrono

Nesta figura o servidor **A** faz uma requisição de confiança para todos os servidores de seu grupo de confiança. Os servidores **B**, **C**, **D** e **F** retornam a confiança ao servidor **A**; o servidor **E** não possui informações sobre o servidor requisitado, então, o servidor **A** envia uma ficha ao servidor **E**, para que **E** seja capaz de gerar sua própria confiança global no servidor pesquisado por **A**. Após receber todas as respostas o servidor **A** deverá

calcular a confiança global (CGS). A seguir é apresentado o algoritmo de propagação global.

Requisita confiança

while *verdadeiro* **do**
  if existe servidor não pesquisado na memória de confiança **then**
    j = seleciona o servidor com maior numero de fichas
    if CLS_j igual a NULO **then**
      $CLS_j = 80$
    endif
    $CGS = CLS_j$
    k = 1
    foreach nó pertencente ao grupo de confiança = i **do**
      C = requisita confiança de j para nó_i
      if C igual a NULO **then**
        envia ficha e endereço do servidor alterado para nó_i
      else
        $CGS = CGS + C$
        k = k + 1
      endif
    end
    $CGS = \frac{CGS}{k}$
    armazena CGS
  else
    aguarda recebimento de ficha
  endif
end

Retorna confiança

while *verdadeiro* **do**
  server = escuta entrada (bloqueante)
  recupera j
  if servidor possui confiança em j **then**
    return CLS_j
  else
    return NULO

endif
end

O primeiro algoritmo fara uma verificação da existência de um servidor na base de confiança que a confiança global ainda esteja com a confiança global NULA (ainda não foi gerada) ou todos os servdores que tenham uma notificação de ajuste pendente. Se não existirem servidores pendentes, este algoritmo ficará aguardando até que existe uma notificação de ajuste de confiança. Do contrário, ele deverá serlecionar o servidor que tenha um número maior de notificações pendentes e deverá requisitar ao seu grupo de confiança a confiança que cada um possui naquele servidor pendente. Caso um membro consultado não possua a confiança naquele servidor, será retornada ao servidor requisitor uma notificação informando que ele não possui uma confiança local formada, do contrário, a confiança deverá ser retornada e somada ao *CGS*. Ao final do recebimento de todas as respostas à requisição, o servidor deverá calcular o *CGS* e gravar no sistema de armazenamento.

O segundo algoritmo deverá receber as requisições, verificar se o servidor requisitado encontra-se na memória de confiança e retornar a *CLS* (se estiver cadastrado) ou então retornar NULO. Por fim, deverá aguarda por novas requisições.

6.6 Benefícios e limitações

A arquitetura apresentada torna possível a diminuição gradativa do tráfego de mensagens de servidores de e-mail que são utilizados por *spammers* como ferramentas de transporte de mensagens maliciosas. A arquitetura traz outros benefícios para a implementação deste sistema em serviços de e-mail:

- cria um modelo capaz de diminuir a quantidade de mensagens a serem recebidas de um determinador servidor, de acordo com um consenso de um determinado grupo de servidores;
- este modelo diminui a quantidade de mensagens não solicitadas armazenadas no servidor, visto que ele diminui a quantidade de mensagens que um possível servidor *spammer* possa enviar;
- a memória de confiança permite com que administradores de serviços de e-mail criem blacklists e disponibilizem informações de possíveis `spammers` em RBLs;

- fácil integração com diversos tipos de filtros para e-mail. Não existe necessidade de alteração do protocolo de comunicação entre servidores;
- facilmente adaptável a outros tipos de algoritmos de Redes de relacionamento e P2P;
- não necessita alteração no protocolo SMTP;
- compatível com o modelo atual de transporte de e-mail;

As principais restrições e deficiências desta proposta são:

- troca de informação entre servidores pode ocasionar perda de desempenho na comunicação dos servidores de e-mail;
- servidores confiáveis podem ser considerados não confiáveis a partir da propagação de confiança global;
- servidores são mais susceptíveis a ataques de negação de serviço (visto que eles aceitam informações de vários servidores da rede).

6.7 Trabalhos correlatos

Nesta seção serão apresentados o *MailRank* e o *Leveraging Social Network to Fight Spam*, dois trabalhos que utilizam redes de relacionamento para criar *white lists* e *black lists* de servidores de e-mail.

6.7.1 MailRank

O *MailRank* [CDN05] é um sistema colaborativo para criação de *white lists* globais. Os dados da lista são coletados através de redes de relacionamento e agrupados em uma única rede global baseada nas atividades dos usuários do MailRank. Cada mensagem enviada a outro servidor é transformada em um voto de confiança.

Representando a rede social de e-mail como um grafo, os autores criaram um algoritmo de iteração global que computa cada valor de reputação para cada endereço de e-mail. Este modelo assume que cada indivíduo de uma rede social não é um *spammer*, e a pontuação deste pode ser usada para identificar endereços de *spammers*.

Arquitetura do sistema

O MailRank é composto por dois módulos principais: proxy MailRank (implementado nos clientes de e-mail) e servidor MailRank (máquina específica da rede).

- **Proxy MailRank**:

  Quando o proxy recebe um e-mail de saída, ele extrai os votos do usuário, utilizando alguns dados de entradas disponíveis (tal como analisando a caixa de mensagens enviadas), envia os votos coletados para o servidor MailRank e repassa a mensagem para o servidor de e-mail local. Quando um e-mail é recebido, o proxy consulta o servidor MailRank verificando o ranque do endereço do emissor, recebe a pontuação já computada e atualiza a pontuação recebida no assunto da mensagem.

  O proxy utiliza os votos que são baseados nos campos "To" do cabeçalho ("To:", "Cc:", "BCc:", etc.). Nenhuma ação adicional por parte do usuário para a criação dos votos é necessária, desde que o proxy consiga extrair automaticamente os votos. Para isto, o usuário do MailRank precisa alterar seu cliente de envio de e-mail para apontar para o MailRank ao invés do servidor de e-mail local.

- **Servidor MailRank**:

  O servidor coleta os dados de entrada de todos os usuários do MailRank utilizando o algoritmo MailRank. O principal objetivo do algoritmo MailRank é atribuir uma classificação para cada endereço de e-mail armazenado no sistema e usar esta classificação para decidir qual endereço de e-mail é de um *spammer* ou não. As etapas deste algoritmo são:

  1. Determinar um conjunto de endereços de e-mail que possuam uma alta reputação na rede social;
  2. Executar o algoritmo no grafo da rede de e-mail, utilizando o conjunto determinado anteriormente para computar a pontuação final de cada endereço de e-mail.

  Existem três formas de se determinar o conjunto de endereços de e-mail:

  1. **Manual**: garante que nenhum *spammer* fará parte do conjunto;
  2. **Automática**: Seleciona os endereços de e-mail que possuem um ranque alto, tal que que a soma dos ranques de cada nó seja igual a 2 do total de ranque do sistema;
  3. **Semi-automática**: Faz uma pré-seleção utilizando a abordagem automática e disponibiliza essa seleção para que seja refinada manualmente.

A partir deste conjunto é gerado um vetor de de pontuação final. Este vetor irá classificar o endereço emissor a partir do proxy de entrada da seguinte forma:

- **não spammer**: se a pontuação final do endereço emissor for maior que o ponto inicial T;
- **spammer**: se a pontuação final do endereço emissor for menor que o ponto inicial T;
- **não identificado**: se o endereço de e-mail ainda não foi identificado pelo sistema.

6.7.2 Leveraging Social Networks to Fight Spam

O *Leveraging Social Networks to Fight Spam* [BR05] é uma ferramenta antispam que explora as propriedades das redes sociais para distinguir mensagens de e-mail não solicitadas (*spam*) de mensagens associadas às pessoas que um usuário conhece. O funcionamento desta ferramenta foi dividido em três etapas apresentadas a seguir:

A primeira etapa é criar uma rede de relacionamento de um único usuário. Essa rede é montada baseada nas informações disponíveis em mensagens de sua caixa de correio: emissores (a partir do campo `From`) e receptores (a partir dos campos `To`, `Cc`, `BCc`, etc). A partir das informações obtidas, são definidos nós que representam todos os endereços coletados. Em seguida são adicionadas conexões entre os pares de endereços que aparecem na mesma mensagem (indivíduos que se comunicaram através do outro). Os nós que representam o próprio usuário deverão ser removidos.

A segunda etapa consiste em utilizar a rede social (grafo) montada para definir *white lists* de usuários, utilizando uma propriedade das redes sociais: Tendência de aglomeração. Por exemplo, se Alice conhece Bob e Eve, Bob possui grandes chances de também conhecer Eve.

A terceira etapa consiste em utilizar fórmulas de classificação definidas em [BR05], nos grupos aglomerados, que analisam se esses grupos de mensagens são *spams* ou não *spams* a partir de seus coeficiente de aglomeramento. Todos os componentes que tenham o coeficiente de aglomeramento maior que uma taxa máxima C_{max} fazem parte do grupo social do usuário. Todos os nós desses componentes deverão ser inseridos em uma *white list*. Componentes que possuam um coeficiente de aglomeramento menor que uma taxa C_{min} serão considerados *spams* e serão inseridos em uma *black list*. Os componentes que estiverem entre a taxa C_{min} e C_{max} não são passíveis de classificação, por isso deverão ser definidos como componentes não classificados.

6.8 Conclusão

Neste capítulo foi apresentada a proposta do trabalho e os trabalhos correlatos que utilizam técnicas de redes de relacionamento para classificar possíveis *spammers* e gerar listas de e-mails maliciosos, visando conter a quantidade de *spams* que trafegam pela rede.

A proposta deste trabalho foi dividida em três módulos:

- Gerência de confiança: forma como a confiança é tratada pelo sistema de confiança;
- Comunicação entre servidores: modelo de propagação de confiança pela rede;
- Sistema de armazenamento: modelo de armazenamento das informações sobre confiança.

A proposta deste trabalho utiliza uma abordagem diferente no uso de redes de relacionamento das que foram apresentadas como trabalhos correlatos.

- O primeiro trabalho utiliza as redes de relacionamento para classificar se uma determinada mensagem é um *spam* ou não. Este trabalho utiliza um sistema de votos que permite classificar e propagar *spammers* e *spams* pela rede.
- O segundo trabalho utiliza as redes de relacionamento para verificar se um determinado usuário apresentado no campo `From` é um *spammer*.
- A proposta deste trabalho utiliza as redes de relacionamento para classificar servidores que são utilizados como emissores de *spam*, diminuindo gradativamente a quantidade de mensagens que esses servidores maliciosos são capazes de enviar para um determinado servidor. Esta proposta apresenta ainda algoritmos para propagar esta confiança para outros servidores da rede.

Além disso, são apontados os principais benefícios e limitações da proposta. No próximo capítulo serão discutidos os detalhes de implementação, os testes efetuados e as análises dos resultados obtidos.

Capítulo 7

Implementação e resultados

O protótipo da proposta apresentada no capítulo anterior foi implementado para a plataforma `Linux`, usando o `Postfix` como servidor de e-mail, o `SpamAssassin` como filtro anti-spam e o `Clamav` como filtro anti-vírus. Foi implementado um programa, denominado `TrustMail`, que realiza a gerência de confiança, armazena as informações e propaga a confiança pela rede a partir do grupo de confiança. As próximas seções descrevem a implementação do `TrustMail`, o sistema de comunicação entre o servidor de e-mail e os demais componentes da proposta e uma avaliação sobre o funcionamento do sistema.

7.1 TrustMail

O `TrustMail` é o sistema de confiança apresentado na proposta. O `TrustMail` implementa os modelos de gerência, armazenamento e propagação de confiança. Foi foi implementado na linguagem C e utiliza o SQLite (biblioteca que contém funções SQL para armazenamento de dados) para armazenar as informações sobre a confiança dos servidores de e-mail. A comunicação do `TrustMail` com as outras ferramentas para recebimento e classificação de e-mail é feita através de filas de mensagens do Linux (explicado a seguir). As filas de mensagens podem ser descritas como listas internas no espaço de endereçamento do Kernel para a troca de mensagens entre processos [Rib05]. Optou-se por este método de IPC (*Interprocess Communication*) para facilitar a integração do `TrustMail` com as demais ferramentas.

O MTA escolhido foi o `Postfix` 2.2, pela facilidade de integração com outras ferramentas e pelo seu código ser simples e legível. O sistema de autenticação de servidor escolhido foi o SPF, por ser fácil de implementar (existe uma biblioteca de desenvolvimento "libspf2" disponível para download no site do autor) e por ser fácil de integrar com o *Postfix*. Tanto a comunicação do MTA com o SPF, quanto a comunicação com

o `TrustMail` são feitas através do `policyd`. O `policyd` é uma implementação SPF desenvolvida pela mesma equipe do libspf2 que utiliza esta biblioteca para verificar a autenticidade dos servidores de e-mail. O código do `policyd` foi modificado para suportar a comunicação com o `TrustMail`. Este programa fará a autenticação completa de um servidor, tornando desnecessária a comunicação direta do `TrustMail` com o `Postfix`. O filtro anti-spam escolhido foi o `SpamAssassin` 2.63. Este filtro possui vários critérios de classificação de mensagens, como por exemplo, análise de cabeçalho, análise de texto, *black lists*, sistema de pontuação de mensagens de acordo com características recuperadas através de análise bayesiana, entre outras. O anti-vírus utilizado foi o `Clamav` 0.70. Este anti-vírus foi desenvolvido em C e possui uma extensa lista de assinaturas de vírus e estão em constante atualização (tanto o Clamav quanto o SpamAssassin rodam como *daemons*, facilitando a integração com outros programas). A comunicação do `Postfix` com os filtros de mensagens é feita através do script `Clamav-Filter`. Este script foi alterado para suportar a notificação de mensagens maliciosas/legítimas para o `TrustMail`. O protótipo da implementação atual está ilustrado na figura 7.1

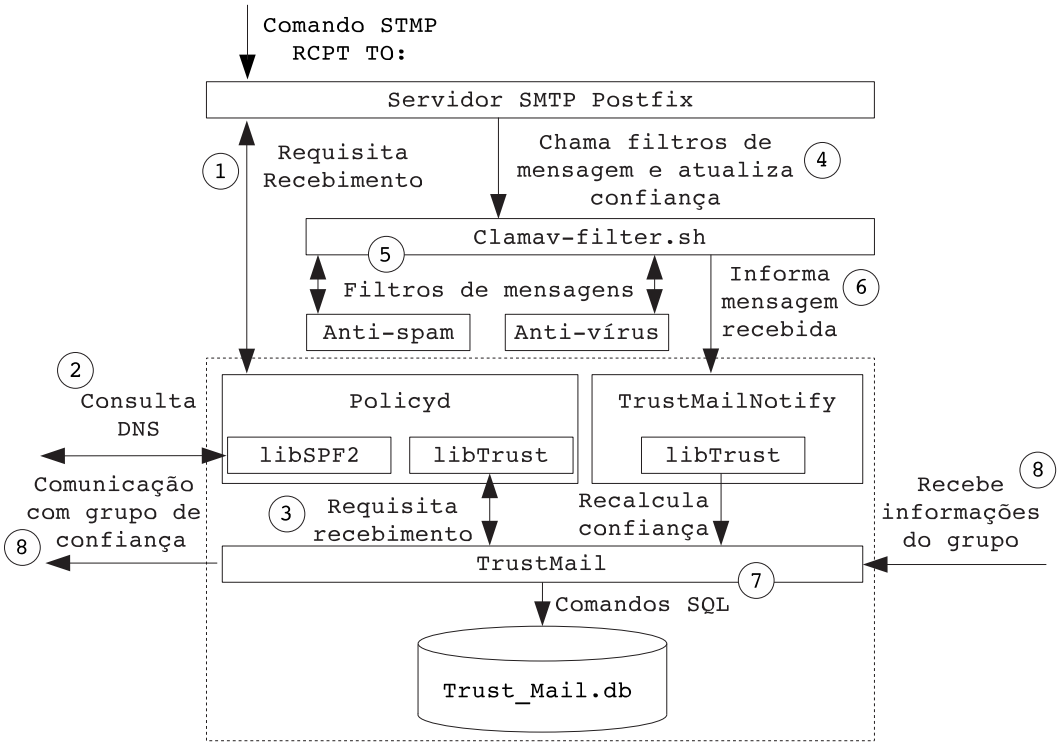

Figura 7.1: Protótipo implementado

Nesta figura é possível observar que o `TrustMail` independe do servidor, do sistema de autenticação e dos filtros de mensagens. Ou seja, este modelo pode ser facilmente integrado com outras ferramentas. A seguir será detalhado o fluxo de comunicação entre

os processos no recebimento de e-mails.

7.1.1 Fluxo do programa

O processo de recebimento deverá seguir as seguintes etapas:

1. O *Postfix* deverá fazer uma chamada ao *policyd* toda vez que receber um comando `RCPT TO` e deverá aguardar o comando de resposta (que será recebido do *policyd*). Os parâmetros passados ao *policyd* são: endereço do emissor, ip do emissor e domínio informado no comando `HELO/EHLO`.

2. O *policyd* recebe os parâmetros de entrada recebidos do `Postfix` e faz uma consulta, utilizando a biblioteca `libspf2`, ao servidor DNS do domínio do emissor, autenticando assim, a procedência do servidor. O retorno desta consulta deverá ser:

   - `Pass`, `Soft Fail`, `Neutral`, `Unknown` ou `None`: O processo continua;
   - `Error`: Retorna ao `Postfix` a mensagem "450 Falha temporária";
   - `Fail`: Retorna um erro e a conexão deve ser finalizada pelo `Postfix`.

3. Se o processo prosseguir, o *policyd* chamará o `TrustMail` (através de Filas de Mensagens) e requisitará se a conexão pode continuar. O `TrustMail` calcula a quantidade de mensagens que o servidor pode receber em um período de tempo (no caso do protótipo, o período estabelecido foi de um dia) e retorna **verdadeiro** se o servidor puder receber, ou **falso**, caso contrário. O `policyd` notifica o `postfix` sobre a resposta recebida. Por fim, o processo de recebimento de mensagem do `Postfix` continua.

4. Quando uma mensagem for recebida, o `Postfix` chamará o script `Clamav-Filter`. Os parâmetros passados ao `Clamav-Filter` são: o endereço ip, o host de origem e o domínio da mensagem.

5. Em seguida, o `Clamav-Filter` recuperará a mensagem de um diretório, chamará o `Clamav` e o `SpamAssassin` e aguardará o retorno.

6. Depois, o `Clamav-Filter` chamará o `TrustMailNotify` que notificará ao `TrustMail` (utilizando filas de mensagens) que uma mensagem maliciosa ou legítima foi recebida.

7. Por fim, o servidor calculará a nova confiança e armazenará as novas informações no `Trust_Mail.db`.

8. A comunicação pela rede é feita pelo processo **TrustMail** que escuta na porta 60002 as notificações de fichas e na porta 60001 a requisição de confiança. Quando a confiança de um servidor é alterada no servidor, ele propaga uma ficha para todos os integrantes do seu grupo de confiança. Uma thread que roda a cada 60 segundos verifica se existem fichas pendentes no servidor para requisitar a confiança. Caso existam fichas pendentes, o servidor faz uma requisição para todos os servidores de seu grupo de confiança para gerar a confiança global do servidor em questão.

Para facilitar a criação do grupo de confiança, foi desenvolvido o `TrustMailAdmin`, que insere, exclui e lista servidores do grupo de relacionamento.

Além do `TrustMailAdmin`, existe um processo batch (`TrustMailTTL`) que é executado pelo `cron.d`. Este processo é responsável por limpar os contadores temporários e diminuir o **TTL** dos servidores.

Quando a confiança do servidor é alterada, um processo de log registra o domínio, CGS, CLS, CMMS e CMLS. Este log será utilizado para a avaliação do protótipo.

7.2 Avaliação do protótipo

O equipamento utilizado para a avaliação do protótipo nos testes foi um Pentium 4 2.8, com 1.5 GB de memória e 120 GB de disco rígido serial ATA e um Pentium 3 Dual 800 com 1024GB de memória e 80 GB de disco rígido. O sistema operacional de ambas as máquinas é o Linux Fedora Core 4 com *kernel* 2.6.3. Para a avaliação dos serviços de e-mail foi definida a utilização de máquinas virtuais. A máquina virtual escolhida foi o UML (*User-Mode Linux*) compilada com o *kernel*2.6.9. A escolha do UML foi devido a facilidade de implementação, suporte a todas as funcionalidades de uma máquina real e por ser suportada pelo *kernel* oficial do Linux.

As próximas seções discutem os aspectos relacionados a massa de teste, funcionamento e escalabilidade, desempenho e a análise dos resultados obtidos.

7.2.1 Massa de teste

Como a proposta da avaliação não é testar a eficácia dos filtros de mensagens e sim características do `TrustMail`, foram definidos dois e-mails para testes. Um e-mail será classificado como spam pelo filtro anti-spam e o outro e-mail será classificado como legítimo.

Mensagem maliciosa

O SpamAssassin classifica a mensagem *GTUBE* como sendo um *spam*. A mensagem maliciosa a ser enviada para o MTA é apresentada a seguir:

```
Subject: Test spam mail (GTUBE)
Message-ID: <GTUBE1.1010101@example.net>
Date: Wed, 23 Jul 2003 23:30:00 +0200
From: Sender <sender@example.net>
To: Recipient <recipient@example.net>
Precedence: junk
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

This is the GTUBE, the
        Generic
        Test for
        Unsolicited
        Bulk
        Email

If your spam filter supports it, the GTUBE provides a test by which you
can verify that the filter is installed correctly and is detecting
incoming
spam. You can send yourself a test mail containing the following string
of
characters (in upper case and with no white spaces and line breaks):

XJS*C4JDBQADN1.NSBN3*2IDNEN*GTUBE-STANDARD-ANTI-UBE-TEST-EMAIL*C.34X

You should send this test mail from an account outside of your network.
```

Mensagem legítima

A mensagem legítima a ser utilizada nos teste será a mensagem apresentada a seguir. Esta mensagem foi submetida aos serviços anti-vírus e anti-spam e foi considerada confiável.

```
From: Leonard Bispo de Oliveira <bispo@ppgia.pucpr.br>
To: sender@example.net
Subject: Mensagem =?iso-8859-1?q?Id=F4nea?=
Date: Thu, 28 Jun 2001 19:33:35 -0300
User-Agent: KMail/1.7.2
X-KMail-CryptoFormat: 15
MIME-Version: 1.0
```

```
8  Content-Type: text/plain;
9    charset="iso-8859-1"
10 Content-Transfer-Encoding: quoted-printable
11 Content-Disposition: inline
12 X-KMail-Recipients: sender@example.net
13 Status: R
14 X-Status: N
15 X-KMail-EncryptionState:
16 X-KMail-SignatureState:
17 X-KMail-MDN-Sent:
18
19 Modelo de mensagem que n=E3o dever=E1 ser classificada como um=20
20 SPAM
21
22 Esta mensagem servir=E1 como massa de testes para o TrustMail.
23
24 Abra=E7o,
25
26 Servidor de teste
```

7.2.2 Análise do funcionamento

Os testes foram realizados utilizadas seis máquinas virtuais. Em cada máquina virtual foi instalado todos os programas necessários para suportar as funcionalidades da proposta. O grupo de confiança de cada `TrustMail` pode ser visto na figura 7.2.

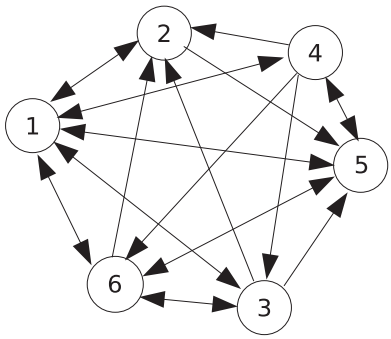

Figura 7.2: Grupo de confiança

Os valores das variáveis do `TrustMail` foram definidos da seguinte forma:

- **QM**: 1000 (foi definido um valor alto para permitir a comunicação de um servidor até o final dos testes);

- **MC**: 10;
- **TTL**: 1;
- **Confiança default inicial:** 80%.

Os testes foram realizados através de um programa que envia automaticamente mensagens para os servidores de e-mail. Este programa foi configurado para enviar mensagens do domínio `teste.com.br`. Metade dos servidores recebeu mensagens maliciosas e a outra metade recebeu mensagens legítimas. De tempos em tempos o programa modificou o envio das mensagens legítimas/maliciosas. Os servidores que estavam recebendo mensagens legítimas passaram a receber mensagens maliciosas e vice-versa. Todas as mensagens recebidas foram tratadas por cada um dos servidores e armazenadas para análise. Ao final dos testes, cada servidor recebeu cerca de duzentas mensagens.

Para facilitar a análise, os resultados obtidos foram plotados em gráficos. A análise foi feita a partir dos gráficos com valores mais expressivos do funcionamento do `TrustMail`. Destes, cinco estão interpretados detalhadamente nesta seção. O último gráfico corresponde a uma visão geral de todos os servidores. A seguir são apresentados os gráficos que representam os resultados obtidos.

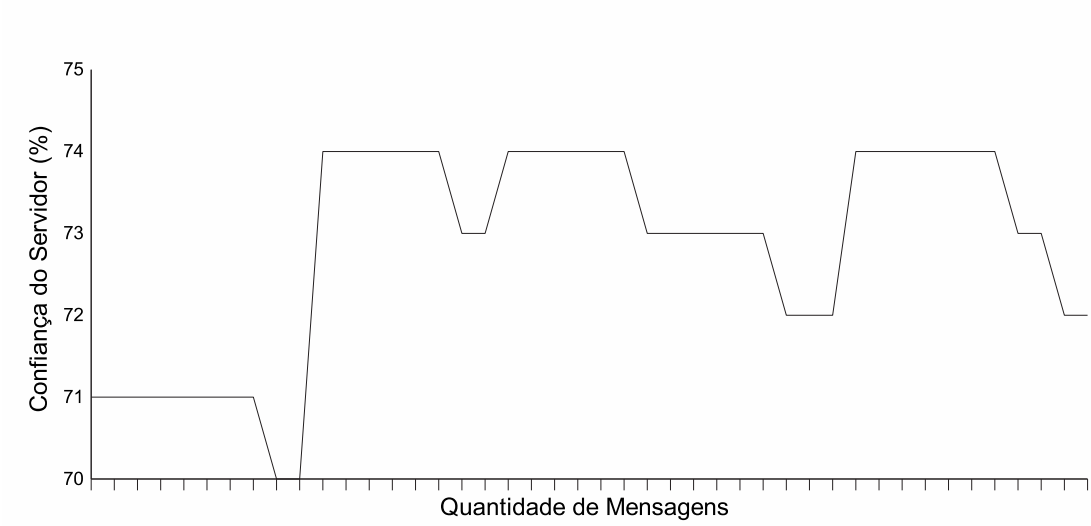

Figura 7.3: Acompanhamento do Grau de Confiança do Servidor 1

Esta figura demonstra que, partindo de um grau de confiança total (CS) de 71% e, estando o grau de confiança global dos servidores (CGS) em 66%, foram necessárias sete mensagens maliciosas para diminuir 1% do grau de confiança local do servidor (CLS) e conseqüentemente o grau de confiança total (CS). O valor de sete mensagens maliciosas foi obtido pela fórmula:

$$QR = 0.7 * 10 = 7$$

Enquanto seriam necessárias três mensagens legítimas para aumentar em 1% o grau de confiança total (CS), conforme demonstrado na fórmula abaixo:

$$QR = (1 - 0.7) * 10 = 0.3 * 10 = 3$$

A linha do gráfico que parte do valor de confiança de 70% para 74%, demonstra o aumento de 4% no grau de confiança total (CS), devido ao recalculo do grau de confiança global dos servidores (CGS) que passou de 66% para 74%. O recalculo dos servidores é realizado a cada 60 segundos.

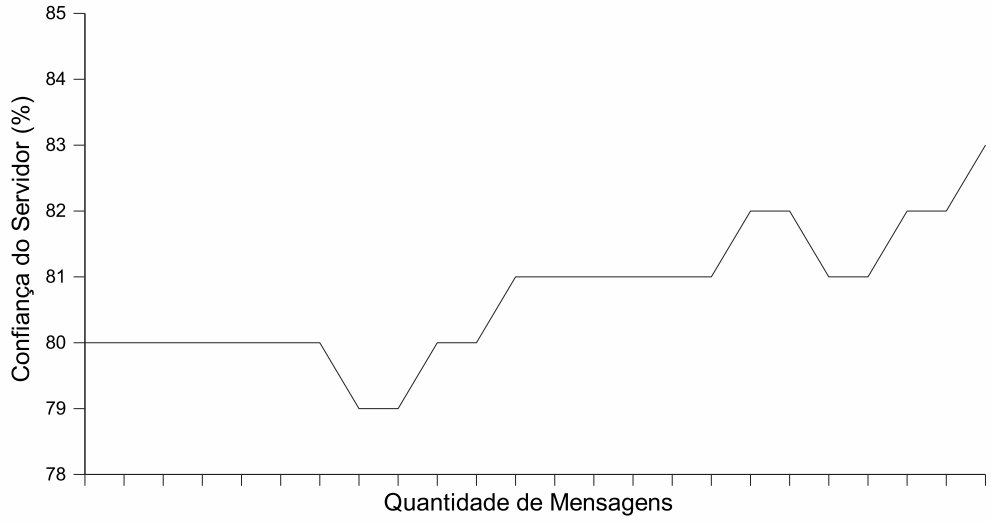

Figura 7.4: Acompanhamento do Grau de Confiança do Servidor 3

Esta figura destaca a variação das condições de aumento da quantidade de mensagens legítimas e aumento da quantidade de mensagens maliciosas. O ponto no gráfico, cujo valor de confiança total (CS) é de 65% sofre uma queda para o valor de 58% quando o servidor recebe o total de seis mensagens maliciosas. O grau de confiança total (CS) volta a aumentar, conforme demonstrado na curva do gráfico, quando se estabiliza a quantidade de mensagens maliciosas e aumenta a quantidade de mensagens legítimas. A cada cinco mensagens legítimas é aumentado 1% do grau de confiança total (CS), que passou de 57% para 58%

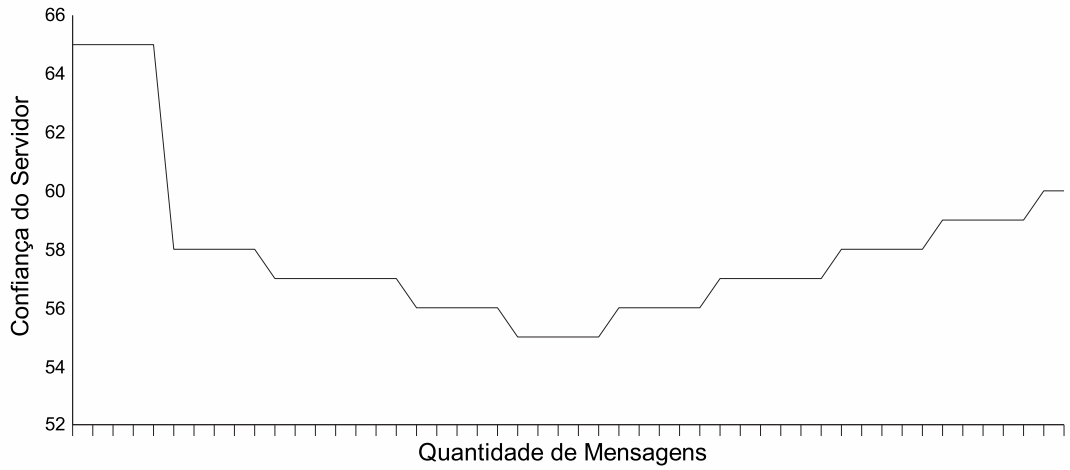

Figura 7.5: Acompanhamento do Grau de Confiança do Servidor 4

Esta figura demonstra a diminuição do grau de confiança total (CS) de 79% para 67% devido à queda do valor de confiança global dos servidores (CGS), que passou de 75% para 51%. Esta diminuição ocorreu mesmo estando estáveis os valores de confiança local do servidor (CLS) e tendo aumentado a quantidade de mensagens legítimas.

Logo em seguida, é demonstrado o aumento do grau de confiança total (CS) de 67% para 78%, também justificado exclusivamente pelo aumento do valor de confiança global dos servidores (CGS).

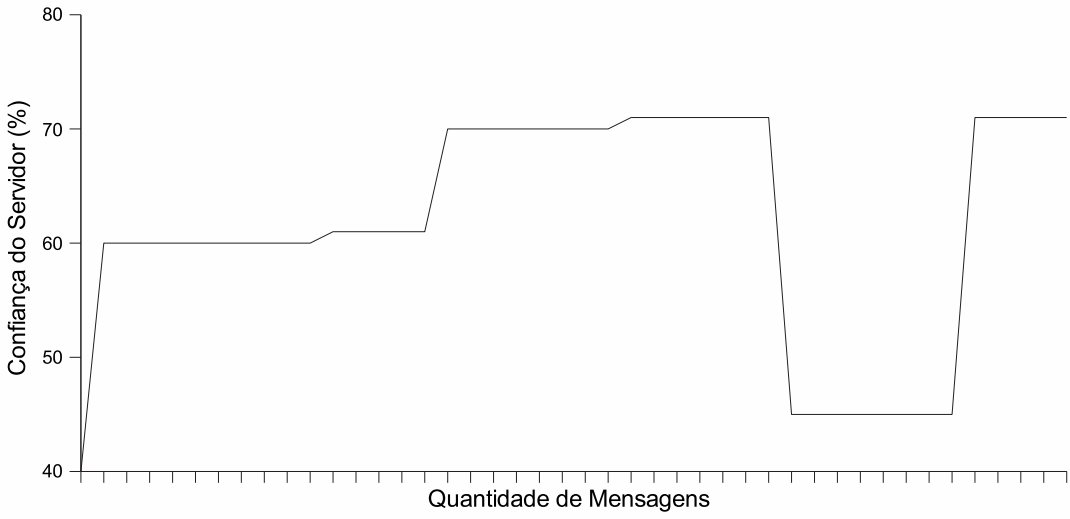

Figura 7.6: Acompanhamento do Grau de Confiança do Servidor 5

Esta figura apresenta uma situação comum do funcionamento do `TrustMail`. Basicamente, iniciando a análise onde o grau de confiança total (CS) é de 79%, este sofre uma queda devido ao recalculo do valor de confiança global dos servidores (CGS) que passou de 77% para 65%. A curva do gráfico sofre pequenas alterações. Essas alterações são decorrentes da fórmula QR onde, estando o valor de 73% de confiança total (CS), foram necessárias três mensagens para aumentar 1% deste valor. Estas pequenas variações ocorrem até 76%, quando sofre uma elevação, que corresponde ao aumento do valor de confiança total (CS) para 81%, também devido ao recalculo do valor de confiança global dos servidores (CGS).

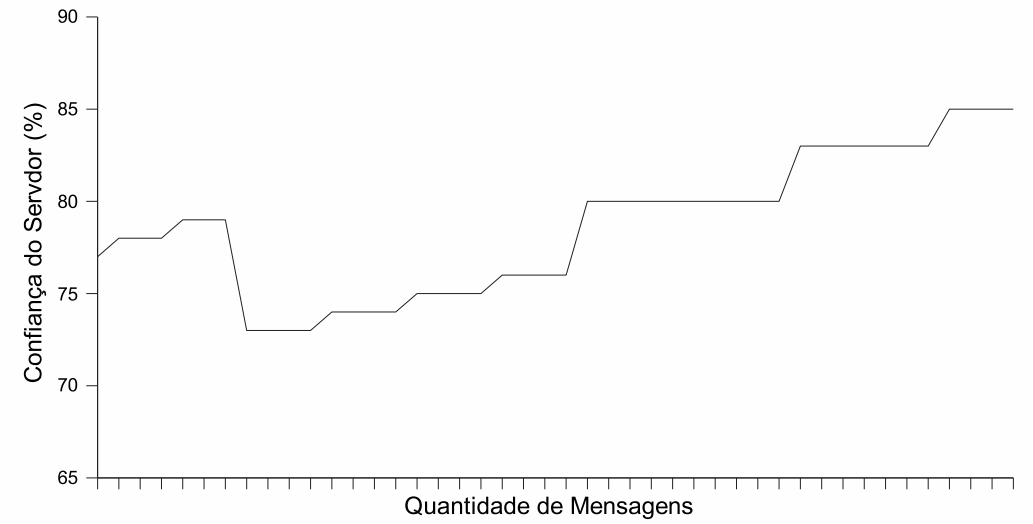

Figura 7.7: Acompanhamento do Grau de Confiança do Servidor 6

Esta figura apresenta uma deficiência no funcionamento do TrustMail. Partindo do ponto, cujo grau de confiança total (CS) é de 80%, foram necessárias oito mensagens maliciosas para diminuir em 1% o grau de confiança total (CS) e somente duas mensagens legítimas para aumentar este mesmo grau. Esta diferença de valores é justificada pela fórmula QR:

Quantidade necessária de mensagens legítimas

$$QR = (1 - 0.8) * 10 = 0.2 * 10 = 2$$

Quantidade necessária de mensagens maliciosas

$$QR = 0.8 * 10 = 8$$

Desta forma, um domínio poderia ter sua confiança diminuída ao enviar oito mensagens maliciosas e sua confiança aumentada posteriormente enviando apenas duas mensagens.

Este último gráfico apresenta uma visão geral de todos os servidores funcionando nas mesmas condições de quantidades de mensagens e propagação de confiança entre os seus respectivos grupos de relacionamento. O gráfico demonstra o comportamento dos servidores no recebimento das vinte e cinco primeiras mensagens.

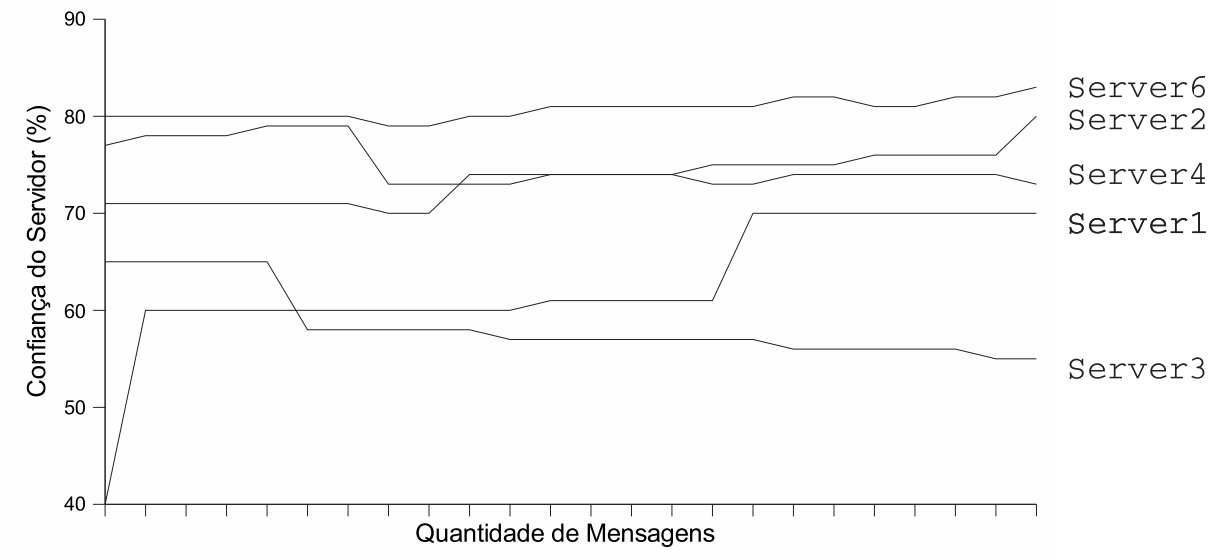

Figura 7.8: Acompanhamento de Grau de Confiança de Servidores

7.2.3 Análise do desempenho

O desempenho foi medido analisando o tempo de resposta do `TrustMail` para o servidor de e-mail, o tempo médio gasto para a propagação de confiança e o tempo médio gasto na comunicação de servidores pertencentes a um grupo comum.

Desempenho local

A avaliação do tempo de resposta do `TrustMail` foi feita utilizando uma única máquina virtual (para não sobrecarregar o processamento da máquina). Neste teste foram enviadas, quinze vezes, vinte e cinco mensagens maliciosas e vinte e cinco mensagens legítimas para o servidor de e-mail da máquina virtual com o `TrustMail` e vinte e cinco mensagens maliciosas e vinte e cinco mensagens legítimas para o mesmo servidor de e-mail da máquina virtual sem o `TrustMail`. Em ambos os caso, o MTA foi integrado com o

SPF, SpamAssassin e Clamav. Todas as mensagens foram enviadas utilizando um único canal de comunicação SMTP. Os resultados do teste são apresentados na tabela 7.1.

Tabela 7.1: Tempo de reposta do `TrustMail` para o servidor de e-mail

	Servidor sem `TrustMail`	**Servidor com** `TrustMail`	Custo (%)
1^o envio	40s	45s	12,5%
2^o envio	38s	45s	18,5%
3^o envio	42s	47s	12,0%
4^o envio	41s	48s	17,0%
5^o envio	39s	53s	36,0%
6^o envio	42s	52s	23,0%
7^o envio	42s	59s	40,5%
8^o envio	41s	58s	41,5%
9^o envio	39s	62s	59,0%
10^o envio	37s	65s	75,5%
11^o envio	42s	65s	54,5%
12^o envio	41s	69s	68,0%
13^o envio	42s	66s	57,0%
14^o envio	39s	70s	79,5%
15^o envio	40s	73s	82,5%

Analisando os resultados, nota-se que os valores de tempo no recebimento de mensagens por um servidor de e-mail sem `TrustMail` é constante. Já o tempo no recebimento de mensagens por um servidor de e-mail com `TrustMail` cresce a cada comunicação. Este aumento de tempo é causado pelas consultas SQL na base de dados de confiança. Quanto maior a base de dados, maior será o tempo de procura.

Desempenho na comunicação entre servidores

Para a avaliação do desempenho remoto, foram realizados três testes. O primeiro teste com quatro máquinas, o segundo teste com doze máquinas e o terceiro teste com vinte máquinas virtuais. Essas máquinas foram divididas em duas máquinas reais. Neste teste, todas as máquinas fazem parte de um grupo de confiança comum. Optou-se por um grupo de confiança comum para avaliar o pior dos casos, onde, todas as máquinas deverão se comunicar entre si. A base de confiança para o teste de comunicação foi gerada a partir dos testes anteriores. A partir desta base de confiança, as máquinas trocaram

informações e geraram um percentual de confiança comum (visto que todas as máquinas possuem a mesma base de confiança). Os resultados deste teste é apresentado na tabela 7.2.

Tabela 7.2: Tempo médio de comunicação entre os servidores do grupo de confiança

	Tempo médio de comunicação
Quatro Máquinas	3s
Doze Máquinas	7s
Vinte Máquinas	13s

Os resultados apresentam um aumento no tempo de comunicação a medida que o número de servidores do grupo aumenta. Esta análise de tempo de comunicação não gera resultados confiáveis, pois as máquinas virtuais encontram-se no mesmo espaço físico e a comunicação entre elas é feita diretamente pela memória. Para serem gerados resultados mais satisfatórios faz-se necessário a montagem de um ambiente real, onde as máquinas de um grupo estejam em locais diferentes, e velocidade de comunicação variável. Contudo, esses resultados permitem tirar a seguinte conclusão: quanto maior o grupo de confiança, mais lenta será a criação da confiança global, visto que a quantidade de máquinas a serem consultadas aumenta. Por este motivo, os grupos de confiança não podem compostos por um conjunto muito grande de servidores/domínios.

Esses resultados não podem ser totalmente considerados confiáveis, visto que os testes foram executados em máquinas virtuais. Para mensurar os tempos de um modo mais confiável será necessário que os testes sejam realizados em máquinas reais.

7.3 Considerações finais

O protótipo construído demonstra que a abordagem é viável, apresenta um desempenho satisfatório e um resultado interessante. Todavia, trabalhos complementares deverão ser realizados, visando melhorar os algoritmos de cálculo de quantidade de mensagens e melhorar o desempenho de comunicação.

O objetivo principal do projeto, criar um mecanismo de classificação e propagação de servidores confiáveis e não confiáveis foi alcançado com o protótipo atual, salvo problemas no cálculo de quantidades de mensagens maliciosas, conforme apresentado na figura 7.7.

7.4 Conclusão

Este capítulo descreveu o protótipo do TrustMail e as alterações no processo de recebimento de e-mail conforme proposta no capítulo 6. A implementação demonstrou como as redes de relacionamento podem ser modelos interessantes a serem utilizados na confiança de serviços da internet.

Conclusão

Este trabalho descreveu a proposta de um modelo de confiança em servidores de e-mail. A base da proposta foi gerar a confiança de cada servidor que se comunica com outro servidor e utilizar esta confiança para aumentar ou diminuir gradualmente a quantidade de mensagens que serão recebidas de um servidor emissor. A confiança é propagada pela rede para tentar atingir o maior número de servidores confiáveis através do modelo de redes de relacionamento, tornando as listas de servidores confiáveis e não confiáveis descentralizadas.

O protótipo implementado, apresentou resultados interessantes, contudo deve ser melhorado. Os cálculos de confiança apresentaram problemas no cálculo de quantidade de mensagens, pois um servidor com um grau de confiança alto precisa de mais mensagens maliciosas para perder a confiança do que para ganhar. Com isso, ele é capaz de enviar uma quantidade **X** de mensagens maliciosas e perder um porcento a confiança e enviar a mesma quantidade de mensagens legítimas e aumentar em dois porcento a confiança.

As principais contribuições deste trabalho foram:

- Estudo de modelos de serviços de autenticação de e-mail: neste estudo foram levantados os principais mecanismos de autenticação de remetente e suas principais fragilidades;
- Estudo de modelos de confiança baseado em seres humanos que podem e vêm sendo utilizados em sistemas computacionais, para tentar resolver problemas relacionados a autenticidade e confiança em sistemas distribuídos;
- Definir um modelo de confiança baseado em redes de relacionamento na tentativa de minimizar a quantidade de *spam* a serem enviados por possíveis servidores *spammers*;
- Implementar o modelo de confiança definido para validar a proposta;

Outros aspectos deste trabalho que podem ser explorados em trabalhos futuros correspondem à reputação dos membros de um grupo de confiança e os melhores valores para as constantes do modelo de confiança. Uma alternativa, é implementar algoritmos de reputação presentes em redes de relacionamento para gerar grupos de confiança de forma automática. Além disso, podem ser realizados testes mais detalhados com o `TrustMail`, alterando os valores constantes do sistema para chegar a valores mais expressivos.

Referências Bibliográficas

[Abd04] Abdul-Kareem Abdullah. Protecting your good name: identity theft and its prevention. In *InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development*, pages 102–106, New York, NY, USA, 2004. ACM Press.

[AK04] E. Allman and H. Katz. Sender id: Authenticating e-mail. Technical report, Microsoft, 2004.

[BR05] P. Oscar Boykin and Vwani P. Roychowdhury. Leveraging social networks to fight spam. *IEEE Computer*, 38(4):61–68, 2005.

[Car61] Fernando Henrique Cardoso. *Homem e sociedade : leituras básicas de sociologia geral*. Sao Paulo: Nacional, 1961.

[CDN05] Paul Alexandru Chirita, Jörg Diederich, and Wolfgang Nejdl. Mailrank: Using ranking for spam detection. *ACM International CIKM Conference*, 2005.

[CL98] Lorrie Faith Cranor and Brian A. LaMacchia. Spam! *Commun. ACM*, 41(8):74–83, 1998.

[Coh90] Fred Cohen. How does a virus spread through a system. *ASP Press*, 1990.

[Cri96] M. Crispin. RFC 2060: Internet Message Access Protocol — Version 4rev1. IETF RFC 2060, December 1996.

[Cro82] D. Crocker. RFC 822: Standard for the format of arpa internet text messages, August 1982.

[Cun05] B. D. Cunningham. Message level protocol. Technical report, MessageLevel, May 2005.

[DY04] Mark Delany and Yahoo. Domain-based email authentication using public-keys: Advertised in the DNS (DomainKeys). Internet Draft, 2004.

[FB96a] N. Freed and N. Borenstein. RFC 2045: Multipurpose internet mail extensions (mime) part one: Format of internet message bodies, November 1996.

[FB96b] N. Freed and N. Borenstein. RFC 2046: Multipurpose internet mail extensions (mime) part two: Media types, November 1996.

[FB96c] N. Freed and N. Borenstein. RFC 2049: Multipurpose internet mail extensions (mime) part five: Conformance criteria and examples, November 1996.

[FKP96] N. Freed, J. Klensin, and J. Postel. RFC 2048: Multipurpose internet mail extensions (mime) part four: Registration precedures, November 1996.

[Fro04] Michael Fromberger. Bayesian classification of unsollicited e-mail. http://thayer.dartmouth.edu/ sting/sw/bayes-spam.pdf, 2004.

[GHR05] Joshua Goodman, David Heckerman, and Robert Rounthwaite. Guerra anti-spam. *Scientific American*, May 2005.

[GKRT04] R. Guha, Ravi Kumar, Prabhakar Raghaven, and Andrew Tomkins. Propagation of trust and distrust. In *Proceedings of WWW 04*, pages 403–412. ACM, ACM, May 2004.

[Gra76] J. Grahagan. *Comportamento interpessoal e de grupo*. Rio de Janeiro: Zahar, 1976.

[Hal98] R. J. Hall. How to avoid unwanted email. *Communications of the ACM*, 41(3):88–95, 1998.

[Han98] Robert A. Hanneman. Introduction to social network methods, 1998.

[HKS01] Brian Hatch, George Kurtz, and Saumil Shah. *Hacking Linux Exposed*. McGraw-Hill Professional, 2001.

[IDC04] IDC. The true cost of spam and the value of antispam solutions. IDC Report, 2004.

[JS04] Jaeyeon Jung and Emil Sit. An Empirical Study of Spam Traffic and the Use of DNS Black Lists. In *Internet Measurement Conference*, Taormina, Italy, October 2004.

[Kle01a] J. Klensin. RFC 2821: Simple mail transfer protocol, April 2001.

[Kle01b] J. Klensin. Simple mail transfer protocol. IETF RFC 2821, April 2001.

[Kop04] Gene J. Koprowski. Beware of 'spoofing' scams. *United Press International*, 2004.

[KSGM03] Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 640–651, New York, NY, USA, 2003. ACM Press.

[Lin99] G. Lindberg. Anti-spam recommendations for SMTP MTAs. IETF RFC 2505, February 1999.

[Lyo05] J. Lyon. Purpoted responsible address in e-mail messages, 2005.

[MMH02] L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation. In *Proceedings of the 35th Hawaii International Conference on System Sciences*, 2002.

[Moo96] K. Moore. RFC 2047: Multipurpose internet mail extensions (mime) part three: Message header extensions for non-ascii text, November 1996.

[MR96] J. Myers and M. Rose. RFC 1939: Post Office Protocol — Version 3. IETF RFC 1939 (also STD0053), May 1996.

[Ost98] E. Ostrom. A behavioral approach to the rational choice theory of collective action. In *he American Political Science Review 92*, pages 1–22, 1998.

[Pos82] J. Postel. IETF RFC 821: Simple mail transfer protocol, August 1982.

[Ram04] B. Ramsdell. RFC 2633: S/mime version 3.1 message specification, April 2004.

[RB56] Russell and Bertrand. *Ética e política na sociedade humana.* Rio de Janeiro: Zahar, 1956.

[Rib05] U Ribeiro. *Sistemas Distribuídos - Desenvolvendo Aplicações de alta performance no Linux.* Axcel Books, 2005.

[Sch94] B Schneier. *Applied cryptography : protocols, algorithms, and source code in C.* New York, Wiley, 1994.

[SDHH98] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk E-mail. In *Learning for Text Categorization: Papers from the*

1998 Workshop, Madison, Wisconsin, 1998. AAAI Technical Report WS-98-05.

[SHF90] Eugene H. Spafford, Kathleen A. Heaphy, and David J. Ferbrache. A computer virus primer. *ACM Press*, pages 316–355, 1990.

[Str03] Prashanth Strinthan. An overview of spam handling techniques. Technical report, George Mason University, 2003.

[VM02] Wesley Vaz and Geovane Cayres Magalhães. Um modelo para derivação de relacionamentos espaciais em equivalentes semânticos relacionais. *IV Simpósio Brasileiro de GeoInformática - Caxambú, MG*, 2002.

[Wei82] P. Weil. *Relações humanas na familia e no trabalho.* ed. Petropolis: Vozes, 1982.

[WML04] Wong, Microsoft, and Lentczner. The SenderID record: Format interpretation. http://www.ietf.org/internet-drafts/draft-ietf-marid-protocol-02.txt, 2004.

[WOB76] D. R. Woods, S. D. Omerod, and M. D. Black. *Como tecer uma rede de relacionamentos e se valer dela.* São Paulo :Nacional, 1976.

[Won04] M. Wong. Sender Policy Framework (SPF): A convention to describe hosts authorized to send SMTP traffic. http://db.org/drafts/internet/mengwong/spf/00/, 2004.

[ZGT02] Cliff Changchun Zou, Weibo Gong, and Don Towsley. Code Red worm propagation modeling and analysis. In *9th ACM conference on Computer and Communications Security*, pages 138–147, 2002.

[Zim95] P. R. Zimmermann. *The Official PGP User's Guide.* The MIT Press, 1995.