VLADIMIR GERASEEV JÚNIOR

Maia Mailguard: Controle de *Spam* Baseado em uma Interface *Web*

Monografia apresentada ao Curso de Administração em Redes Linux (ARL) da Universidade Federal de Lavras como parte das exigências da disciplina Monografia para obtenção do título de Especialista em Administração em Redes Linux.

Orientador

Prof. Heitor Augustus Xavier Costa

LAVRAS
MINAS GERAIS – BRASIL
2008

VLADIMIR GERASEEV JÚNIOR

Maia Mailguard: Controle de *Spam* Baseado em uma Interface *Web*

Monografia apresentada ao Curso de Administração em Redes Linux (ARL) da Universidade Federal de Lavras como parte das exigências da disciplina Monografia para obtenção do título de Especialista em Administração em Redes Linux.

Aprovada em ____ de Abril de 2008.

Prof._____

Prof._____

_____

Prof. Heitor Augustus Xavier Costa

(Orientador)

LAVRAS
MINAS GERAIS – BRASIL

Dedico este trabalho à minha esposa
e à minha mãe, pelo apoio, carinho
e compreensão dedicados em todos
os momentos importantes de minha
vida.

Agradeço a Profª M.Sc. Maria do Carmo Soares pela dedicação e sugestões que contribuíram para a realização deste trabalho.

Ao Prof. Dr. Heitor Augustus Xavier Costa pela orientação e disponibilidade.

A equipe do Laboratório de Integração e Testes pelo apoio e recursos disponibilizados.

RESUMO

Nos dias atuais, o correio eletrônico se tornou uma ferramenta essencial para realizar comunicação ágil e eficiente, porém ela vem sendo mal utilizada, para fins de envio de propagandas, geralmente não solicitada pelos usuários, conhecidas como *spam*. Atualmente, existem técnicas que podem ser usadas para filtrar o *spam*, porém a identificação das mensagens de *spam* é um processo extremamente complicado. Caso não seja possível classificar corretamente uma mensagem, a ocorrência de um falso positivo é mais grave que um falso negativo, ou seja, visualizar uma mensagem com propaganda é melhor do que a perda de uma mensagem importante. Existem diversos filtros *antispam* que são boas soluções para evitar o recebimento do *spams*, mas esbarram no problema do falso positivo. Focando nesta contexto, este trabalho apresenta uma ferramenta que auxilia no combate ao *spam*, Maia Mailguard, e, principalmente, previne o falso positivo, pois coloca a decisão final do que deve ou não ser filtrado para o usuário decidir. Para a elaboração deste trabalho, foi feita uma revisão teórica da problemática do *spam* e das técnicas usadas para sua filtragem e foram descritos os principais recursos do Maia Mailguard e detalhados os passos necessários para sua aplicação em um ambiente de produção. Finalmente, foram colhidos os resultados do uso da ferramenta no período de 54 meses em um ambiente de produção e, através destes resultados, foi feita uma análise para obter uma conclusão significativa das medidas de eficiência, precisão e popularidade do Maia Mailguard.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1 – Introdução

Um dos primeiros casos reportados de *spam* aconteceu em maio de 1978, quando um funcionário da DEC, uma empresa americana pioneira na indústria da computação, foi contratado para fazer propaganda do novo sistema DEC 20 e enviou uma mensagem aos usuários da Arpanet sobre o referido sistema (TEMPLETON, 2003).

Com o crescimento exponencial da Internet, a proliferação de mensagens não-solicitadas (*spams*) se deu na mesma proporção. Isso é possível observar nas estatísticas coletadas pelo CERT.BR (Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil), que recebeu mais de 3.403.430 de casos de reclamações de envio de *spams* no ano de 2006 (CERT.BR, 2007)**.**

Com relação à prática abusiva do *spam*, muitas soluções de combate são possíveis, mas pode-se concluir que não existe uma receita capaz de eliminá-lo de modo permanente, muito menos uma única solução que resolva o problema dos administradores e dos usuários. Mas, existem alternativas capazes de reduzir o impacto causado. A mais usada é a filtragem, para o qual se tem uma variedade de filtros disponíveis em diversos níveis.

Os filtros permitem uma seleção nos *e-mails* recebidos, separando as mensagens não-válidas dos *e-mails* válidos. Existem três principais métodos de filtragem: i) listas de permissão; ii) lista de bloqueio; e iii)

classificação de conteúdo. A lista de bloqueio e a lista de permissão verificam o cabeçalho das mensagens recebidas em busca dos endereços IP (*Internet Protocol*), domínios ou endereços de *e-mail* que devem ser bloqueados ou permitidos. Porém, a classificação de conteúdo usa uma abordagem diferente. Esta abordagem analisa o cabeçalho da mensagem à procura de remetentes suspeitos e o conteúdo inteiro da mensagem em busca de padrões; com base na identificação destes padrões, uma análise estatística é realizada para classificar o que é ou não *spam*.

O *spam* é um incômodo realmente difícil de combater. Embora os filtros de *spam* usem com relativo sucesso heurísticas para separar o joio do trigo, os *spammers* estão um passo à frente das armas de defesa e continuamente desenvolvendo novos e engenhosos métodos para contorná-las. Um dos tipos mais eficientes de *antispam* é Bayesiano, uma excelente solução para controlar o recebimento de *spam*, porém com certa margem de erros.

1.1. Objetivo

O objetivo deste trabalho é apresentar os problemas ocasionados pelo recebimento de mensagens não solicitadas e mostrar uma solução com base em filtros *Bayesianos* (Maia Mailguard), que atua junto ao MTA (*Mail Transfer Agent*) local e configurável através de uma interface *Web*. Além disso, são apresentados os resultados obtidos após o uso da ferramenta em um ambiente de produção.

1.2. Justificativa

A importância deste trabalho é devido ao gasto desnecessário de tempo e de dinheiro resultante do recebimento de *spams* e à necessidade de testar uma ferramenta que traga resultados satisfatórios na classificação das mensagens. São muitas as ferramentas disponíveis para o combate ao *spam*, mas poucas com uma solução eficaz que resulte em uma boa classificação das mensagens recebidas. Este trabalho apresenta uma ferramenta com uma abordagem diferente de outras, pois o processo de classificação das mensagens é assistido pelo usuário final.

Assim, este trabalho abordará a questão do *spam* de modo geral, mas focará em apresentar uma ferramenta para auxiliar na gestão das mensagens eletrônicas, permitindo o administrador gerenciar, satisfatória e intuitivamente, o sistema de correio eletrônico.

1.3. Metodologia

Este trabalho consistiu inicialmente de levantamentos bibliográficos específicos na área de mensagens eletrônicas não solicitadas, *spam*, e comparação dos métodos e ferramentas usados para o seu combate, visando apresentar uma breve discussão sobre o assunto. Além disso, são apresentados os recursos da ferramenta escolhida para este trabalho, mostrando suas principais características e os procedimentos necessário para sua instalação e operação, tendo por base uma implementação em um ambiente de produção. Finalmente, foi feita uma coleta dos dados estatísticos com base no uso diário da ferramenta, através dos quais foi

possível fazer análise da eficácia da ferramenta.

1.4. Organização do Trabalho

Do ponto de vista estrutural, este trabalho é constituído por sete capítulos, sendo o primeiro reservado a introdução, objetivos do trabalho, escopo e justificativa do assunto. O Capítulo 2 apresenta a fundamentação teórica, sendo o principal foco a caracterização do *spam,* as técnicas e as ferramentas consagradas usadas para seu combate. Os Capítulos 3, 4 e 5 apresentam a ferramenta Maia Mailguard, como é feita sua instalação, configuração e uso de suas principais funções. O Capítulo 6 apresenta dados estatísticos colhidos após o uso da ferramenta em um ambiente de teste e uma análise destes dados, visando obter uma conclusão da aplicabilidade da ferramenta. O Capítulo 7 apresenta as considerações finais do trabalho.

Capítulo 2 – Fundamentação Teórica

2.1. Mensagens Eletrônicas Não Solicitadas (*Spam*)

2.1.1. Caracterização

A definição encontrada na Cartilha de Segurança para Internet do CERT.BR*:* *spam* é o termo usado para se referir aos *e-mails* não solicitados, que geralmente são enviados para um grande número de pessoas. Quando o conteúdo é exclusivamente comercial, este tipo de mensagem é referenciada como UCE (*Unsolicited Commercial E-mail*).

O termo *spam* é a marca de um presunto enlatado americano, não tendo qualquer relação com o envio de mensagens eletrônicas não solicitadas, exceto pelo episódio da série de filmes de comédia de Monty Python, onde alguns Vikings desajeitados pediam repetidas vezes o referido presunto (TEMPLETON, 2003).

Na Internet, *spam* é um ato considerado abusivo e se refere ao envio de um grande volume de mensagens não solicitadas, ou seja, o envio de mensagens indiscriminadamente a vários usuários, sem que estes tenham requisitado tal informação.

O conteúdo do *spam* pode ser propaganda de produtos e de

serviços, pedido de doações para obras assistenciais, correntes da sorte, propostas de ganho de dinheiro fácil, boatos desacreditando o serviço prestado por determinada empresa, dentre outros.

Teixeira (2001) faz uma comparação entre *spam* e a antiga mala direta, com a seguinte afirmação:

> As pessoas questionam se *spam* não segue a ordem natural das coisas, afinal seria o mesmo caso dos vários folhetos de propaganda distribuídos na rua, enviados pelo correio tradicional ou dos serviços de *telemarketing*. Não, *spam* não pode ser classificado na mesma categoria que tais serviços, a diferença básica é quem paga a conta.

Pode-se concluir que existe grande diferença entre *spam* e as propagandas distribuídas nos panfletos e há algumas diferenças de opinião encontradas na literatura sobre o que é *spam*. Sendo assim, para este trabalho, *spam* é definido como o termo usado para mensagens eletrônicas enviadas em massa para fins comerciais.

2.1.2. Problemas Causados pelo *Spam*

Como relatado pelo CERT.BR (2006), *spam* pode afetar os usuários do serviço de correio eletrônico de diversas formas. Alguns exemplos são:

- **Não recebimento de *e-mails*:** Boa parte dos provedores de Internet limita o tamanho da caixa postal do usuário no seu servidor. Caso o número de *spams* recebidos seja grande, ele corre o risco de ter sua caixa postal lotada com mensagens não solicitadas. Se isto ocorrer, ele não receberá *e-mails* e, até que possa liberar espaço em sua caixa postal, as mensagens recebidas serão devolvidas ao remetente. Outro problema é o usuário deixa de receber *e-mails* nos casos em que regras *antispam* ineficientes

são utilizadas, por exemplo, classificando como *spam* mensagens legítimas;

- **Gasto desnecessário de tempo:** Para cada *spam* recebido, o usuário gasta um determinado tempo para ler, identificar o *e-mail* como *spam* e removê-lo da caixa postal;
- **Aumento de custos:** Independente do tipo de acesso a Internet utilizado, quem paga a conta pelo envio do *spam* é quem o recebe. Por exemplo, para um usuário que utiliza acesso discado a Internet, cada *spam* representa alguns segundos a mais de ligação que ele pagará;
- **Perda de produtividade:** Para quem usa *e-mail* como ferramenta de trabalho, o recebimento de *spams* aumenta o tempo dedicado à sua leitura, além de existir a chance de mensagens importantes não serem lidas, serem apagadas por engano ou lidas com atraso;
- **Conteúdo impróprio ou ofensivo:** Como a maior parte dos *spams* é enviada para conjuntos aleatórios de endereços de *e-mail*, é provável que o usuário receba mensagens com conteúdo que julgue impróprio ou ofensivo;
- **Prejuízos financeiros causados por fraude:** *Spam* tem sido amplamente utilizado como veículo para disseminar esquemas fraudulentos, que tentam induzir o usuário a acessar páginas clonadas de instituições financeiras ou a instalar programas maliciosos, projetados para furtar dados pessoais e financeiros. Esse tipo de *spam* é conhecido como *phishing/scam*. O usuário

pode sofrer prejuízos financeiros, caso forneça as informações ou execute as instruções solicitadas nesse tipo de mensagem fraudulenta.

2.1.3. Dificuldade de Detecção do *Spam*

Uma das características que torna difícil a detecção automática de *spam* é a capacidade de adaptação das pessoas que os distribuem na Internet (os *spammers*) (REAL, 2003).

Para confundir os filtros de *spam*, os *spammers* podem enviar mensagens que assumem diferentes formas: i) imagens em arquivos anexos com as propagandas inseridas dentro das imagens; ii) mensagens com frases curtas apenas com *links* para *sites* na *Web*, onde está a propaganda em si; iii) palavras obscuras entre textos, entre outras.

A dificuldade de detecção resulta em dois tipos de classificações errôneas feitas pelos sistemas de filtragem de mensagens:

- **Falsos Negativos:** São mensagens com conteúdo de propaganda não corretamente classificado como *spam.* Este tipo de classificação errônea, quando em uma pequena escala, não é tão grave, pois só gera o desconforto do usuário receber uma mensagem não solicitada;
- **Falsos Positivos**: São mensagens válidas erroneamente classificadas como *spam*. O falso positivo é mais grave que o falso negativo, porque pode resultar na perda de mensagens válidas do usuário, classificando-se como um potencial problema para o

administrador.

2.2. Técnicas de Combate ao *Spam*

2.2.1. Listas de Bloqueio / Permissão

Ferramentas baseadas em lista de bloqueio são conhecidas como lista negra (*blacklist*) e a de permissão como lista branca (*whitelist*). Os dois tipos de listas verificam o cabeçalho das mensagens recebidas e identificam os endereços IP, os domínios ou os endereços de *e-mail* que devem ser bloqueados ou permitidos, respectivamente (FABRE, 2005).

As listas negras são alimentadas das seguintes formas: i) automática, através de bancos de dados mantidos por entidades responsáveis pela lista, baseada em denúncias enviadas; e ii) manual, possibilitando ao usuário indicar que uma mensagem é *spam* e o remetente da mensagem ser adicionado à lista-negra.

A filtragem de *spam*, baseada em listas, tem grande precisão contanto que exista uma constante atualização das listas de bloqueio e permissão; porém, as listas de bloqueio tendem a ficar ineficazes à medida que os *spammers* protegem-se dessa filtragem, mudando freqüentemente de endereço eletrônico e provedor e utilizando endereços de *e-mails* falsos.

Nas listas de permissão, são inseridos os endereços eletrônicos de indivíduos ou domínios que são considerados confiáveis. Assim que é feito o recebimento de uma mensagem, as duas listas são consultadas. Se

a presença do endereço de origem for detectada na lista de bloqueio, a mensagem é imediatamente classificada como *spam*. Por outro lado, caso o endereço esteja na lista de permissão, a mensagem é aceita. Estas listas inicialmente eram alimentadas manualmente, ou seja, cada usuário inseria ou removia os endereço dos servidores de mensagem da lista.

Um aprimoramento do processo foi o uso de um sistema centralizado, onde existe o papel de um gerenciador central, que controla a entrada e a saída de endereços das listas. Isto motivou a criação de um dos primeiros sistemas que implementa o uso de listas distribuídas, *Real-time Blackhole List* (RBL).

Através do RBL, é possível, de uma forma automática, consultar a lista dos endereços IP de servidores usados para o envio de *spam*. Porém, a grande desvantagem é estes endereços necessitarem de ser adicionados manualmente.

Uma nova proposta lançada foi o ORBS (*Open Relay Behavior-modification System*), cuja principal diferença do RBL é a execução automática de testes, visando identificar servidores que permitem o envio de mensagens sem controle, os chamados *open-relay*. Quando estes servidores são identificados, o endereço IP é adicionado de forma automática na lista de bloqueio e as mensagens originadas são imediatamente classificadas como *spam*. No entanto, para que seja feita remoção dessa lista, é necessário um processo manual, ou seja, o administrador do domínio bloqueado deve contactar o administrador da entidade responsável pela gerência da lista.

A consulta a estas listas distribuídas é feita através do uso do protocolo DNS (*Domain Name System*). Sendo assim, elas são chamadas de modo genérico de DNS-BLs (*Domain Name Sytem Black Lists*).

2.2.2. Listas Cinzas (*Greylist*)

Segundo HARRIS (2003), a denominação "listas cinzas" é devido ao fato de ser uma mistura entre listas negra e branca, com a maior parte de sua manutenção automática.

A eficácia da lista cinza baseia-se no fato de *spams* não serem retransmitidos quando o servidor notifica algum erro no envio das mensagens. Esse comportamento é adotado para aumentar a eficácia do envio de mensagens pelos *spammers*, uma vez que, se um determinado endereço reportou um erro quando a mensagem foi enviada, um *spammer* não enviará a mensagem uma próxima vez, para não perder tempo (TAVEIRA *et al,* 2006).

Endereços para os quais a mensagem não foi entregue na primeira vez podem ser excluídos das listas dos *spammes*, assumindo que esses não existem ou são inválidos. O funcionamento da lista cinza se baseia em dois tipos de listas: i) lista de permissão; e ii) lista de bloqueio.

Após o recebimento de uma mensagem, o servidor de mensagens faz uma consulta para verificar se o endereço de origem está na lista de permissão. Caso a resposta seja afirmativa, a mensagem é encaminhada para o destinatário diretamente. Todavia, quando o par destinatário/origem não se encontra na lista de permissão, o servidor

reporta um erro para o remetente da mensagem informando que houve um problema temporário no envio da mensagem.

Servidores que realmente seguem a RFC 821 (POSTEL, 1982), que diz respeito ao envio de mensagens eletrônicas através do protocolo SMTP (*Simple Mail Transfer Protocol*), tentam enviar novamente a mensagem depois de um determinado tempo. Após o servidor de mensagens enviar a mensagem de erro temporário, a tripla endereço de origem/destinatário/endereço do servidor de origem são adicionados à lista cinza junto com a informação do tempo em que esta entrada foi adicionada.

A informação relativa ao tempo que a entrada foi adicionada é importante para estabelecer um período de quarentena, para que *spammers* que usam programas de envio automático não tentem reinviar a mensagem no momento seguinte que foi recebido o erro temporário do destinatário. Neste caso, a mensagem será rejeitada novamente com um erro temporário.

Caso a mensagem seja enviada em um tempo maior que o período de quarentena, ela é encaminhada para o destinatário e o endereço do remetente e o seu endereço do servidor de mensagem são adicionados a uma lista branca, permitindo que, em futuras comunicações entre o destinatário e o remetente, o endereço do servidor de mensagens do remetente não seja adicionado a lista cinza. Esta entrada do remetente/servidor de mensagens é expirada da lista branca depois de algum tempo para evitar manter registros antigos por muito tempo.

É importante manter em uma lista de permissão, os endereços IP de servidores que não devem ser filtrados no processo de *greylisting*, ou porque são máquinas consideradas confiáveis (da própria rede, de redes conhecidas, etc) ou porque seus servidores de mensagens não conseguem tratar corretamente erros temporários.

2.2.3. Filtros Bayesianos

Fabre (2005) define:

> Os filtros *Bayesianos* usam diferente abordagem dos anteriores que fazem classificação baseados no cabeçalho da mensagem, ele faz uma classificação baseada no conteúdo da mensagem. Este método analisa o conteúdo da mensagem, isto é, o texto completo em busca de padrões suspeitos e, com base na identificação de determinados padrões, utiliza estatística e probabilidade para fazer uma classificação do que é ou não *spam*.

A abordagem do filtro *Bayesiano* é mais abrangente do que a dos filtros baseados em listas, pois, mesmo que o *spammer* consiga usar mais de um endereço de origem para o envio de mensagens, o texto das mensagens pode ser usado para classificar a mensagem como *spam* ou válida. Esta abordagem não é tão precisa por ser baseada em uma análise estatística, possivelmente suscetível a erros; assim, poderá ter o risco da classificação de uma mensagem válida como *spam* (falso-positivo) ou de um *spam* como mensagem válida (falso-negativo).

Um dos principais aspectos do filtro *bayesiano* é seu processo de aprendizagem. No início da sua utilização, é complexo definir parâmetros e padrões de texto para que seja possível determinar o que é ou não *spam*. Sendo assim, as ferramentas que trabalham com o filtro providenciam meios para que o usuário classifique manualmente as mensagens como

spam ou válidas. Quanto maior o número de mensagens classificadas, maior será a precisão do filtro automático.

Outro ponto que vale salientar é o correto funcionamento do filtro depender fortemente da correta classificação do usuário, que poderá classificar erroneamente uma mensagem válida como *spam* ou vice-versa, inserindo uma inconsistência na análise estatística feita pelo filtro.

2.2.4. Correio Identificado por Chaves de Domínio (*Domain Key Identifyed Mail*)

Segundo Nunes (2006), *Domain Key Identifyed Mail* (DKIM) consiste em um método para efetuar uma assinatura das mensagens eletrônicas com uma chave pública, certificada ou não, para garantir a autenticidade do remetente.

O *Domain Key Identifyed Mail* é um método *antispam* desenvolvido em conjunto entre as empresas Yahoo e Cisco, que baseia-se no uso de chave pública para autenticar o domínio do remetente (ALLMAN *et al*., 2007). Este metódo é utilizado para evitar a fraude nos endereços dos remetentes, prática largamente utilizada por *spammers*, que enviam mensagens colocando um usuário legitimo no remetente para que essas sejam enviada em nome de um determinado domínio.

Para o funcionamento do DKIM, é necessário ter um estrutura de chave pública e privada, colocar uma chave privada no MTA do remetente, para que seja feita a verificação com a chave pública e disponibilizar a chave pública através do servidor DNS. A cada mensagem enviada pelo MTA, é feita sua assinatura pela chave privada e

essa assinatura é incluída como um campo no cabeçalho da mensagem.

Quando o MTA do destinatário recebe uma mensagem, ele verifica o domínio contido no campo *from* da mensagem e procura obter a chave pública deste domínio, publicada através do DNS, para confirmar se a mensagem foi mesma originada deste domínio e se o conteúdo da mensagem não foi alterado. Uma vez obtida a chave pública através do servidor DNS, é feita uma comparação da assinatura digital da mensagem com a assinatura enviada no cabeçalho da mensagem. Após esta verificação, o MTA do destinatário pode ter as seguintes decisões: i) a assinatura é válida, significando que a mensagem vem mesmo do domínio indicado no campo *from*, então esta mensagem pode passar por outros processos para o reconhecimento de *spam*; ii) a assinatura não é válida. Neste caso, o MTA do destinatário pode recusar a mensagem; e iii) o domínio do remetente não possui um registro da implementação do DKIM. Neste caso, o DKIM não é usado como um critério de verificação de autenticidade da mensagem.

Maior atenção deve ser dada na implementação de sistemas que usam este método pois o custo computacional no servidor MTA do remetente pode ser elevado quando houver um grande volume de mensagens, em vista da necessidade de efetuar uma assinatura eletrônica para cada mensagem enviada.

2.2.5. Política de verificação de remetente (*Sender Policy Framework*)

O *Sender Policy Framework – SPF,* assim como o DKIM, é um

método que tem como objetivo principal garantir a autenticidade do remetente das mensagens em um sistema de correio eletrônico.

Através deste método, um administrador de um domínio pode especificar quais servidores tem direito de enviar mensagens em nome de seu domínio, estabelecendo políticas de envio publicadas através de um registro no servidor DNS (WRONG *et al*., 2006). Essas informações são posteriormente acessadas quando um servidor de correio eletrônico recebe uma mensagem com o endereço do domínio em questão no remetente e, através de uma consulta ao servidor DNS, pode validar se a mensagem foi realmente gerada pelo domínio identificado no remetente. Resumidamente, o SPF funciona do seguinte modo:

- Uma política SPF é estabelecida pelo administrador de um domínio, onde, entre outras informações, ele relaciona quais servidores podem enviar mensagens eletrônicas em nome de seu domínio;
- Quando um servidor de correio eletrônico recebe uma mensagem, esta é verificada, que faz uma consulta ao servidor DNS do domínio contido no campo *from* da mensagem, em busca de uma política SPF para aquele servidor;
- Caso exista uma política, é feita verificação se o servidor emissor da mensagem está autorizado a enviar uma mensagem em nome do domínio consultado;
- Se o servidor estiver autorizado, a mensagem é recebida e pode passar por outros processos de verificação de *spam*. Caso

contrário, a política SPF pode definir qual ação será tomada para esta mensagem, entre elas: i) rejeitar a mensagem; ii) aceitar a mensagem, mas passá-la por processos rigorosos de verificação de *spam*; ou iii) rejeitar a mensagem com um código SMTP de erro temporário. Estas ações, de acordo com a especificação do SPF, são apenas recomendações feitas na política SPF, o cliente não é obrigado a tomar a ação sugerida (WRONG *et al*., 2006).

A política SPF não é uma técnica de combate ao *spam*, pois, no final de seu processamento, ela não define se a mensagem é ou não *spam*, mas se o domínio do remetente foi ou não forjado. Porém, como a forjamento de endereços é uma prática comum de *spammers*, está técnica pode ser usada como um instrumento que ajuda a diminuir a quantidade de *spams* recebidos.

2.3. Ferramentas para Controle de *Spam* e Vírus

2.3.1. SpamAssassin

O SpamAssassin[1] é um software livre para filtrar *spam* escrito em Perl, que interage com o servidor de *e-mail* (MTA), ele suporta diferentes plataformas de hardware como: Linux, Solaris, FreeBSD, dentre outros.

Ele aplica uma série de testes nas mensagens, procurando

1 *SpamAssassin* - http://spamassassin.apache.org/

características que as identifiquem como *spam* e designa uma pontuação para cada característica encontrada. Quando a mensagem atinge uma nota de borda designada pelo usuário, ela é classificada como *spam*. Dentre desses testes executados pelo SpamAssassin, tem-se:

- **Análise de Cabeçalho**: Os *spammers* custumam usar técnicas para mascarar sua identidade e esconder o servidor de origem das mensagens. O SpamAssassin tenta identificar indícios do uso desses truques;
- **Análise de Texto**: As mensagens com *spam* tem um estilo próprio, geralmente propagandas tentando convencer que a venda de algum produto ou a compra de um remédio milagroso são as soluções para seus problemas. O SpamAssassin tenta identificar as mensagens recebidas com base no estilo das mensagens com *spam*, verificando a ocorrência de palavras ou textos usados nas mensagens com *spam*;
- **Listas de Bloqueio**: O SpamAssassin suporta consulta a listas de bloqueio distribuídas como mail-abuse.org e ordb.org e pode ignorar mensagens vindas de domínios reconhecidamente abusados por *spammers*;
- **Utilização de Filtros *Bayesianos***: A principal vantagem do SpamAssassin é o uso da filtragem *bayesiana*, pois utiliza estatísticas e probabilidades da ocorrência de padrões que classifiquem a mensagem como válida ou *spam*. Para funcionar corretamente, é necessário o treino do filtro por uma quantidade

razoável de mensagens (o filtro fica ativo quando treinado pelo menos com mais de 200 mensagens) válidas e de *spams*.

A Figura 2.1 apresenta um exemplo de pontuação aplicada em uma mensagem pelo SpamAssassin.

Pontuação	Regra Acionada	Explicação
5.000	BAYES_99	Probabilidade bayesiana de ser spam é de 99%
4.500	RAZOR2_CHECK	Listado na lista negra Razor2
0.300	BR_PERDER_TEMPO	Fala sobre não perder tempo
0.100	BR_CURSO_SUBJ	Curso no subject

Total: 9,900

Figura 2.1 – Exemplo de Pontuação do SpamAssassin

Neste exemplo, o filtro bayesiano determinou que esta mensagem tem 99% de chance de ser *spam*, a mesma mensagem também foi reconhecida na lista distribuída de bloqueio Razor. Além disso, foi verificada na mensagem o texto "não perder tempo", geralmente usado em *spams*, e a palavra "curso" no assunto da mensagem.

Para cada regra que o SpamAssassin reconheceu a característica na mensagem, foi dada uma pontuação, no total 9.9, consultada a nota limite de corte e se o total for maior que a nota limite, a mensagem é classificada como *spam*.

2.3.2. Clamav

O *Clamav*[1] Antivírus é um pacote sobre licença GPL (*General Public License*) escrito especificamente para análise de mensagens em servidores de correio eletrônico. Este pacote contém várias utilidades como suporte a paralelismo (*multi-threaded*) flexível e escalável, um utilitário de análise para linha de comandos e uma ferramenta avançada para atualização automática da base de dados. O núcleo do pacote é um motor anti-viral disponível como biblioteca. A seguir, são listadas as suas principais funções:

- Analisador de linha de comandos;
- Serviço rápido e com paralelismo;
- Interface milter para o sendmail;
- Atualizador avançado para a base de dados com suporte para *scripts* e assinaturas digitais;
- Biblioteca C para análise viral;
- Múltiplas atualizações diárias da base de dados de vírus;
- Suporte incorporado a executáveis ELF e arquivos executáveis portáveis comprimidos com UPX, FSG, Petite, NsPack, wwpack32, MEW, Upack entre outros;
- Suporte incorporado para formatos de documentos comumente usados, como Microsoft Office e MacOffice, HTML, RTF e PDF.

1 Clamav - http://www.clamav.net/

2.3.3. Amavisd-new

O Amavisd-new[1] utiliza uma gama de programas antivírus para a detecção de vírus nas mensagens recebidas no servidor, por exemplo: *Clamav*, FRISK F-Prot[2], Mcafee[3], entre outros. Além disso, é feita a descompressão de mensagens com arquivos compactados nos formatos: compress, gzip, bzip, bzip2, zip, rar, arj, entre outros, e verificado o conteúdo do arquivo compactado.

Ele também faz verificação de extensões proibidas (Ex: exe, scr), tipos MIME (*Multipurpose Internet Mail Extensions*) proibidos (Ex. x-msdos-program) e uma característica interessante é a possibilidade de desabilitar individualmente cada tipo de checagem feito pelo amavisd-new para um endereço eletrônico específico ou um domínio.

O Amavisd-new integra as ferramentas de verificação de vírus e de reconhecimento de *spam*, através do seguinte processo: o MTA externo recebe a mensagem na porta 25 e envia a mensagem para o filtro de conteúdo (amavisd-new) que atua como um SMTP na porta 10024. O Amavisd-new invoca o *Clamav* e SpamAssassin para as verificações de vírus e *spam*, respectivamente. Depois de feita as verificações, caso a mensagem não tenha sida classificada como *spam* ou não contenha vírus, o Amavisd-new faz a re-injeção da mensagem no Postfix na porta 10025 (MTA interno) onde a mensagem é entregue ao destinatário. Caso contrário, a mensagem pode tomar as seguintes direções, de acordo com a

1 amavisd-new - http://www.ijs.si/software/amavisd/
2 F-Prot – http://www.f-prot.com/
3 Mcafee - http://www.mcafee.com/

configuração do Amavisd-new: i) ser aceita; ii) ser marcada contendo vírus ou *spam*; e iii) ser rejeitada e enviada para um área de quarentena no sistema, geralmente uma conta de *e-mail*, onde poderá ser recuperada posteriormente (Figura 2.2).

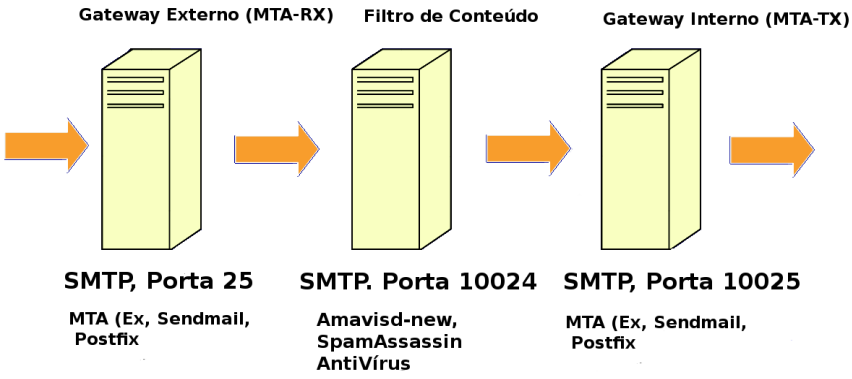

Figura 2.2 – Funcionamento do Amavisd-new

Capítulo 3 – Maia Mailguard

O Maia Mailguard[1] é uma interface *Web* baseada no popular filtro de mensagens amavisd-new e no *SpamAssasin*. Escrito em Perl e PHP, o Maia Mailguard permite ao usuário o controle de como suas mensagens serão processadas pelos antivírus e filtros de *spam* e, ao mesmo tempo, capacita o administrador a configurar padrões e limites para o seu *site*.

O Maia Mailguard é um projeto de Robert Le Blank, que inicialmente teve o objetivo do desenvolvimento de um *front-end* para o Amavis-new com uma interface acessível via *Web*. Na versão 20030616 do Amavisd-new implementada por Mark Martinec, foi introduzida uma nova característica, a possibilidade do armazenamento das configurações de cada usuário do Amavisd-new em bancos de dados. Nessa época, foram feitas algumas sugestões na lista de usuários para a criação de uma interface em PHP para gerenciar as configurações dos usuários que estavam armazenadas no banco de dados. Assim, surgia a primeira versão do Maia Mailguard que proporcionou a possibilidade de cada usuário do sistema configurar estas opções através de uma interface *Web*.

Porém, com o amadurecimento do Maia Mailguard, surgiram aprimorações que possibilitaram o gerenciamento de outras funções proporcionadas pelo Amavisd-new, como o gerenciamento das mensagens em quarentena. A idéia é os usuários serem capazes de acessar, visualizar e gerenciar mensagens na quarentena do sistema

(BLANK, 2004).

Isto traria uma solução para um dos maiores problemas encontrados pela comunidade de combate ao *spam*, uma vez que, enquanto os diferentes filtros de mensagem se aprimoravam, o medo de falsos positivos era grande, então um sistema que descartasse as mensagens classificadas como *spam* seria uma solução que a maioria das organizações não aceitavam. Com esta preocupação, os sistemas de controle de *spam* passaram a armazenar no servidor as mensagens da quarentena, enviando para uma conta do sistema, que seria usada para uma posterior análise feita pelo administrador, para a recuperação de um falso positivo relatado por um usuário.

Este esquema ainda é usado por muitas organizações, mas traz a tona dois problemas: i) a privacidade do usuário que precisa a todo o momento ser violada para a verificação das mensagens em quarentena; e ii) a constante necessidade da intervenção do administrador.

Focando este problema, Robert Le Blank desenvolveu uma solução amigável ao usuário que coloca o gerenciamento das mensagens em quarentena em suas mãos, além de prover ao usuário o controle de como suas mensagens serão processadas pelos programas antivírus e filtro de *spam*. Ao mesmo tempo, a solução permite ao administrador configurar limites de borda, onde é possível estabelecer padrões e limites para o domínio das mensagens.

3.1. Licença

O Maia Mailguard tem uma licença[1] baseada nas licenças de software de código aberto (*open source*) e é basicamente constituida dos seguintes termos: i) o código fonte pode ser livremente modificado e distribuído desde que os termos da licença sejam incluídos de alguma forma junto com o código (em forma de manuais ou na interface *Web)*; ii) se o Maia Mailguard for distribuído dentro de algum código binário, este deve incluir os termos da licença junto com a documentação ou com qualquer material distribuído em conjunto com o software; iii) se for fornecida a documentação para o usuário final esta deve incluir a frase "software desenvolvido por Robert Le Blank".

Adicionalmente, são aplicadas pelo menos uma das três seguintes restrições:

- Deve se colocado no topo das páginas da interface *Web o* Logo do Maia Mailguard (Figura 3.1), com uma ligação para o *site* principal do projeto Maia Mailguard;
- Deve ser incluída a imagem "equipado com Maia Mailguard" (Figura 3.2), em qualquer posição nas páginas da interface *Web*, com uma ligação para o *site* principal do projeto Maia Mailguard;
- Poderá ser pago um valor definido pelo autor do projeto para se obter uma licença onde é possível remover o símbolo do Maia.

1 Licença do MaiaMailguard - http://www.maiamailguard.com/license.php

Figura 3.1 – Logo do Maia Mailguard

Figura 3.2 – Equipado com Maia Mailguard

3.2. Traduções

A interface *Web* do Maia Mailguard tem nativamente versões disponíveis nos idiomas inglês, português de portugal, espanhol e francês. Uma nova tradução da interface para o idioma português do Brasil está em processo de desenvolvimento pelo autor deste trabalho.

3.3. Funcionamento

A filtragem de conteúdo executada pelo Maia Mailguard é uma derivação do Amavisd-new, preservando a maior parte de suas características.

Como o Amavisd-maia é derivado do Amavisd-new, pode-se fazer uma comparação do gerenciamento do processo de filtragem feito por

estes dois programas.

Com o Maia Mailguard, o processo é um pouco diferente do Amavisd-new, a mensagem é recebida pelo MTA externo e enviada ao filtro de conteúdo, porém o filtro de conteúdo consulta um banco de dados com as configurações de cada usuário, sendo possível determinar quais as verificações (*spam*, vírus, anexos proibidos) serão feitas para aquele usuário específico ou para seu domínio. Além disso, é verificada uma lista específica para cada usuário do sistema, se o endereço de origem está inserido na lista de permissão ou de bloqueio, fazendo com que a mensagem seja aceita ou rejeitada automaticamente.

Quando a mensagem é bloqueada, ela é armazenada integralmente no banco de dados, onde é possível, através da interface *Web*, o usuário visualizar e recuperar sua mensagem. Uma característica importante nesse ponto é o usuário auxiliar na precisão do filtro à medida que usa interface *Web* para indicar as mensagens válidas ou *spams* que ele recebeu, confirmando para o sistema a sua classificacação. Este processo é importante principalmente nos casos de falsos positivos e falsos negativos, pois, quando é feita a classificação pelo usuário, a mensagem é treinada no filtro *Bayesiano*, que aumenta sua estatística sobre as mensagens recebidas no sistema e evita que o sistema cometa os mesmos erros (Figura 3.3).

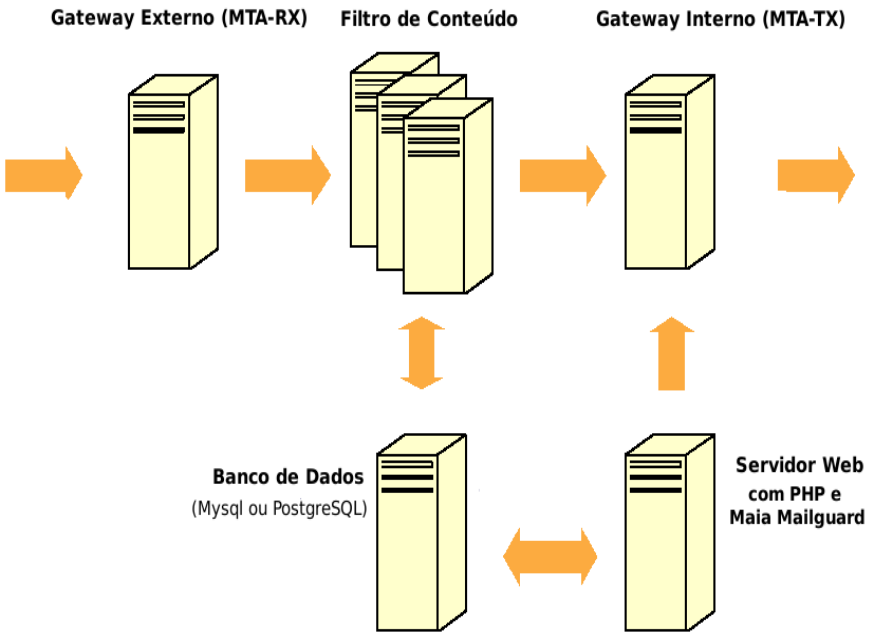

Figura 3.3 – Funcionamento do Maia Mailguard

Capítulo 4 – Instalação e Configuração do Maia Mailguard

O sistema operacional utilizado para a instalação do servidor foi o Linux, com a distribuição Gentoo. Como a instalação dos pacotes pode ser feita de mais de uma maneira, ela não será abordada; assim, pressupõe-se que os pacotes descritos (Apache, PHP, PostgreSQL, SpamAssassin, Perl e Postfix) estejam devidamente instalados e operacionais.

4.1. Amavisd-maia

Para a instalação com sucesso Amavisd-maia, são necessários alguns módulos do Perl. A Tabela 4.1 apresenta estes módulos e a versão mínima requerida. Os arquivos fontes do Maia Mailguard podem ser encontrados em (MAIA, 2007), onde estão embutidos os arquivos fonte com a versão modificada do Amavisd-new (Obs: O módulo DBD::Pg é usado para acesso ao banco de dados PostgreSQL, caso seja utilizado o banco de dados MySQL será usado o módulo DBD::mysql.). Além dos módulos Perl, é necessário instalar:

- file, versão >=4.16
- BerkeleyDB, versão >= 0.26
- DBI, versão >=1.50
- DB_File, versão >= 1.810
- URI, versão >= 1.31
- Template, versão >= 2.14

Tabela 4.1 – Módulos Perl – Pré-Requisitos para o Maia Mailguard

Módulo Perl	**Versão**	**Pacote Gentoo**
Archive::Tar	>=1.10	Archive-Tar
Archive::Zip	>=1.16	Archive-Zip
Compress::Zlib	>=1.41	Compress-Bzip2
Convert::TNEF	>=0.17	Convert-TNEF
Convert::UUlib	>=1.06	Convert-UUlib
Crypt::Blowfish	>=2.09	Crypt-Blowfish
Crypt::CBC	>=2.12	crypt-cbc
Data::UUID	>=0.13	Data-UUID
DBD::Pg	>=1.31	DBD-Pg
Digest::MD5	>=2.33	perl-Digest-MD5
Digest::SHA1	>=2.07	Digest-SHA1
File::Spec	>=3.01	File-Spec
HTML::Parser	>=3.35	HTML-Parser
HTTP::Date	>=1.46	libwww-perl
IO::Stringy	>=2.109	IO-stringy
IO::Zlib	>=1.03	IO-Zlib
LWP::UserAgent	>=2.032	libwww-perl
Mail::Address	>=1.64	Email-Address
Mail::Internet	>=1.64	MailTools
Mail::SpamAssassin	>=3.1.1	SpamAssassin
MIME::Base64	>=3.05	MIME-Base64
MIME::Parser	>=5.420	MIME-tools
MIME::QuotedPrint	>=3.03	MIME-Base64
Net::DNS	>=0.57	Net-DNS
Net::Server	>=0.93	net-server
Net::SMTP	>=2.29	perl-libnet
Pod::Usage	>=1.16	PodParser
Time::HiRes	>=1.61	perl-Time-HiRes
Unix::Syslog	>=0.99	Unix-Syslog

A instalação dos módulos e dos programas citados foram feitos

através dos pacotes disponíveis no Gentoo Linux, de acordo com o nome do pacote apresentado na Tabela 4.1, através do seguinte comando:

[root]# emerge nome_pacote

Após a instalação dos pacotes, é necessária a criação de um usuário e de um grupo para o processo do *daemon* Amavisd-maia. Assim, será criado um usuário e um grupo com o nome "amavis".

[root]# useradd amavis
[root]# passwd amavis
[root]# groupadd amavis

O arquivo Amavisd-maia na distribuição do Maia Mailguard é uma versão modificada do Amavisd-new 2.2.1. Este arquivo pode ser obtido junto com os arquivos de instalação do Maia em (Maia, 2007).

Após a descompactação dos arquivos de instalação do Maia Mailguard, devem ser feitas a cópia e a configuração das permissões de execução para o arquivo de execução do Amavisd-maia, através dos seguintes comandos:

[root]# cp amavisd-maia /usr/sbin/
[root]# chown root /usr/sbin/amavisd-maia
[root]# chmod 755 /usr/sbin/amavisd-maia

4.1.1. Configuração do Banco de Dados

O Maia Mailguard utiliza um banco de dados para armazenar as regras do SpamAssassin, as configurações do usuário e as mensagens em quarentena. Sendo assim, é necessária a criação do banco de dados antes de iniciar o *daemon* do Amavisd-maia. Os bancos de dados suportados

pelo Maia Mailguard nativamente são o PostgreSQL e o MySQL. Para este trabalho, foi escolhido o PostgreSQL; assim, serão mostrados os procedimentos para a criação do banco de dados com ele.

Primeiro, deve ser criado o usuário que o Amavisd-maia usará para efetuar conexão ao banco de dados:

[root]# psql template1

template1=# CREATE USER maia WITH PASSWORD 'senha' NOCREATEDB NOCREATEUSER;

Depois de inserido o usuário "maia", deve-se criar um banco de dados com o nome "maia" configurado para as permissões para o usuário maia:

template1=# CREATE DATABASE maia WITH OWNER = maia;

Após a criação do banco de dados, geram-se as tabelas do banco de dados. O arquivo contendo os comandos SQL necessários para a criação da estrutura inicial é o maia-pgsql.sql que pode ser obtido junto aos arquivos de instalação do Maia Mailguard. Para o PostgreSQL, o seguinte comando criará as tabelas iniciais do banco de dados:

[root]# psql -f maia-pgsql.sql maia

4.1.2. Instalação dos *Scripts* de Manutenção

O Maia Mailguard baseia-se em uma coleção de *scripts* em Perl que auxiliam na atualização do banco de dados com as configurações do SpamAssassin, além de outras funções. Estes *scripts* são instalados no mesmo diretório do usuário amavis:

[root]# mkdir /var/amavisd/maia

[root]# mkdir /var/amavisd/maia/scripts
[root]# mkdir /var/amavisd/maia/templates

Voltando aos arquivos da instalação do Maia Mailguard, deve-se copiar os arquivos dos sub-diretórios script e templates para o diretório /var/amavisd/maia/scripts e /var/amavisd/maia/scripts, respectivamente. Depois, deve-se configurar o dono dos diretórios para o usuário do sistema amavis, com as permissões de acesso corretas:

[root]# chown -R amavis /var/amavisd/maia
[root]# chgrp -R amavis /var/amavisd/maia
[root]# chmod 640 /var/amavisd/maia/templates/.tpl*
[root]# chmod 750 /var/amavisd/maia/scripts/.pl*

Antes de qualquer *script* ser utilizado, deve-se copiar o arquivo maia.conf.dist, que contém as configurações dos parâmetros usados nos *scripts* de manutenção, para o diretório /etc/maia.conf e configurar algumas informações básicas, por exemplo, como deve ser feita a conexão ao banco de dados:

[root]# cp maia.conf.dist /etc/maia.conf
[root]# chown amavis /etc/maia.conf
[root]# chgrp amavis /etc/maia.conf
[root]# chmod 640 /etc/maia.conf

As linhas a serem modificadas no arquivo maia.conf são apresentadas na Figura 4.1, onde são configurados os parâmetros para a conexão ao banco de dados e o diretório onde estão localizados os *scripts* de manutenção.

```
# Configura a string de conexão ao banco de dados
$dsn = "DBI:Pg:dbname=maia;host=localhost;port=5432";

# Nome do usuário para a conexão ao banco de dados
$username = "amavis";

# A senha do usuario do banco de dados PostgreSQL
$password = "senha";

# Configura o diretório onde estão localizados os scripts de manutenção
$script_dir = "/var/amavisd/maia/scripts";
```

Figura 4.1 – Alterações no Arquivo maia.conf

4.1.3. Testando a Configuração

O SpamAssassin e o Amavisd-maia baseiam-se em um conjunto de módulos Perl e utilitários do sistema para seu funcionamento correto. O Maia Mailguard disponibiliza um *script* em Perl para a verificação dos módulos instalado no sistema, assim como a sua versão, o configtest.pl. Este *script* verifica se estes módulos estão de acordo com a versão mínima requerida pelo SpamAssassin e Amavisd-maia, bem como um teste de conexão ao banco de dados. Caso seja acusada a ausência de algum módulo ou as versões sejam inferiores ao recomendado, deve ser feita a instalação do módulo de acordo com a versão mínima recomendada e, em seguida, a verificação com o configtest.pl. A Figura 4.2 apresenta a saída do configtest.pl.

Application/Module	Version	Status
Perl	: 5.8.3 :	OK
file(1)	: 4.16 :	OK
Archive::Tar	: 1.10 :	OK
Archive::Zip	: 1.16 :	OK
BerkeleyDB	: 0.26 :	OK
Compress::Zlib	: 1.41 :	OK
Convert::TNEF	: 0.17 :	OK
Convert::UUlib	: 1.06 :	OK
Crypt::Blowfish	: 2.09 :	OK
Crypt::CBC	: 2.12 :	OK
Crypt::OpenSSL::RSA	: 0.22 :	OK
Data::UUID	: 0.13 :	OK
DB_File	: 1.810 :	OK
DBD::mysql	: 3.0002:	OK
DBD::Pg	: 1.31 :	OK
DBI	: 1.50 :	OK
Digest::MD5	: 2.33 :	OK
Net::DNS	: 0.57 :	OK
Net::Server	: 0.93 :	OK
Net::SMTP	: 2.29 :	OK
Pod::Usage	: 1.16 :	OK
Template	: 2.14 :	OK
Time::HiRes	: 1.61 :	OK
Unix::Syslog	: 0.99 :	OK
URI	: 1.31 :	OK
Database DSN test	**: PASSED**	

Figura 4.2 – Saída do *Script* configtest.pl

4.1.4. Carregando as Regras do SpamAssassin

O Maia Mailguard precisa ter acesso às regras do SpamAssassin instaladas no sistema; então, estas regras, suas descrições e suas pontuações serão carregadas em uma tabela do banco de dados do Maia

Mailguard para serem utilizadas posteriormente na verificação das mensagens do servidor SMTP feitas pelo Amavid-maia. Este processo de carregar as regras pode ser feito com auxilio de um dos *scripts* Perl localizados no sub-diretório dos *scripts* chamado load-sa-rules.pl. A Figura 4.3 apresenta a saída do comando.

```
2007-08-31 10:45:24 Maia: [load-sa-rules] Adding new rule:
SARE_SUB_REPRESENT_REQ (Possible phishing subject)
2007-08-31 10:45:24 Maia: [load-sa-rules] Adding new rule:
SARE_SUB_SINCERE (Spam topic found in subject)
2007-08-31 10:45:25 Maia: [load-sa-rules] Adding new rule:
SARE_SUB_MEDS_LEO (obfuscated subject header)
2007-08-31 10:45:25 Maia: [load-sa-rules] Adding new rule:
SARE_SUB_PHARM_LEO (obfuscated subject header)
2007-08-31 10:45:25 Maia: [load-sa-rules] Adding new rule:
SARE_SUB_PHARM_LEO2 (obfuscated subject header)
```

Figura 4.3 – Saída do *Script load-sa-rules.pl*

Este *script* procura no sistema por arquivos com a extensão *.cf e pelas configurações do SpamAssassin, lê as regras, as descrições e as pontuações que ele encontrar nestes arquivos e armazena as informações no banco de dados do Maia Mailguard. Este *script* deve ser executado sempre que for adicionada alguma regra ao SpamAssassin ou quando for feito qualquer atualização em suas pontuações.

4.1.5. Configuração do Amavisd-maia

Para a configuração do Amavid-maia, o arquivo usado é o amavisd.conf distribuído junto com o Maia Mailguard. Este arquivo contém as configurações do Amavisd-maia e deve ser copiado para o diretório /etc e configurado de acordo com os seguintes itens:

- Altere a seguinte linha de acordo com seu domínio

  $mydomain = 'meudominio.com.br';

- A próxima linha determinar o acesso ao banco de dados, de acordo com o banco de dados, o usuário e a senha anteriormente criados no PostgreSQL

  #Formato ([DBI:Pg:nomebanco:hostname', 'usuário', 'senha'])
  @lookup_sql_dsn=(['DBI:Pg:maia:localhost','amavis','senha']);

- Em seguida, iniciar o amavisd-maia, de preferência em modo *debug* para verificar se as configurações estão corretas. Para isso, basta executar o comando

  [root]# amavisd debug

4.2. Postfix

A configuração do Postfix para o funcionamento do Amavisd-maia é feita através do arquivo master.cf (Figura 4.4), onde o postfix deve ser configurado para executar um serviço SMTP adicional na porta 10025. O seu objetivo é fazer a reinjeção das mensagens que passarão pelo filtro Amavisd-maia e foram aceitas, ou seja, que não será feita verificação adicional. É necessário direcionar a verificação de conteúdo do serviço SMTP normal, geralmente executado na porta 25, para o Amavisd-maia. A Figura 4.4 apresenta as modificações necessárias no arquivo master.cf.

Após as modificações, as configurações do postfix devem recaregadas através do comando *postfix reload.* Através do comando netstat, pode-se verificar se a porta 10025/tcp foi iniciada:

 # netstat -apn | grep "10025"
tcp 0 0.0.0.0:10025 0.0.0.0: OUÇA smtpd*

```
#redirecionando a verificação de conteúdo para o amavisd-maia, porta 10024
smtp      inet  n       -       n       -       -       smtpd
 -o content_filter=smtp-amavis:[127.0.0.1]:10024

#Serviço smtp-amavis, onde será feita a reinjeção da mensagem após a
verificação pelo amavisd, não deve ser feita a verificação de conteúdo para este
serviço

smtp-amavis unix -      -       n       -       2  smtp
   -o smtp_data_done_timeout=1200
   -o smtp_send_xforward_command=yes
   -o disable_dns_lookups=yes

#Configurando as opções para o serviço SMTP na porta 10025, o principal é a
opção -o content_filter sem nenhum parâmetro que significa que não será feita a
verificação de conteúdo da mensagem

0.0.0.0:10025 inet n    -       n       -       -  smtpd
   -o content_filter=
   -o local_recipient_maps=
   -o relay_recipient_maps=
   -o smtpd_restriction_classes=
   -o smtpd_client_restrictions=
   -o smtpd_helo_restrictions=
   -o smtpd_sender_restrictions=
   -o mynetworks=127.0.0.0/8,150.163.13.244
   -o strict_rfc821_envelopes=yes
   -o smtpd_error_sleep_time=0
   -o smtpd_soft_error_limit=1001
   -o smtpd_hard_error_limit=1000
   -o smtpd_client_connection_count_limit=0
   -o smtpd_client_connection_rate_limit=0
   -o receive_override_options=no_header_body_checks
```

Figura 4.4 – Alterações no Arquivo master.cf

4.3. Clamav

A configuração do clamav é simples e é feita através de dois arquivos de configuração: i) arquivo de configuração do Amavis-maia, que direciona a verificação de vírus para o clamav; e ii) arquivo de configuração do clamav, onde deve ser configurado para o *daemon* do clamav seja executado com o mesmo usuário do Amavisd-maia (usuário amavis), para que o Maia Mailguard seja capaz de utilizar o banco de dados de vírus do clamav.

As seguintes linhas devem ser descomentadas do arquivo amavisd .conf:

```
## http://www.clamav.net/
['ClamAV-clamd',
  \&ask_daemon, ["CONTSCAN {}\n", "/var/run/clamav/clamd.sock"],
  qr/\bOK$/, qr/\bFOUND$/,
  qr/^.*?: (?!Infected Archive)(.*) FOUND$/ ],
```

Em seguida, é necessário modificar a seguinte linha no arquivo /etc/ clamavd.conf:

```
# Usuário no qual será executado o processo do daemon do clamd
User amavis
```

Adicionalmente, o diretório /var/lib/clamav e seus sub-diretórios devem ser configurados com a permissão para o usuário amavis. Por final, é necessário iniciar o *daemon* do clamav, através da execução do comando:

```
[root]# /etc/init.d/clamd start
```

4.4. PHP

Após a instalação do Maia Mailguard no servidor SMTP, a próxima fase é a instalação dos arquivos com os *scripts* PHP. Para que sejam acessíveis, os arquivos devem ser localizados em um diretório abaixo da árvore da *Web*. No caso do Gentoo, o diretório padrão é o /var/www/localhost/htdocs/; para a instalação do Maia Mailguard, será criado um subdiretório nomeado maia, sendo o caminho completo resultante /var/www/localhost/htdocs/maia. Copia-se o conteúdo do subdiretório /php para dentro da árvore de instalação do Maia Mailguard para este diretório, sendo como resultante os sub-diretórios apresentados na Tabela 4.2.

Tabela 4.2 – Diretórios da Instalação dos Arquivos PHP do Maia Mailguard

maia/admin	Testes de configuação e upgrades
maia/images	Imagens
maia/libs	Bibliotecas PHP usadas pelo Maia Mailguard
maia/locale	Pacotes de linguagem
maia/themes	Temas da página HTML

4.4.1. Classe de *Templates* Smarty

O Maia Mailguard usa o sistema de *templates* Smarty, uma classe de *templates*. Ele funciona de uma forma que separa interface da lógica de programação e tem o objetivo de facilitar e de melhorar o desenvolvimento de aplicações em PHP (Feitosa, 2006). Para que o Maia Mailguard consiga usar as classes Smarty, elas devem ser instaladas no

sistema e no gentoo da seguinte forma:

[root]# emerge smarty

O Smarty provê um *cache* das páginas renderizadas e, para isso, deve ter acesso de escrita ao diretório themes.

4.4.2. Pacotes PEAR

A interface *Web* do Maia Mailguard tem como pré-requisitos alguns pacotes da biblioteca PEAR para a execução de seus *scripts* PHP. O PEAR é uma plataforma e um sistema de distribuição que promove a reutilização de código e padronização da escrita de códigos em PHP.

A instalação dos pacotes da biblioteca PEAR poderão ser feitos após a instalação do PEAR, através do seguinte comando:

[root]# emerge pear

Após a instalação do PEAR, seus pacotes podem ser instalados no sistema através do comando:

pear install nome_do_pacote

A Tabela 4.3 apresenta os pacotes necessários para a interface *Web* do Maia Mailguard.

4.4.3. Configuração do PHP

A maior parte da configuração do Maia Mailguard é feita através da *Web*, contudo existem alguns parâmetros que precisam ser configurados através do arquivo de configuração maia/config.php. Dentro deste arquivo, são configuradas a conexão ao banco de dados, a linguagem padrão e como será feita a autenticação do usuário do maia. São frisadas

as principais configurações deste arquivo que pode ser visualizado integralmente no Anexo 2.

Tabela 4.3 – Pacotes PEAR – Pré-Requisitos para o Maia Mailguard

Nome do Pacote	Versão
Archive_Tar	1.3.1
DB	1.7.6
DB_Pager	0.7
Log	1.9.3
Mail_Mime	1.3.1
Net_IMAP	1.0.3
Net_POP3	1.3.6
Net_SMTP	1.2.8
Net_Socket	1.0.6
PEAR	1.4.9

- **$default_display_language**: Configura a linguagem padrão usada no maia. Algumas opções disponíveis junto ao Maia Mailguard são: “en” inglês, “pt” português lusitano, “fr” francês;
- **$maia_sql_dsn**: Configura a *string* de conexão que as funções PEAR::DB usarão para efetuar conexão ao banco de dados do Mail Mailguard. A *string* de conexão tem a seguinte sintaxe: "dbtype://dbuser:passwd@tcp(hostname:port)/dbname", onde os campos são: i) **dbtype:** mysql para bancos de dados MySQL ou *pgsql* para bancos de dados PostgreSQL; ii) **dbuser:** usuário do banco de dados; iii) **passwd:** senha do usuário do banco de dados;

e iv) **dbname:** nome da base de dados. No caso, o tipo de banco de dados é “postgres”, nome do banco de dados “maia”, usuário “amavis” e conexão ao *localhost*, a *string* de conexão deve ser a seguinte: $maia_sql_dsn = "pgsql://amavis:senha@tcp (localhost:432) /maia";

- **$auth_method**: Determina o método de autenticação usado pelo Maia Mailguard para a verificação das credenciais de *login* de seus usuários. A Tabela 4.4 apresenta os métodos disponíveis. O método de autenticação escolhido foi o pop3, ou seja, sempre que o usuário efetuar *login*, será enviado um pedido de autenticação a um servidor pop3. Para que esta opção seja configurada no arquivo de configuração, as seguintes linhas devem ser alteradas: i) $auth_method = "pop3"; ii) $auth_pop3_host = "ssl://localhost"; e iii) $auth_pop3_port = 995;
- **$auth_pop3_host**: Configura o endereço do *hostname* do servidor pop3. Caso seja usado pop3 sobre SSL (*Secure Sockets Layer*), deve ser configurado ssl://hostname;
- **$auth_pop3_port**: Configura a porta de conexão ao servidor pop3.

Tabela 4.4 – Métodos de Autenticação do Usuário

$auth_method	Método Usado
pop3	Autenticação ao servidor Pop3 ou Pop3 SSL
imap	Autenticação ao servidor IMAP ou IMAP SSL
ldap	Autenticação ao servidor LDAP
sql	Autenticação ao servidor a um Banco de dados SQL
internal	Autenticação direta ao banco de dados do maia
external	Aplicação ou script externo

Capítulo 5 – Utilizando o Mail Mailguard

A interface *Web* do Maia Mailguard difere de acordo com o nível de hierarquia do usuário conectado no sistema. No Maia Mailguard, existem três níveis de hierarquia: i) super usuário; ii) administrador do domínio; e iii) usuário comum. Estes três diferentes modos de operação são apresentados a seguir:

- **Modo Usuário**: Este é o modo operado pelo usuário de correio eletrônico comum, ou seja, o usuário final dentro de um domínio de uma empresa ou instituição. As principais tarefas executadas por ele são:

  Gerenciamento das mensagens em quarentena. Esta função consiste em confirmar que as mensagens estão em quarentena, são *spams* ou que foram erroneamente classificadas (falso positivo). No caso de falso positivo, após o usuário confirmar como uma mensagem válida, ela é enviada diretamente para a caixa postal do usuário. A Figura 5.1 e a Figura 5.2 apresentam, respectivamente, o gerenciamento de *spams* recebidos e de mensagens válidas;

  Gerenciamento das listas brancas/negras. Nesta função, o usuário tem possibilidade de inserir endereços de *e-mail* ou domínio de um remetente na lista branca ou negra,

sendo que este endereço não passará por verificações de *spam*. A Figura 5.3 apresenta o gerenciamento das listas branca e negra;

Possível Spam

Confirmar o Estado destes Items

Pontuação	Recebido	De	Assunto	Spam?	Não-spam?	Eliminar
21.243	2008-04-20 06:37:...	trinhsutruongthi@...	Cheap and convini...	◉	○	○
30.929	2008-04-19 22:39:...	neal@www.prinet-j...	Looking for the b...	◉	○	○
7.954	2008-04-19 11:21:...	treinamentos@hebm...	Análise e Viabili...	◉	○	○
22.067	2008-04-19 02:17:...	apet@informativod...	SEMINÁRIO ESPECIA...	◉	○	○
12.287	2008-04-19 01:57:...	marketing@aslan.c...	A MAIOR LOJA VIRT...	◉	○	○

Figura 5.1 – Lista de *Spams* Armazenados em Quarentena

Não-spam por confirmar

Confirmar o Estado destes Items

Pontuação	Recebido	De	Assunto	Spam?	Não-spam?	Eliminar
-2.599	2008-04-18 17:08:...	fonseca@amr.com.br	RES: Cotação p/ g...	○	◉	○
-2.599	2008-04-18 14:40:...	barbosa@lit.inpe.br	[Primeira fotogr...	○	◉	○
-2.599	2008-04-18 07:52:...	fonseca@amr.com.br	RES: Cotação p/ g...	○	◉	○
0.000	2008-04-17 17:23:...	barbosa@lit.inpe.br	[Fwd: Enviando em...	○	◉	○
0.000	2008-04-17 10:42:...	marcos@unicacompu...	RES: Cotação Unic...	○	◉	○
-2.416	2008-04-17 09:42:...	marcos@unicacompu...	Cotação Unica Com...	○	◉	○

Figura 5.2 – Mensagens Válidas Armazenadas no *Cache*

Configurações do endereço eletrônico. Nestas configurações, o usuário pode habilitar ou desabilitar os tipos de filtro que a mensagem enviada para seu endereço eletrônico. Estas configurações estão limitadas às

configurações estabelecidas pelo super usuário. A Figura 5.4 apresenta as configurações do endereço eletrônico;

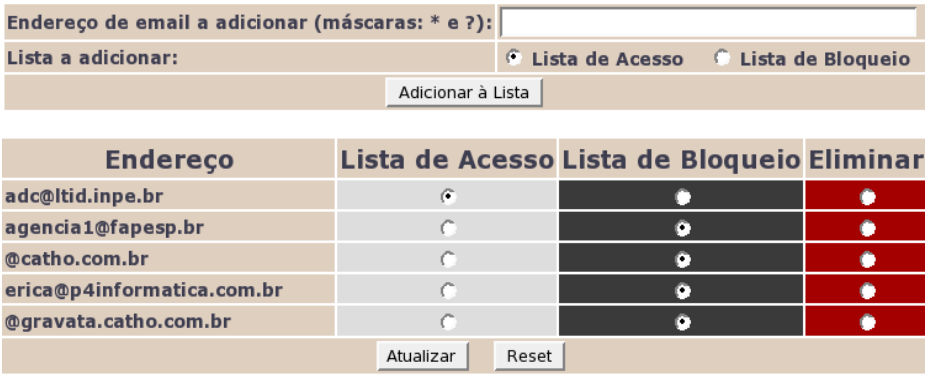

Figura 5.3 – Gerenciamento das Listas Branca e Negra

Figura 5.4 – Configurações do Endereço Eletrônico

Estatísticas do Sistema. Nesta opção, o usuário pode visualizar as estatísticas das mensagens recebidas, classificadas pelo sistema como *spam* e válidas, a

quantidade de falsos positivos e negativos encontrados após a classificação das mensagens pelo usuário e outras informações. Podem ser visualizadas as estatísticas do próprio usuário, ou de todos os usuários do sistema (Figura 5.5).

Estatísticas de todos os Usuários

	Items			Pontuação			Tamanho (kB)		
Tipo de Mail	Contagem	Items/dia	Percentagem	Mínimo	Máximo	Média	Mínimo	Máximo	Média
Não-spam por confirmar	18038	1151.6	1.7%	-7.898	8.985	-0.519	0.5	963.6	78.9
Não-spam confirmado	244626	228.6	22.6%	-16.546	38.812	-0.569	0.4	976.4	64.4
Falsos Positivos	10798	10.1	1.0%	0.000	112.772	10.941	0.3	258.4	26.9
Possível Spam	9501	606.4	0.9%	0.000	112.772	15.008	0.3	252.9	14.0
Spam Confirmado	550050	513.9	50.8%	-2.312	135.826	17.166	0.3	440.4	7.5
Falsos Negativos	16906	15.8	1.6%	-16.129	34.290	2.042	0.3	956.5	29.6
Items na lista de acesso	113687	106.2	10.5%	-	-	-	0.3	975.5	45.5
Items na lista de bloqueio	2040	1.9	0.2%	-	-	-	0.5	933.0	54.6
Vírus/Malware	14566	13.8	1.3%	-	-	-	1.1	908.5	45.1
Anexos banidos	2545	2.4	0.2%	-	-	-	0.6	958.6	96.7
Cabeçalhos de email inválidos	100709	94.2	9.3%	-	-	-	0.4	976.2	41.2
Items excessivamente grandes	74688	70.2	6.9%	-	-	-	952.8	61514.6	2813.8

Eficiência 96.63% Falso Positivo 1.31% Falso Negativo 2.06%
Sensibilidade 97.02% PPV 98.07% Especificidade 95.77% NPV 93.54%

Figura 5.5 – Estatísticas dos Usuários do Sistema

- **Modo Administrador Domínio**. Este é o modo operado pelo administrador de um domínio de uma empresa ou instituição. Neste modo, o usuário pode executar as funções de um usuário normal, além de algumas funções administrativas, são elas:

  Gerenciamento de usuário. Permite criar, excluir ou procurar os usuário do domínio dentro do banco de dados do Maia Mailguard;

Configurações padrões. Configura quais filtros estarão ou não habilitados na criação de um novo usuário. Estas configurações podem ser mudadas pelo usuário.

- **Modo Super Usuário**. Este é modo do administrador do sistema Maia Mailguard. Neste modo, o usuário pode executar as funções do administrador do domínio, além de permitir a criação de domínios, configurações globais do Maia Mailguard e delegar o privilégio de administrador de domínio a um usuário.

A relação entre estes três modos de operação do Maia Mailguard são representados através de um Diagrama de Classes na Figura 5.6.

A seguir, será apresentado um exemplo de como configurar um domínio na interface *Web* e como definir as principais configurações do Maia Mailguard. Para o funcionamento da ferramenta, é necessário que seja feita a correta configuração do servidor MTA, que o banco de dados esteja corretamente criado com as tabelas do Maia Mailguard e o PHP esteja apontando corretamente para estas opções.

Após a criação do ambiente correto para a utilização da ferramenta, poderá ser feito o *login* no sistema usando o modo de autenticação escolhido; no caso, o pop3-ssl. Para o acesso ao Maia Mailguard, neste caso feito através da máquina com o servidor PHP instalado, deve-se abrir um navegador e apontar o endereço de acesso para http://127.0.0.1/maia. Com isso, é possível visualizar a tela de *login*, conforme ilustrado na Figura 5.7.

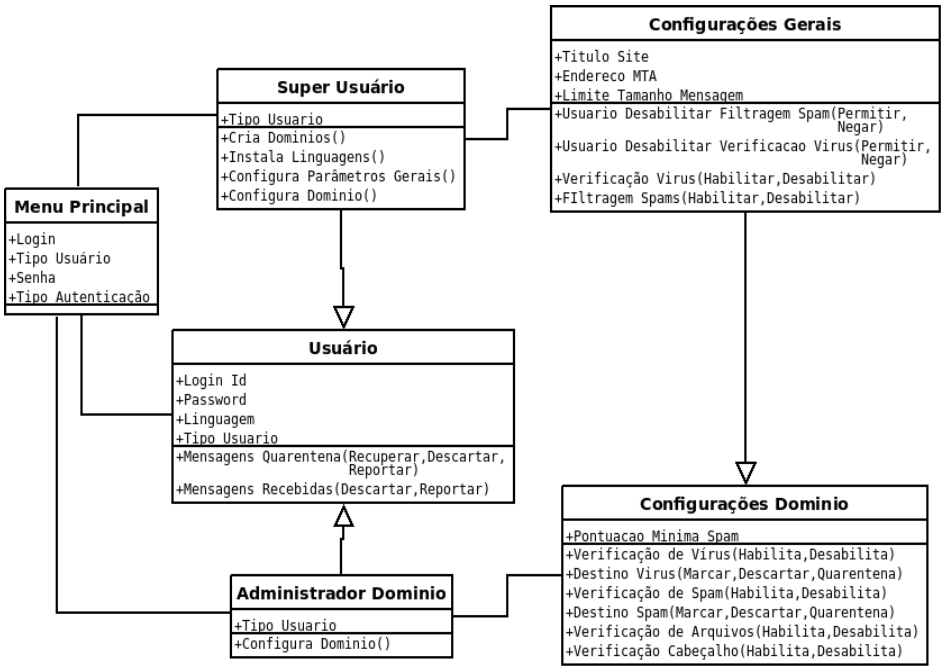

Figura 5.6 – Diagrama de Classes da Interface *Web*

No primeiro *login* do sistema, deve ser designado qual será o super usuário. Este procedimento poderá ser executado acessando o seguinte endereço http://127.0.0.1/maia/login.php?super=register. Este endereço levará a uma tela normal de *login* e, após a correta autenticação, o usuário terá os privilégios de super usuário. Este procedimento precisa ser feito uma única vez; depois disto, o usuário terá os privilégios de super usuário.

Para este trabalho, foi escolhido o **maia_adm** para ser o usuário com os privilégios de super usuário. Como o modo de autenticação escolhido foi o pop3-ssl, este usuário precisa ter um *login* e senha devidamente criados no servidor pop3 para que seja efetuada a correta autenticação no Maia Mailguard.

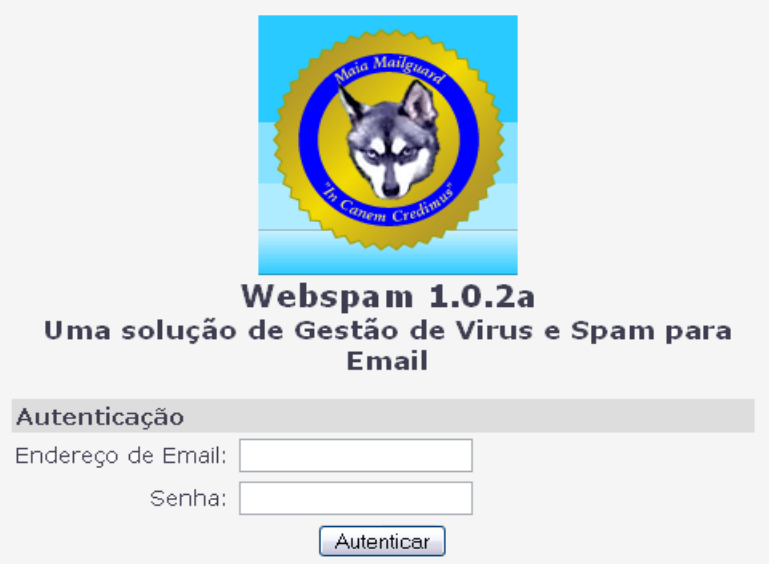

Figura 5.7 – Tela de *Login* do Maia Mailguard

Após efetuado o *login*, o super usuário é levado a tela principal (Figura 5.8). No centro desta tela, é possível verificar as mensagens: i) em quarentena classificadas como *spam*; ii) em quarentena nas quais foram identificados algum arquivo contendo vírus; iii) que têm extensões proibidas; e iv) com cabeçalho corrompido.

No *menu* lateral, é possível acessar as opções disponíveis pelo Maia Mailguard, sendo elas: i) **Estatísticas**, mostra as estatísticas das mensagens classificadas para este usuário; ii) **Lista de acesso/bloqueio**, permite cadastrar ou excluir endereços e domínios na lista branca ou negra; iii) **Definições**, permite configurar as características de filtragem para esta conta de *e-mail*; iv) **Administração**, mostra o *menu* de administração; e v) **Ajuda**, mostra uma página com o auxilio de uso da ferramenta.

Figura 5.8 – Tela Principal do Maia Mailguard

O primeiro passo é adicionar um domínio para que ele seja reconhecido e a mensagem de seus usuários passe a ser classificada pelo Maia Mailguard. Para isso, seleciona a opção **Administração** (Figura 5.9) e acessa a opção **Domínios**. Nesta tela, é possível adicionar um novo domínio; a partir deste momento, as mensagens recebidas no servidor SMTP com destino a este domínio são filtradas pelo Maia Mailguard. Além disso, pode-se remover um domínio existente que, apesar do servidor SMTP ainda fazer a entrega das mensagens, estas não passarão pelo filtro do Maia Mailguard.O endereço do domínio deve ser informado no formato "@meudominio.com.br".

Após adicionar o domínio, é preciso configurar as opções gerais do

Maia Mailguard. Isto deve ser feito retornando ao *menu* **Administração** e selecionando a opção **Configuração de Sistema** (Figura 5.10). As principais opções disponíveis são:

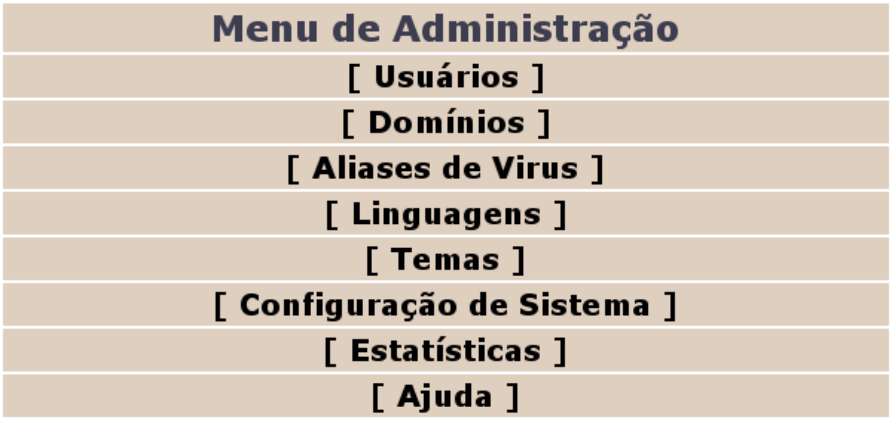

Figura 5.9 – Opções de Administração

- **Ativar criação automática de contas de usuários**. Deixa o Maia Mailguard criar novos usuários sempre que uma mensagem for recebida para um usuário local que não esteja registrado no banco de dados. Se esta opção for desabilitada, contas de usuário serão criadas após o usuário efetuar *login*. É recomendável que esta opção fique em ***no***;
- **Ativar monitoração de falsos negativos**. Habilita ou desabilita o *cache* das mensagens válidas, um mecanismo do Maia Mailguard para reportar falsos negativos. Esta opção deve estar habilitada, pois melhora a ação do filtro de *spam*, fazendo o constante treinamento do filtro;
- **Ativar verificação de vírus**. Habilita ou desabilita globalmente a verificação de vírus;

- **Permitir que os usuários ativem/desativem a verificação de vírus**. Configura se o usuário será capaz de habilitar ou desabilita a verificação de vírus;

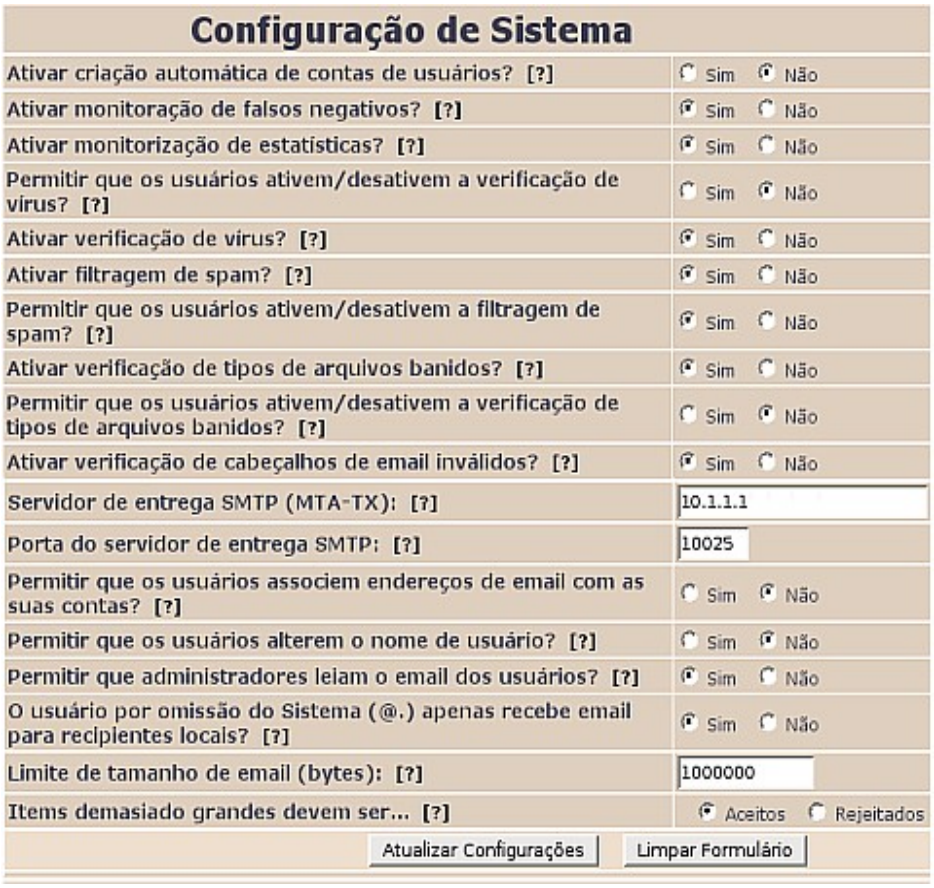

Figura 5.10 – Configurações Globais do Maia Mailguard

- **Ativar filtragem de spam**. Habilita ou desabilita globalmente a filtragem de *spams*;
- **Permitir que os usuários ativem/desativem a filtragem de spam**. Configura se o usuário será capaz de habilitar ou desabilita

a filtragem de *spams*;

- **Servidor de entrega SMTP (MTA-TX)**. Configura o *hostname* ou endereço IP do servidor SMTP que o Maia Mailguard deve usar para fazer a recuperação das mensagens em quarentena que foram reportadas como válidas (falsos positivos);
- **Porta do servidor de entrega SMTP**. Configura a porta de conexão ao servidor SMTP que o Maia Mailguard usa para fazer a recuperação das mensagens em quarentena.

Capítulo 6 – Resultados

Para os teste apresentados nesta seção, foram usados os resultados obtidos no uso do Maia Mailguard em um ambiente de produção, no Laboratório de Integração e Testes (LIT) do Instituto Nacional de Pesquisas Espaciais (INPE) onde a ferramenta está em operação desde outubro de 2003, completando um total de 54 meses. Para a análise dos dados, foi usado o resultado do levantamento estatístico das mensagens eletrônicas recebidas (Anexo 3) extraído da ferramenta desde o período acima citado, onde é possível verificar o número de itens que foram classificados com *spam*, válidas, vírus entre outras classificações.

6.1. Ambiente de Teste

O LIT é um complexo laboratorial situado dentro do INPE especialmente projetado e construído com o objetivo de permitir o desenvolvimento de satélites espaciais, no que diz respeito à montagem, à integração e ao teste. O parque de máquinas funciona em um ambiente misto com estações e servidores com os sistemas operacionais Windows 95/98, Windows NT, Windows 2000, Linux e AIX.

O servidor de mensagens eletrônicas tem aproximadamente 200 contas usuários de correio eletrônico, funciona em uma máquina e recebe aproximadamente 2600 mensagens diárias.

6.2. Teste de Precisão

O primeiro gráfico (Figura 6.1) retrata a quantidade de mensagens recebidas mensalmente cujo *status* foi confirmado pelo usuário, ou seja, as mensagens que tiveram a interação do usuário, através da interface *Web*, para indicar se a mensagem recebida era *spam* ou válida. Através do gráfico, é possível observar que apenas 1% das mensagens foram falsos positivos, mensagens válidas classificadas como *spam*, sendo que estas foram recuperadas e enviadas ao usuário. Também, é possível observar que apenas 2% das mensagens são falsos negativos, ou seja, *spams* que foram erroneamente classificados como mensagens válidas, mas trazem o desconforto de receber um *spam* em sua caixa de mensagens.

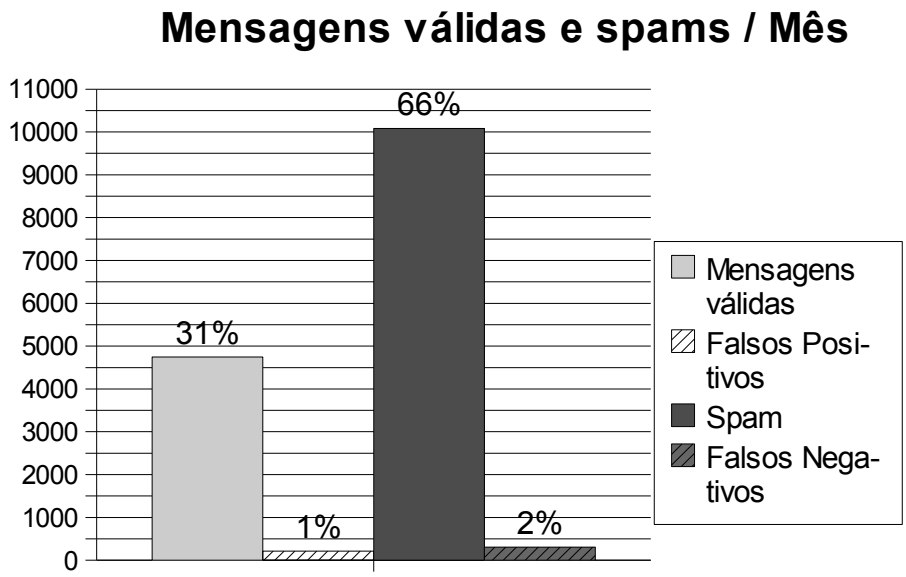

Figura 6.1 – Gráfico Mensal das Mensagens Válidas e *Spams*

6.3. Teste de Eficiência

Para a medida de eficiência, foi usada a quantidade de mensagens classificadas desde o início do uso da ferramenta (Figura 6.2). É possível observar que 65% das mensagens recebidas pelo servidor de correio eletrônico são *spam*, sendo que elas foram barradas pelo filtro, além dos 2% das mensagens que contêm arquivos com vírus e 2% que receberam outros tipos de classificações como anexos banidos ou cabeçalho inválido.

Neste cenário, apenas 32% das mensagens são mensagens válidas; sendo assim, o número de mensagens filtradas é aproximadamente 67% do total de mensagens recebidas, aumentando em mais de 200% a produtividade no tempo gasto para a leitura das mensagens do sistema de correio eletrônico.

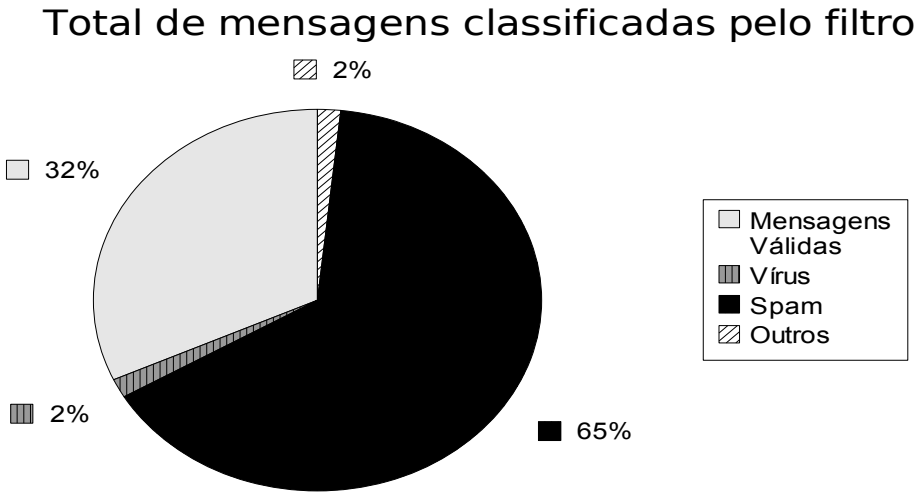

Figura 6.2 – Total de Mensagens Classificadas pelo Filtro

6.4. Teste de Popularidade

A mais importante característica observada no Maia Mailguard é sua interatividade com o usuário na classificação das mensagens. Para que o processo de classificação de mensagens fique mais preciso, o Maia Mailguard conta com a interação do usuário que deve confirmar se as mensagens recebidas são realmente *spam* ou válidas. Sendo assim, foi feita uma comparação entre o número de mensagens classificadas e não classificadas. Observando a Figura 6.3, é possível concluir que apenas 3% das mensagens não tiveram seu tipo confirmado pelo usuário, isto representa a grande interação do usuário com o sistema através da interface *Web*.

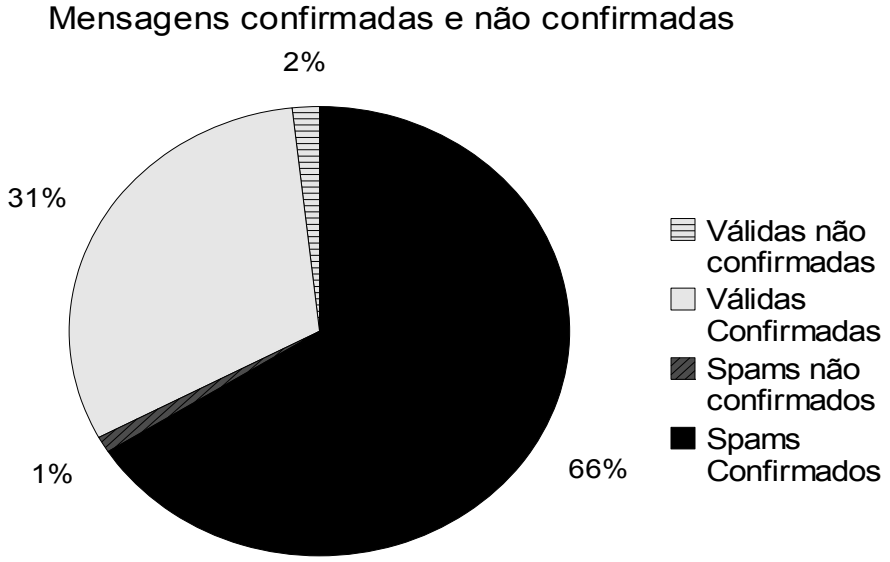

Figura 6.3 – Mensagens Confirmadas pelo Usuário

Capítulo 7 – Considerações Finais

Após a execução da pesquisa bibliográfica e do estudo dos resultados obtidos no uso do Maia Mailguard em um ambiente de produção (LIT/INPE), foi possível saber como usar várias ferramentas em conjunto para obter uma melhor eficiência no uso das mensagem eletrônicas. Assim, é possível afirmar que o Maia Mailguard é uma ferramenta poderosa no auxílio ao combate ao *spam*. Além de fornecer ferramenta ao administrador do sistema para o controle das ações do usuário, Maia Mailguard possibilita ao usuário a liberdade de escolher quais mensagens serão ou não bloqueadas pelo filtro.

Além da execução deste projeto no LIT/INPE, a instalação deste sistema foi vendida para duas empresas do ramo comercial, caracterizando a aceitação deste sistema em empresas/instituições com diferentes áreas de atuação.

Pode-se afirmar que os objetivos propostos no início deste trabalho foram obtidos, ou seja, apresentar os problemas ocasionados pelo recebimento de mensagens não solicitadas, bem como mostrar uma solução livre com base em filtros *bayesianos* configurável via *Web*.

Uma grande contribuição, feita pelo autor deste trabalho, é o esforço na tradução do Maia Mailguard para o idioma português do Brasil, que nos moldes da filosofia do software livre, pretende disponibilizar esta tradução para a comunidade assim que ela for

concluída.

O uso do Maia Mailguard no Laboratório de Integração e Teste (LIT) ajudou a melhorar a produtividade na troca de mensagens eletrônicas, classificando, ao longo de aproximadamente cinco anos de uso, mais de 64% das mensagens recebidas como *spam* e possibilitando ao usuário recuperar a mensagem que foi erroneamente classificada.

Este trabalho é apenas o começo, pois uma grande contribuição é o uso de um banco de dados *Bayesiano* individual para cada usuário do sistema. Afinal, cada usuário tem uma particular necessidade e, pensando neste propósito, o treinamento do filtro *Bayesiano* do SpamAssassin pode ser individual, gerando maior acuracidade na classificação das mensagens válidas e *spams*.

Referências Bibliográficas

Allman, E. et al., **DomainKeys Identified Mail Signatures** (DKIM) [on-line]. Internet Engineering Task Force, Request for Comments 4871, Maio de 2007. Disponível em: <http://www.ietf.org/rfc/rfc4871.txt> Acesso em 01/05/2008.

BLANK,R. L. **Maia The Big Picture** [on-line], Janeiro de 2004. Disponível em: <http://www.maiamailguard.com/bigpicture.php>. Acesso em 15/07/2007.

CERT.BR. **Cartilha de Segurança para Internet** [on-line]**,** v. 3.1, Outubro de 2006. Disponível em:<http://www.cert.br/stats/spam/ 2006.html>. Acesso em 10/06/2007.

CERT.BR. **Estatísticas de Notificações de Spam Reportadas no Ano de 2006** [on-line], Julho de 2007. Disponível em: <http://www.cert.br /stats/spam/ 2006.html>. Acesso em 24/07/2007.

EVAN, H. The Next Step in the Spam Control War: **Greylisting** [on-line],Agosto de 2003.Disponível em: <http://projects.puremagic.com/ greylisting/whitepaper.html> Acesso em 10/07/2007.

FABRE, R. C. **Métodos Avançados para Controle de Spam**, Fevereiro de 2005. Unicamp, Trabalho Final de Mestrado.

FEITOSA , C. **Smarty e PHP, tudo a ver** [on-line]. Janeiro, 2006. Disponível em: <http://cirofeitosa.com.br/post/smarty-e-php-tudo-a-ver>. Acesso em 18/07/2007.

LESSING, L. **The Spam Wars** [on-line], Dezembro de 1998. Disponível em: <http://www.dotcomeon.com/thespamwars.html>. Acesso em: 15/08/2007.

MAIA MAILGUARD . Maia:**Arquivos de Instalação** [on-line]**,** versão 1.02a, Agosto de 2007. Disponível em: <http://www.maiamailguard.com/files/maia-1.0.2a.tar.gz> Acesso em 15/08/2007.

Nunes , D. Spam e Fraudes por E-mail: **Técnicas de Mitigação para Administradores,** 8° Simpósio Segurança em Informática, novembro de 2006, São José dos Campos -- SP

POSTEL, J. B. **Simple Mail Transfer Protocol** [on-line], Internet Engineering Task Force, Request for Comments 821, Agosto de 1982. Disponível em: <http://www.faqs.org/rfcs/rfc821.html> Acesso em 19/07/2007.

REAL, R. **Redes Bayesianas Aplicadas a Reconhecimento de SPAM** [on-line], Março de 2003. Disponível em: <http://www.inf.ufrgs.br/procpar/disc/cmp135/trabs/rodrigo/T2/html_spam/index.html> Acesso em 10/07/2007.

TAVEIRA, D. M. et al. **Técnicas de defesa contra spam**. Em Livro Texto dos Mini-cursos do VI Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (2006), Sociedade Brasileira de Computação, pág. 202–250.

TEIXEIRA, R. C. **O Pesadelo do SPAM**. News Generation, Rio de Janeiro, RJ, v.5, n.1, Jan.2001.

TEMPLETON, B. **Origin of the term spam to mean net abuse** [on-line], Março de 2003. Disponível em:<http://www.templetons.com/brad/spamterm.html>. Acesso em 12/06/2007.

WONG, M; SCHLITT, W. - **Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail** [on-line]. Internet Engineering Task Force, Request for Comments 4408, Abril de 2007. Disponível em <http://www.ietf.org/rfc/rfc4408.txt>. Acesso em: 10 maio. 2008

Anexo 1 – Arquivo maia.conf

```
#######      Maia Mailguard Script Configuration File            #

# Configure your Maia database DSN here
$dsn = "DBI:Pg:dbname=maia;host=localhost;port=5432";

# Your Maia database user's login name
$username = "maia";

# Your Maia database user's password
$password = "senha";

# The directory where Maia's Perl scripts can be found.
$script_dir = "/var/amavis/maia/scripts";



### load-sa-rules.pl

# The directory where SpamAssassin's local.cf file can be found.
#$local_cf_dir = "/etc/mail/spamassassin";
$local_cf_dir = undef; # default: let the script find it

# The directory where SpamAssassin's core rules can be found.
# If you wish to specify the directory yourself, you can use the
# %%VERSION%% macro to replace the actual SpamAssassin version number.
#$system_rules_dir = "/usr/share/SpamAssassin";
```

#$system_rules_dir = "/var/lib/spamassassin/%%VERSION%%"; # sa-update

$system_rules_dir = undef; # default: let the script find it

The directory where your amavis user's user_prefs file can be found.

#$user_rules_dir = "/var/amavisd/.s*pamassassin*";

#$user_rules_dur = "~/.*SpamAssassin*";

$user_rules_dir = undef; # default: let the script find it

Anexo 2 – Arquivo config.php

```
<?php

   // These names may be misleading, but for now...
   //Possible levels: LOG_DEBUG, LOG_INFO, LOG_NOTICE, LOG_WARNING,
LOG_ERR ...
   $loglevel = PEAR_LOG_DEBUG;          // messages >= this level are sent to php log
   $debug_popup = false;                // allow a debug popup to be shown
   $debuglevel = PEAR_LOG_DEBUG;        // messages >= this level are displayed
in popup window



   // Language to use for text display (en, fr, ja, de, etc.).
   // Locale-specific language files should be installed
   // under ./locale/<lang>. ./local/en (English) is provided
   // by default; translations to other languages may be
   // provided by other parties in the future. Visit the
   // Maia Mailguard website for more details about "supplemental
   // files" (http://www.renaissoft.com/maia/download.html).
   ##$default_display_language = "en";
   $default_display_language = "pt";



   // Default character set
   $default_charset = "ISO-8859-1";


   //$maia_sql_dsn = "dbtype://dbuser:passwd@tcp(hostname:port)/maia";
   //$maia_sql_dsn = "mysql://amavis:passwd@tcp(localhost:3306)/maia";
```

```
$maia_sql_dsn = "pgsql://maia:senha@cp(localhost:5432)/maia";
//$maia_sql_dsn = "mysql://dbuser:passwd@unix(/var/tmp/mysql.sock)/maia";

// Select an authentication method from one of the following:
//
//    "pop3" - use a POP3 server to authenticate
//    "imap" - use an IMAP server to authenticate
//    "ldap" - use an LDAP server to authenticate
//    "exchange" - use an Exchange Server to authenticate
//    "sql" - use a SQL database server to authenticate
//    "internal" - use Maia's internal SQL database to authenticate
//$auth_method = "internal";
$auth_method = "pop3";

// *** AUTHENTICATING VIA POP3 ***
//
// POP3 support is provided by the Net_POP3 PEAR module
// http://pear.php.net/package/Net_POP3
// If you need ssl support, set the port to 995 and add "ssl://" to the hostname

// Standard POP3
#$auth_pop3_host = "localhost";
#$auth_pop3_port = 110;
$auth_pop3_host = "ssl://localhost";
$auth_pop3_port = 995;
?>
```

Anexo 3 – Levantamento Estatístico de Mensagens Eletrônicas Recebidas

Levantamento estatístico de mensagens eletrônicas recebidas, coletado através do Maia Mailguard implementada no LIT do INPE, após o uso da ferramenta pelo período de 54 meses com aproximadamente 200 contas de usuários de correio eletrônico. Este estatística foi coletada em 01/03/2008.

Estatísticas de todos os Usuários			
Items			
Tipo de Mail	**Contagem**	**Items/dia**	**%**
Msg válidas por confirmar	15508	998,2	1,30%
Msg válidas confirmadas	275303	222,9	23,20%
Falsos positivos	12343	10	1,00%
Spam por confirmar	10319	662,5	0,90%
Spam confirmado	584762	473,4	49,30%
Falsos negativos	17819	14,4	1,50%
Items na lista de acesso	134791	109,1	11,40%
Items na lista de bloqueio	2908	2,4	0,20%
Vírus/Malware	14788	12,1	1,20%
Anexos banidos	2860	2,3	0,20%
Cabeçalhos de msg inválidos	114199	92,5	9,60%
Items excessivamente grandes	92956	75,6	7,80%
Eficiência 96,61% Falso Positivo 1,39% Falso Negativo 2,0% Sensibilidade 97,04%			