UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Alteração no protocolo SMTP para redução de spam

Pablo Marques de Oliveira

Itajubá, Dezembro de 2014

UNIVERSIDADE FEDERAL DE ITaAJUBÁ
PROGRAMA DE PÓS GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Pablo Marques de Oliveira

Alteração no protocolo SMTP para redução de spam

Dissertação submetida ao Programa de Pós-Graduação em Ciência e Tecnologia da Computação como parte dos requisitos para obtenção do Título de Mestre em Ciência e Tecnologia da Computação

Área de Concentração: SISTEMAS DE COMPUTAÇÃO

Orientador: Prof. Dr. Otávio Augusto Salgado Carpinteiro
Coorientador: Prof. Dr.Edmilson Marmo Moreira

Dezembro de 2014
Itajubá - MG

Agradecimentos

Agradeço primeiramente a Deus, em seguida a minha esposa, que é minha força, e seu apoio possibilita-me, não só a realização desse trabalho de mestrado, mas também de todos os meus projetos.

Agradeço a meus pais pela educação e minha formação, e aos meus irmãos pela amizade que nos une.

Agradeço, de forma profunda, ao meu orientador, Professor Dr. Otávio Augusto Salgado Carpinteiro e ao meu coorientador Professor Dr.Edmilson Marmo Moreira, a orientação nesse trabalho, pois sem ela não seria possível concluí-lo. Ao Professor Dr. Otávio Augusto Salgado Carpinteiro, agradeço ainda o valoroso auxílio a mim direcionado, desde meu primeiro dia de ingresso na UNIFEI. Deixo aqui, Professor, meus mais sinceros agradecimentos.

Finalmente agradeço a todo o corpo do Hospital Dante Pasinesi, onde as primeiras páginas dessa dissertação foram escritas.

Nós nascemos, vivemos por um breve instante, e morremos. Sempre assim aconteceu durante imenso tempo. A tecnologia não muda muito isso - se é que muda alguma coisa.".

Steve Jobs

Resumo

Um dos principais problemas encontrados no serviço de correio eletrônico (e-mail) é o recebimento de mensagens não solicitadas, conhecidas como spam. Spam causa sérios prejuízos às instituições, sobrecarregando servidores, links de comunicação e ativos de rede.

Esta dissertação propõe uma modificação no Simple Mail Transfer Protocol (SMTP) para redução de spam. A modificação no protocolo produz três consequências vantajosas. A primeira, consiste na rejeição de e-mails indesejados, assim definidos pelo destinatário, evitando-se o desperdício de seus recursos computacionais e de rede. A segunda, consiste no retorno do e-mail indesejado ao spammer, causando-lhe custos, uma vez que seu servidor efetuará processamentos e armazenamentos extras para tratar o spam recusado. A terceira consequência consiste no fato de que, em virtude da recusa, o spammer remove o endereço do destinatário de suas listas de distribuição.

A modificação do SMTP foi implementada em um servidor de e-mail Zimbra e avaliada exaustivamente. Os resultados são promissores. O servidor Zimbra modificado demonstrou desempenho e custo computacionais equivalentes ao do servidor Zimbra original quando recebe e-mails legítimos. Quando recebe spam porém, ele apresenta melhor desempenho e custo computacionais que os do servidor Zimbra original.

Abstract

One of the key problems found in the e-mail service is the receipt of unsolicited messages, known as spam. Spam causes serious losses to institutions, overloading servers, communications links and network appliances.

This dissertation proposes to modify the Simple Mail Transfer Protocol (SMTP) in order to reduce spam. The modification in the protocol produces three advantageous consequences. The first consists in rejecting undesired e-mails, thus defined by the recipient, avoiding a waste of his/her computational and network resources. The second consists in the return of the undesired e-mail to the spammer, penalizing him/her as his/her server will use more CPU and memory to handle the rejected spam. The third consequence consists in the fact that owing to the refusal, the spammer removes the recipient from his/her distribution lists.

The modification in the SMTP was implemented in a Zimbra e-mail server and assessed exhaustively. Results are promising. The modified Zimbra server showed computational performance and costs similar to those of the original Zimbra server when receiving legitimate e-mails. When receiving spam, however, it showed better computing performance and costs than the original Zimbra server.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

O correio eletrônico (e-mail), na forma como é conhecido hoje em dia, foi proposto por Louis Pouzin, Glenda Schroeder e Pat Crisman, pesquisadores do MIT (VLECK, 2012). Na forma proposta, as mensagens de cada usuário eram anexadas a um arquivo chamado "*mailbox*", de acesso privado, de forma que somente o usuário proprietário poderia ler ou excluir mensagens.

Com a expansão da Internet, o correio eletrônico tornou-se uma das formas mais ágeis e populares de comunicação, tendo a capacidade de abranger, de forma simples, milhões de usuários. Infelizmente, foi logo explorado indevidamente, com o surgimento de spams. Spams são mensagens eletrônicas que, em sua maior parte, têm fins publicitários ou maliciosos. São enviadas sem o consentimento prévio do destinatário (GEORGALA; PALIOURAS, 2014).

No decorrer dessa dissertação, as mensagens (e-mails) não solicitadas são denominadas spam, enquanto que as mensagens (e-mails) legítimas são denominadas ham.

1.1 Origem do spam

Segundo Templetom (TEMPLETOM, 2014) (acessado em 13/08/2014), o mais antigo spam documentado foi enviado em 03 de maio de 1978. Esta primeira mensagem (e-mail) de spam foi emitida por um funcionário da *Digital Equipment Corporation* (DEC). A DEC foi comprada pela Compaq e é agora uma unidade da HP, mas outrora era o principal fabricante de microcomputadores. Seus computadores forneceram a plataforma para o desenvolvimento do Unix, C e de grande parte dos softwares que hoje compõem a Internet.

Em 1978, o correio eletrônico já era usado por um grande número de pessoas em universidades, instituições governamentais e empresas de tecnologia na rede Arpanet, precursora da Internet. Alguns anos antes, Dave Farber tinha criado o "MsgGroup", a primeira lista de discussão da Arpanet.

O funcionário Gary Thuerk, identificado apenas como Thuerk na DEC, decidiu enviar um aviso para todos os usuários da Arpanet na costa oeste dos EUA. O conteúdo da mensagem descrevia as vantagens do computador DEC-20 sobre computadores mainframes, nascendo assim o primeiro e-mail spam, embora este nome só tenha sido cunhado realmente quinze anos mais tarde.

O termo spam, segundo Templetom (TEMPLETOM, 2014), vem de uma discussão inflamada sobre envio de mensagens (e-mails) não solicitadas, onde a cena remete a uma lembrança de um programa do grupo *Monty Python Flying Circus*, que mostrava um restaurante que possuía muitos pratos, tendo SPAM como ingrediente principal no cardápio. A cada pedido feito, uma garçonete repetia várias vezes o vocábulo spam. Um grupo de Vikings, acomodados em uma mesa lateral deste restaurante, ouvindo o cantarolar repetido da garçonete, criou a seguinte canção: "spam, spam, spam de spam, spam, spam de spam, spam, amamos spam! Spam é Maravilhoso!". Assim, conclui-se que a ideia, subentendida do termo spam é designar algo que se repete e que causa grande aborrecimento. A Figura 1[1] ilustra a ideia da propaganda.

Figura 1: Ilustração da Propaganda vinculada ao spam

O produto em questão, denominado SPAM® é fabricado pela Hormel Foods desde 1930, que não aprova a associação de sua marca como algo tão nocivo à Internet e a seus

[1]Ilustração disponível em http://www.antispam.br

usuários. No site oficial do SPAM®, existe um texto esclarecendo que spam, grafado com letras minúsculas, diz respeito ao envio de mensagens não solicitadas, enquanto que SPAM®, grafado com letras maiúsculas, refere-se marca registrada pela Hormel Foods[2].

1.2 Cenário atual

Atualmente, spam significa prejuízos, transtornos e falhas de segurança. Sua quantidade cresceu rapidamente no decorrer dos anos, o que fez nascerem várias iniciativas ao combate do mesmo. Estas iniciativas vão desde grupos e associações, como antispam.br, até empresas que se dedicam a desenvolver sistemas anti-spam, como a Barracuda Tecnologia e Canit Corporation.

Segundo estudos feitos pela Kaspersky (KASPERSKY..., 2014), ilustrado pela Figura 2[3] mais de 70% do tráfego global de mensagens (email) é algum tipo de spam.

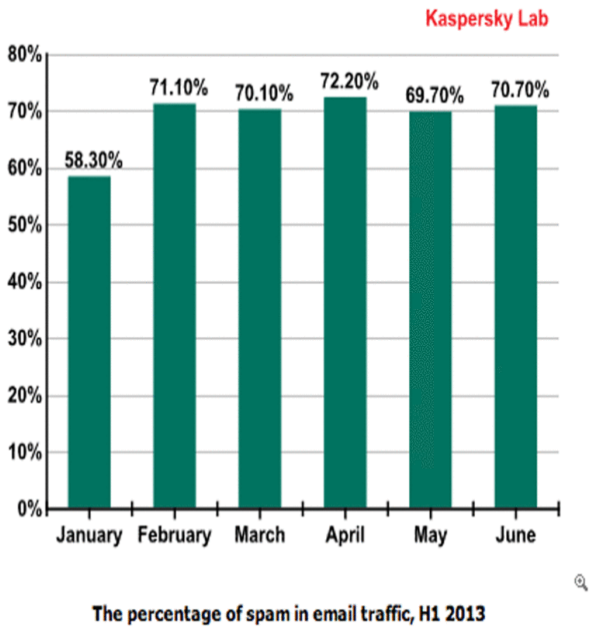

Figura 2: Gráfico do estudo da Kaspersky labs

O mesmo estudo aponta que as principais fontes de spam são máquinas infectadas

[2]Hormel Foods(http://www.hormel.com)
[3]Kaspersky Labs (http://www.kaspersky.com/about/news/spam)

que formam *botnets*. Seu principal objetivo é a propagação de software malicioso e roubo de informações.

1.3 Principais problemas causados pelo spam

Dentre os prejuízos causados pelas mensagens spam incluem-se (BUJANG, 2013):
Perda da credibilidade do sistema de mensagens: devido ao fato das mensagens legítimas aparecerem misturadas com as mensagens spam nas mailboxes dos usuários;

Não recebimento ou atraso na entrega de mensagens: devido à sobrecarga nos servidores e nos ativos de rede (*network appliances*), as mensagens podem sofrer atrasos ou mesmo não serem entregues;

Desperdício de tempo: o usuário corporativo perde tempo de trabalho produtivo para localizar mensagens legítimas em meio às spam;

Conteúdo impróprio: muitas das mensagens spam possuem conteúdo impróprio, tais como pedofilia, pornografia, ofensas raciais e apologia a drogas;

Fraudes: spammers utilizam uma técnica, chamada *phishing*, para, através de engenharia social,"pescar" dados sigilosos de usuários, como senhas, números de cartão de crédito e de documentos pessoais. Estes dados são utilizados em fraudes que podem lesar o usuário ou a instituição em que trabalha.

Custos para combate ao spam: para minimizar os prejuízos causados pelas mensagens spam, empresas lançam mão de sistemas anti-spam, sistemas estes que têm um alto custo. Para exemplificar, em uma rede como a da UNIFEI, com aproximadamente 8000 mailboxes de usuários, o spam tem um custo em torno de R$ 27.000,00 anuais somente com a licença de uso do sistema anti-spam Canit. Além do custo com sistemas anti-spam, há igualmente custos indiretos, tais como gastos com links de Internet, manutenção de servidores, energia elétrica, dentre outros, levando-se em conta a demanda extra gerada pelo spam.

1.4 A razão da existência de spam

Para se entender o porquê do envio de spams, é necessário diferenciar os tipos de spams. Os tipos mais comuns são:

Spam de marketing: visa divulgar um produto, lícito ou não, para o maior número de pessoas possíveis. Estes spams são geralmente enviados por empresas como "estratégia de marketing";

Spam incluindo software malicioso: visa disseminar um software que tem o intuito de roubar dados do usuário, ou mesmo atacar outra máquina.

Spam de *phishing*: visa obter dados pessoais ou sigilosos de usuários, através de engenharia social. Neste caso, o spammer passa-se por outra pessoa ou empresa e tenta convencer usuários a informarem-lhe dados pessoais ou sigilosos.

A razão para a existência de spam varia segundo a motivação do spammer. Qualquer que seja sua motivação, um spammer é favorecido pelo fato de que o custo para o envio de uma mensagem spam é muito baixo, quando não inexistente (KRISHNAMURTHY; BLACKMOND, 2010). O spammer consegue acesso fácil a milhões de endereços de e-mail e dissemina seus spams de forma simples, através de programas feitos para esta finalidade, deixando todo o custo do processamento das mensagens spam com a estrutura do sistema de mensagens. É comum vermos muitos usuários de um único domínio no campo "To" de uma mensagem spam. Outra estratégia comum aos spammers é o envio de spam para listas de distribuição. Neste caso, todos os usuários da lista recebem-no. Por estes motivos, fica fácil perceber que, muitas vezes, o spammer envia uma única mensagem e esta chega facilmente a milhares de usuários.

1.5 Proposta de trabalho

A proposta desse trabalho decorre da observação de um fato ocorrido no antigo CPD da Escola Federal de Engenharia de Itajubá (EFEI), hoje Universidade Federal de Itajubá (UNIFEI). Na época, a organização do domínio da então EFEI, domínio "efei.br",

era desmembrado pelos institutos da escola federal. Assim, cada instituto possuía seu subdomínio, na forma "*instituto*.efei.br" e, consequentemente, seu servidor de mensagens. Os subdomínios da EFEI eram alvos maciços de spam, consumindo não só uma grande quantidade de recursos computacionais, bem como tornando o serviço de mensagem precário. No ano de 2001, a Escola Federal de Engenharia de Itajubá tornou-se Universidade Federal de Itajubá. Com isto, foram extintos os subdomínios dos institutos, agrupando-os no novo domínio "unifei.edu.br".

Os antigos servidores dos institutos foram então desativados. Após algumas semanas de operação do novo servidor de e-mails com o novo domínio, um professor de um dos institutos teve necessidade de receber um e-mail que lhe era endereçado no seu antigo subdomínio. Para isto, ativou-se novamente o servidor do referido instituto, que, curiosamente, recebia a maior quantidade de spam dentre os antigos subdomínios. De forma surpreendente, esta mensagem foi recebida junto com outras mensagens legítimas pelo servidor, sem a presença de qualquer mensagem spam. Após uma breve pesquisa nos logs de tráfego interno e junto à Rede Nacional de Pesquisa (RNP), órgão que provê o link de Internet às instituições federais de ensino e pesquisa, comprovou-se que, pelo fato do subdomínio deste instituto não estar mais ativo, as mensagens de spam eram devolvidas para seus remetentes e estes, com o intuito de evitar o tráfego de mensagens devolvidas e o consequente aumento de processamento em suas máquinas, retiraram os destinatários do subdomínio de suas listas de distribuição.

A observação deste caso fez com que surgisse a ideia da proposta desta dissertação. A proposta é, portanto, desenvolver um servidor de mensagens que permita aos usuários marcar quais remetentes lhes são indesejados. Desta forma, as mensagens spam são devolvidas aos remetentes, tal como o caso ocorrido, ou seja, como se o usuário ou domínio não existam.

A grande vantagem oferecida por esta proposta decorre do fato de que as mensagens spam com endereço de remetente fixo não são processadas, armazenadas e enviadas aos usuários. De fato, são recusadas prontamente durante a negociação, através do protocolo SMTP, entre os servidores de envio e destino. Assim, onera-se o *spammer* e não à instituição destino do spam.

Faz-se necessário ressaltar o fato de que muitos e-mails spam decorrem de malas dire-

tas de empresas de comércio eletrônico, que possuem um endereço fixo de e-mail. Ao efetuar uma compra nestas empresas, por exemplo, o cliente tem seu e-mail automaticamente cadastrado, sem seu consentimento, nas malas diretas destas empresas. A redução deste tráfego indesejado, por si só, justifica a implementação da proposta desta dissertação.

1.5.1 Conteúdo da dissertação

A dissertação é dividida em cinco capítulos. O capítulo 2 apresenta os conceitos teóricos que embasam esta dissertação. O capítulo 3 apresenta uma revisão de trabalhos relacionados à área de sistemas antispam. O capítulo 4 apresenta a metodologia empregada, os estudos realizados e os resultados obtidos. Por fim, o capítulo 5 conclui o trabalho, apontando direções para pesquisas futuras.

2 Revisão teórica

2.1 Zimbra

O Zimbra é um servidor de e-mails de código fonte aberto. É usado em mais de 5.000 instituições do setor público e privado e por mais de 100 milhões de usuários em mais de 140 países. Implementa todos os protocolos de e-mail (IMAP, IMAPS, POP, SMTP, SMTPS, dentre outros), catálogo de endereços, calendário/agenda e compartilhamento de arquivos e tarefas. Pode ser acessado a partir do cliente Zimbra Web, clientes de e-mail que implementam o protocolo POP ou IMAP.

Por implementar uma solução completa, incluindo módulos de gerência e de *backup*, o Zimbra é considerado uma plataforma de excelência em correio eletrônico, como descrito nos estudos de casos da Dell (DELL..., 2014) Texas Intruments (TEXAS..., 2014). Conta com a versão gratuita Zimbra Communit e a versão paga Zimbra Collaboration. A diferença entre as duas versões é que a versão paga conta com um módulo de backup de mensagens avançado e suporte técnico de implantação e administração pela Zimbra Software (ZIMBRA..., 2014). Neste trabalho foi usada a versão gratuita. No final de 2012, a Zimbra Software foi adquirida pela VMware.

2.1.1 Arquitetura do Zimbra

O Zimbra possui uma arquitetura modular de software. Os módulos foram desenvolvidos tanto por terceiros quanto pela Zimbra Software.

O módulo Zimbra server implementa os componentes para serviço *mailbox*, incluindo os sub-componentes auxiliares. Cada conta é associada com uma mailbox que contém todas as mensagens de correio, arquivos anexos, documentos, contatos, calendário e arquivos de

colaboração para a conta de correio.

O módulo Zimbra Core inclui as bibliotecas, utilitários, ferramentas de monitoramento e arquivos de configuração básica.

O modulo Zimbra LDAP usa o software OpenLDAP, um serviço de diretório de código aberto para a autenticação do usuário, lista de endereços e atributos de configuração.

O modulo Zimbra MTA Postfix é um agente de transferência de correio – *Mail Transfer Agent* – (MTA) de código aberto que recebe e-mail via SMTP e encaminha cada mensagem para o caixa de correio apropriada no servidor Zimbra usando o protocolo LMTP (*Local Mail Transfer Protocol*)

e Finalmente o modulo Zimbra Logger é o módulo responsável por integrar todos os registro do servidor Zimbra. Utiliza a ferramenta de código aberto syslog para esta tarefa

A Figura 3[4] apresenta os módulos principais citados acima e os módulos opcionais do Zimbra.

[4]Disponível em http://files.zimbra.com/website/docs/8.0/Zimbra_NE_Admin_Guide_8.0.6.pdf

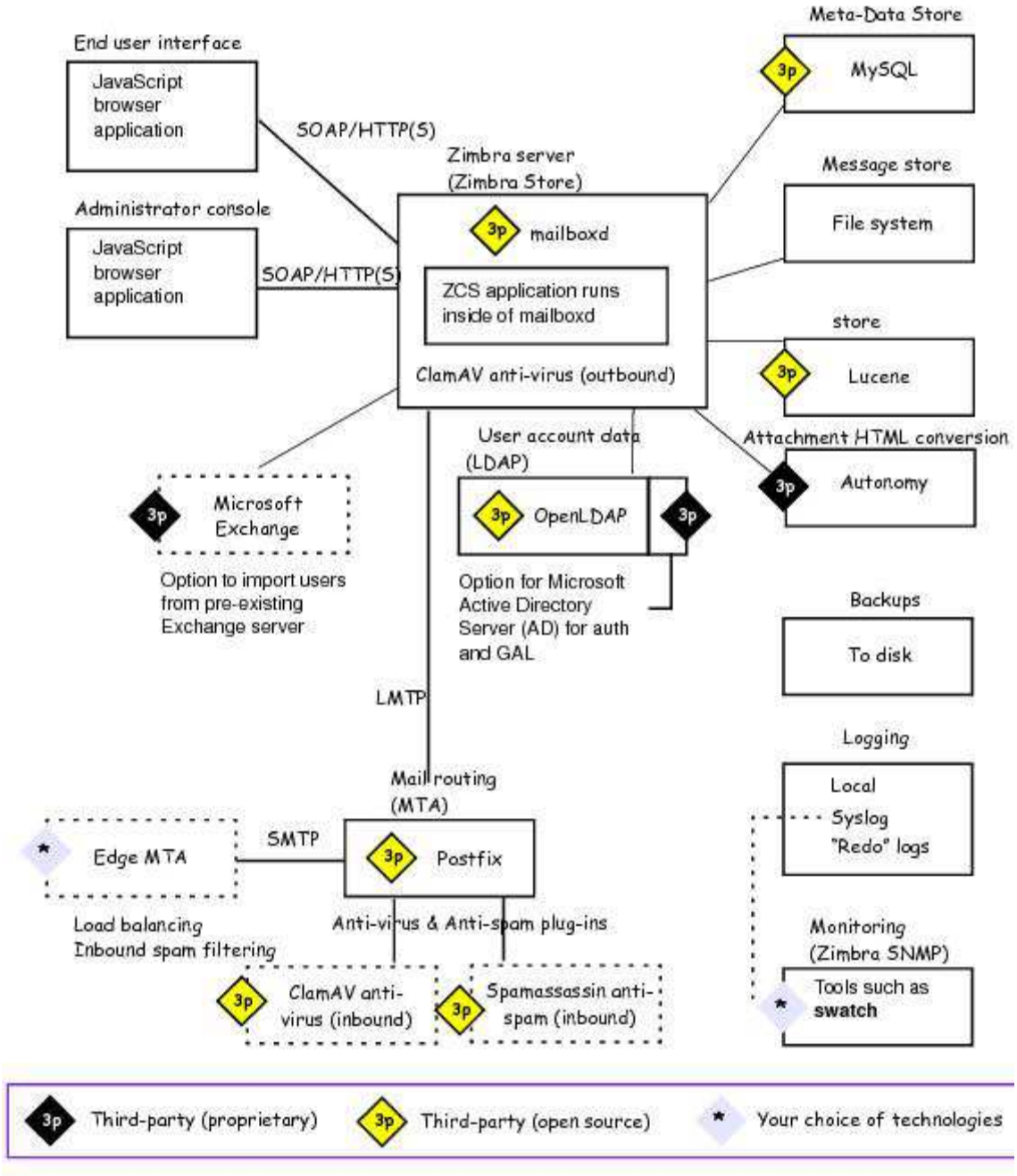

Figura 3: Arquitetura do Servidor Zimbra

A Figura 4[5] apresenta a sequência de recebimento de mensagens realizada pelo Zimbra. Neste trabalho, o filtro anti-spam *Edge MTA* foi desabilitado. Assim, as mensagens são recebidas diretamente pelo módulo Zimbra MTA, que inclui o módulo Mail routing.

[5]Disponível em http://files.zimbra.com/website/docs/8.0/Zimbra_NE_Admin_Guide_8.0.6.pdf

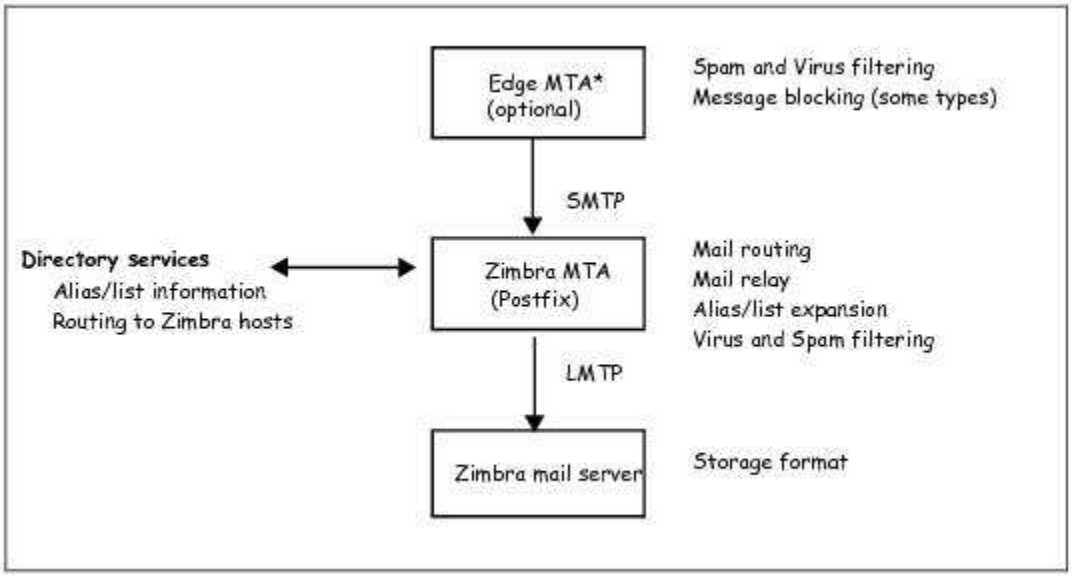

Figura 4: Fluxo de recebimento de mensagens no Zimbra

2.2 Postfix

O Postfix é um servidor de e-mail desenvolvido por Wietse Venema. É um servidor rápido e seguro. Pode ser executado sobre diversos sistemas operacionais UNIX e sobre o Linux. É implementado na linguagem C e faz uso de funções da biblioteca POSIX.1 e de sockets BSD. Além disto, o Postfix exige certos requisitos do sistema de arquivo instalado, requisitos estes relacionados no anexo A.

2.2.1 Arquitetura do Postfix

A arquitetura do Postfix é dividida em dois estágios (POSTFIX, 2010). O primeiro é responsável pelo recebimento das mensagens e o segundo, pela entrega das mesmas.

Recebimento de e-mails pelo Postfix

Quando o Postfix recebe uma mensagem, ele a encaminha para sua fila de entrada. A Figura 5[6] mostra os principais processos do estágio de recebimento de mensagens.

[6]Disponível em http://www.postfix.org/OVERVIEW.html

Figura 5: Arquitetura de recebimento de mensagens do Postfix

O Postfix recebe mensagens externas através do processo deamon smtpd ou qmqpd. Mensagens locais são recebidas pelo processo sendmail que as envia para a fila local maildrop. As mensagens locais ou externas são validadas, em relação a correção de seu formato, pelo processo *cleanup*. Por fim, o processo clanup encaminha as mensagens corretas para a fila de entrada (*incoming queue*).

2.2.2 Entrega de e-mails pelo Postfix

A Figura 6[7] mostra os principais processos do estágio de entrega de mensagens.

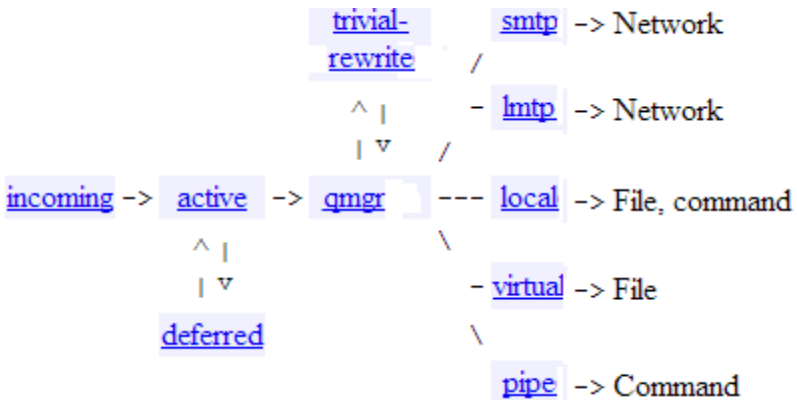

Figura 6: Arquitetura de entrega de mensagem do Postfix

O processo qmgr é responsável pela entrega das mensagens aos processos smtp, lmtp,

[7]Disponível em http://www.postfix.org/OVERVIEW.htm

local, virtual e pipe que as encaminham para os destinatários. Os processos *discard* e *erro*, não mostrados na Figura 7, simplesmente descartam ou devolvem as mensagens.

2.3 MySQL Server

O MySQL Server é o sistema gerenciador de banco de dados mais utilizado. Atualmente, é mantido pela Oracle®.

MySQL é um sistema flexível. Pode ser usado tanto em aplicações embarcadas quanto em sistemas redundantes de alta disponibilidade. (SCHWARTZ PETER ZAITSEV, 2012). InnoDB e MyISAM são os dois mais populares mecanismos de armazenamento do MySQL.

O InnoDB é o motor padrão do MySQL das versões 5.2 em diante. Foi projetado para processar muitas transações de curta duração. Seu desempenho e suas funcionalidades para recuperação automática de falhas fizeram-no popular também para armazenamento não transacional (SCHWARTZ PETER ZAITSEV, 2012).

O MyISAM é o motor padrão do MySQL nas versões anteriores à 5.2. Não suporta transações ou bloqueios (locks) de linhas de tabelas. Sua maior deficiência é indubitavelmente o fato de não oferecer quaisquer garantias contra falhas. É usado em aplicações não transacionais e de leitura de dados em bancos de dados de pequeno porte(SCHWARTZ PETER ZAITSEV, 2012).

2.4 O protocolo SMTP

SMTP é o protocolo para troca de mensagens de e-mail entre servidores. É capaz de transportar mensagens através de múltiplos servidores em múltiplos domínios. Esta capacidade é conhecida como "SMTP *mail relay*"(INTERNET ENGINEERING TASK FORCE, 2008b). A estrutura básica do protocolo SMTP pode ser vista na Figura 7[8]

[8]Retirada da RFC 5322

Figura 7: Estrutura básica do Protocolo SMTP

Para transmitir mensagens, um cliente SMTP estabelece uma conexão com o servidor SMTP de seu domínio. O encaminhamento da mensagem do cliente para o servidor pode ser feito de várias formas, segundo às necessidades de cada instituição.

Como mencionado anteriormente, um servidor SMTP pode ser o servidor de destino da mensagem ou um servidor intermediário. Caso seja um servidor intermediário, ele assume o papel de um cliente SMTP, fazendo a retransmissão da mensagem para o servidor de destino ou um outro servidor intermediário. Comandos do protocolo SMTP são enviados pelo cliente SMTP para o servidor SMTP. As respostas aos comandos são enviadas do servidor para o cliente.

A Figura 8[9] apresenta uma série de comandos emitidos pelo cliente para especificar a origem, destino do e-mail além de comandos para transmitir o conteúdo do mesmo. O servidor responde a todos os comandos do cliente. Estas respostas indicam que o comando foi aceito, que comandos adicionais são esperados, ou que ocorreu uma condição de erro. Uma vez transmitida a mensagem, o cliente SMTP encerra a conexão com o servidor.

[9]Disponível em http://resources.infosecinstitute.com/phishing-techniques-similarities-differences-and-trends-part-ii-targeted-phishing/

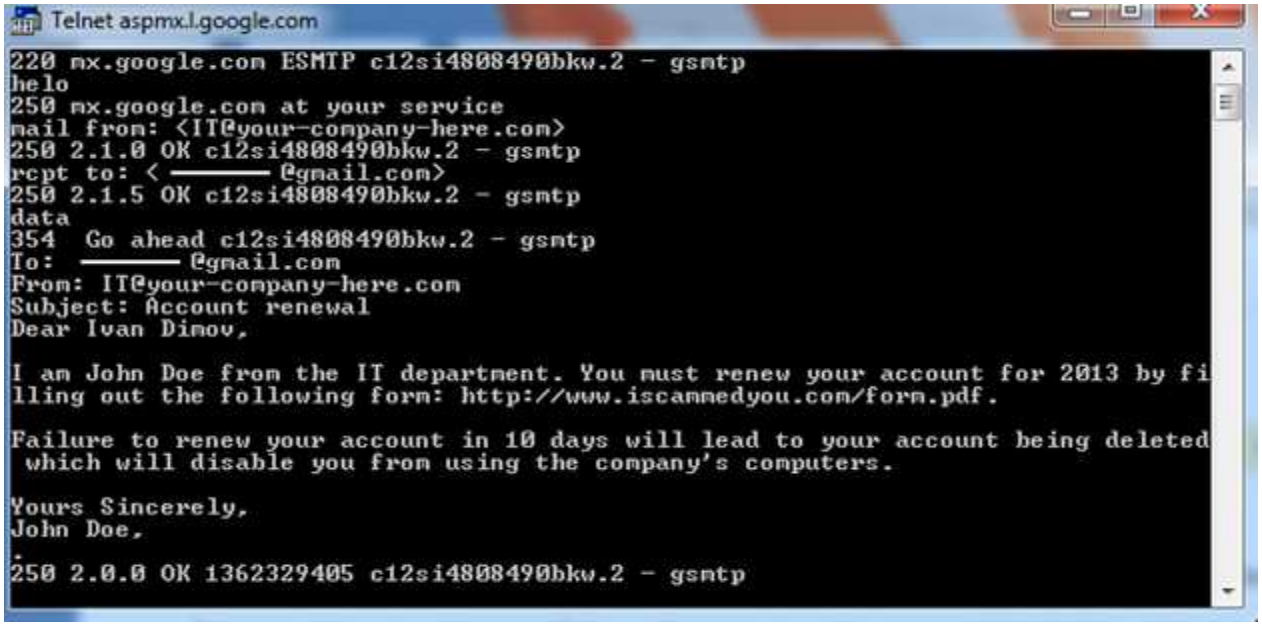

Figura 8: Exemplo de envio de e-mail via comandos do protocolo SMTP

3 Revisão bibliográfica

Tran e Armitage (TRAN; ARMITAGE, 2004) afirmam que a monitoração, em tempo real, do tráfego TCP/IP na porta SMTP de um servidor permite a detecção de mensagens spam. O tráfego é monitorado a fim de detectar o aumento do número de pacotes causados pelo tráfego spam em cada requisição SMTP. Para a monitoração, utilizam a ferramenta MT Proxy, proveniente do FreeBSD, que faz uso do ipfw — um firewall nativo do FreeBSD — e do *dummynet* — uma ferramenta que prioriza ou posterga o tráfego. O monitoramento é realizado por um servidor proxy que faz a interface entre o tráfego de entrada e o servidor de e-mail. O proxy recebe as requisições SMTP em lugar do servidor SMTP. Desta forma, o proxy limita as requisições SMTP detectadas como spam. As requisições detectadas como spam são armazenadas em listas negras que podem ser tanto acessadas localmente ou através da Internet. Os autores também afirmam que a detecção de um e-mail spam pela ferramenta MT Proxy depende do tamanho do e-mail. E-mails com tamanho inferior a 20 Kbytes são mais difíceis de serem detectados. Já para e-mails de com tamanho de 200 Kbytes foi observado uma redução, resultado da filtragem dos spam, de 196608Kb por segundos na taxa de entrega de e-mail.

Wang e Chen (WANG; CHEN, 2007) analisam os cabeçalhos das mensagens típicas de spam para usá-los como padrões para a detecção de spam. Os cabeçalhos possuem campos, tais como endereço do remetente, do destinatário e data de envio, que fornecem informações relevantes para a detecção de spam. Os autores verificam que o uso de um único campo é ineficiente para a detecção, propondo, assim, o uso combinado dos mesmos. Com diversas combinações de campos do cabeçalho, chegaram a uma eficiência de filtragem de 88.5%, resultados próximos de sistemas que fazem uso do conteúdo da mensagem para a detecção de spam.

Krishnamurthy e Blackmond (KRISHNAMURTHY; BLACKMOND, 2010) propõem uma abordagem que atribui custos ao envio de mensagens. Nesta abordagem, mensagens ham te-

riam seu custo ressarcido enquanto as spam não. A abordagem é análoga ao sistema de correio tradicional, onde o envio de cada mensagem está associado a uma pequena taxa de envio, representada por um selo. Na abordagem dos autores, os selos são emitidos e gerenciados por uma entidade autônoma, independente dos provedores ou usuários. Os provedores de e-mails compram lotes de selos que são repassados a seus usuários. Um selo é gasto para cada mensagem enviada. Caso esta seja uma mensagem de spam, o selo é perdido. Caso contrário, é restituído ao usuário. Os autores também abordam aspectos de segurança no uso dos selos, sugerindo formas de cancelamento e reutilização dos mesmos. Uma dos problemas desta abordagem consiste em sua adoção em âmbito mundial, devido à diversidade das leis políticas e econômicas existentes.

Agrawal, Kumar, e Molle (AGRAWAL; MOLLE, 2005) propõem o uso de roteadores para a filtragem de tráfego spam. A filtragem pode ser realizada pelos próprios roteadores ou por servidores específicos. Neste caso, os servidores recebem o tráfego de mensagens, analizam-no e devolvem ao roteador somente as mensagens legítimas. A detecção de spam é realizada por meio de um modelo Bayesiano. A taxa de detecção obtida em seus experimentos alcança 60%.

Ramachandran e Feamster (RAMACHANDRAN; FEAMSTER, 2006) analisam as quatro técnicas de envio de spam mais empregadas — spam direto, open relays e proxies, botnets e BGP spectrum agility. Com isto, concluem que a maioria dos spams é originada de botnets. No estudo das botnets, observam que a ação de worms e a emissão de spam possuem faixas (ranges) de endereçamento IP similares. Observam, igualmente, que 70% dos servidores que disseminam spam exploram worms instalados em máquinas infectadas. No entanto, somente 25% destas máquinas infectadas são saneadas, ou sejam, têm seus worms removidos. Por fim, os autores afirmam que o uso de listas negras em sistemas anti-spam é ineficiente devido ao tempo necessário para a atualização dos endereços IP que catalogam.

Marsono, El-Kharashi, e Gebali (MARSONO; GEBALI, 2009b) propõem um sistema que detecta spam, analisando os datagramas IP, durante as sessões SMTP. O tráfego classificado como spam é enviado para uma fila com prioridade inferior à do tráfego normal, retardando o tráfego de spam. Segundo os autores, a classificação através dos datagramas IP possui baixo índice de falsos positivos e permite uma rápida classificação do e-mail. Deste modo, a maioria dos e-mails spam não é enfileirada por MTAs, reduzindo-se o atraso devido ao enfileiramento e a probabilidade de perda de e-mails legítimos. Os resultados

obtidos pelos autores confirmam que o sistema proposto diminui sensivelmente o atraso no tráfego das mensagens legítimas.

Em um outro estudo, Marsono, El-Kharashi, e Gebali (MARSONO; GEBALI, 2009a) propõem aplicar seu sistema de detecção de spam, descrito no artigo anterior, tanto na entrada quanto na saída do servidor de e-mails. Os autores observam que atualmente a detecção de spam na saída do servidor raramente é realizada devido aos custos. Através da simulação de seu sistema sobre conjuntos de e-mails, os autores observam que a classificação de e-mails na camada 3 apresenta uma taxa de falsos positivos inferior a 0,5%.

Goodman (GOODMAN, 2004) afirma que o uso de listas negras não é conveniente. Uma de suas desvantagens no seu emprego é o fato de que bloqueiam domínios inteiros da Internet, e não somente os spammers que atuam nestes domínios. O autor sugere duas alternativas mais eficientes. A primeira, consiste na consulta ao servidor DNS do domínio do remetente da mensagem, de forma a garantir sua autenticidade. No entanto, esta alternativa, já implementada pelo SPF[10], falha devido a sua pouca adesão pelos domínios da Internet. A segunda alternativa consiste na inserção, pelo servidor de origem, de uma marca encriptografada do cabeçalho do pacote SMTP, de forma a garantir a procedência do e-mail.

Harker (HARKER, 1997) propõe o uso de regras implementadas no servidor de e-mails *sendmail* para filtrar e-mails spam. As regras fazem uso de um banco de dados para armazenar as informações extraídas dos e-mails, tais como, endereços de remetentes, nomes de servidores de e-mail, seus domínios ou endereços IP. O autor destaca que, apesar de eficazes, os filtros têm de passar por alterações constantes. Além disto, há dois outros sérios inconvenientes na proposta. Primeiro, devido a inerente complexidade das regras, os procedimentos de criação e manutenção delas são difíceis para a maioria dos usuários. Segundo, a proposta não reduz os custos devidos ao recebimento dos e-mails spam, tais como uso dos links de Internet e sobrecarga nos ativos de rede, uma vez que o filtro, implementado no servidor, só opera após o recebimento integral dos e-mails.

Lawrence e Bloom (PFLEEGER; BLOOM, 2005) manifestam suas preocupações com o fato de que o problema causado por spam atinge dimensões não previstas, sendo assunto de órgãos reguladores e de órgãos governamentais. Segundo o autor, é difícil determinar o que é spam e o que é propaganda ou marketing legal, pois as definições são vagas e cir-

[10]Sender Permitted From. http://spf.pobox.com.

cunstanciais, variando conforme opiniões pessoais. Portanto, sugere que seja o usuário a definir o que deseja ou não em sua caixa postal. Critica, igualmente, o fato de que as leis usualmente possuem um embasamento técnico deficiente, responsabilizam, de forma indevida, provedores de serviços de Internet, propõem soluções impraticáveis, causam custos adicionais e têm abrangência estadual ou nacional. Como o spam atinge dimensões globais, as leis têm praticamente nenhum efeito. Por fim, o autor analisa propostas que prevêem ônus financeiro aos *spammers*, afirmando que estas são eficazes.

Clayton (CLAYTON, 2004) propõe a criação de arquiteturas de sistemas anti-spam baseadas nas arquiteturas de sistemas de detecção de intrusão. Destaca que tais arquiteturas sejam mais eficientes devido às formas empregadas pelos *spammer*s para ofuscar suas mensagens. Sistemas de detecção de intrusão analisam logs do sistema monitorado para aprender seu comportamento padrão. Ao detectar um desvio neste padrão, uma ação é executada. Assim, propõe o uso dos logs dos servidores de e-mail para identificar o comportamento padrão dos e-mails legítimos e, com isto, conseguir distingui-los dos *spams*. A proposta possui, no entanto, dois sérios inconvenientes. Primeiro, é necessário que os clientes façam uso de endereços IP estáticos, o que não é comumente usado por grandes provedores. Segundo, a proposta não reduz os custos devidos ao recebimento dos e-mails spam, tais como uso dos links de Internet e sobrecarga nos ativos de rede, uma vez que a detecção dos *spams* é realizada após o recebimento dos mesmos.

4 A proposta

Durante o processo de recepção do e-mail, conforme estabelece o protocolo SMTP (INTERNET ENGINEERING TASK FORCE, 2008a), é verificada a existência do domínio e do usuário de destino. A Figura 9 descreve o processo de recepção.

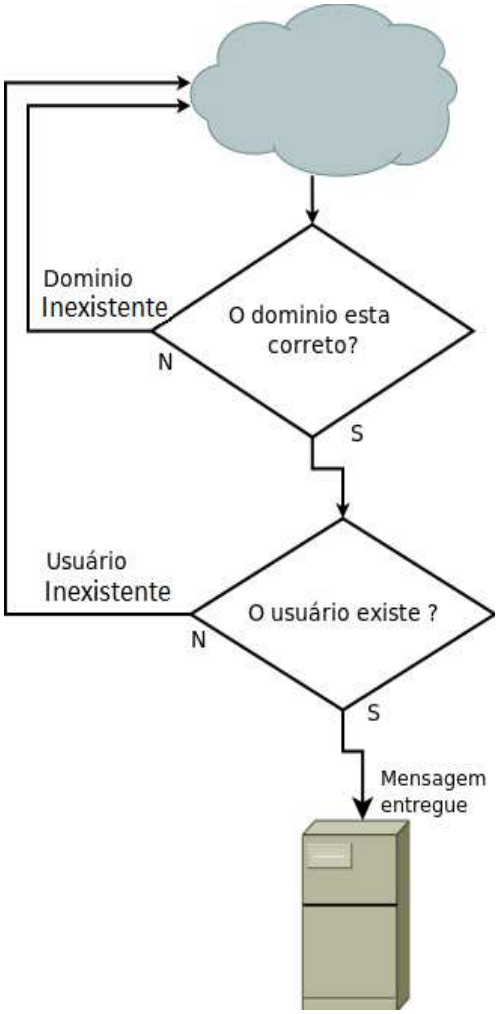

Figura 9: Fluxograma simplificado de verificação de recebimento de mensagem do protocolo SMTP

A proposta envolve a modificação do processo de recepção do e-mail, acrescentando-lhe mais uma verificação. Esta verificação consiste em consultar uma lista negra, definida pelo usuário, para averiguar a existência do endereço de e-mail do remetente. Caso exista, o e-mail é recusado, gerando ao remetente a mensagem de "destinatário inexistente". A Figura 10 descreve o processo de recepção modificado.

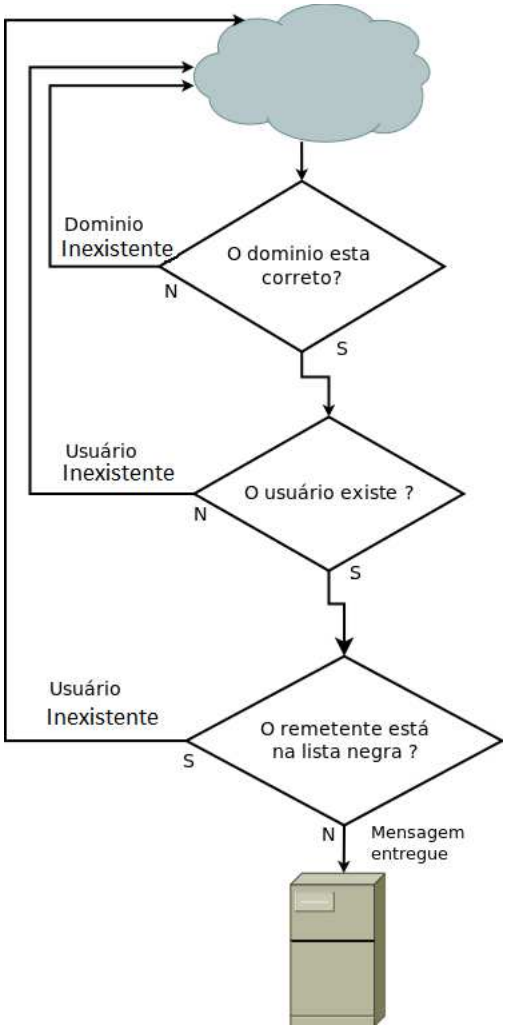

Figura 10: Fluxograma simplificado com a modificação sugerida na verificação de recebimento de mensagem do protocolo SMTP

A modificação do protocolo SMTP apresenta duas vantagens. A primeira, consiste na geração de custos ao spammer, uma vez que seu servidor efetuará processamentos extras para tratar o spam recusado. A segunda vantagem consiste no fato de que, em consequência da recusa, os spammers tendem a remover o endereço do usuário de suas listas.

5 Experimentos e Resultados

5.1 O software

A proposta foi implementada no servidor de e-mail Zimbra 8.0 (ZIMBRA..., 2014). Os filtros de anti-spam e anti-vírus do servidor foram desativados. O Zimbra 8.0 utiliza o Postfix 2.7.5 (POSTFIX..., 2014) como *mail transfer agent*(MTA). A implementação consistiu na criação da função check_spam no arquivo smtpd_check.c do Postfix. A função check_spam verifica a existência do endereço do remetente na lista negra. A lista negra é implementada como uma tabela, denominada spam, no sistema gerenciador de banco de dados MySQL 5.1 do Zimbra. A função check_spam é chamada pela função do Postfix "reject_unknown_address", que verifica se o domínio e o usuário existem.

A Figura 11 apresenta o banco de dados do Zimbra com a inclusão da nova tabela spam. As modificações no código fonte do Postfix descritas no apêndice A. No total, foram incluídas 30 linhas no código fonte original.

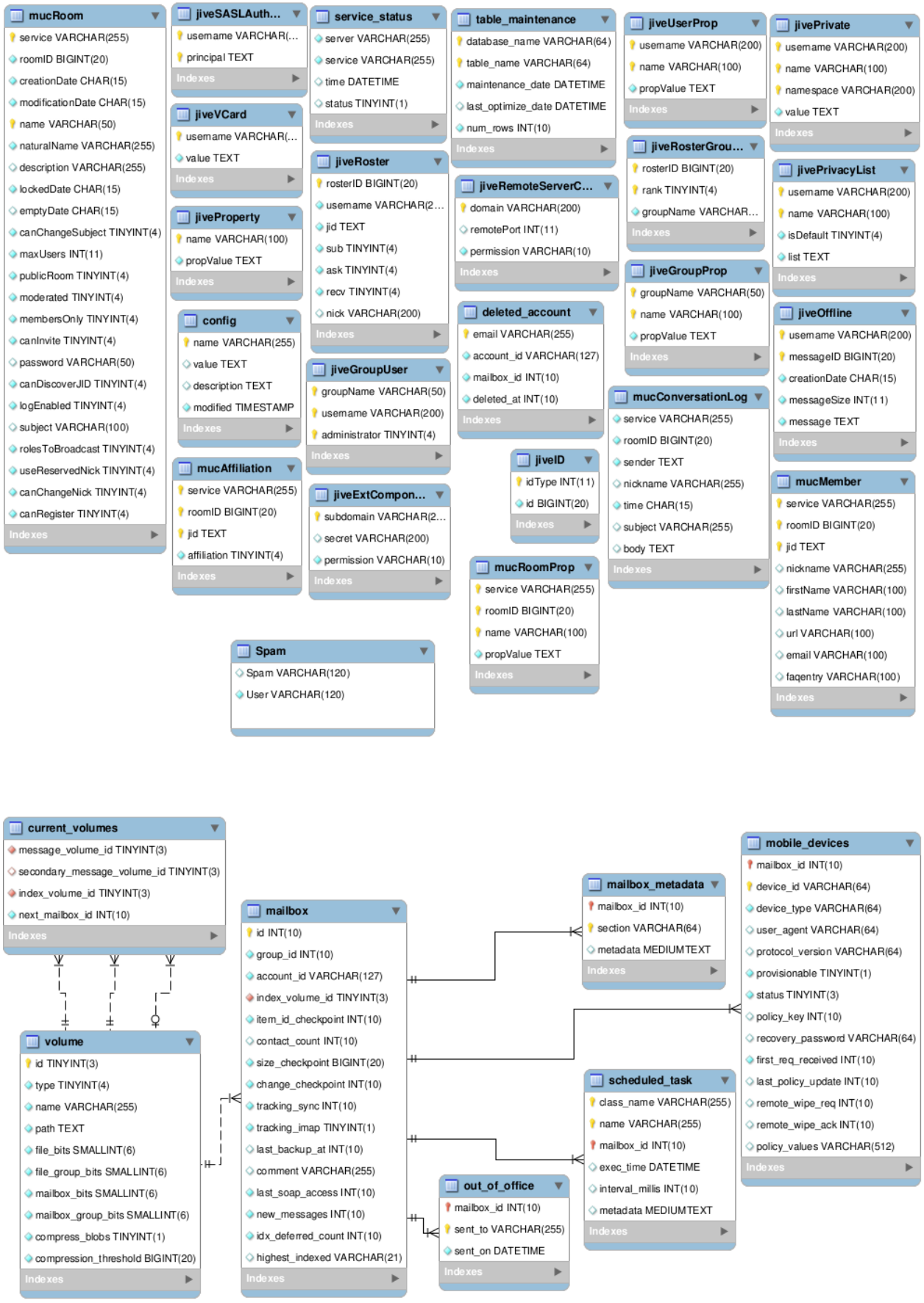

Figura 11: Modelo do Banco de Dados Zimbra modificado para suportar a Implementação proposta

Além da modificações no servidor Zimbra, foram também criadas diversas ferramentas para manipulação de dados. Algumas destas ferramentas são descritas a seguir.

A ferramenta *monitor* foi criada para o monitoramento sincronizado dos dois servidores e do *host*. Esta ferramenta monitora o tempo de *uptime*, uso de memória, uso de processador, estado da interface de rede e tráfego de entrada e saída dos três computadores. Além disso, ela monitora o tamanho das filas *incoming*, *maildrop*, *active*, *deferred*, *hold* e *corrupt* dos dois servidores Zimbra. Os dados obtidos de cada computador são gravados em um arquivo *logmail*. A ferramenta é executada a cada dois segundos, através de um daemon. O código da ferramenta *monitor* e a inicialização de seu daemon monitord estão descritos no apêndice D.

A ferramenta *enviaemail* foi criada para envio de e-mails. É executada no *host*. Realiza duas tarefas. A primeira, consiste na inicialização do arquivo *logmail* em cada computador. A segunda tarefa consiste no envio de e-mails do *host* para os dois servidores Zimbra, de forma sincronizada. O envio de e-mails é definido através de entradas em um arquivo de configuração *catalogo*. As entradas neste arquivo definem a quantidade de e-mails a serem enviados, a quantidade de destinatários e a de remetentes. Os diversos experimentos realizados foram definidos através de diversas configurações deste arquivo.

Os e-mails são enviados através de sequências de comandos SMTP dentro de uma conexão telnet na porta 25 do MTA Postfix que executa no *host*. Dois arquivos de logs são criados para registrar o sucesso ou o insucesso no envio de cada e-mail. A ferramenta *enviaemail* e exemplos de arquivos *catalogo* são apresentados no apêndice E.

A ferramenta *verificaatraso* foi criada para calcular o atraso na entrega de mensagens entre os servidores. Utiliza os dados armazenados no arquivo *mailbox.log*. O arquivo *mailbox.log* é gerado pelo Zimbra. Registra os eventos ocorridos nas caixas postais dos usuários. O horário de entrega dos e-mails é o único evento de interesse. A ferramenta *verificaatras*o está disponível no apêndice G.

A ferramenta *geraestatisticas* foi criada para geração dos resultados dos experimentos. Utiliza os dados armazenados no arquivo *logmail*. A ferramenta extrai os dados contidos no arquivos logmail dos três computadores. Seleciona os dados de uso de processamento, quantidade de memória livre, quantidade de processos em execução, tráfego de entrada e

saída da interface de rede, tamanho das filas de e-mail em cada experimento e calcula a média desses dados em cada experimento. A ferramenta *geraestatisticas* está disponível no apêndice G.

5.2 O hardware

A arquitetura de hardware é descrita na Figura 11. Consiste em dois servidores DELL T110 idênticos, com processador Intel Xeon de 4 núcleos de 2Ghz e 32Gb de memória RAM, e um *host* Itautec, com processador Pentium 4 de 3 GHz e 512Mb de memória RAM. Os servidores e o *host* utilizam o sistema operacional Ubuntu Server 10.04.

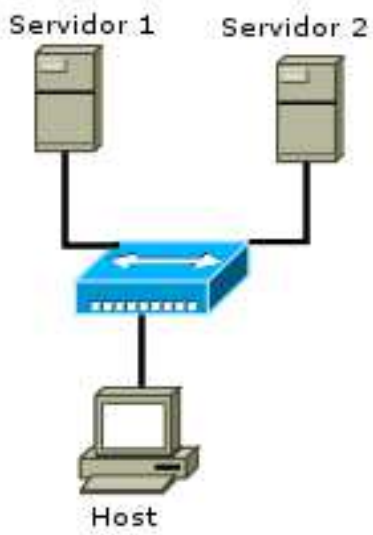

Figura 12: Arquitetura de hardware

Ambos os servidores executam o Zimbra Server 8.0, que inclui o Postfix 2.7.5 como MTA. O módulo smtpd do Postfix que executa no servidor 1 contém a modificação do protocolo SMTP. O servidor 2 executa o código original do Postfix. O *host* executa o MTA Postfix 2.7.5.

5.3 Experimento preliminar

Este experimento teve por objetivo verificar se as funcionalidades do Zimbra foram mantidas após as mudanças no módulo *smtpd* do postfix. Três e cinco mil e-mails foram enviados, de forma simultânea, aos dois servidores. Para este teste nenhum e-mail foi cadastrado como spam. O uso de processamento, quantidade de memória livre, tráfego de entrada e saída da interface de rede e o tamanho das filas de e-mail foram medidos. Os resultados são apresentados na tabela 1 e, graficamente, nas Figuras de 13 a 17.

Tabela 1: Médias dos resultados do experimento preliminar- Mysql eng. MySAM

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
3000	431	349322	17	4578	0
5000	755	611314	19	4583	0
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
3000	442	349650	18	4497	0
5000	774	611888	21	4521	0
Atraso de entrega no servidor 2 em relação ao servidor 1					
Envios	Tempo (segundos)				
3000	3				
5000	7				

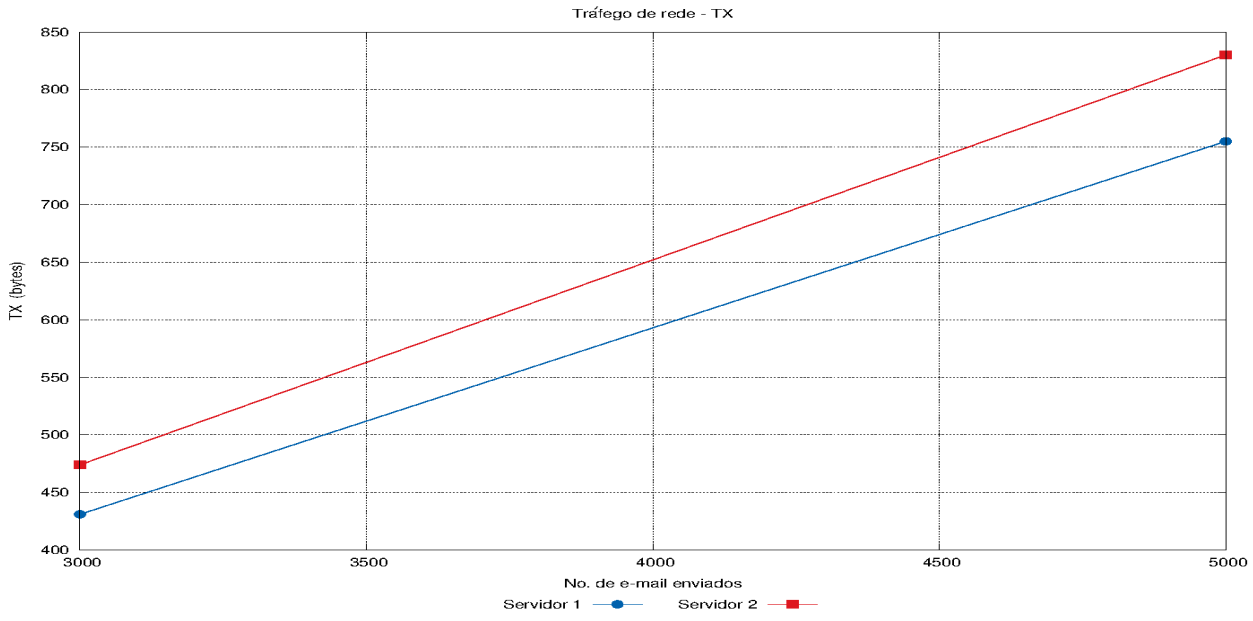

Figura 13: Média de transmissão das interfaces de rede

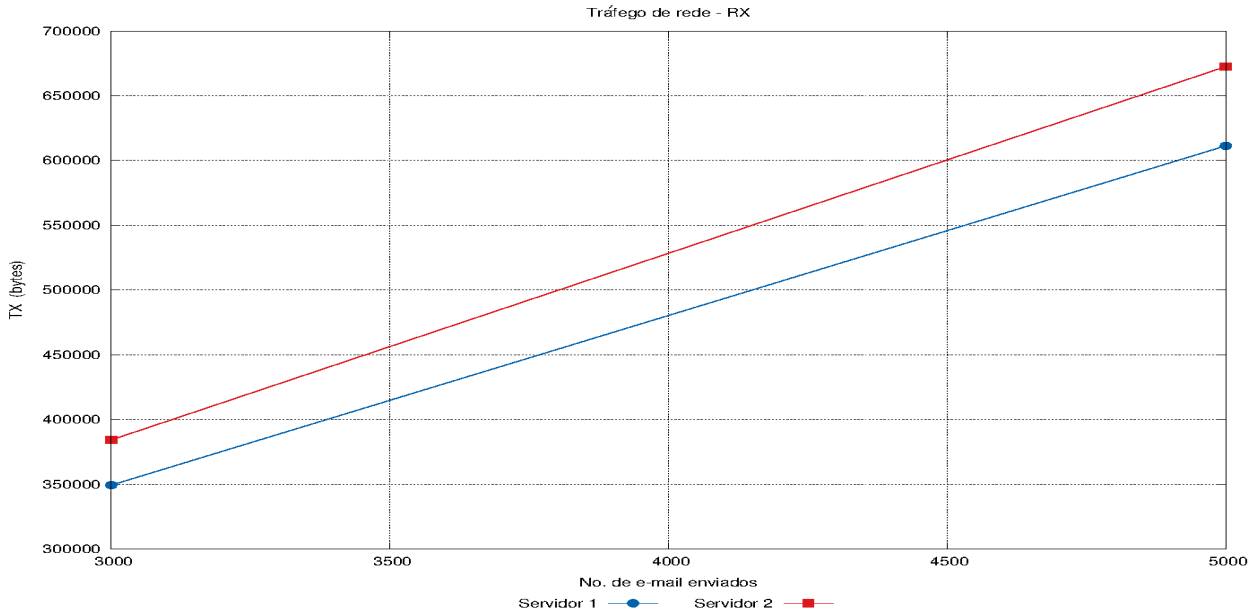

Figura 14: Média de recepção das interfaces de rede

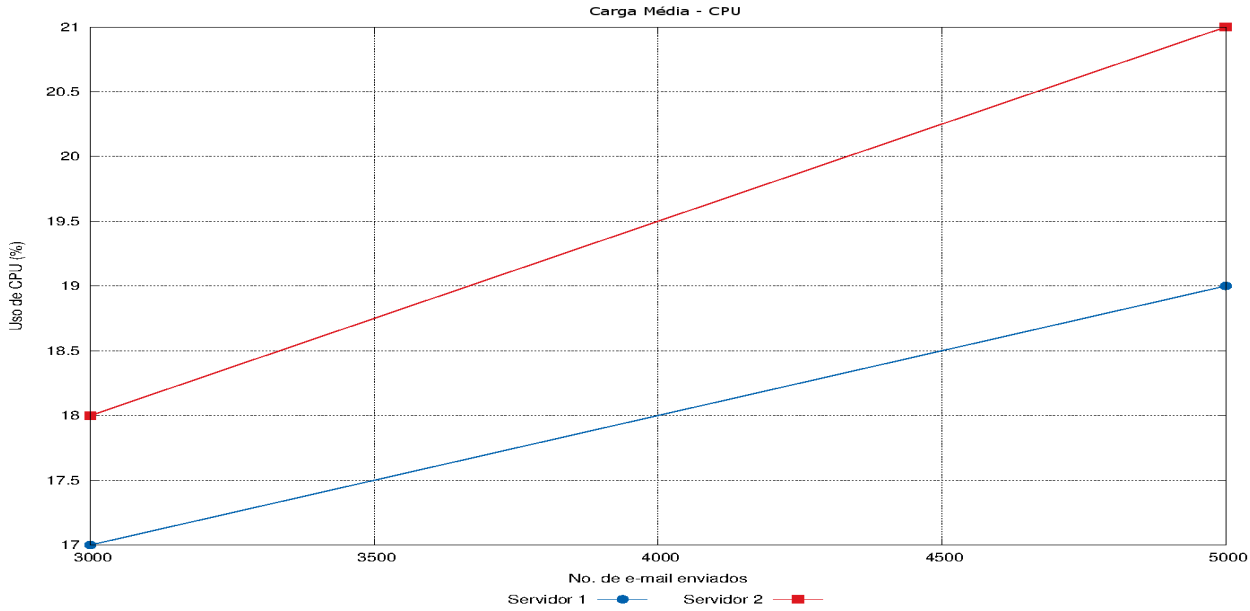

Figura 15: Carga média de uso de CPU

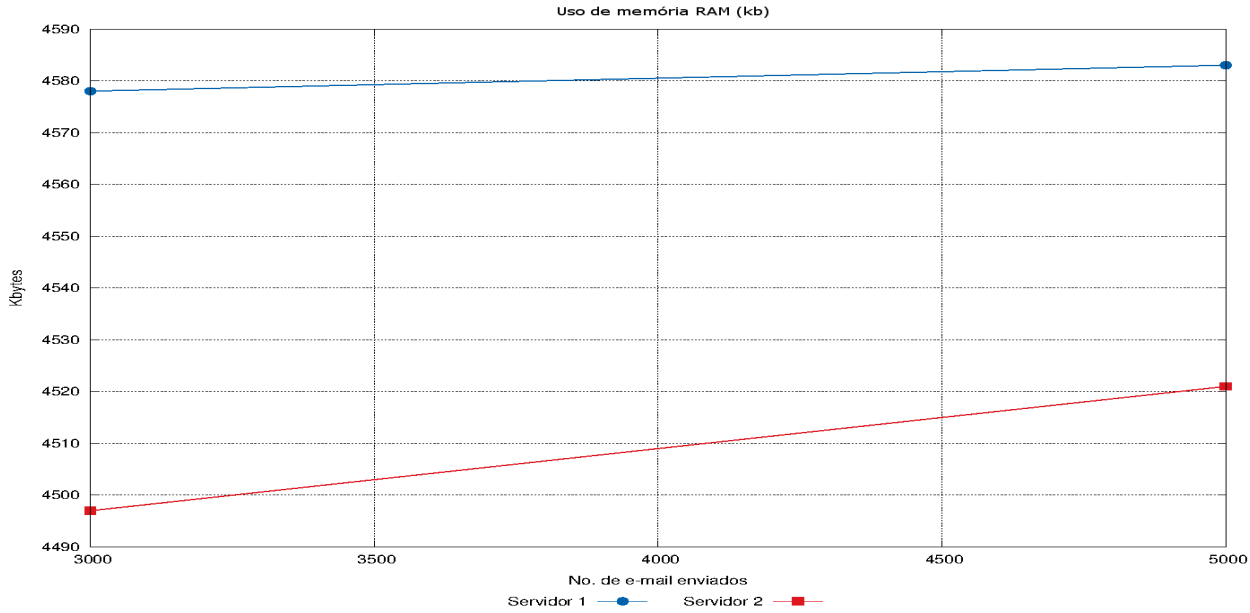

Figura 16: Média de uso de memória RAM

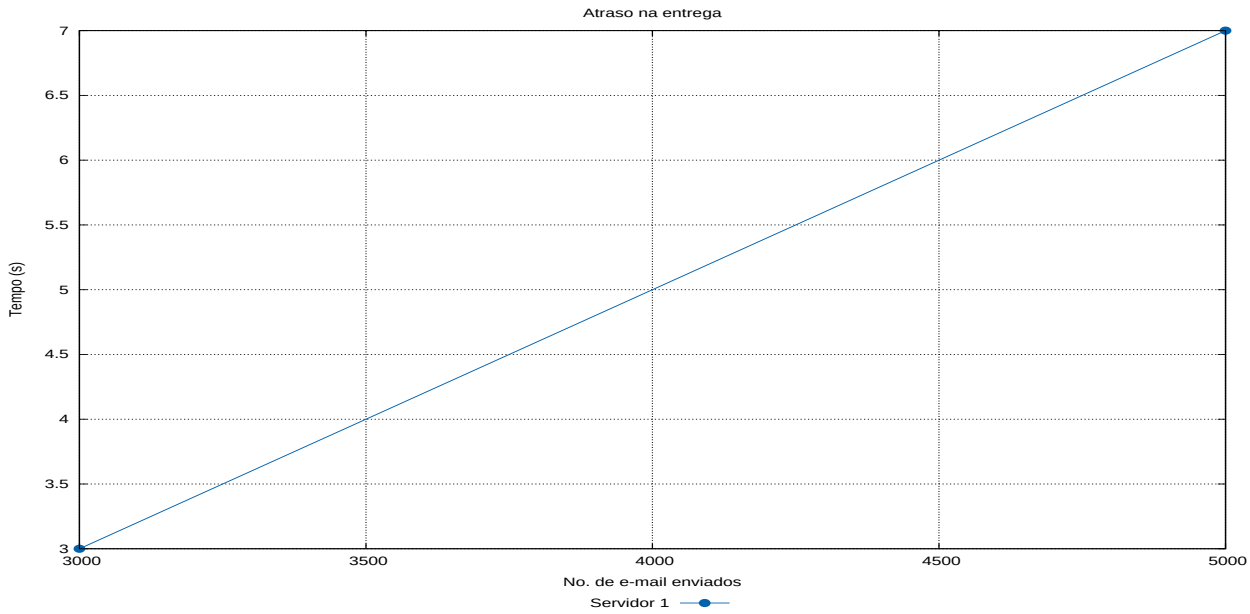

Figura 17: Atraso médio na entrega das mensagens

Como observado nas Figuras de 13 a 16, o uso de processamento, quantidade de memoria livre e o tráfego de entrada e saída da interface de rede, nos dois servidores, são similares. A Figura 17, porém, mostra que o servidor 2 gasta um tempo sensivelmente maior que o servidor 1 para entregar os e-mails recebidos. Como apresentado nas Figuras 5 e 6, os módulos *smtpd*, *cleanup*, *incoming* e *active* são responsáveis por entregar os e-mails nas *mailboxes* dos usuários. Após seguidos testes com o módulo *smtpd*, o único

modificado, verificou-se que o responsável pelo atraso era o motor MyISAM do SGBD MySQL 5.1. Este motor, padrão do MySQL 5.1, possui fraco desempenho (SCHWARTZ PETER ZAITSEV, 2012). O experimento foi repetido com o uso do motor InnoDB. Como pode ser visto na tabela 2 e, graficamente, nas Figuras de 18 a 21, todos os resultados são agora similares.

Tabela 2: Médias dos resultados do experimento preliminar- Mysql eng. MySAM

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
3000	456	366502	16	4576	0
5000	754	606623	18	4579	0
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
3000	470	369380	18	4502	0
5000	778	611388	19	4531	0
Atraso de entrega no servidor 2 em relação ao servidor 1					
Envios	Tempo (segundos)				
3000	0				
5000	0				

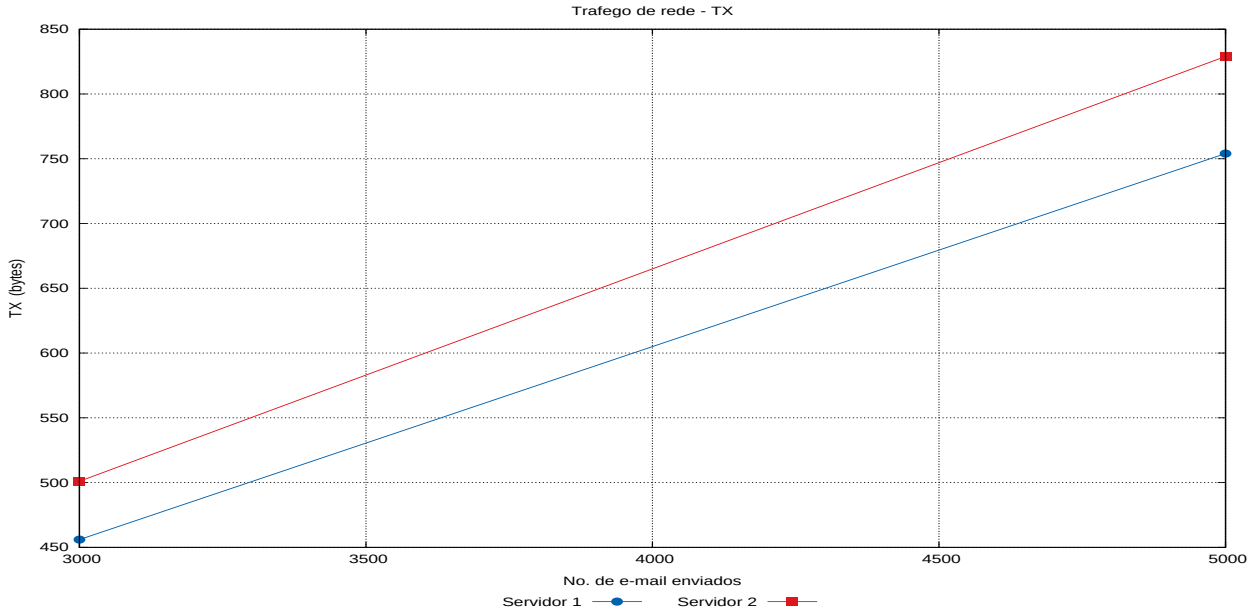

Figura 18: Média de transmissão das interfaces de rede

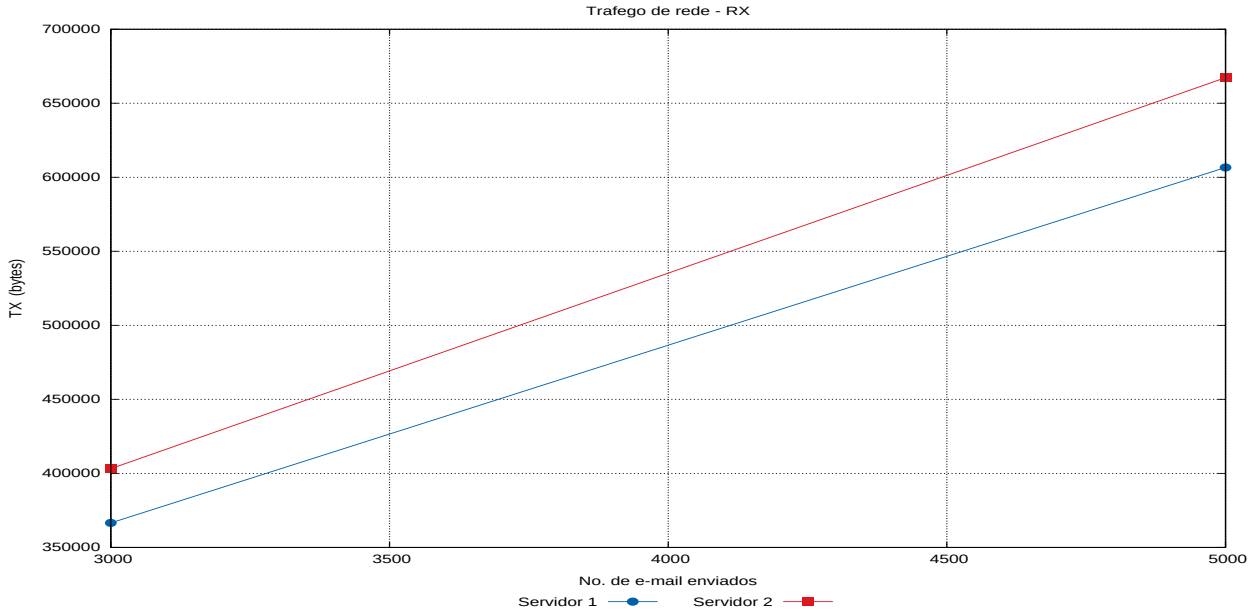

Figura 19: Média de recepção das interfaces de rede

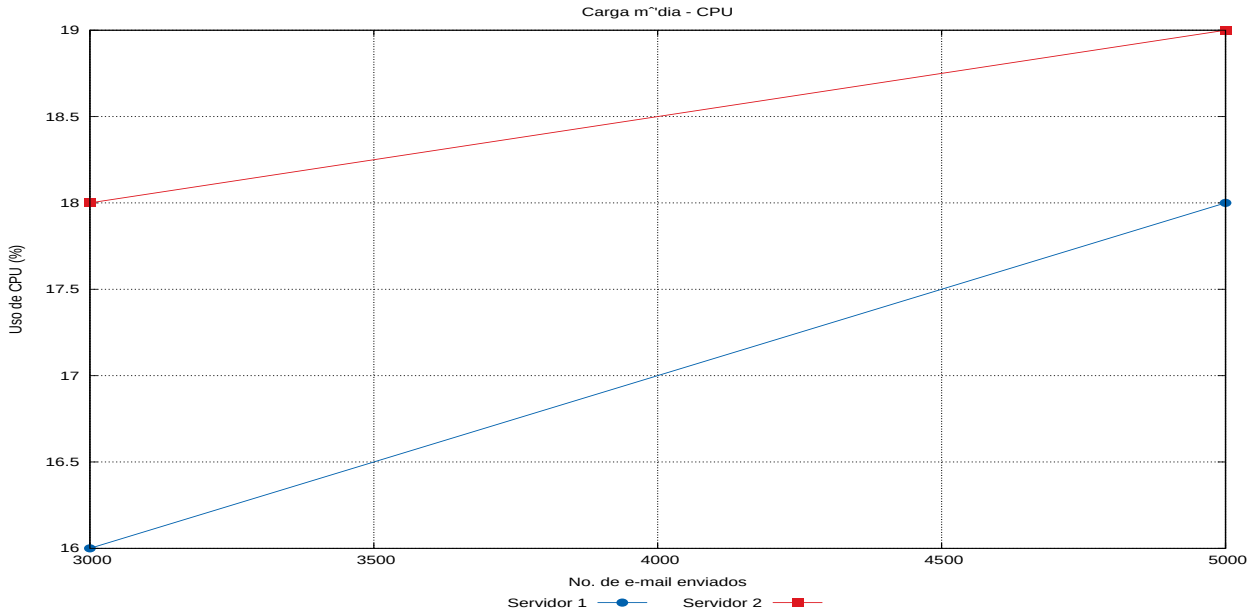

Figura 20: Carga média de uso de CPU

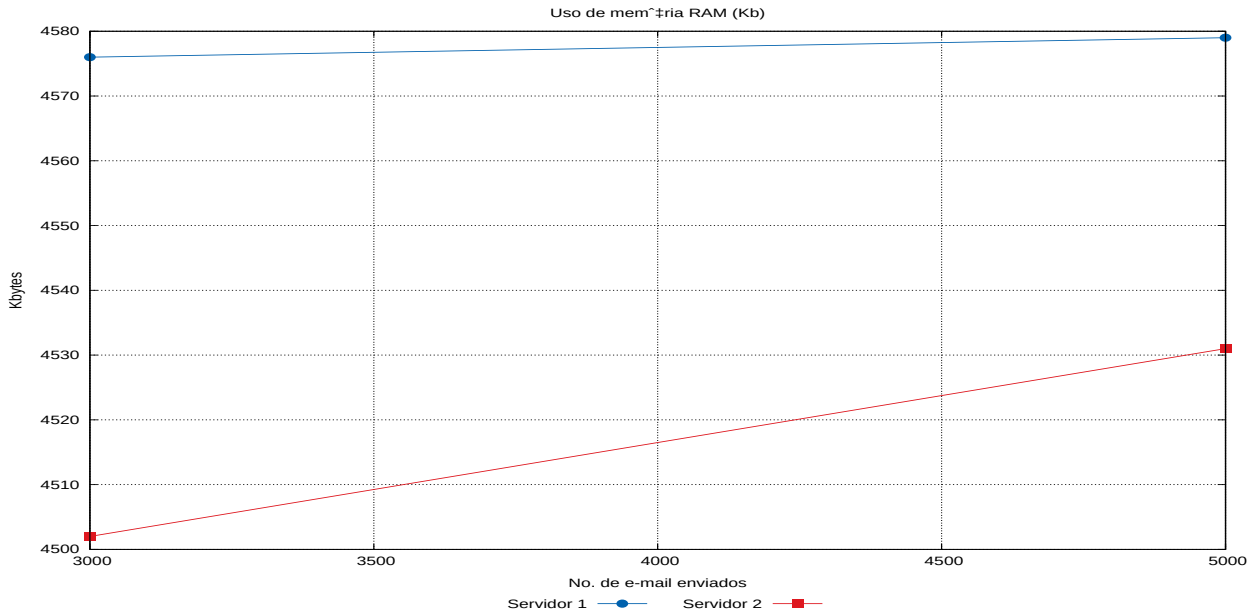

Figura 21: Média de uso de memória RAM

5.4 Experimentos

Os experimentos visam medir o uso de processamento, quantidade de memória livre, quantidade de processos em execução, tráfego de entrada e saída da interface de rede e tamanho das filas de e-mail nos servidores 1 e 2. O servidor 1 executa o módulo smtpd original e o 2, o modificado. O *host* envia simultaneamente 100, 1000, 10000 e 100000 e-mails para cada um dos servidores. Estes e-mails enviados são somente ham, somente spam ou uma combinação de ham e spam. Os e-mails são enviados para um ou cem usuários contendo 30 ou 100 endereços cadastrados como spam na tabela Spam do MySQL. A tabela 3 descreve as características de cada experimento.
Os resultados dos experimentos são apresentados e avaliados a seguir.

5.4.1 Experimento 1

Os resultados do experimento 1 estão descritos na tabela 4 e, graficamente, nas Figuras 22 a 27.

Tabela 3: Parâmetros utilizados nos experimentos

Exp.	No. e-mails enviados	Tipo e-mails	No. usuários	No. end. cadastrados	Total end. cadastrados
1	100-1000-10000-100000	ham	1	30	30
2	100-1000-10000-100000	ham	1	100	100
3	100-1000-10000-100000	ham	100	30	3000
4	100-1000-10000-100000	ham	100	100	10000
5	100-1000-10000-100000	spam	1	30	30
6	100-1000-10000-100000	spam	1	100	100
7	100-1000-10000-100000	spam	100	30	3000
8	100-1000-10000-100000	spam	100	100	10000
9	100-1000-10000-100000	ham e spam	1	78	78
10	100-1000-10000-100000	ham e spam	100	78	7800

Tabela 4: Médias dos resultados do experimento 1

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	43	12256	11	4198	0
1000	154	124758	12	4231	0
10000	1458	1364878	24	4352	98
100000	15589	12635847	57	5695	746
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	46	12288	11	4217	0
1000	158	124875	13	4241	0
10000	1489	1365248	26	4396	101
100000	15647	12648951	58	5785	749

Atraso de entrega no servidor 2 em relação ao servidor 1	
Envios	Tempo (segundos)
100	0
1000	0
10000	1
100000	1

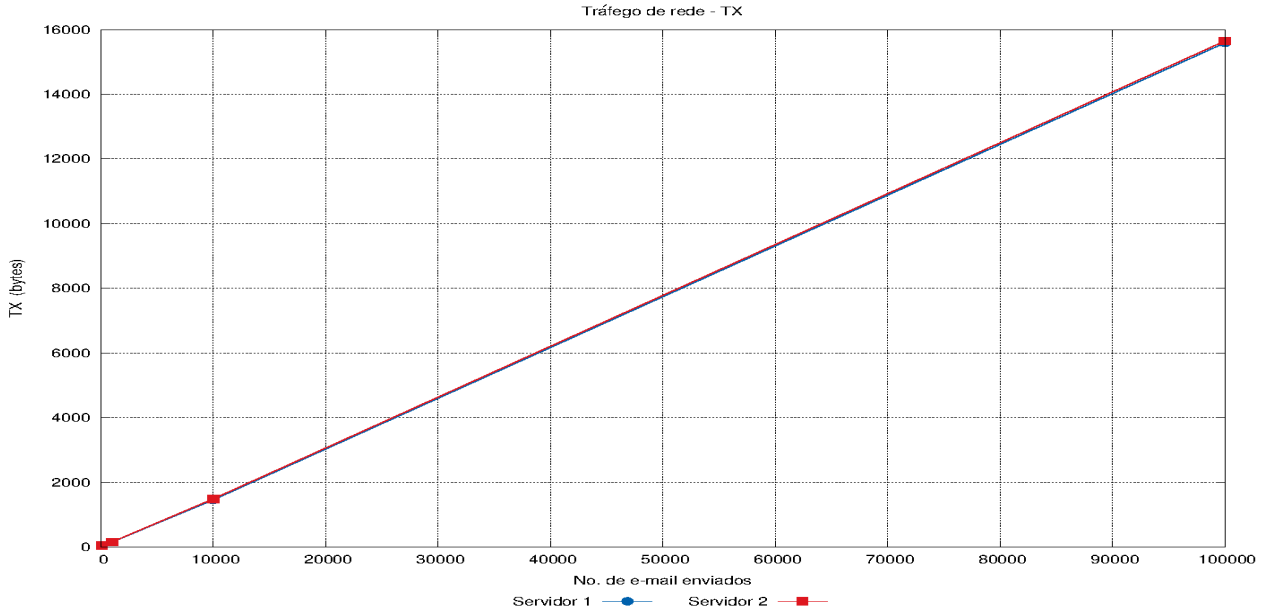

Figura 22: Média de transmissão das interfaces de rede

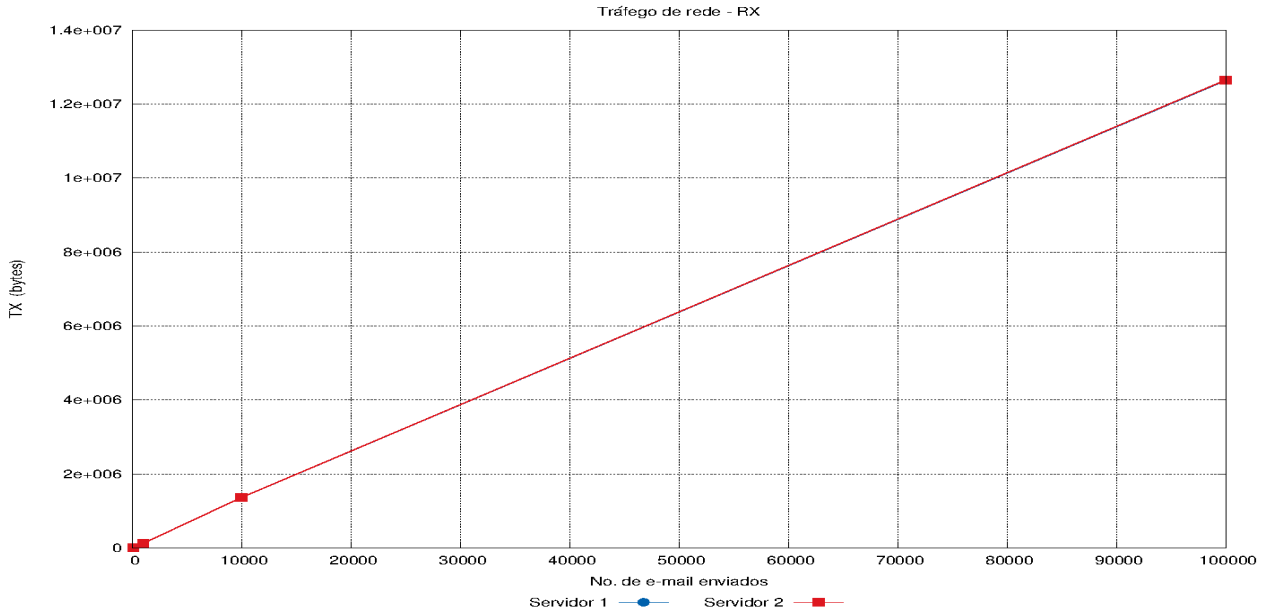

Figura 23: Média de recepção das interfaces de rede

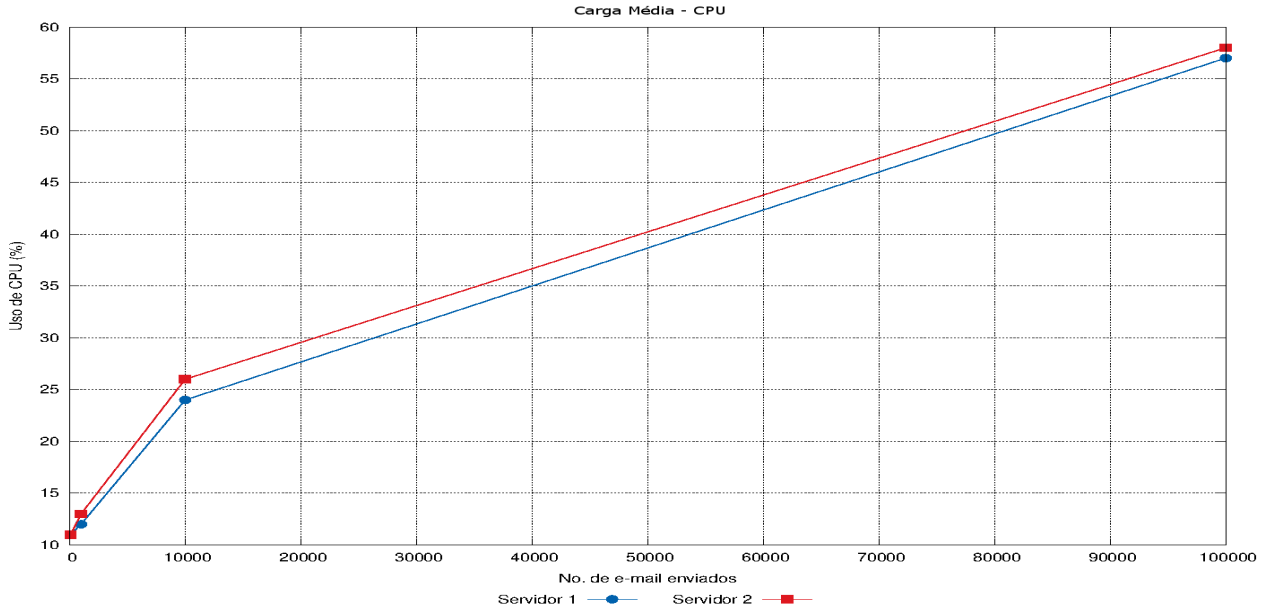

Figura 24: Carga média de uso de CPU

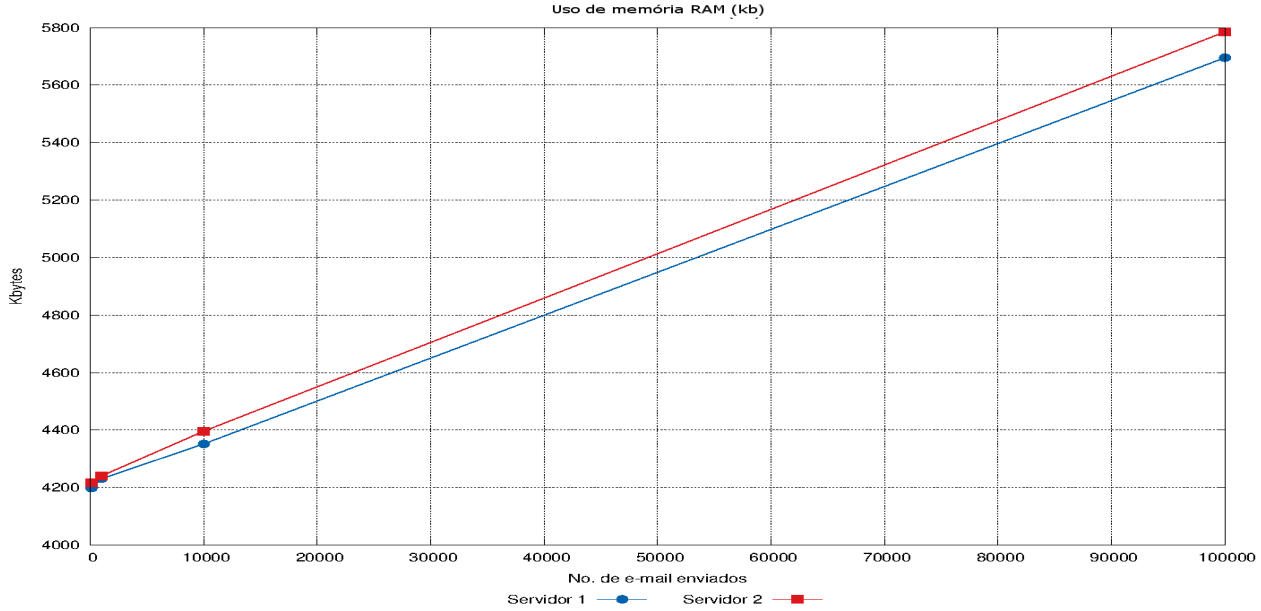

Figura 25: Média de uso de memória RAM

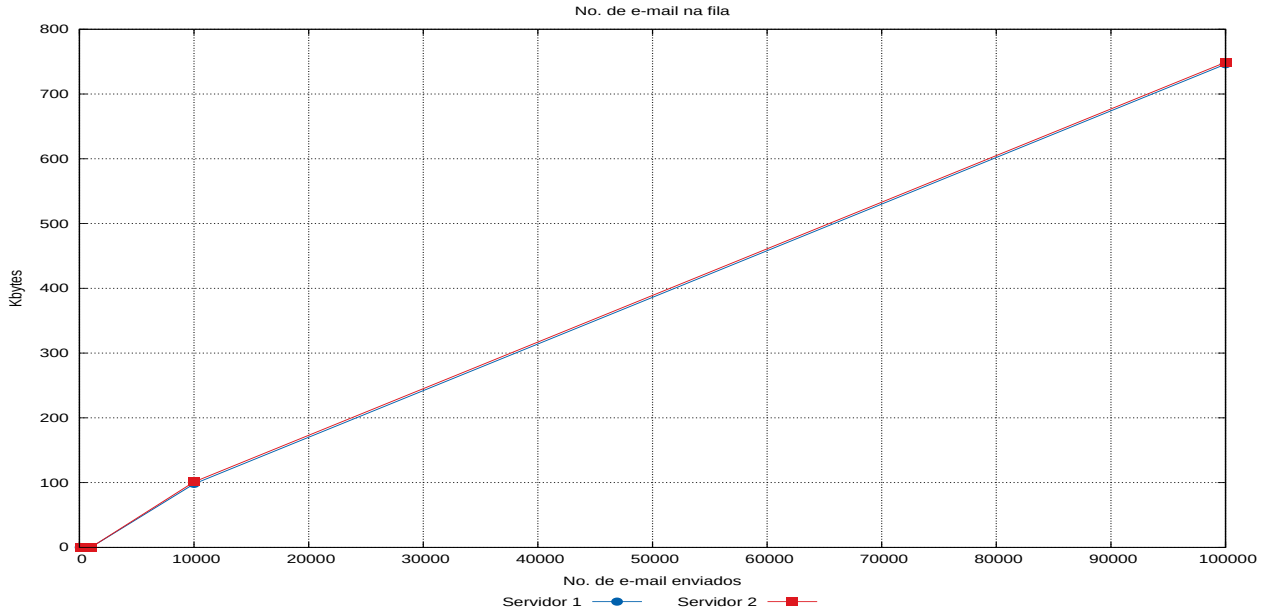

Figura 26: Média das filas de recepção de e-mail

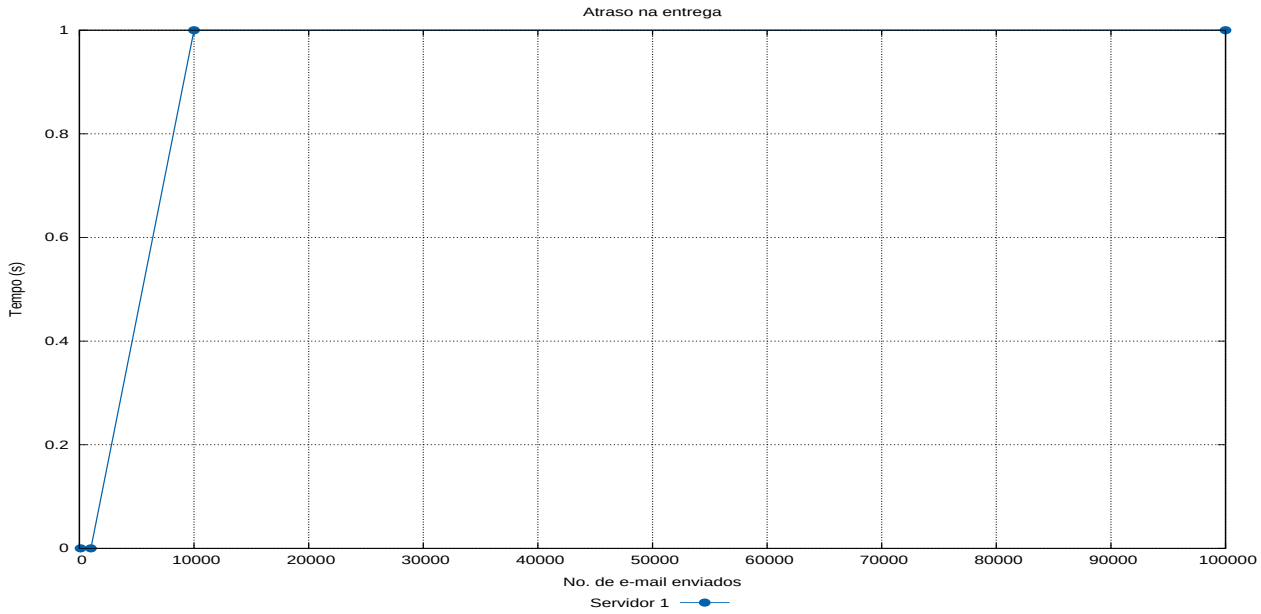

Figura 27: Atraso médio na entrega das mensagens

5.4.2 Experimento 2

Os resultados do experimento 2 estão descritos na tabela 5 e, graficamente, nas Figuras 28 a 33

Tabela 5: Médias dos resultados do experimento 2

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	44	12379	11	4240	0
1000	156	126006	13	4273	0
10000	1473	1378527	23	4396	103
100000	15745	12762205	58	5752	753
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	45	12411	12	4259	0
1000	160	126124	14	4283	0
10000	1504	1378900	26	4440	104
100000	15803	12775441	59	5843	756

Atraso de entrega no servidor 2 em relação ao servidor 1	
Envios	Tempo (segundos)
100	0
1000	0
10000	1
100000	2

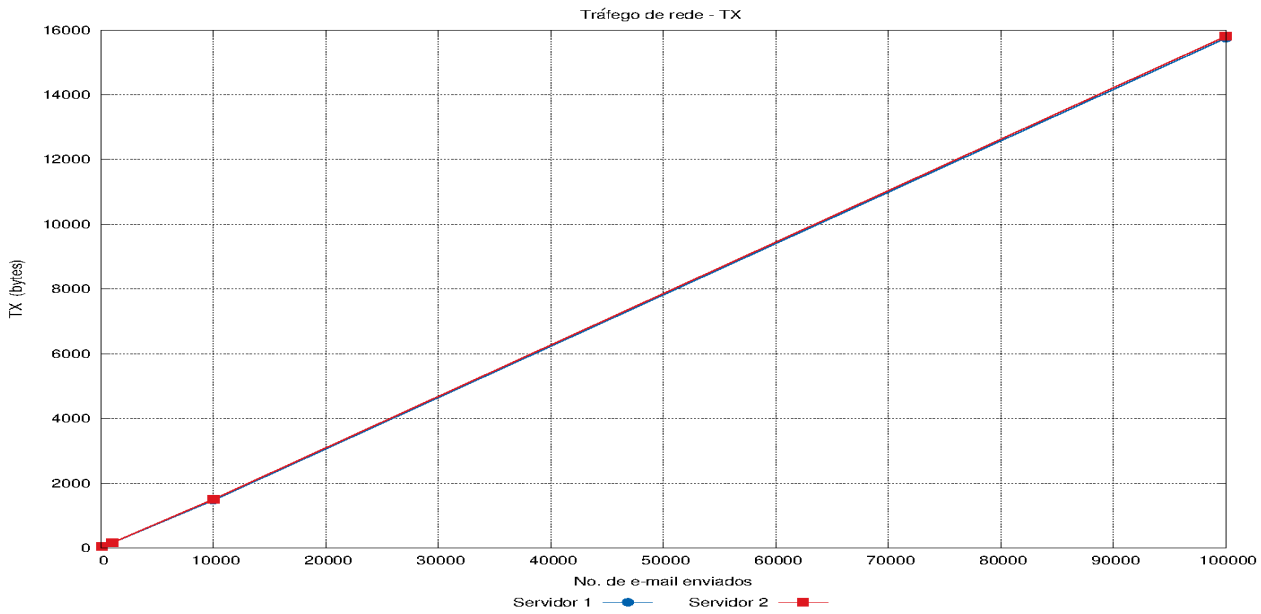

Figura 28: Média de transmissão das interfaces de rede

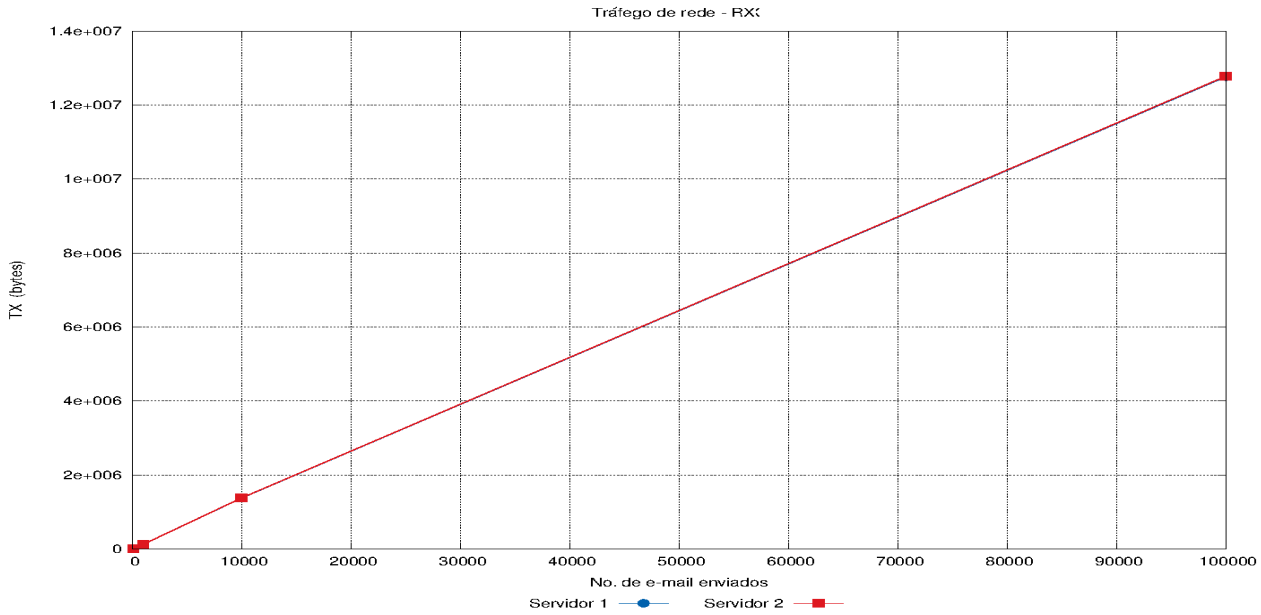

Figura 29: Média de recepção das interfaces de rede

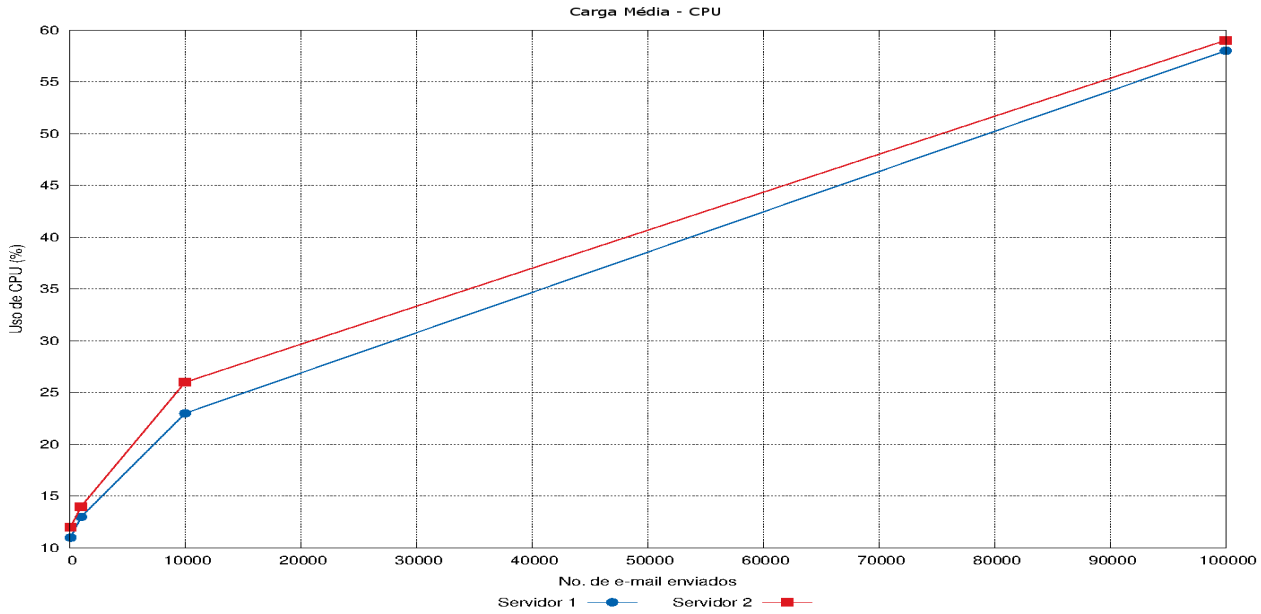

Figura 30: Carga média de uso de CPU

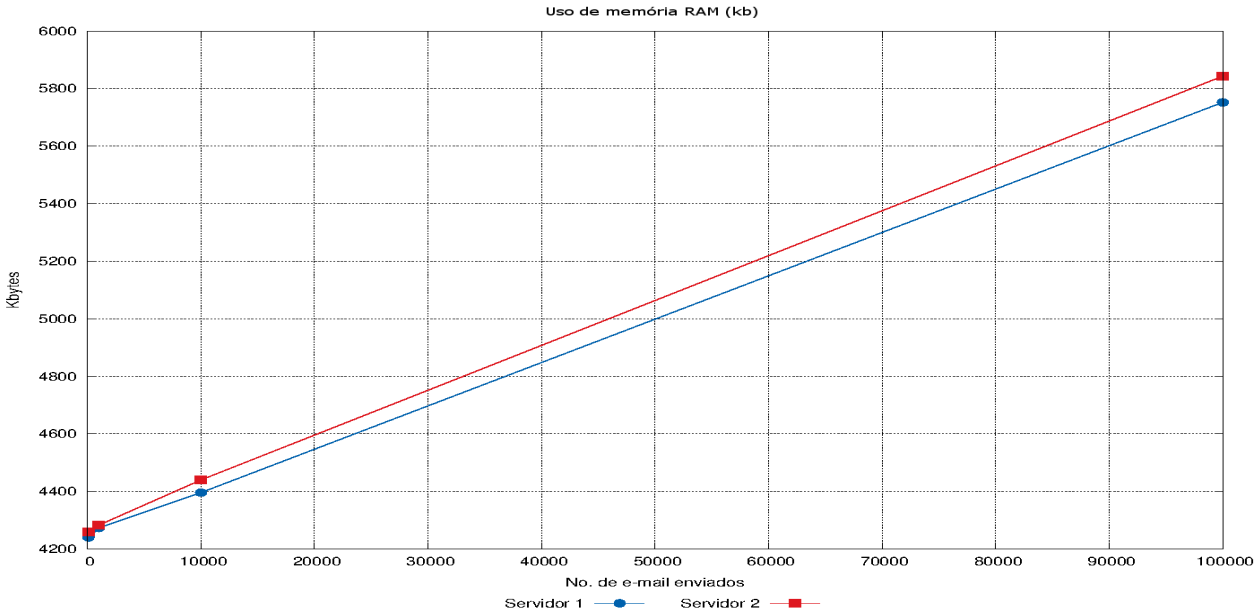

Figura 31: Média de uso de memória RAM

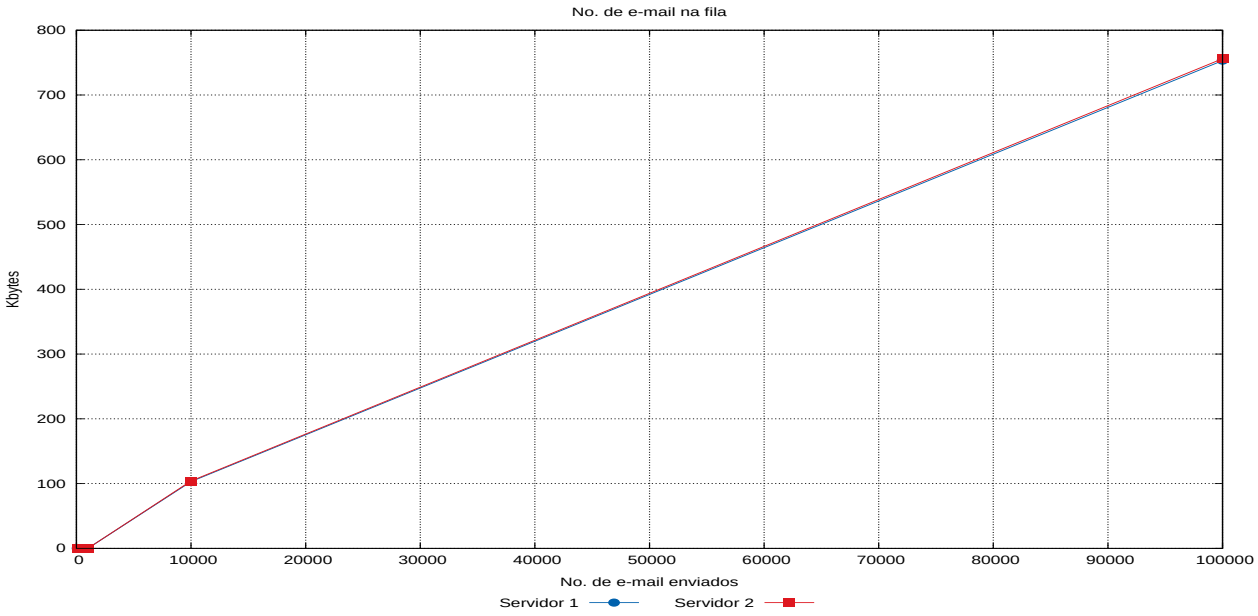

Figura 32: Média das filas de recepção de e-mail

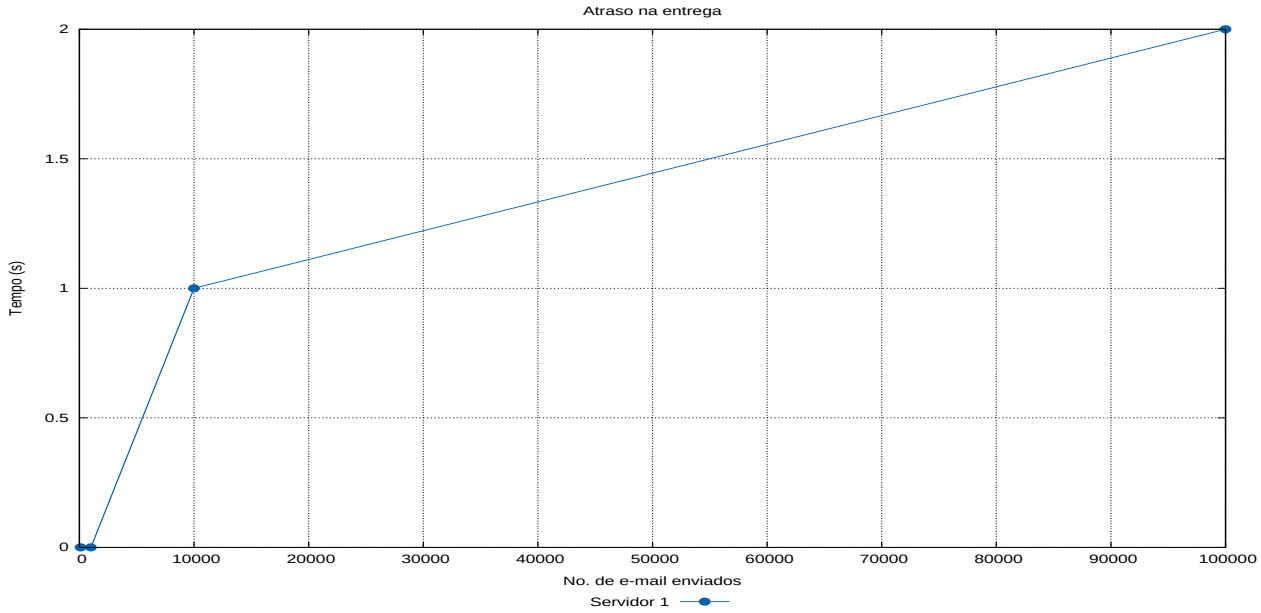

Figura 33: Atraso médio na entrega das mensagens

5.4.3 Experimento 3

Os resultados do experimento 3 estão descritos na tabela 6 e, graficamente, nas Figuras 34 a 39

Tabela 6: Médias dos resultados do experimento 3

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	43	12256	11	4198	0
1000	154	124758	12	4231	0
10000	1458	1364878	24	4352	98
100000	15589	12635847	57	5695	746
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	46	12288	11	4217	0
1000	158	124875	13	4241	0
10000	1489	1365248	26	4396	101
100000	15647	12648951	58	5785	749

Atraso de entrega no servidor 2 em relação ao servidor 1	
Envios	Tempo (segundos)
100	0
1000	0
10000	1
100000	1

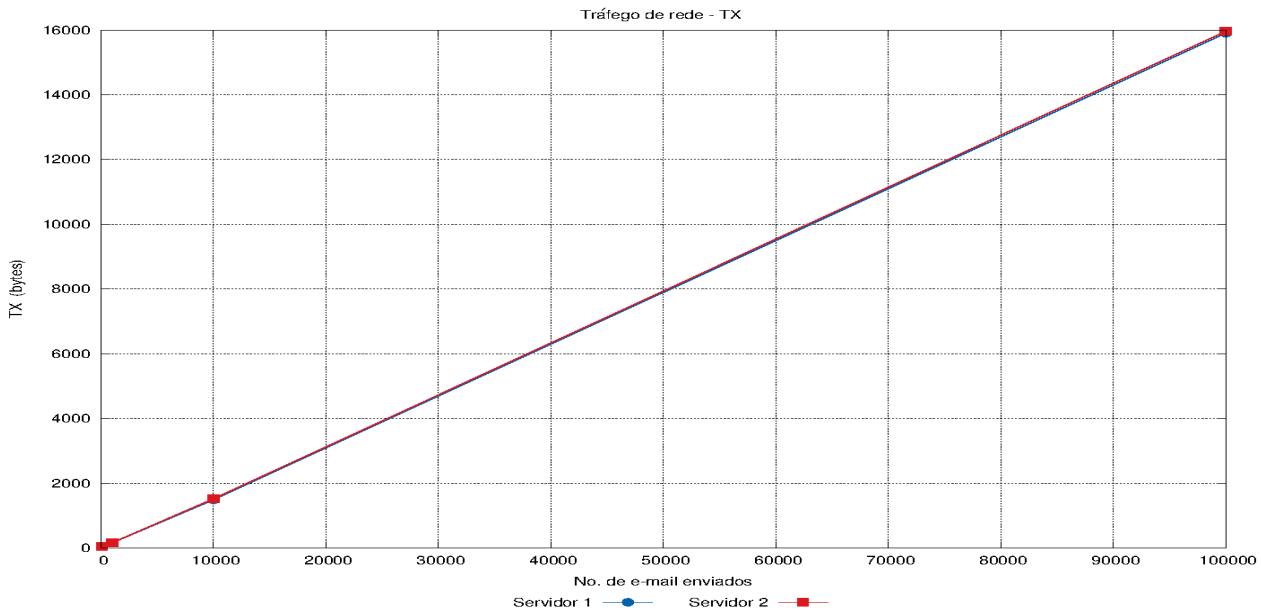

Figura 34: Média de transmissão das interfaces de rede

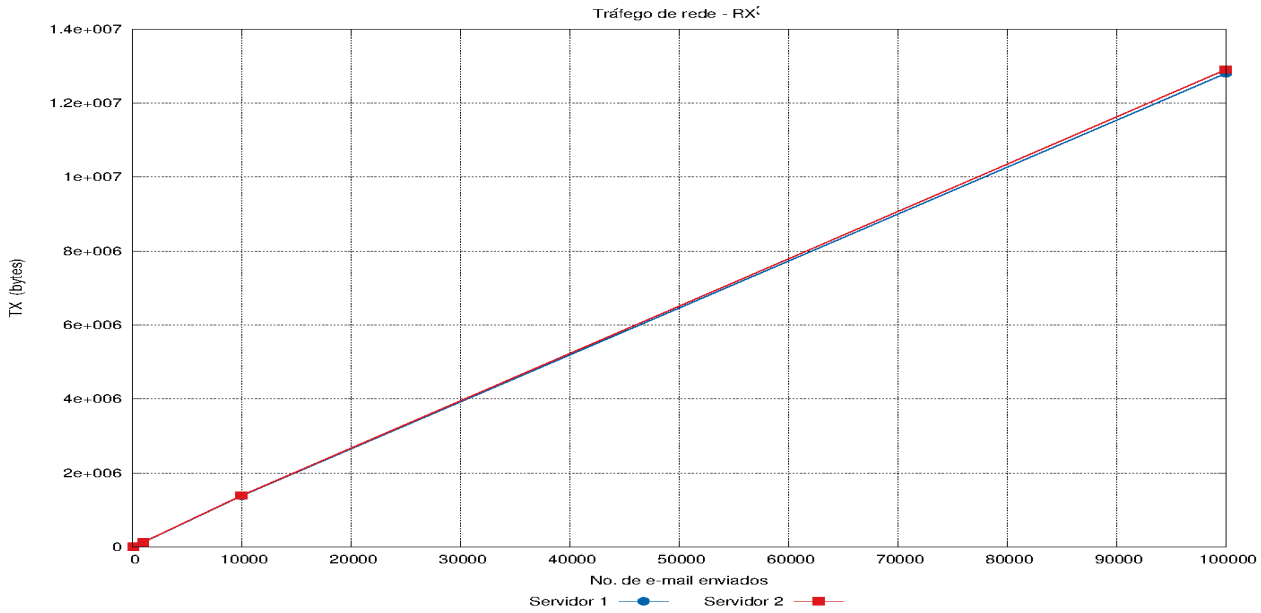

Figura 35: Média de recepção das interfaces de rede

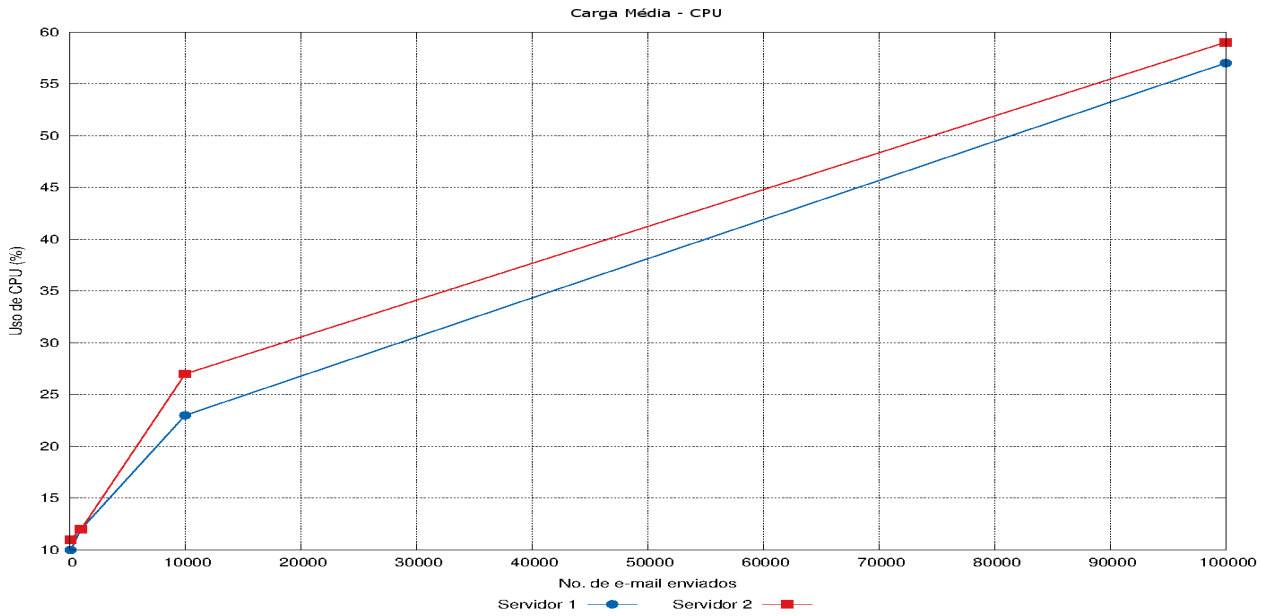

Figura 36: Carga média de uso de CPU

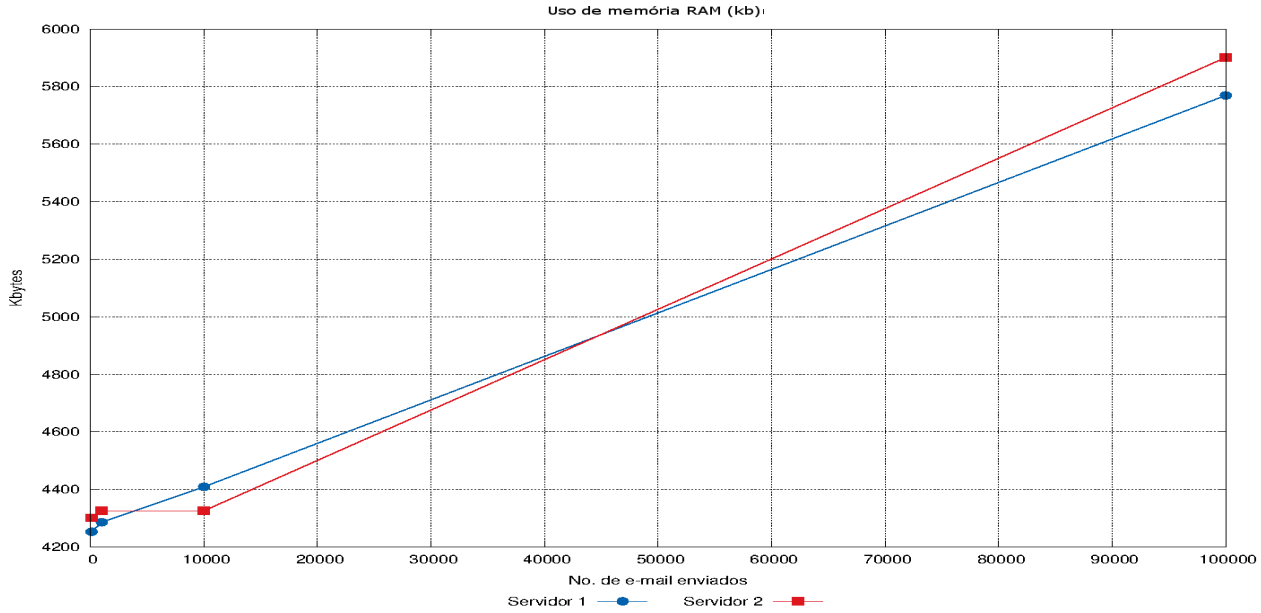

Figura 37: Média de uso de memória RAM

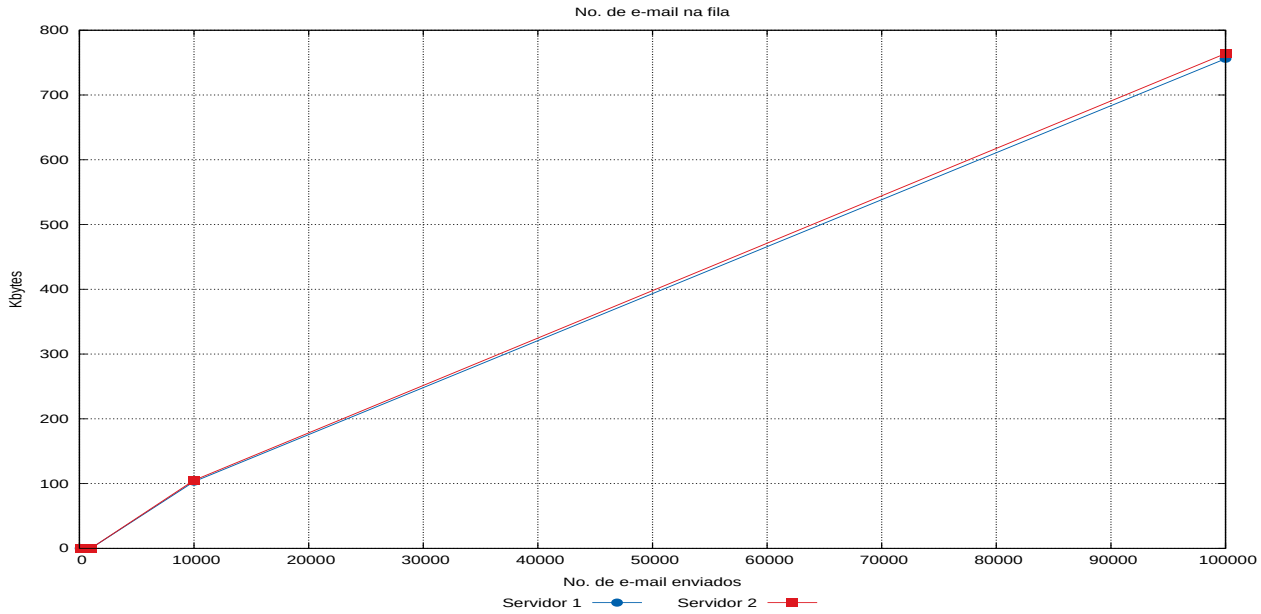

Figura 38: Média das filas de recepção de e-mail

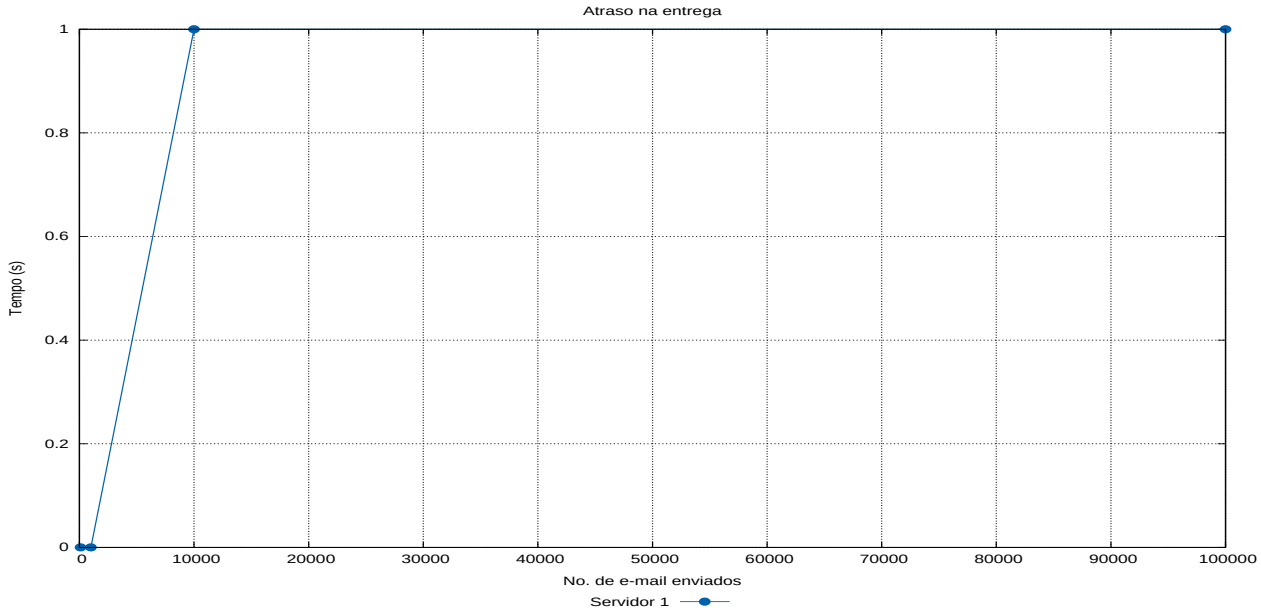

Figura 39: Atraso médio na entrega das mensagens

5.4.4 Experimento 4

Os resultados do experimento 4 estão descritos na tabela 7 e, graficamente, nas Figuras 40 a 45

Tabela 7: Médias dos resultados do experimento 4

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	48	12403	11	4248	0
1000	157	126255	12	4282	0
10000	1487	1381257	24	4404	102
100000	15901	12787477	58	5763	755
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	47	12534	11	4301	0
1000	162	127373	14	4326	0
10000	1519	1392553	28	4484	103
100000	15960	12901930	61	5901	761

Atraso de entrega no servidor 2 em relação ao servidor 1	
Envios	Tempo (segundos)
100	0
1000	0
10000	1
100000	1

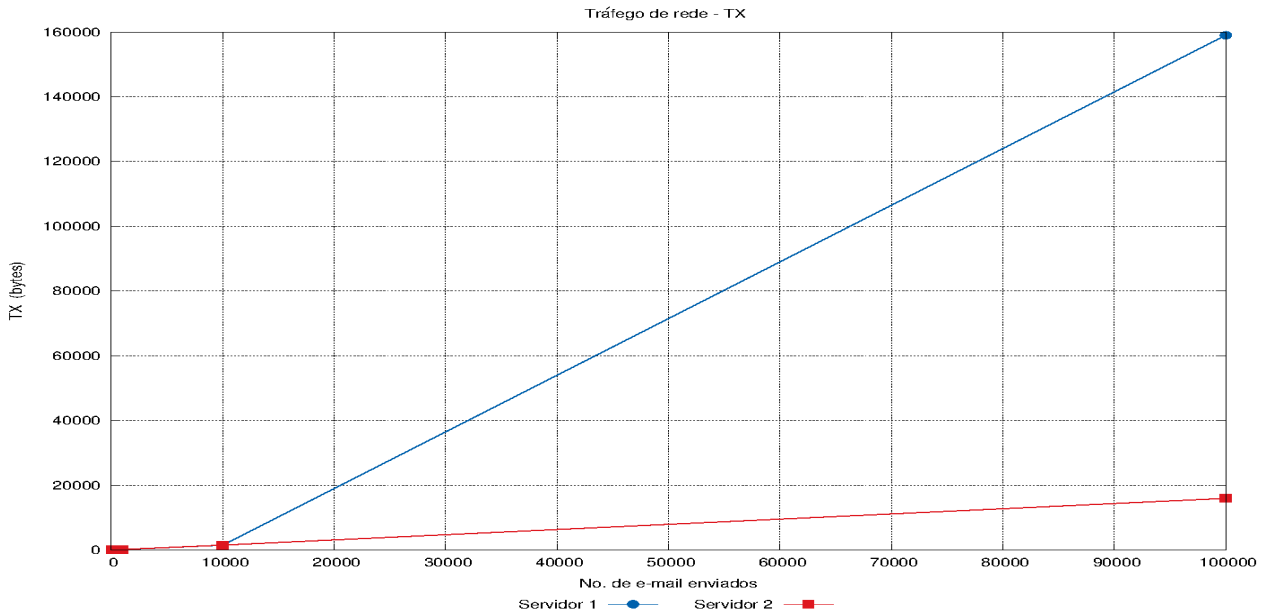

Figura 40: Média de transmissão das interfaces de rede

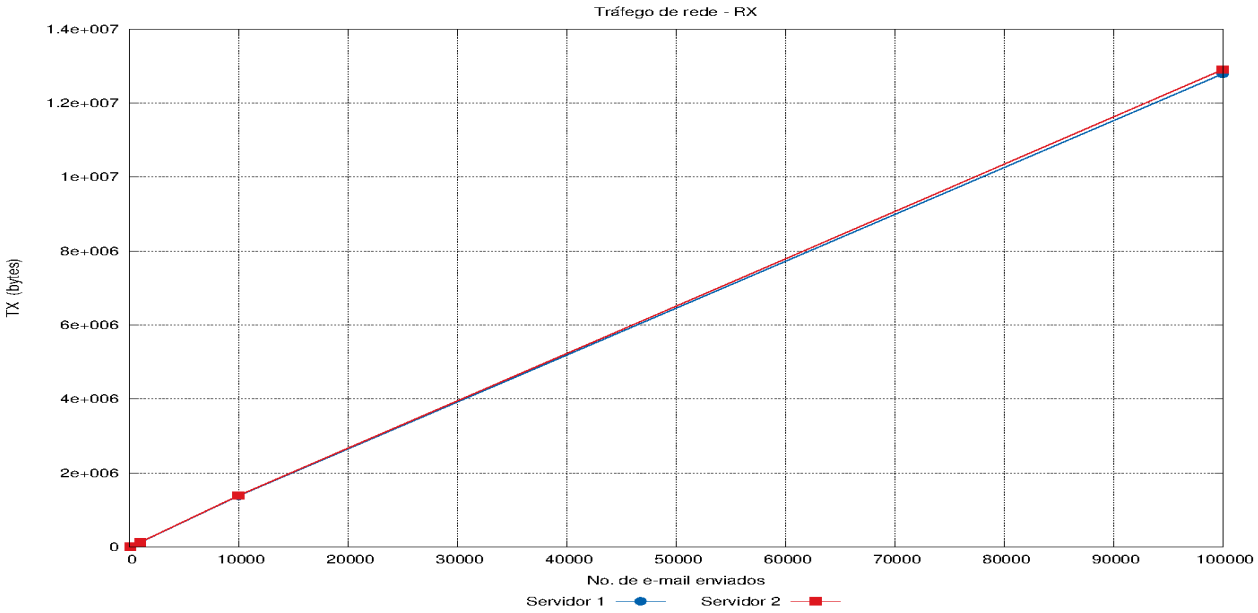

Figura 41: Média de recepção das interfaces de rede

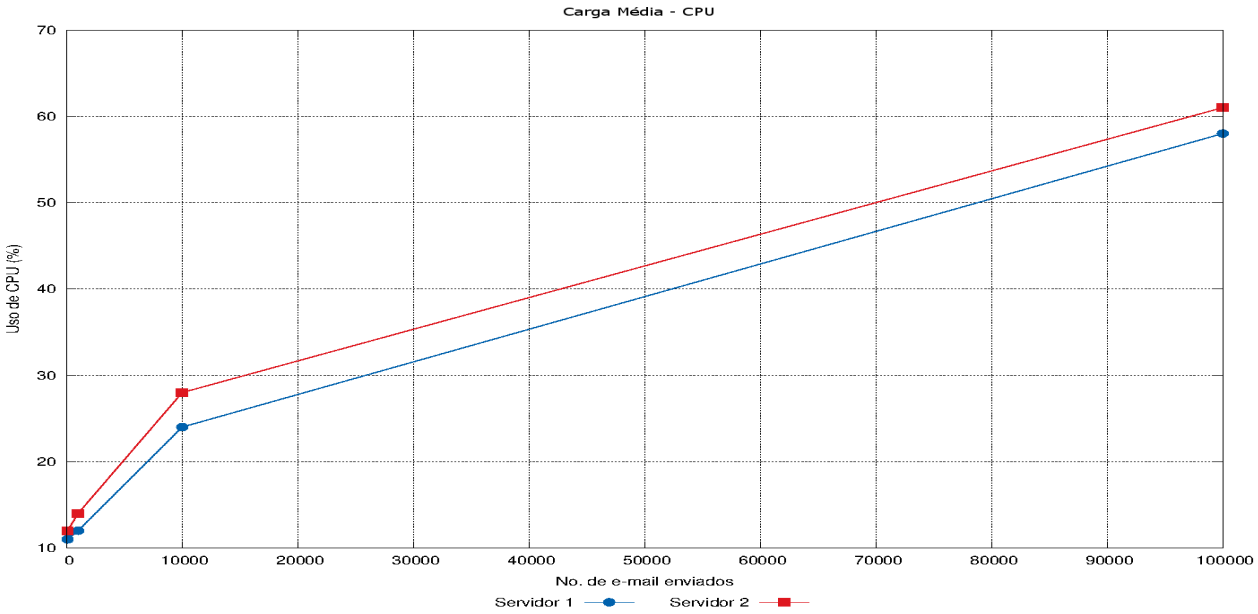

Figura 42: Carga média de uso de CPU

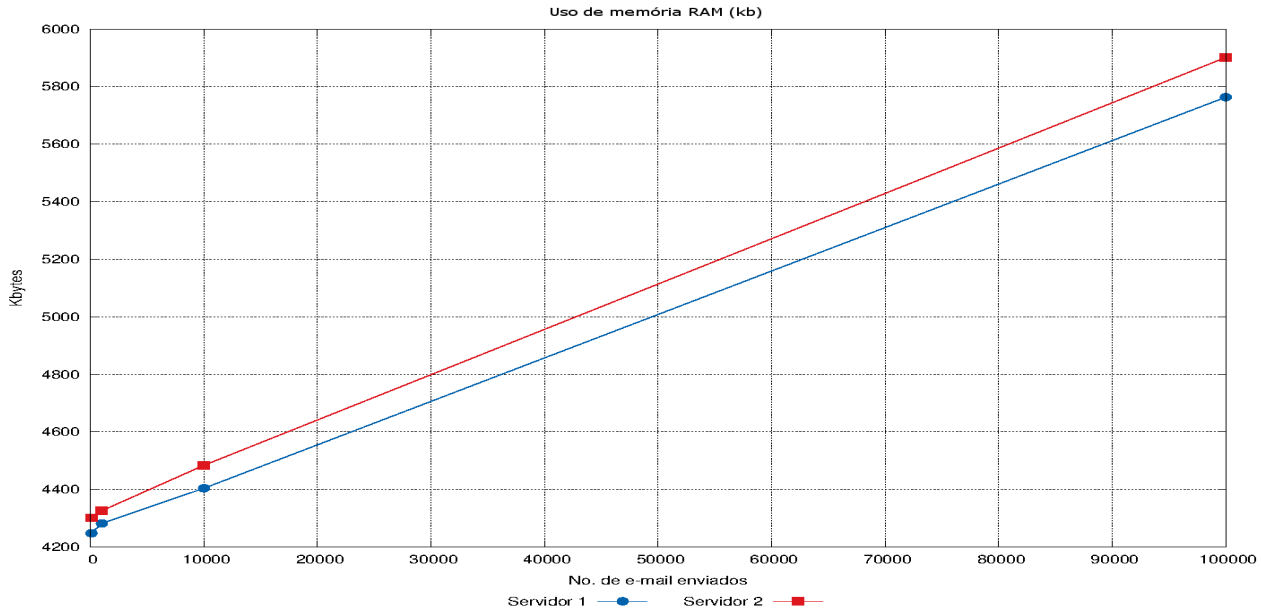

Figura 43: Média de uso de memória RAM

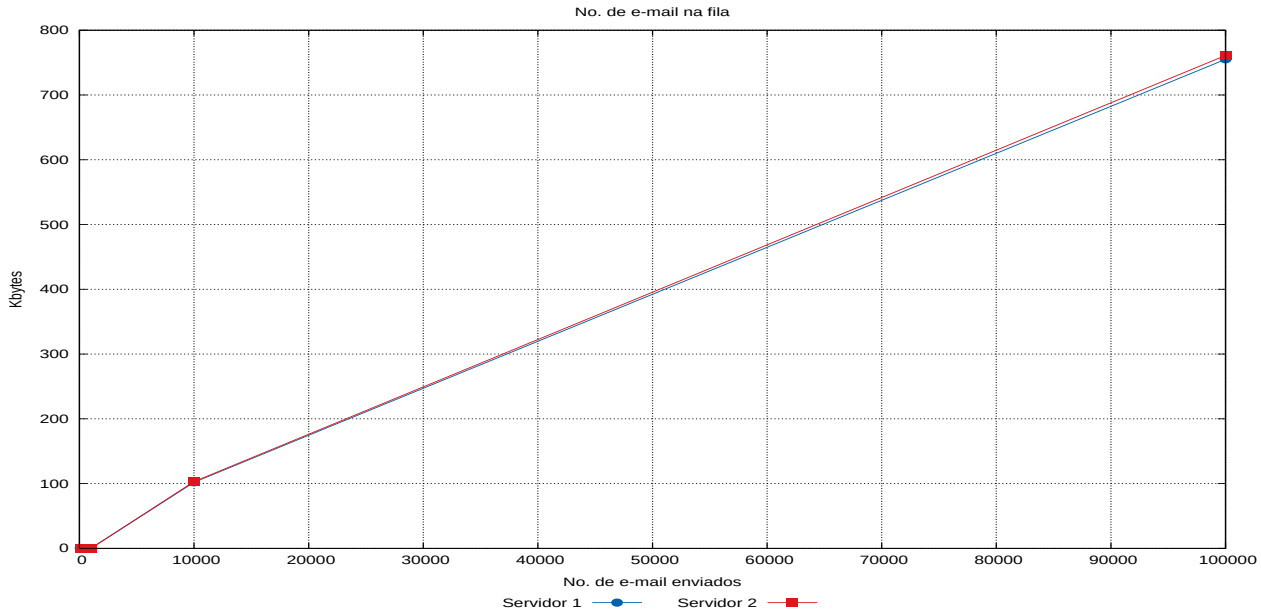

Figura 44: Média das filas de recepção de e-mail

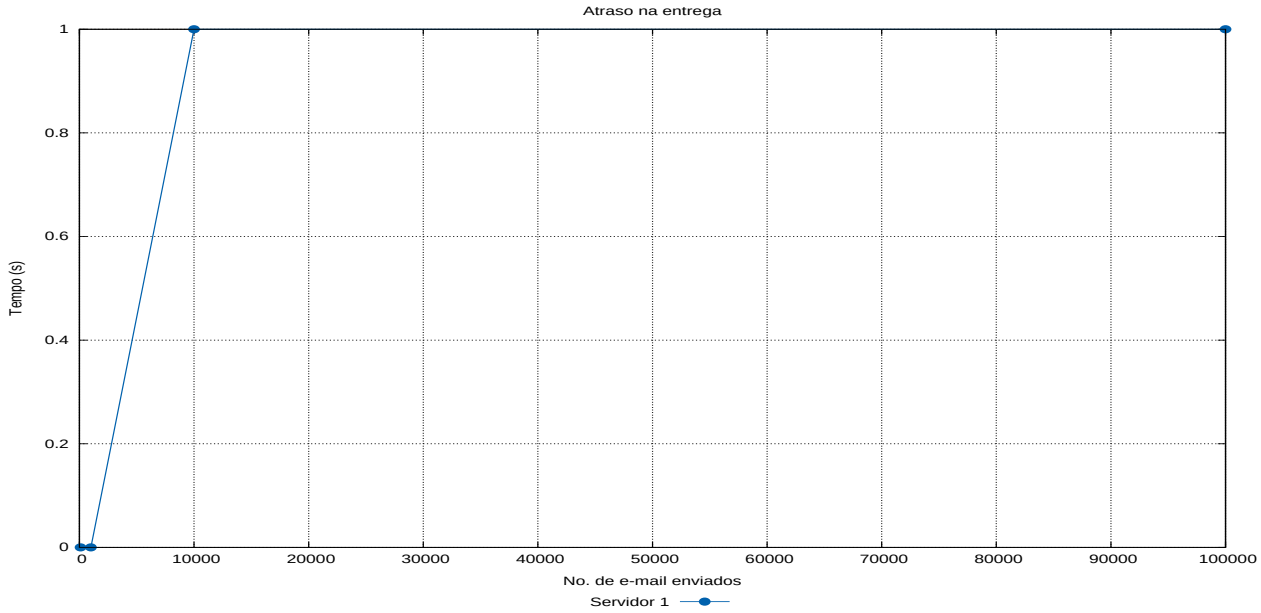

Figura 45: Atraso médio na entrega das mensagens

5.4.5 Avaliação dos experimentos 1 a 4

Os experimentos 1 a 4 envolvem o envio, aos servidores, de e-mails ham somente. O servidor 2 consulta a tabela *BlackList* do MySQL para entregar cada e-mail ham na mailbox do usuário. Por isto, espera-se que os resultados obtidos pelo servidor 2 sejam ligeiramente inferiores aos obtidos pelo servidor 1.

De fato, em todas as seis métricas, o servidor 2 apresentou desempenho ligeiramente inferior ao do servidor 1. A taxa de uso de CPU do servidor 2 foi, em média, 6.66% superior à do servidor 1. As taxas de uso de memória RAM, de bytes transmitidos e de bytes recebidos pelo servidor 2 foram, em média, respectivamente, 1.22%, 2.22% e 0.52% superiores à do servidor 1. O número máximo de e-mails presentes na fila *incoming* do servidor 2 e o tempo gasto por ele para entregar todos os e-mails nas mailboxes dos usuários cresceram, em média, 0.61% e 0.01%, respectivamente.

Os resultados indicam que a consulta à tabela *BlackList*, durante o recebimento dos e-mails, acrescenta custos computacionais insignificantes ao servidor Zimbra modificado. Os custos computacionais de um servidor em relação ao outro, medidos através de cada uma das seis métricas, estão todos descritos na Tabela 8.

Tabela 8: Resultados de um servidor em relação ao outro — experimentos 1–4

Experimento 1						
No. e-mails	CPU (S2/S1)	RAM (S2/S1)	Tx (S2/S1)	Rx (S2/S1)	Fila (S2/S1)	Entrega (S2/S1)
100	1.0000	1.0045	1.0698	1.0026	1.0000	1.0000
1000	1.0833	1.0024	1.0260	1.0009	1.0000	1.0000
10000	1.0833	1.0101	1.0213	1.0003	1.0306	1.0001
100000	1.0175	1.0158	1.0037	1.0010	1.0040	1.0001
Experimento 2						
No. e-mails	CPU (S2/S1)	RAM (S2/S1)	Tx (S2/S1)	Rx (S2/S1)	Fila (S2/S1)	Entrega (S2/S1)
100	1.0909	1.0045	1.0227	1.0026	1.0000	1.0000
1000	1.0769	1.0023	1.0256	1.0009	1.0000	1.0000
10000	1.1304	1.0100	1.0210	1.0003	1.0097	1.0001
100000	1.0172	1.0158	1.0037	1.0010	1.0040	1.0001
Experimento 3						
No. e-mails	CPU (S2/S1)	RAM (S2/S1)	Tx (S2/S1)	Rx (S2/S1)	Fila (S2/S1)	Entrega (S2/S1)
100	1.0000	1.0125	1.0682	1.0106	1.0000	1.0000
1000	1.0833	1.0103	1.0318	1.0089	1.0000	1.0000
10000	1.0800	1.0182	1.0215	1.0082	1.0194	1.0001
100000	1.0172	1.0239	1.0037	1.0090	1.0119	1.0001
Experimento 4						
No. e-mails	CPU (S2/S1)	RAM (S2/S1)	Tx (S2/S1)	Rx (S2/S1)	Fila (S2/S1)	Entrega (S2/S1)
100	1.0000	1.0125	0.9792	1.0106	1.0000	1.0000
1000	1.1667	1.0103	1.0318	1.0089	1.0000	1.0000
10000	1.1667	1.0182	1.0215	1.0082	1.0098	1.0001
100000	1.0517	1.0239	1.0037	1.0090	1.0079	1.0001
Média						
	CPU	RAM	Tx	Rx	Fila	Entrega
Média	*1.0666*	*1.0122*	*1.0222*	*1.0052*	*1.0061*	*1.0001*

A Figura 46 apresenta um gráfico comparando a média das métricas nos dois servi-

dores.

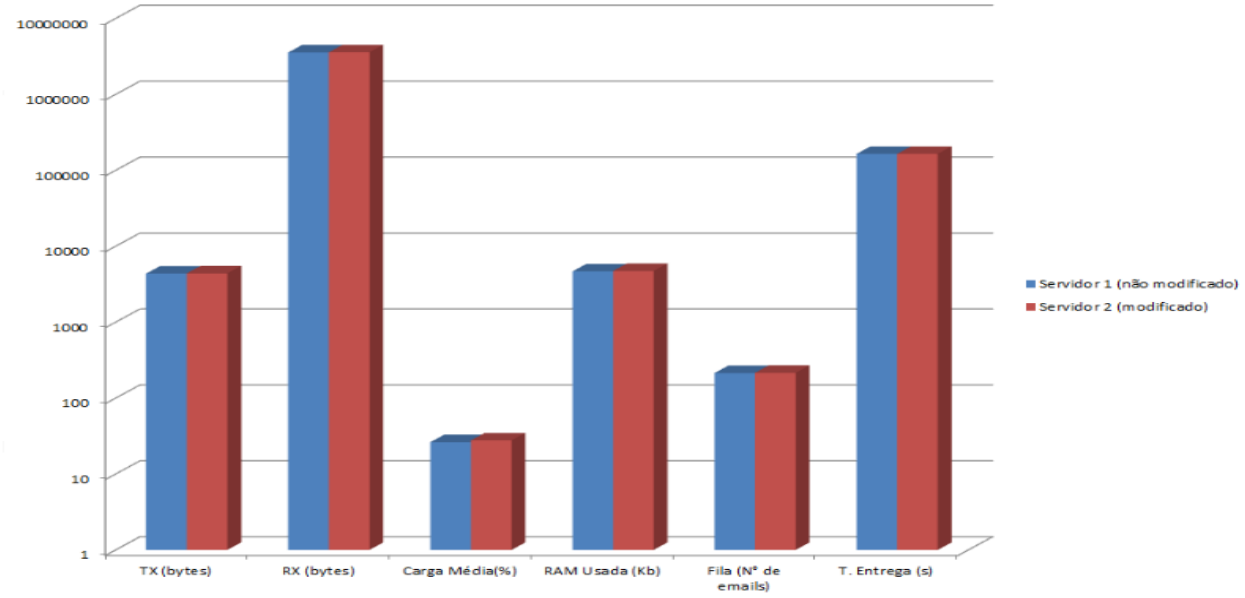

Figura 46: Média das métricas nos servidores

5.4.6 Experimento 5

Os resultados do experimento 5 estão descritos na tabela 8 e, graficamente, nas Figuras 47 a 51.

Tabela 9: Médias dos resultados do experimento 5

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	39	11958	12	4111	0
1000	148	123957	12	4198	0
10000	1422	1335486	27	4309	98
100000	14983	12956984	59	5591	812
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	11548	96	12	4213	0
1000	117534	163	13	4221	0
10000	1316954	1597	11	4403	0
100000	12945837	15324	11	5428	0

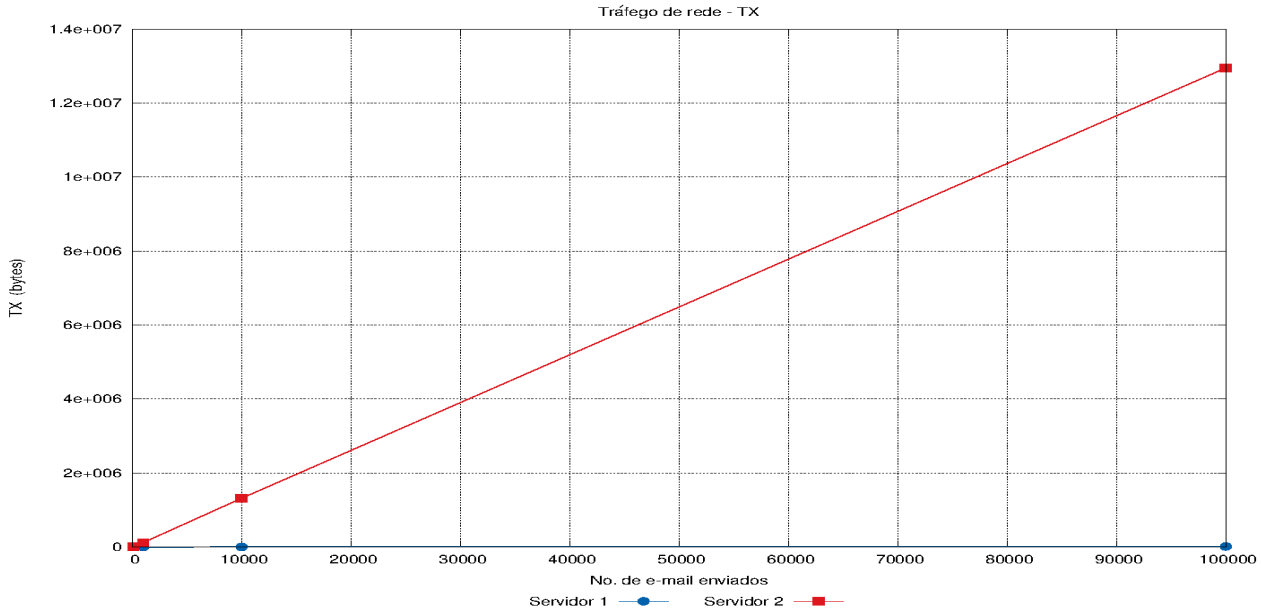

Figura 47: Média de transmissão das interfaces de rede

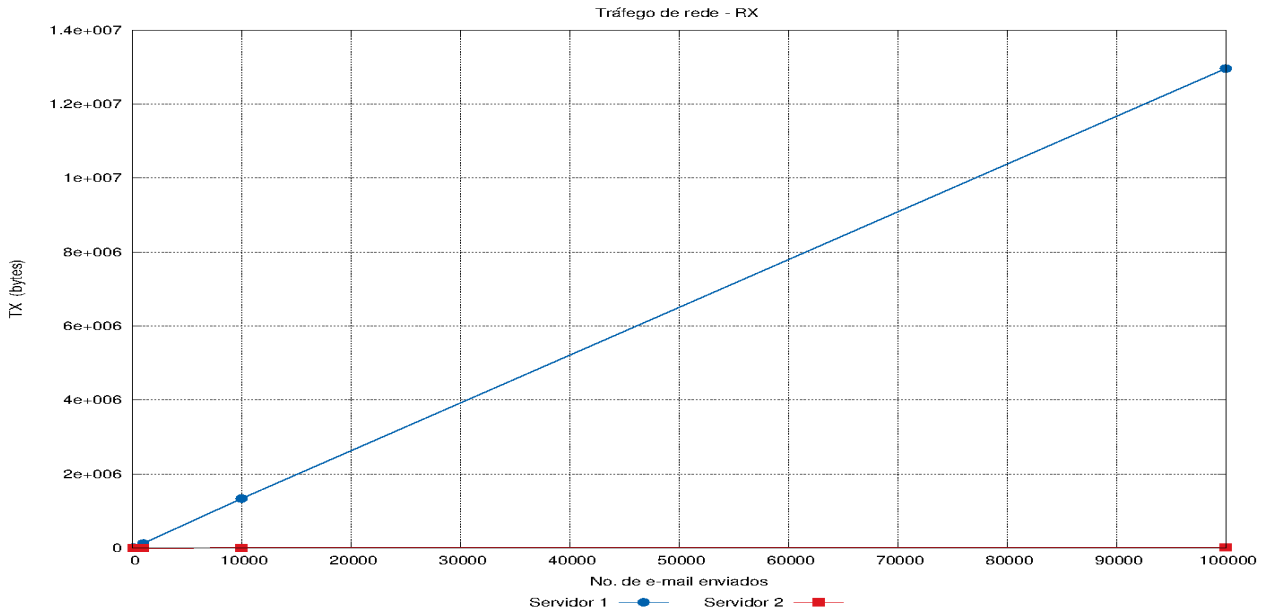

Figura 48: Média de recepção das interfaces de rede

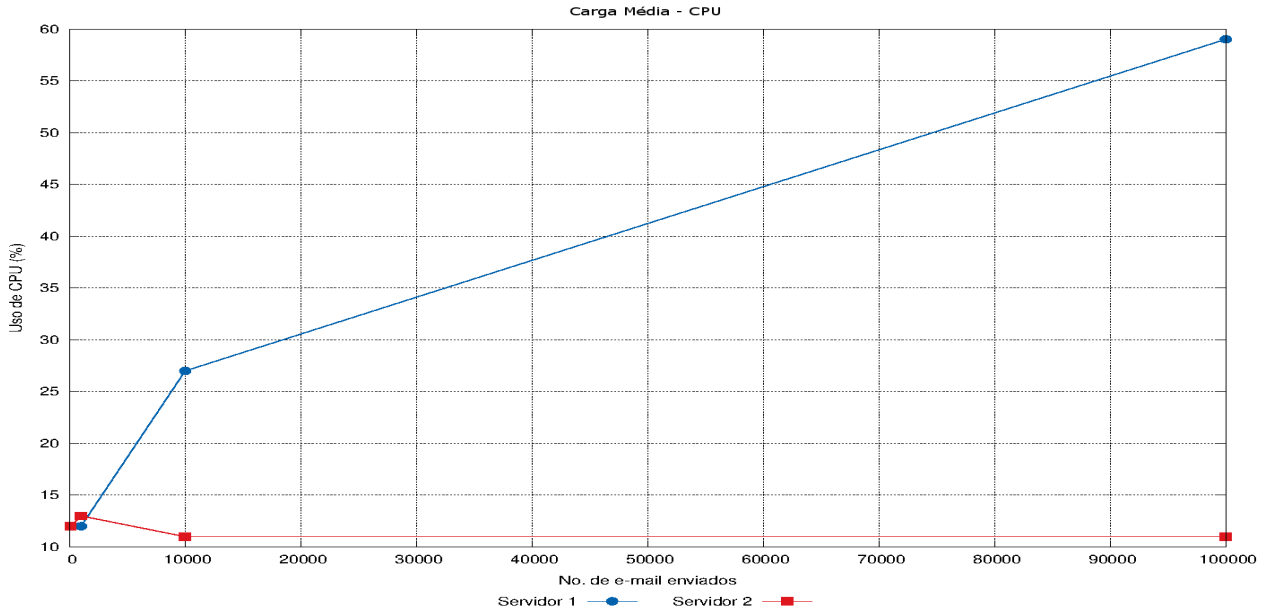

Figura 49: Carga média de uso de CPU

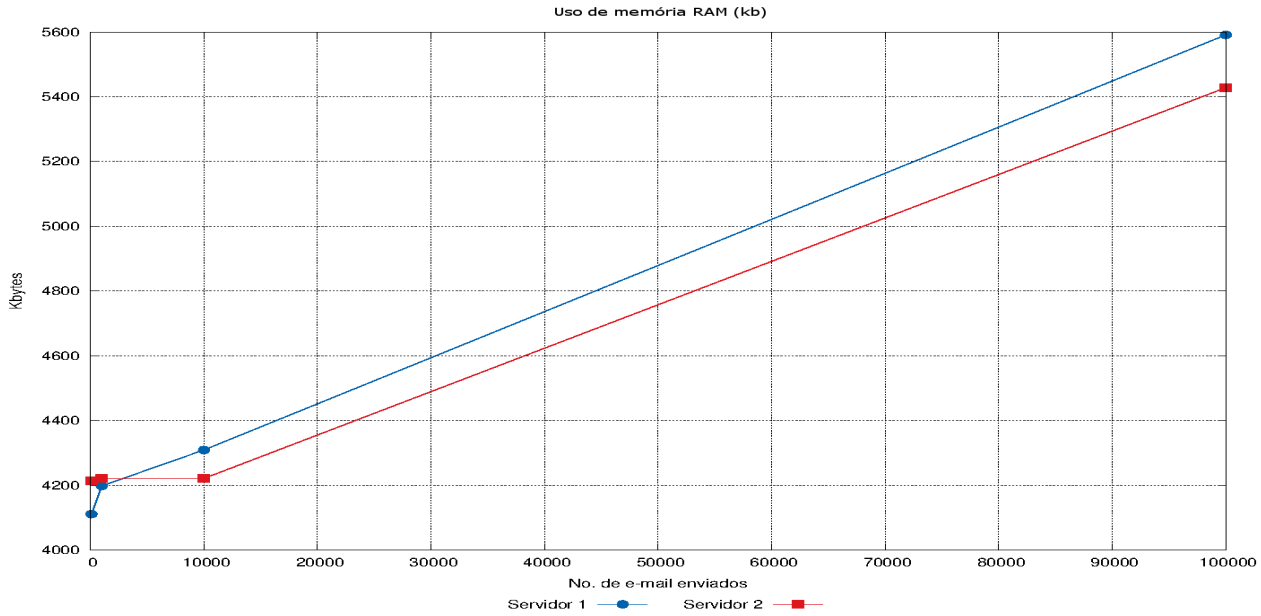

Figura 50: Média de uso de memória RAM

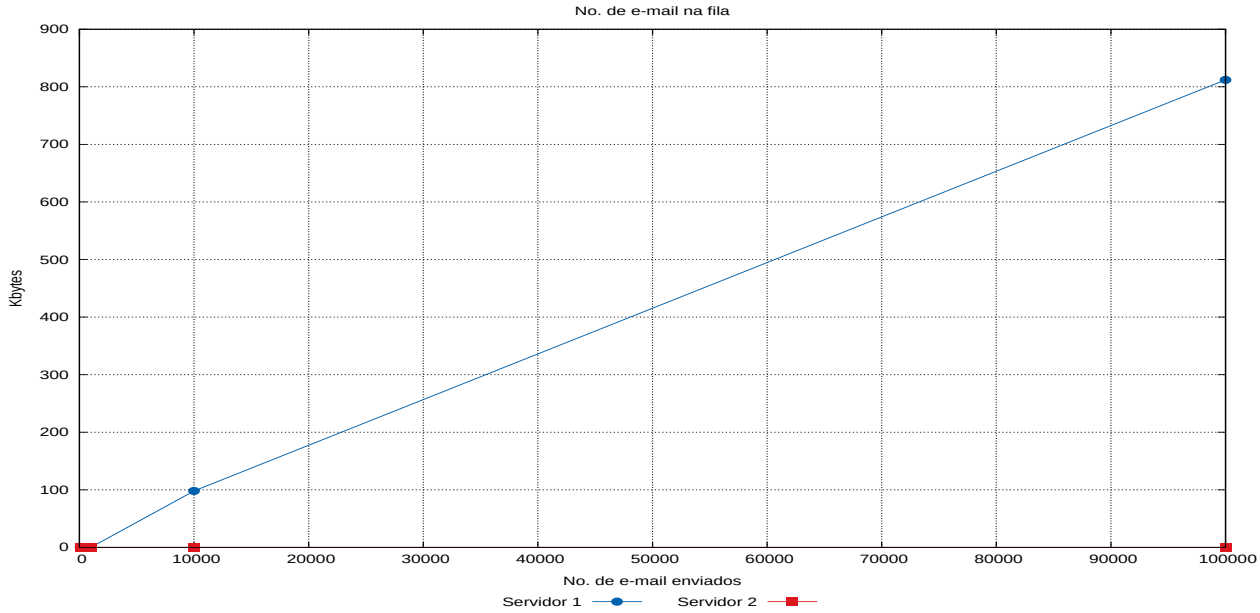

Figura 51: Média das filas de recepção de e-mail

5.4.7 Experimento 6

Os resultados do experimento 6 estão descritos na tabela 9 e, graficamente, nas Figuras 52 a 56

Tabela 10: Médias dos resultados do experimento 6

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	39	12078	12	4152	0
1000	149	125197	12	4240	0
10000	1436	1348841	27	4352	99
100000	15133	13086554	60	5647	820
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	11663	97	11	4255	0
1000	118709	165	12	4263	0
10000	1330124	1613	13	4447	0
100000	13075295	15477	13	5482	0

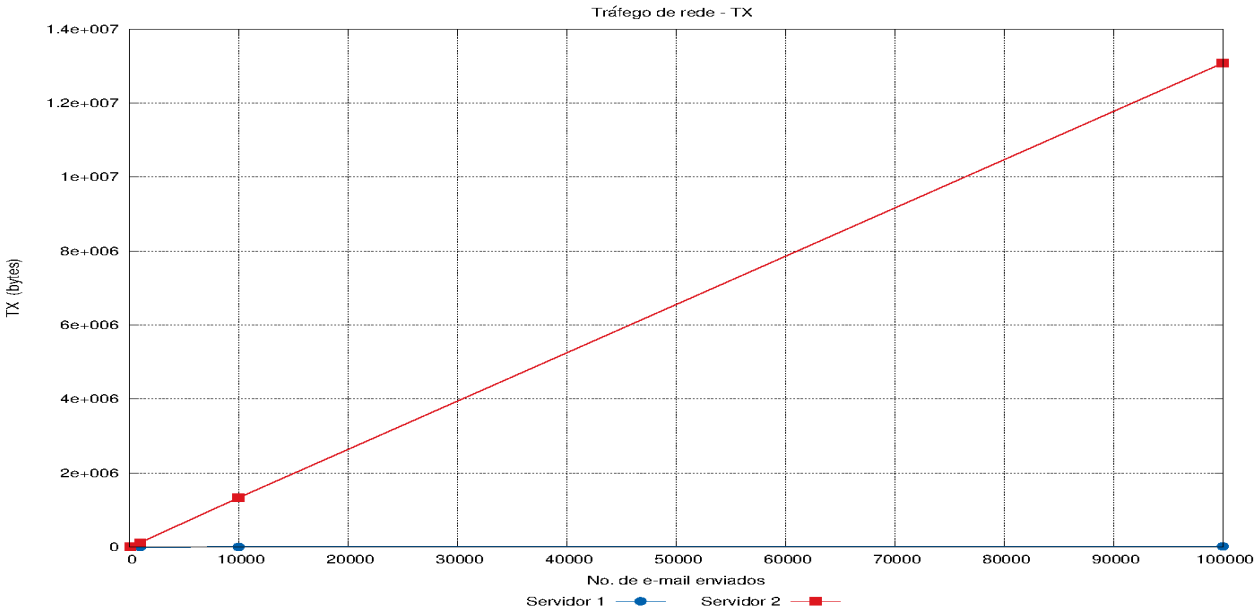

Figura 52: Média de transmissão das interfaces de rede

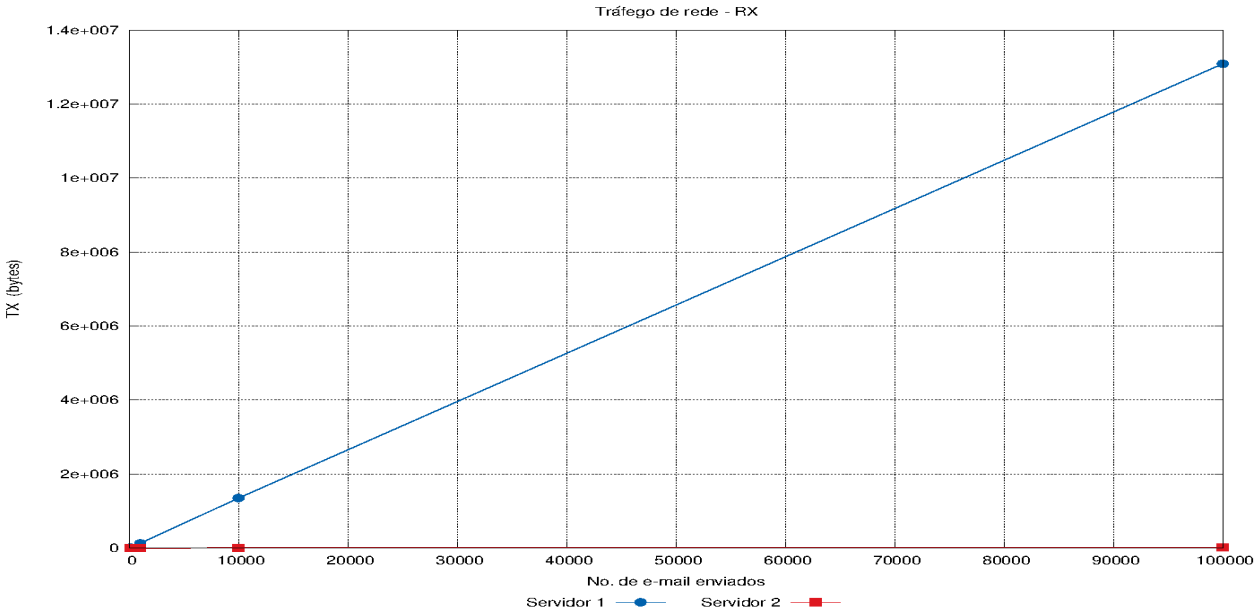

Figura 53: Média de recepção das interfaces de rede

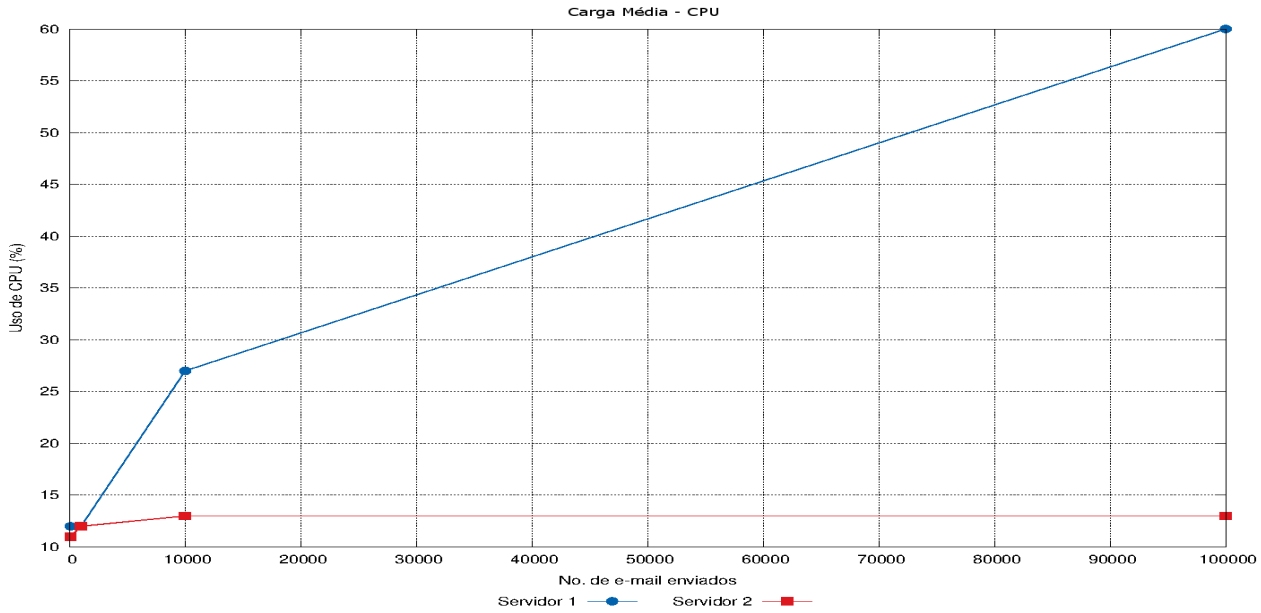

Figura 54: Carga média de uso de CPU

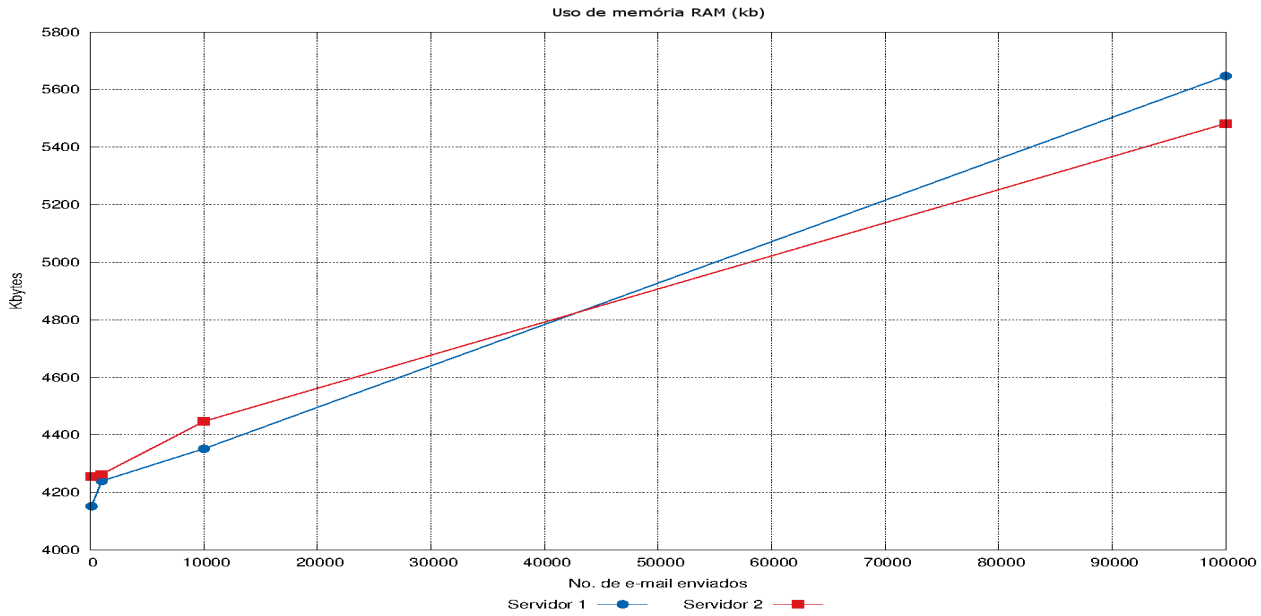

Figura 55: Média de uso de memória RAM

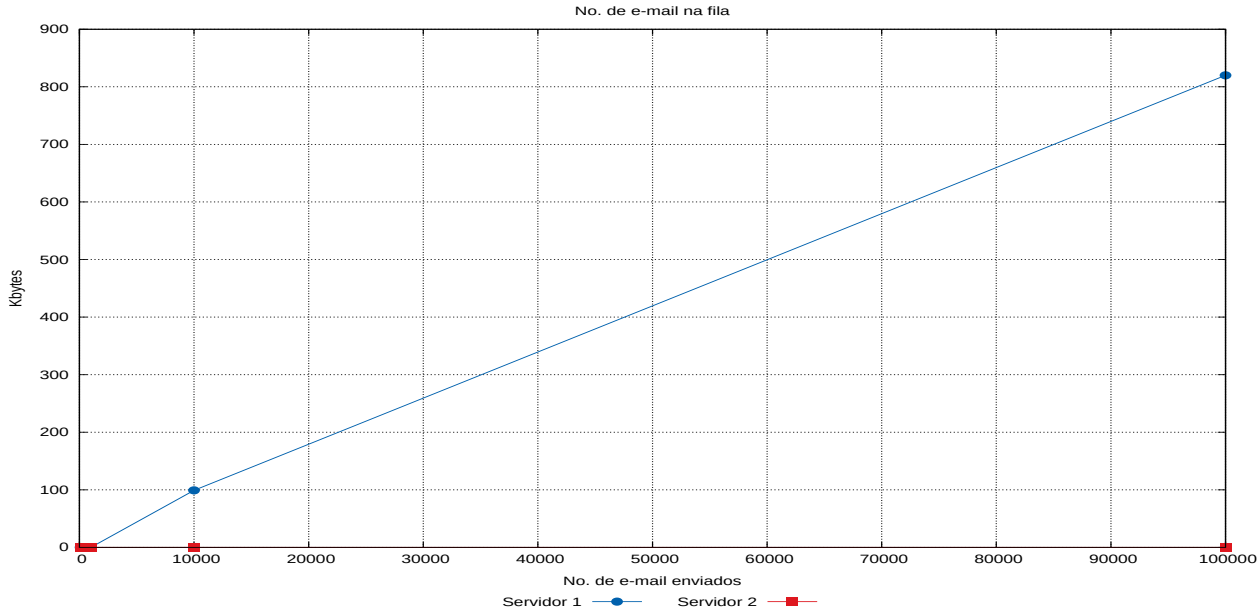

Figura 56: Média das filas de recepção de e-mail

5.4.8 Experimento 7

Os resultados do experimento 7 estão descritos na tabela 10 e, graficamente, nas Figuras 57 a 61

Tabela 11: Médias dos resultados do experimento 7

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	40	12197	12	4193	0
1000	151	126436	12	4282	0
10000	1450	1362196	28	4395	100
100000	15283	13216124	60	5703	828
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	11779	98	12	4297	0
1000	119885	166	13	4305	0
10000	1343293	1629	12	4491	0
100000	13204754	15630	12	5537	0

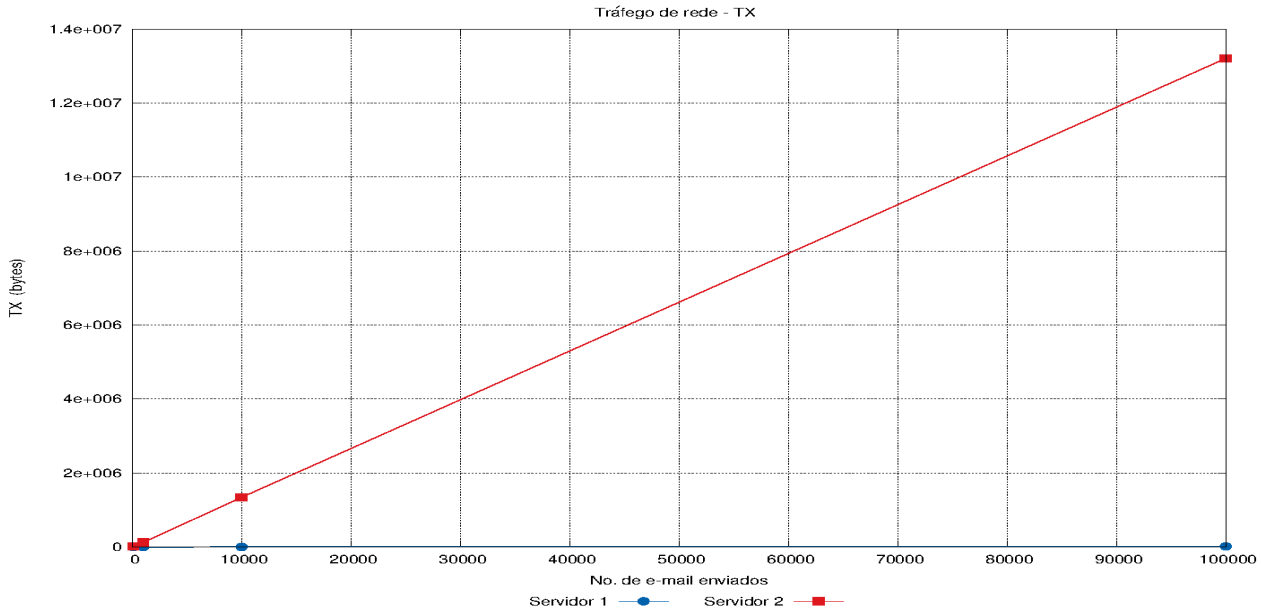

Figura 57: Média de transmissão das interfaces de rede

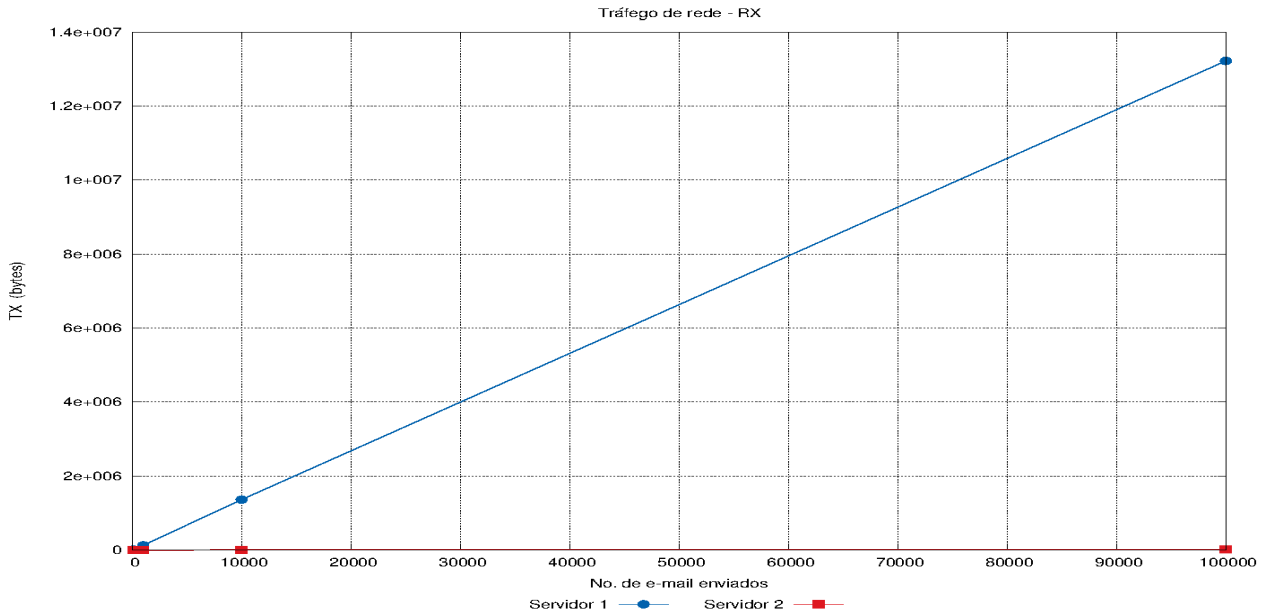

Figura 58: Média de recepção das interfaces de rede

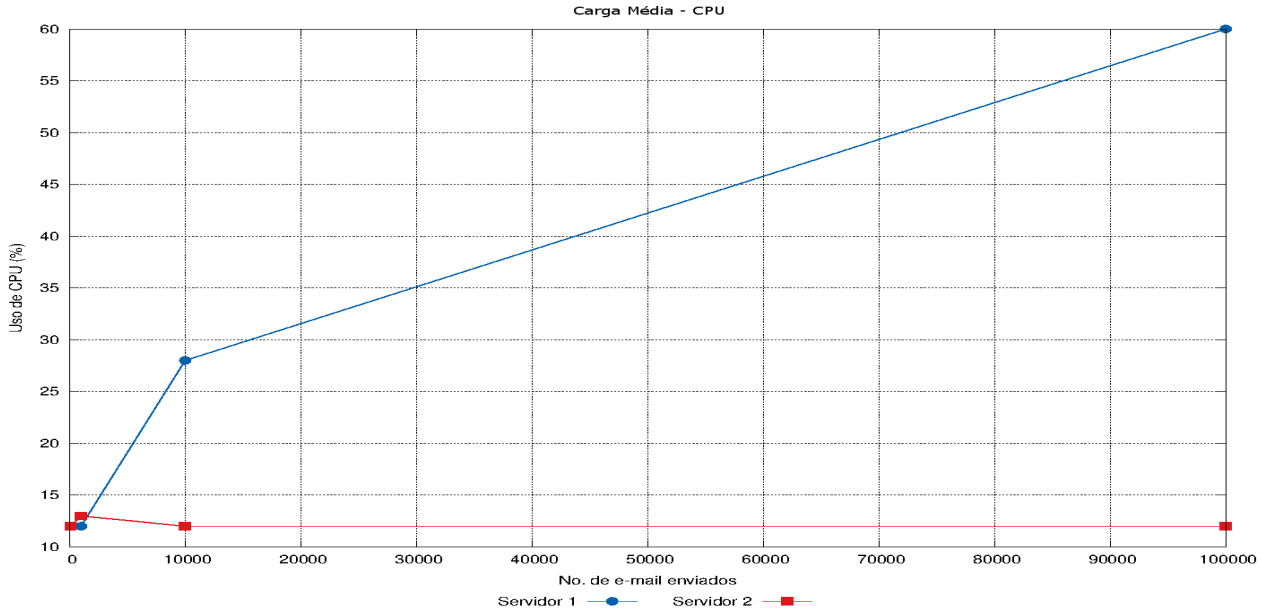

Figura 59: Carga média de uso de CPU

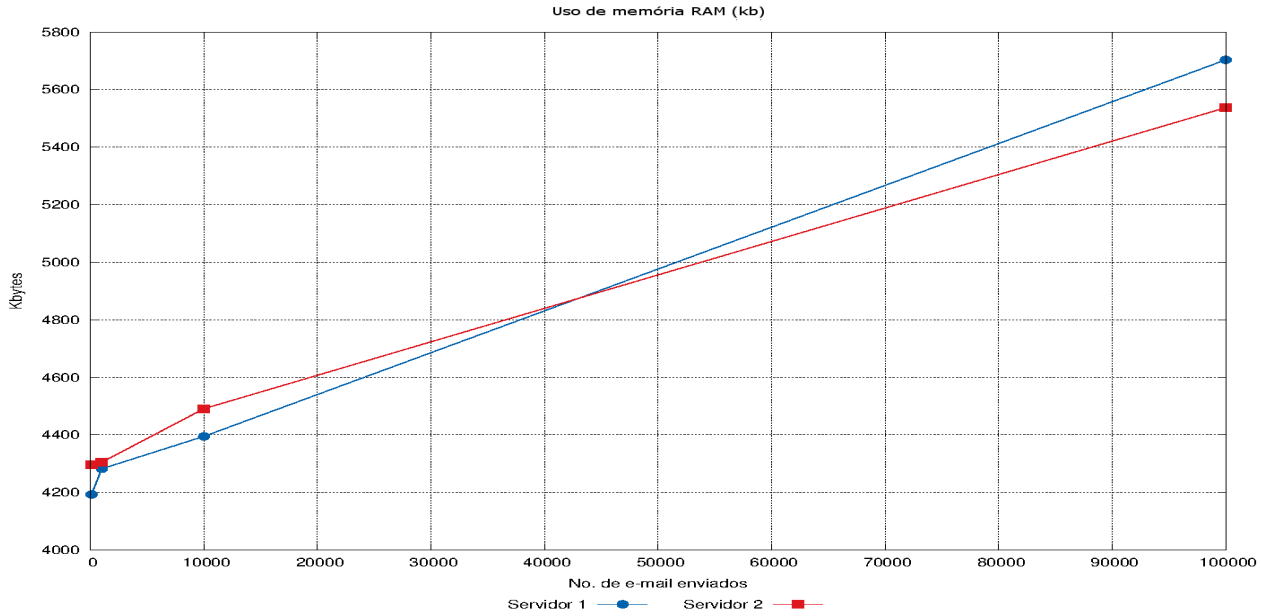

Figura 60: Média de uso de memória RAM

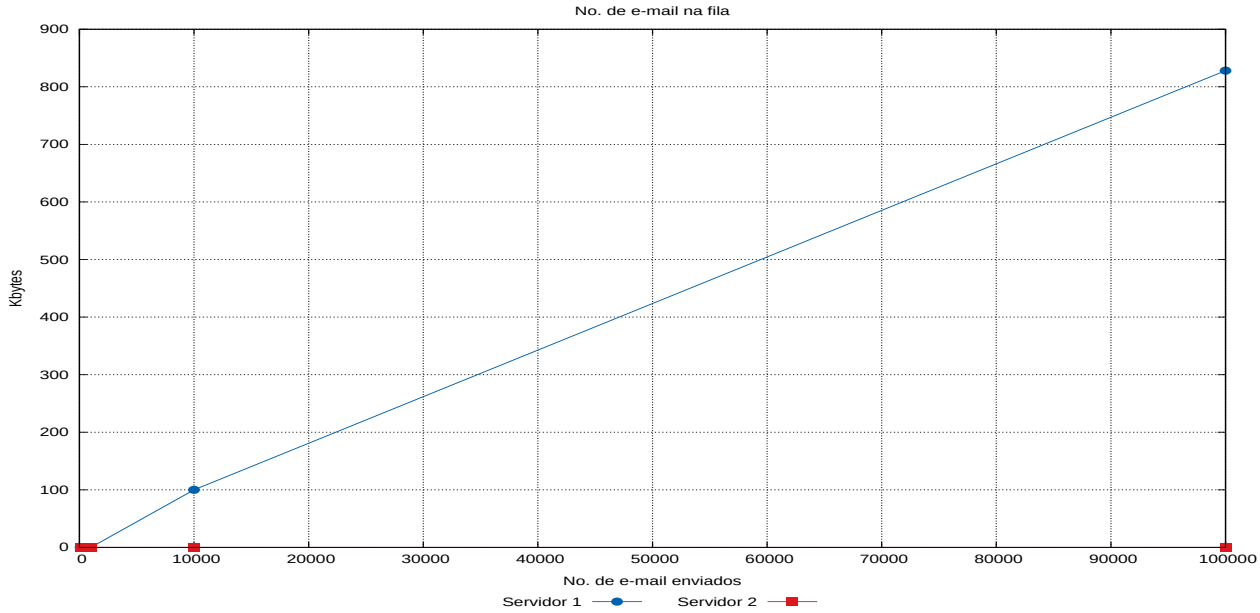

Figura 61: Média das filas de recepção de e-mail

5.4.9 Experimento 8

Os resultados do experimento 8 estão descritos na tabela 11 e, graficamente, nas Figuras 62 a 66

Tabela 12: Médias dos resultados do experimento 8

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	40	12317	12	4234	0
1000	152	127676	12	4324	0
10000	1465	1375551	28	4438	101
100000	15432	13345694	61	5759	836
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	11894	99	12	4339	0
1000	121060	168	13	4348	0
10000	1356463	1645	11	4535	0
100000	13334212	15784	11	5591	0

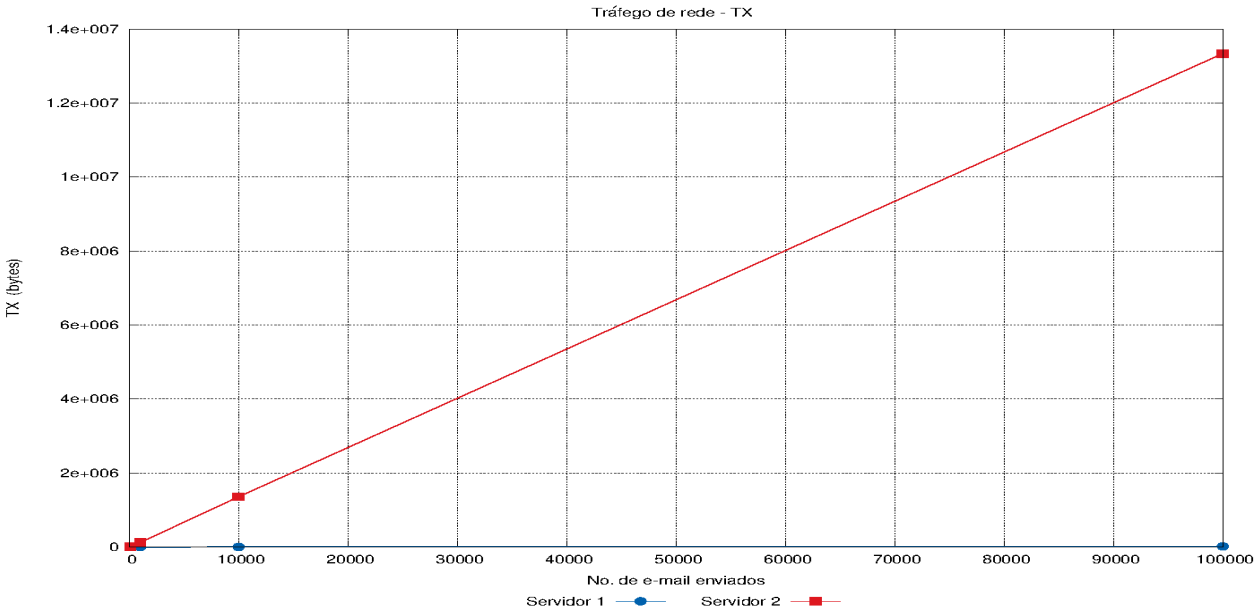

Figura 62: Média de transmissão das interfaces de rede

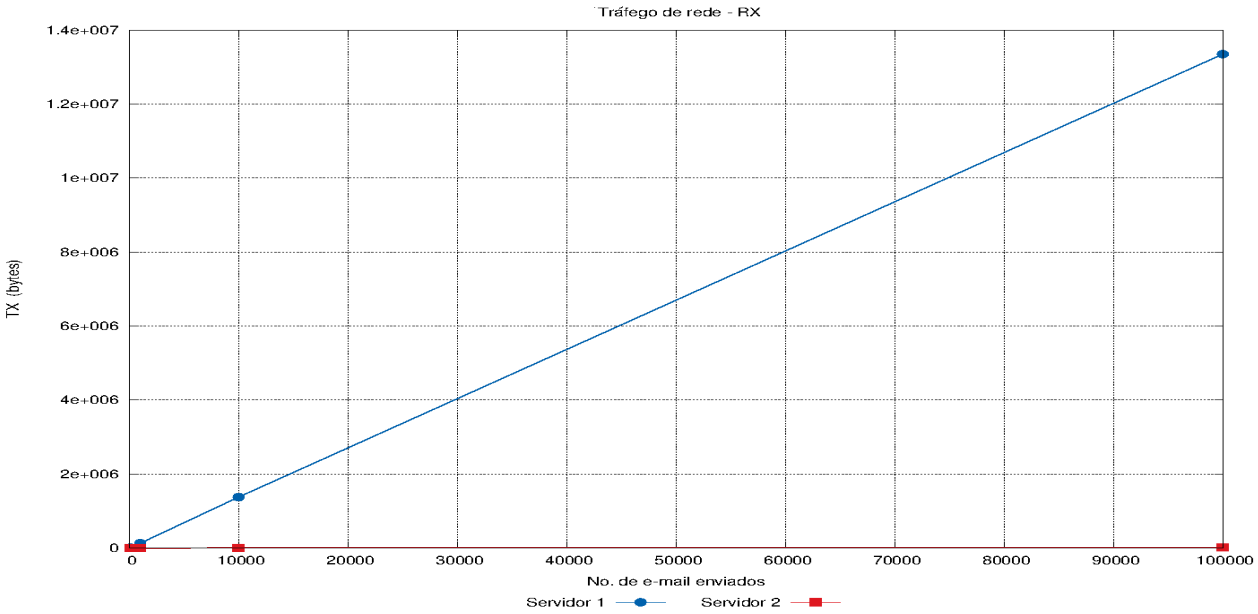

Figura 63: Média de recepção das interfaces de rede

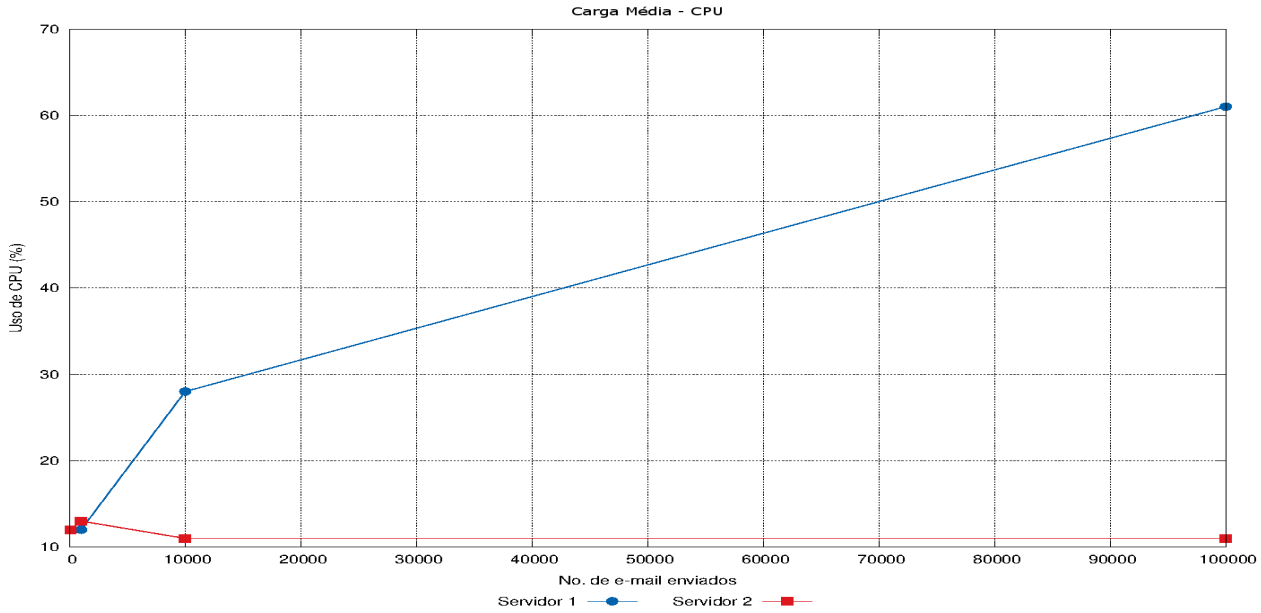

Figura 64: Carga média de uso de CPU

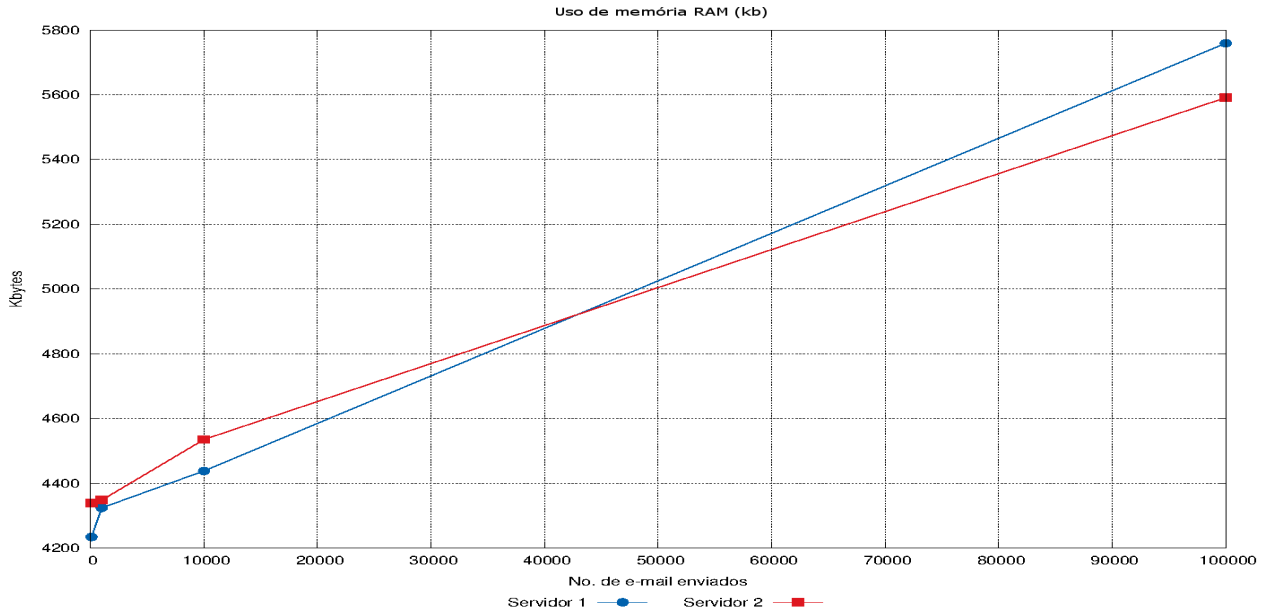

Figura 65: Média de uso de memória RAM

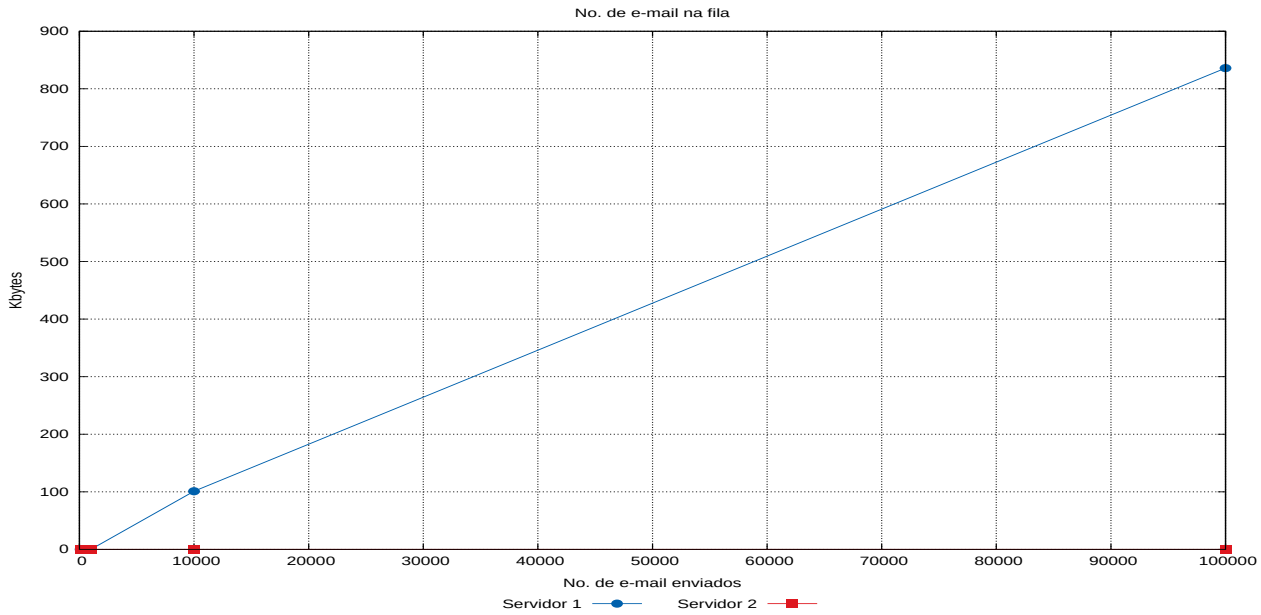

Figura 66: Média das filas de recepção de e-mail

5.4.10 Avaliação dos experimentos 5 a 8

Os experimentos 5 a 8 envolvem o envio, aos servidores, de e-mails spam somente. O servidor 2 recusa e-mails spam, gerando ao remetente a mensagem "Undelivered Mail Returned to Sender". Assim, espera-se que os resultados obtidos pelo servidor 2 sejam inferiores aos obtidos pelo servidor 1 apenas em termos da taxa média de bytes transmitidos (Tx) na interface de rede. Por outro lado, espera-se que os resultados obtidos pelo servidor 2 sejam superiores ou, ao menos, similares aos obtidos pelo servidor 1, em todas as demais métricas.

De fato, o servidor 2 transmite ao , em média, 720.57 vezes mais bytes que o servidor 1. Por outro lado, o servidor 1 recebe do *host*, em média, 641.62 vezes mais bytes que o servidor 2. Além disto, o servidor 1 usa, em média, 2.36 vezes mais CPU que o servidor 2 e o tamanho máximo alcançado por sua fila *incoming* é, em média, 231.88 vezes maior que a do servidor 2. As taxas médias de uso de memória RAM dos dois servidores foram similares.

Os resultados indicam que, ao rejeitar e-mails spam, o servidor 2 reduz sensivelmente o uso de sua CPU, bem como a quantidade máxima de e-mails em sua fila *incoming*. A transmissão (Tx) de bytes por sua interface de rede aumenta, devido à devolução imediata dos e-mails spam ao remetente. Este aumento é, entretanto, compensado pela redução na quantidade de bytes recebidos (Rx) em sua interface de rede. Os custos computacionais

de um servidor em relação ao outro, medidos através de cada uma das cinco métricas, estão todos descritos na Tabela 13 e podem ser comparadas graficamente na Figura 67. host

Tabela 13: Resultados de um servidor em relação ao outro — experimentos 5–8

Experimento 5					
No. e-mails	CPU (S1/S2)	RAM (S2/S1)	Tx (S2/S1)	Rx (S1/S2)	Fila (S1/S2)
100	1.0000	1.0248	296.10	124.56	1.00
1000	0.9231	1.0055	794.15	760.47	1.00
10000	2.4545	1.0218	926.13	836.25	99.00
100000	5.3636	0.9708	864.04	845.54	813.00
Experimento 6					
No. e-mails	CPU (S1/S2)	RAM (S2/S1)	Tx (S2/S1)	Rx (S1/S2)	Fila (S1/S2)
100	1.0909	1.0248	299.05	124.52	1.00
1000	1.0000	1.0054	796.70	758.77	1.00
10000	2.0769	1.0218	926.27	836.23	100.00
100000	4.6154	0.9708	864.03	845.55	821.00
Experimento 7					
No. e-mails	CPU (S1/S2)	RAM (S2/S1)	Tx (S2/S1)	Rx (S1/S2)	Fila (S1/S2)
100	1.0000	1.0248	294.48	124.46	1.00
1000	0.9231	1.0054	793.94	761.66	1.00
10000	2.3333	1.0218	926.41	836.22	101.00
100000	5.0000	0.9709	864.02	845.56	829.00
Experimento 8					
No. e-mails	CPU (S1/S2)	RAM (S2/S1)	Tx (S2/S1)	Rx (S1/S2)	Fila (S1/S2)
100	1.0000	1.0248	297.35	124.41	1.00
1000	0.9231	1.0056	796.45	759.98	1.00
10000	2.5455	1.0219	925.91	836.20	102.00
100000	5.5455	0.9708	864.06	845.52	837.00
Média					
	CPU	RAM	Tx	Rx	Fila
Média	*2.3622*	*1.0057*	*720.57*	*641.62*	*231.88*

A Figura 67 apresenta um gráfico comparando a média das métricas nos dois servi-

dores.

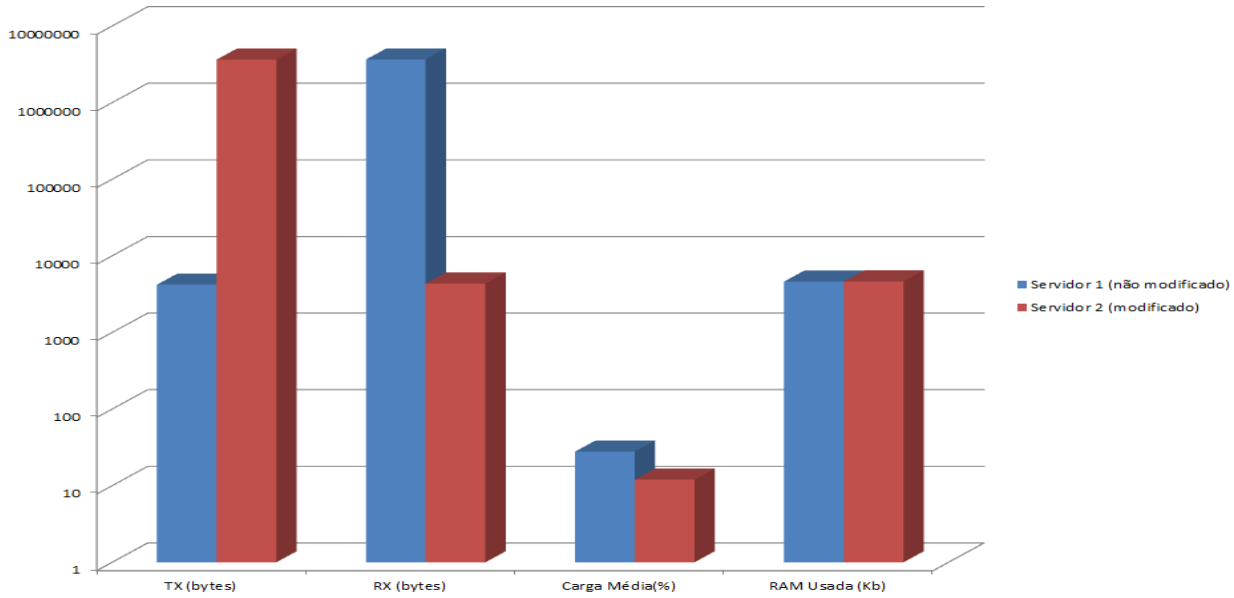

Figura 67: Média das métricas nos servidores

5.4.11 Experimento 9

No experimento 9, 78% dos e-mails enviados pelo *host* eram spam. Este percentual é igual ao percentual médio de recebimento de spam pelo servidor da universidade no período de julho de 2013 a junho de 2014. Os resultados do experimento 9 estão descritos na tabela 12 e, graficamente, nas Figuras 68 a 72.

Tabela 14: Médias dos resultados do experimento 9

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	35	1064	11	3659	0
1000	132	11032	11	3736	0
10000	1266	118858	24	3835	87
100000	13335	1153171	53	4976	723
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	8017	415	11	3793	0
1000	81592	3472	12	3864	0
10000	914229	37254	18	3976	64
100000	8987000	361275	39	4654	528

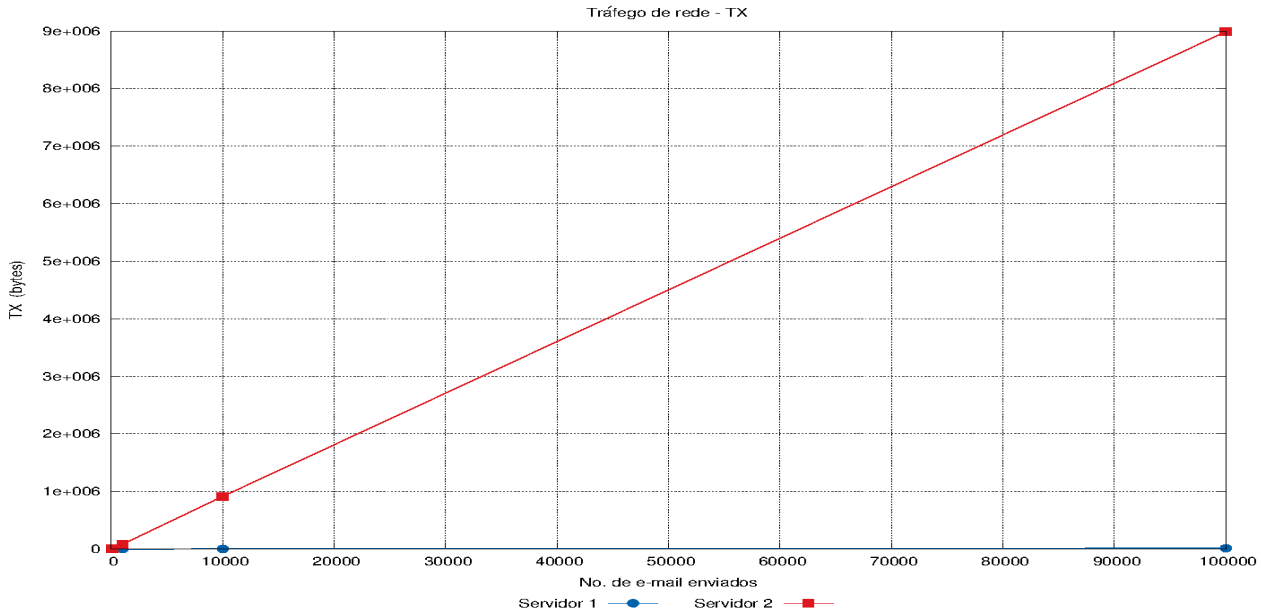

Figura 68: Média de transmissão das interfaces de rede

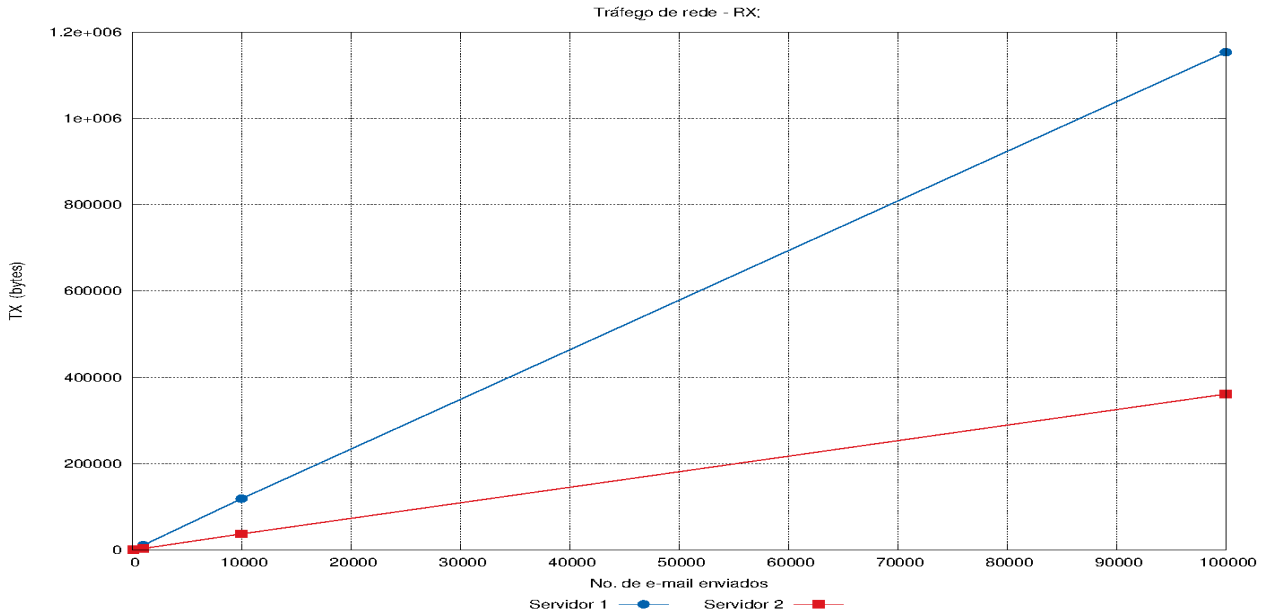

Figura 69: Média de recepção das interfaces de rede

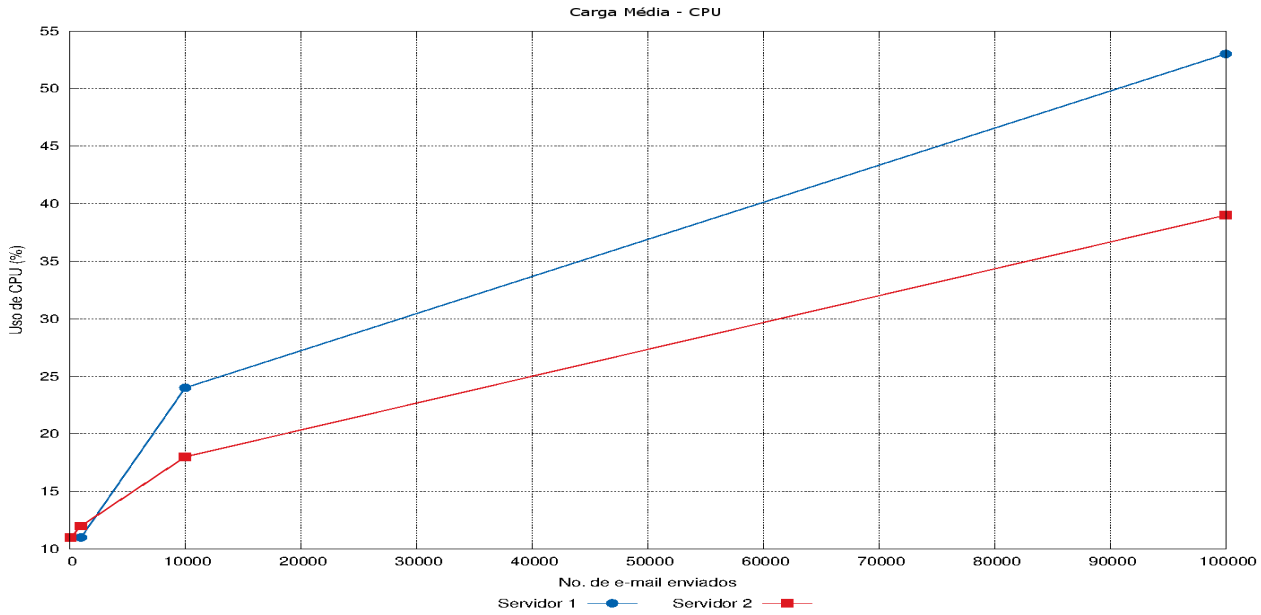

Figura 70: Carga média de uso de CPU

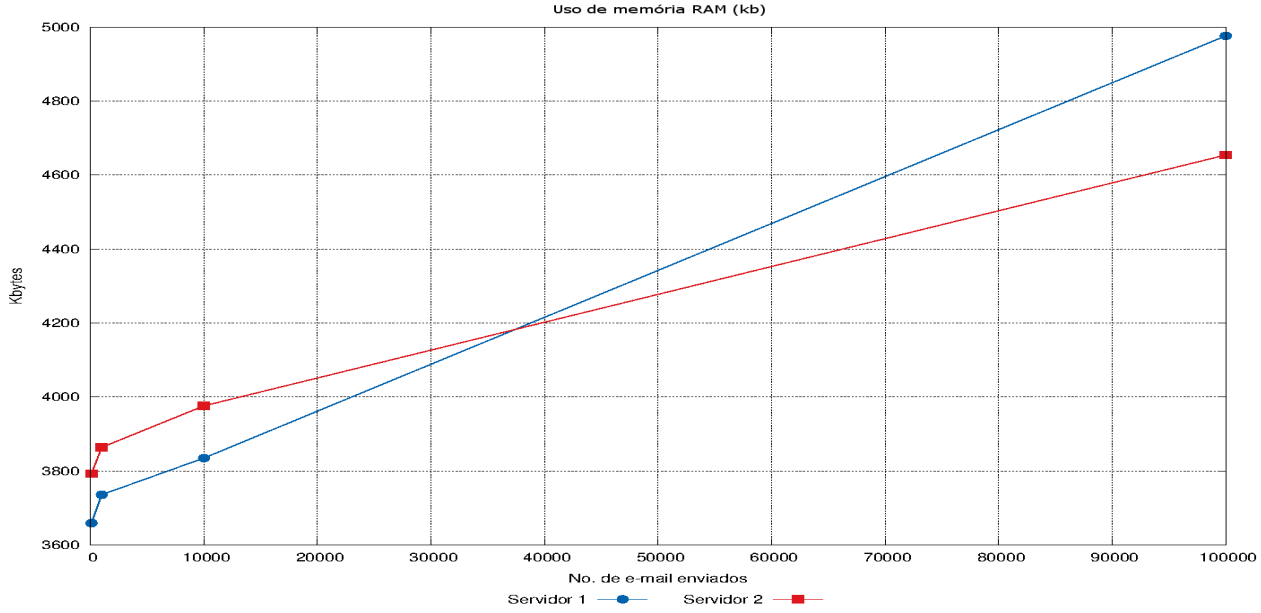

Figura 71: Média de uso de memória RAM

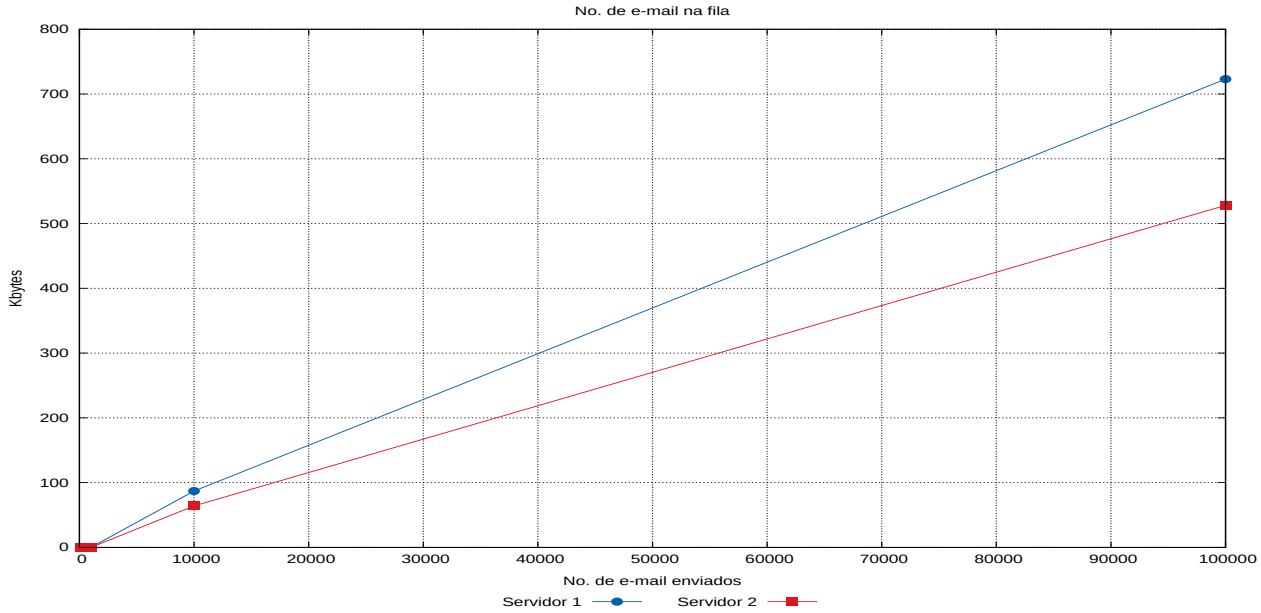

Figura 72: Média das filas de recepção de e-mail

5.4.12 Experimento 10

No experimento 10, 78% dos e-mails enviados pelo *host* também eram spam. Os resultados do experimento 10 estão descritos na tabela 13 e, graficamente, nas Figuras 73 a 77.

Tabela 15: Médias dos resultados do experimento 10

Servidor 1 (Não modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	37	11360	11	3905	0
1000	141	117759	11	3988	0
10000	1351	1268712	26	4094	93
100000	14234	12309135	56	5311	771
Servidor 2 (Modificado)					
Envios	TX (bytes)	RX (bytes)	Carga Média(%)	RAM Usada (Kb)	Fila (N° de emails)
100	8197	3504	11	3793	0
1000	83426	35491	12	3864	0
10000	934774	382211	19	3976	68
100000	9188955	3708064	42	4654	563

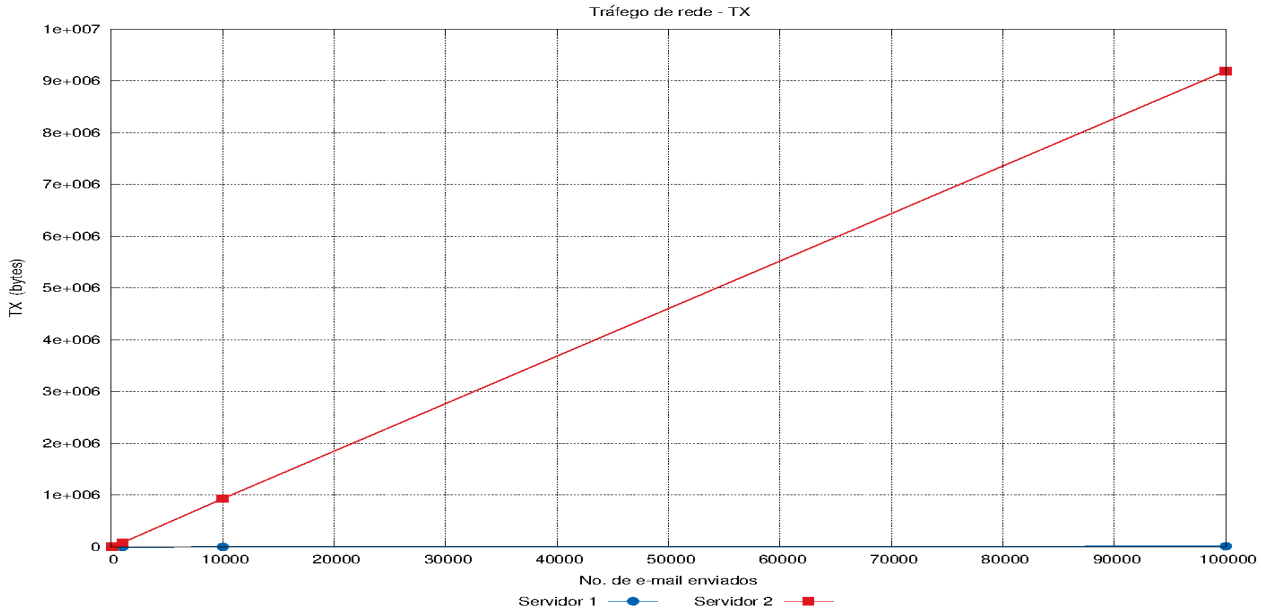

Figura 73: Média de transmissão das interfaces de rede

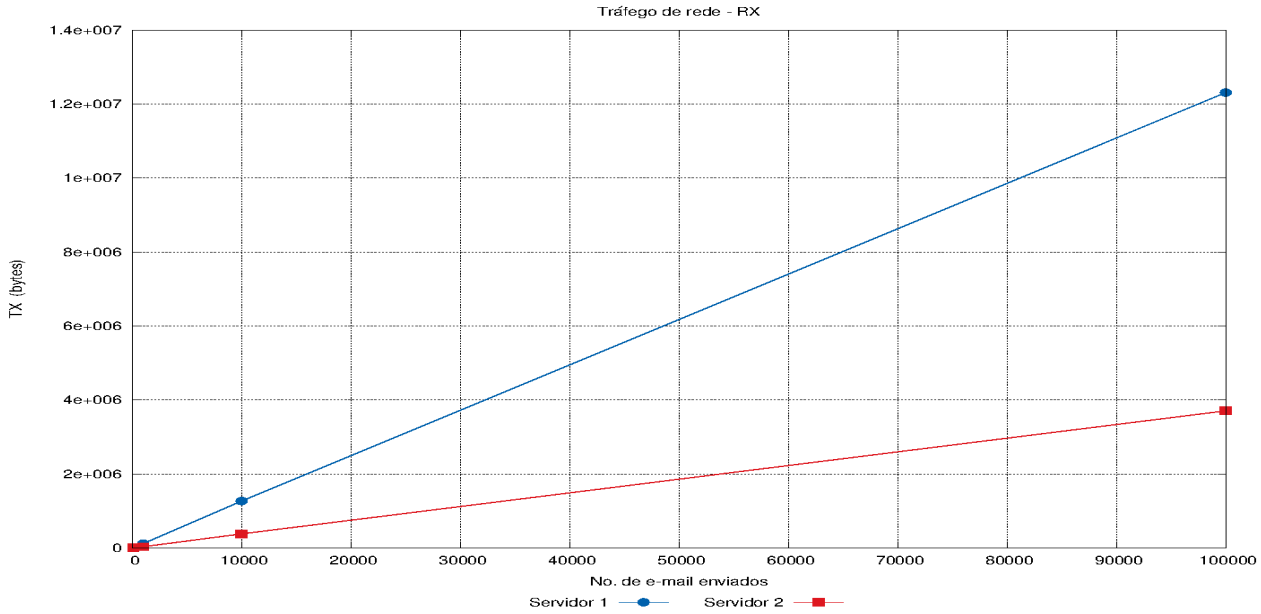

Figura 74: Média de recepção das interfaces de rede

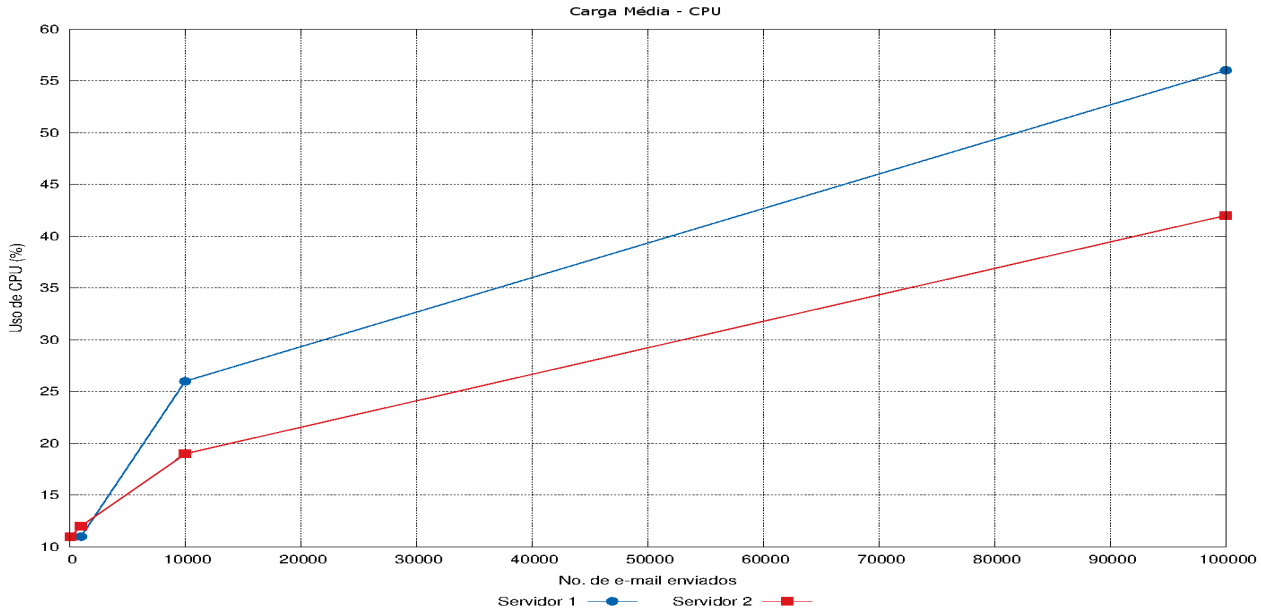

Figura 75: Carga média de uso de CPU

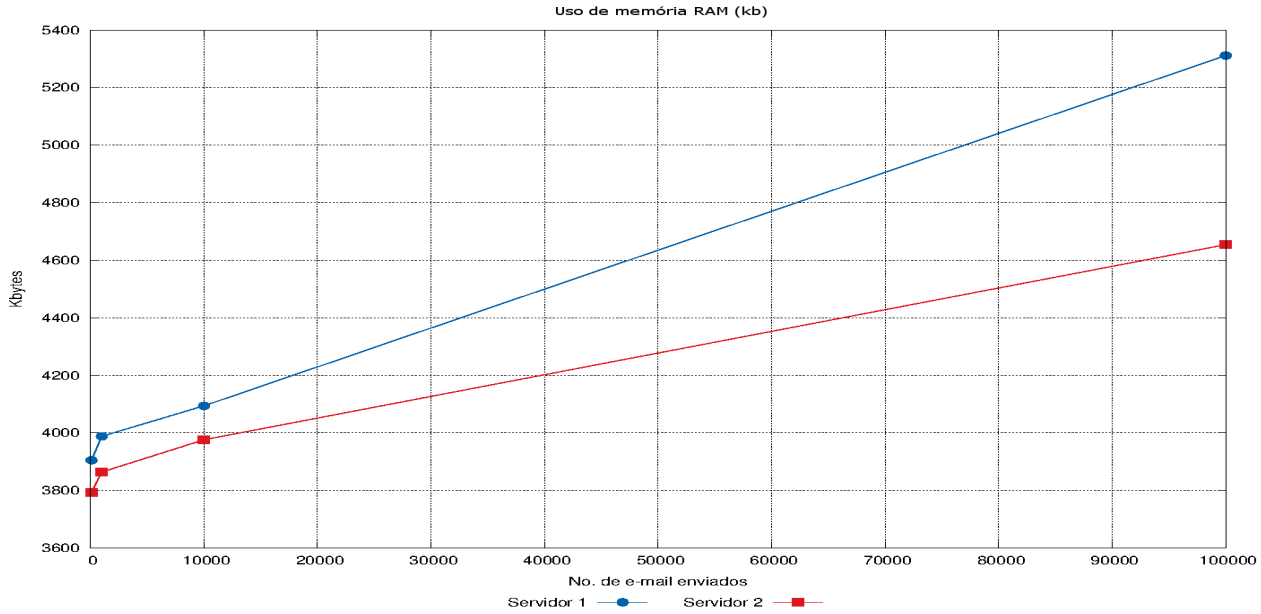

Figura 76: Média de uso de memória RAM

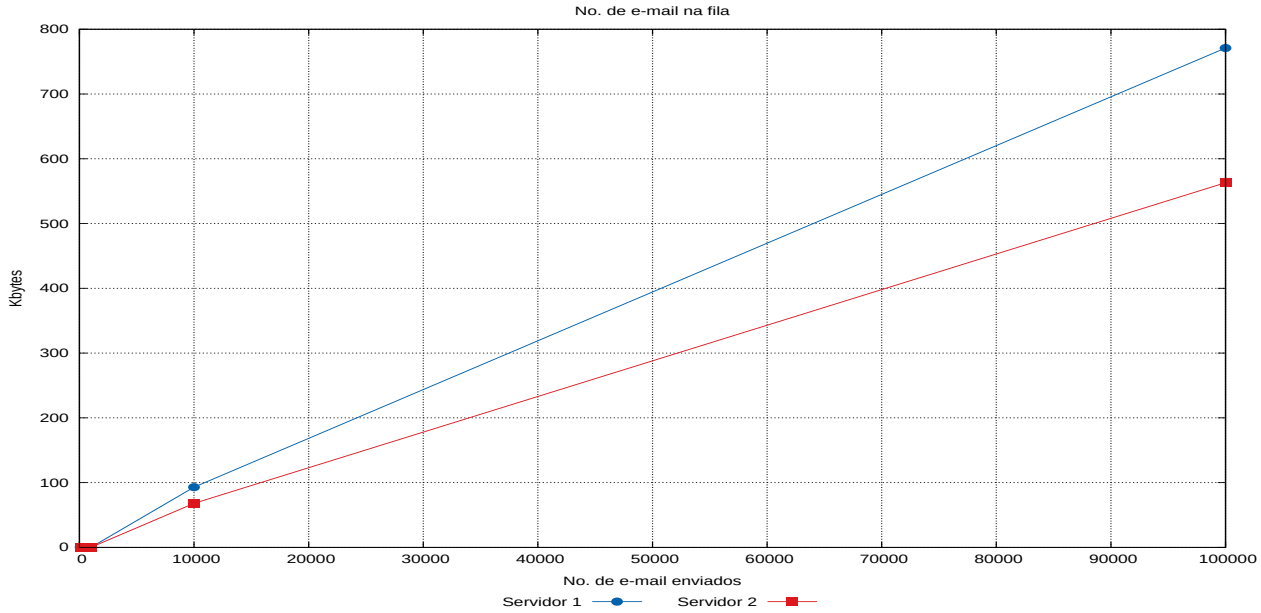

Figura 77: Média das filas de recepção de e-mail

5.4.13 Avaliação dos experimentos 9 e 10

Os experimentos 9 e 10 envolvem o envio, aos servidores, de 22% de e-mails ham e 78 % de e-mails spam. Este percentual é igual ao percentual médio de recebimento de spam do servidor da universidade no período de julho de 2013 a junho de 2014. Assim, espera-se que os resultados obtidos pelos servidores situem-se entre os valores dos resultados dos experimentos 1–4 e os dos resultados dos experimentos 5–8.

De fato, em quatro das métricas — taxa média de uso de CPU, taxas médias de bytes transmitidos (Tx) e recebidos (Rx) na interface de rede e número máximo de e-mails presentes na fila *incoming* —, os resultados situaram-se entre os dos experimentos 1–4 e os dos experimentos 5–8. As taxas médias de uso de memória RAM dos dois servidores foram similares às dos demais experimentos. Os custos computacionais de um servidor em relação ao outro, medidos através de cada uma das cinco métricas, estão todos descritos na Tabela 16 e podem ser comparadas graficamente na Figura 78.

Tabela 16: Resultados de um servidor em relação ao outro — experimentos 9–10

Experimento 9					
No. e-mails	CPU (S1/S2)	RAM (S1/S2)	Tx (S2/S1)	Rx (S1/S2)	Fila (S1/S2)
100	1.0000	0.9647	229.06	2.56	1.00
1000	0.9167	0.9669	618.12	3.18	1.00
10000	1.3333	0.9645	722.14	3.19	1.36
100000	1.3590	1.0692	673.94	3.19	1.37
Experimento 10					
No. e-mails	CPU (S1/S2)	RAM (S1/S2)	Tx (S2/S1)	Rx (S1/S2)	Fila (S1/S2)
100	1.0000	1.0295	221.54	3.24	1.00
1000	0.9167	1.0321	591.67	3.32	1.00
10000	1.3684	1.0297	691.91	3.32	1.37
100000	1.3333	1.1412	645.56	3.32	1.37
Média					
	CPU	RAM	Tx	Rx	Fila
Mean	*1.1534*	*1.0247*	*549.24*	*3.16*	*1.18*

A Figura 78 apresenta um gráfico comparando a média das métricas nos dois servidores.

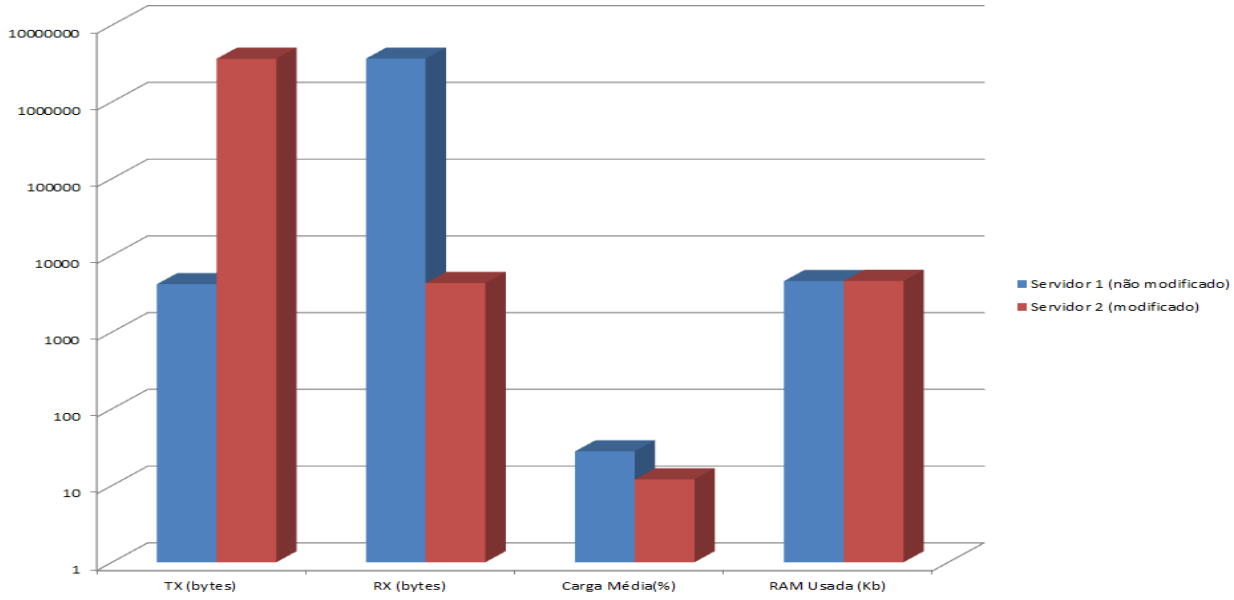

Figura 78: Média das métricas nos servidores

5.4.14 O desempenho do host

Em todos os experimentos realizados, as taxas de uso de CPU, de memória RAM e de transmissão (Tx) e recepção (Rx) de bytes na interface de rede do *host* foram medidas. A Tabela 17 apresenta as taxas médias obtidas pelo *host* nos dez experimentos, discriminadas tanto pelo tipo de e-mail enviado quanto pela quantidade de e-mails enviados.

Tabela 17: Resultados do host nos dez experimentos

Experimentos 1–4 — somente ham				
No. e-mails	CPU (%)	RAM (Kbyte)	Tx (byte)	Rx (byte)
100	10	2624	27285	99
1000	11	2673	277514	355
10000	12	2740	3035048	3349
100000	13	3271	28108821	35492
Mean	*11.50*	*2827.00*	*7862167.00*	*9823.75*

Experiments 5–8 — Somente spam				
No. e-mails	CPU (%)	RAM (Kbyte)	Tx (byte)	Rx (byte)
100	19	5211	13126	12491
1000	21	5308	135165	126863
10000	23	5442	1456066	1421225
100000	26	6496	14126675	13971929
Mean	*22.25*	*5614.25*	*3932758.00*	*3883127.00*

Experiments 9–10 — ham/spam				
No. e-mails	CPU (%)	RAM (Kbyte)	Tx (byte)	Rx (byte)
100	18	4904	16343	8957
1000	20	4996	167754	90910
10000	22	5122	1807034	1018391
100000	24	6114	17531646	10011938
Mean	*21.00*	*5284.00*	*4880694.00*	*2782549.00*

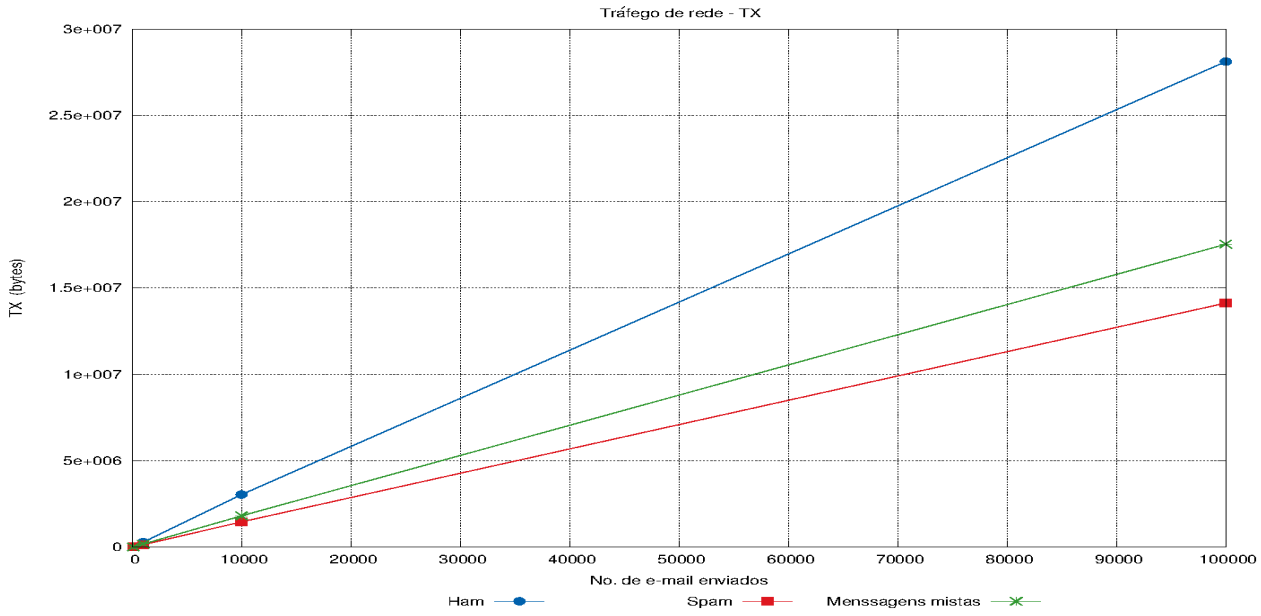

Figura 79: Média de transmissão das interfaces de rede

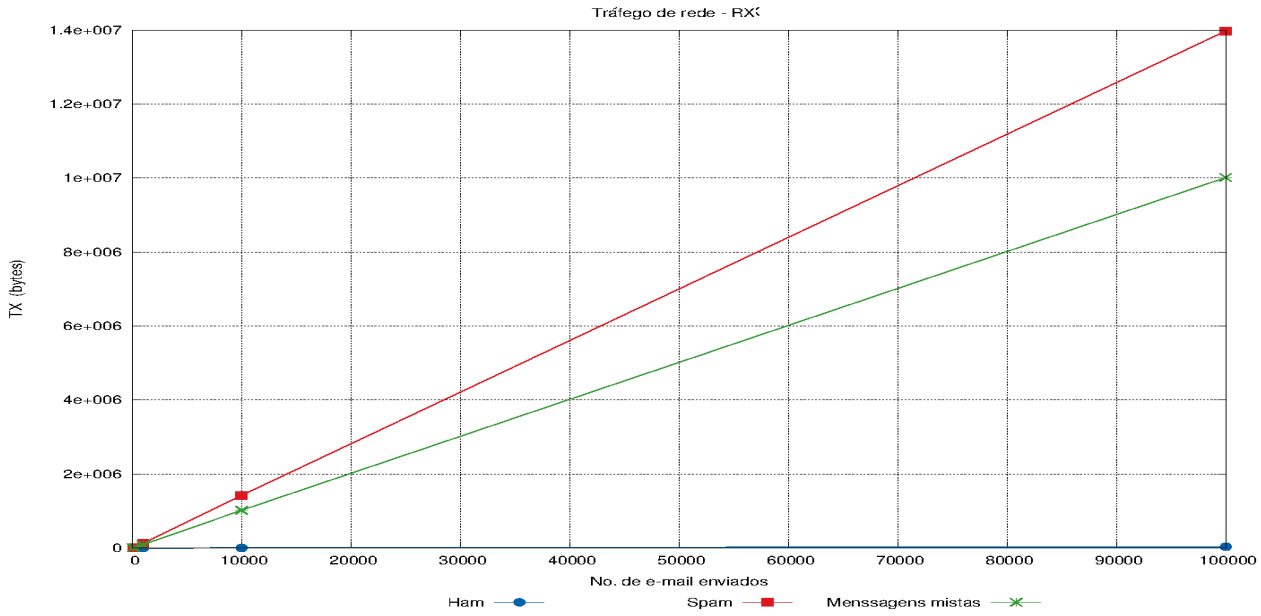

Figura 80: Média de recepção das interfaces de rede

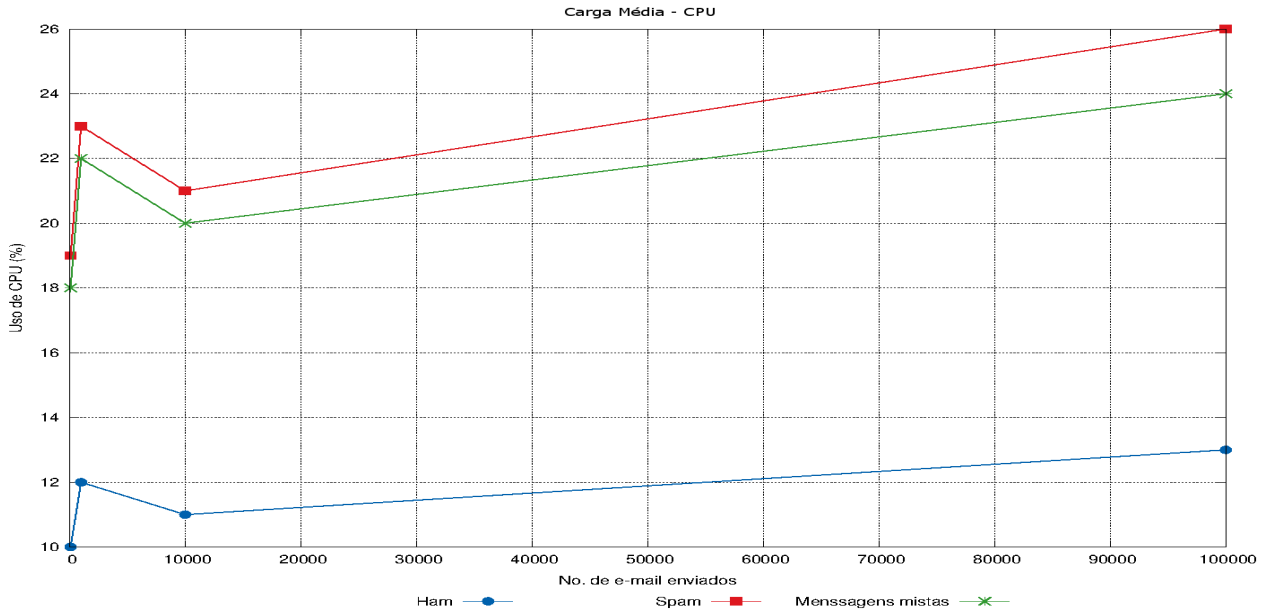

Figura 81: Carga média de uso de CPU

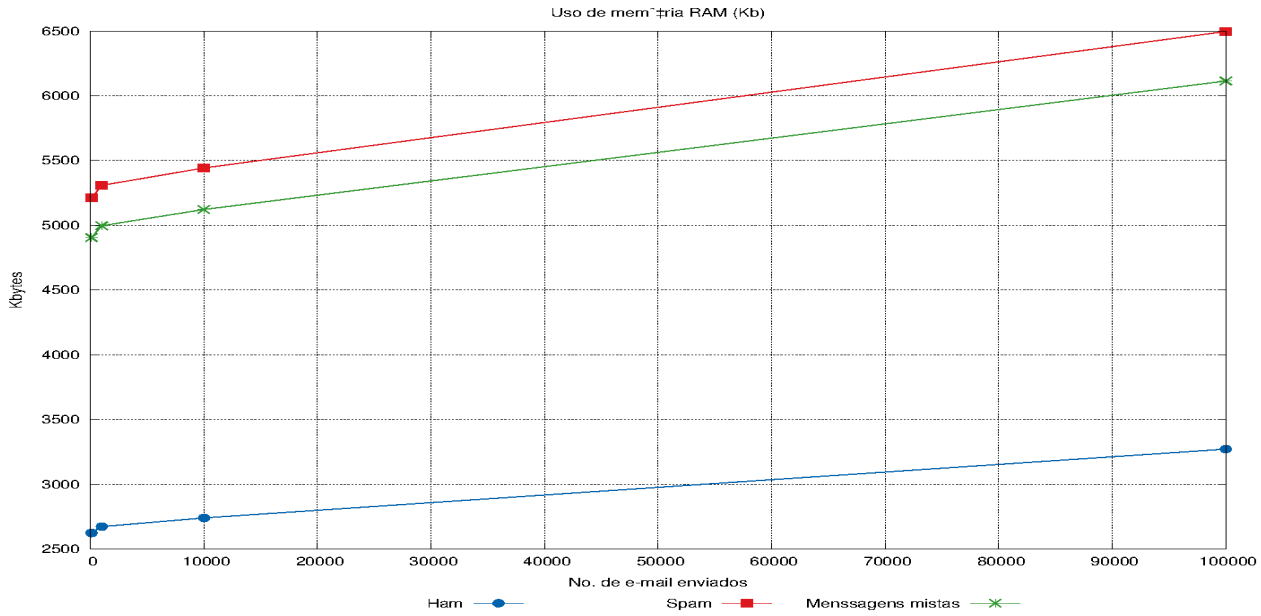

Figura 82: Média de uso de memória RAM

É possível observar nos resultados descritos na Tabela 17 e nas Figuras 77 até 80, que há um custo computacional considerável no *host* quando este envia e-mails spam. Com o envio de e-mails spam somente, suas taxas médias de uso de CPU, de memória RAM e de recepção (Rx) de bytes por sua interface de rede sofrem um aumento de, respectivamente, 93.48%, 98.59% e de quase 40000% em relação às taxas médias obtidas quando envia e-mails ham somente. No entanto, com o envio de e-mails spam somente, a taxa média

de transmissão (Tx) de bytes por sua interface de rede sofre uma redução de 50% em relação à taxa média obtida quando do envio de e-mails ham somente A ideia é comparada graficamente na Figura 83.

A Figura 83 apresenta um gráfico comparando a média das métricas nos dois servidores.

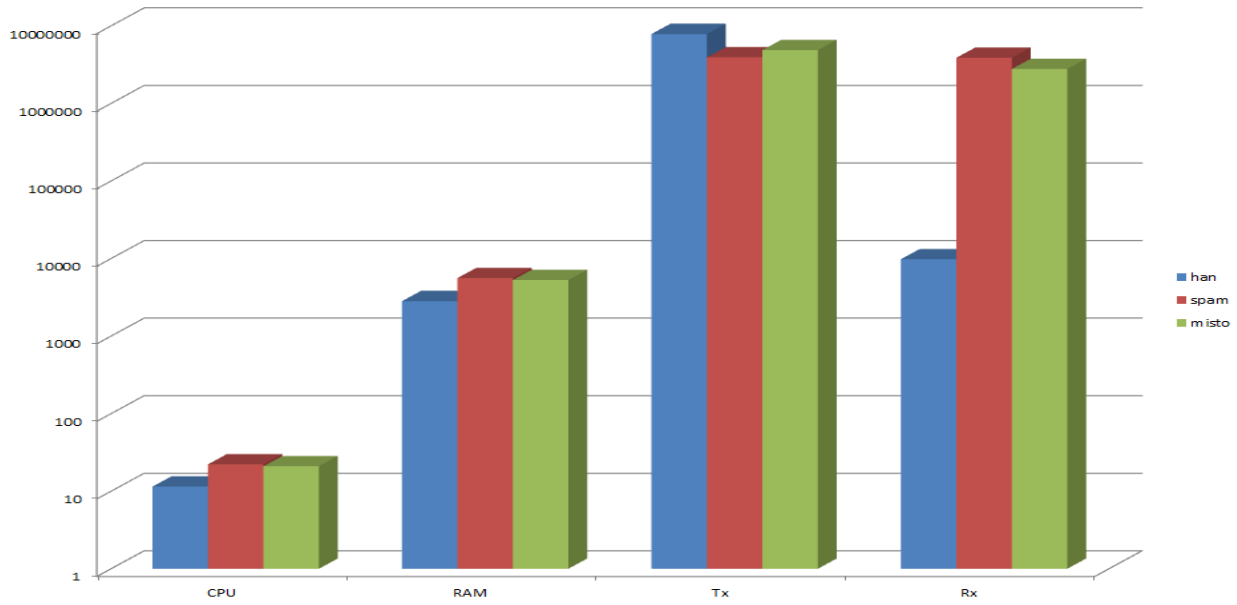

Figura 83: Média das métricas nos servidores

5.4.15 Plug-in para cliente de e-mail

Como complemento a este trabalho de dissertação, o aluno Mateus de Bem Vieira, em seu trabalho de iniciação científica, desenvolveu um *plug-in* para o cliente de e-mail *Mozilla Thunderbird*. O *plug-in* cria uma interface para o usuário cadastrar, de forma simples, endereços que considere de spam. O *plug-in*, escrito em *javascript*, comunica-se com o servidor de e-mail Zimbra através de um *web service*. O *web service*, escrito em php, contém as credenciais para a comunicação com o banco de dados do Zimbra. Assim, o *plug-in* pode ser distribuído livremente posto que não contém informações confidenciais.

A interface do plug-in permite ao usuário adicionar, listar e excluir facilmente endereços de e-mail. As Figuras 82 e 83 apresentam a interface do plug-in.

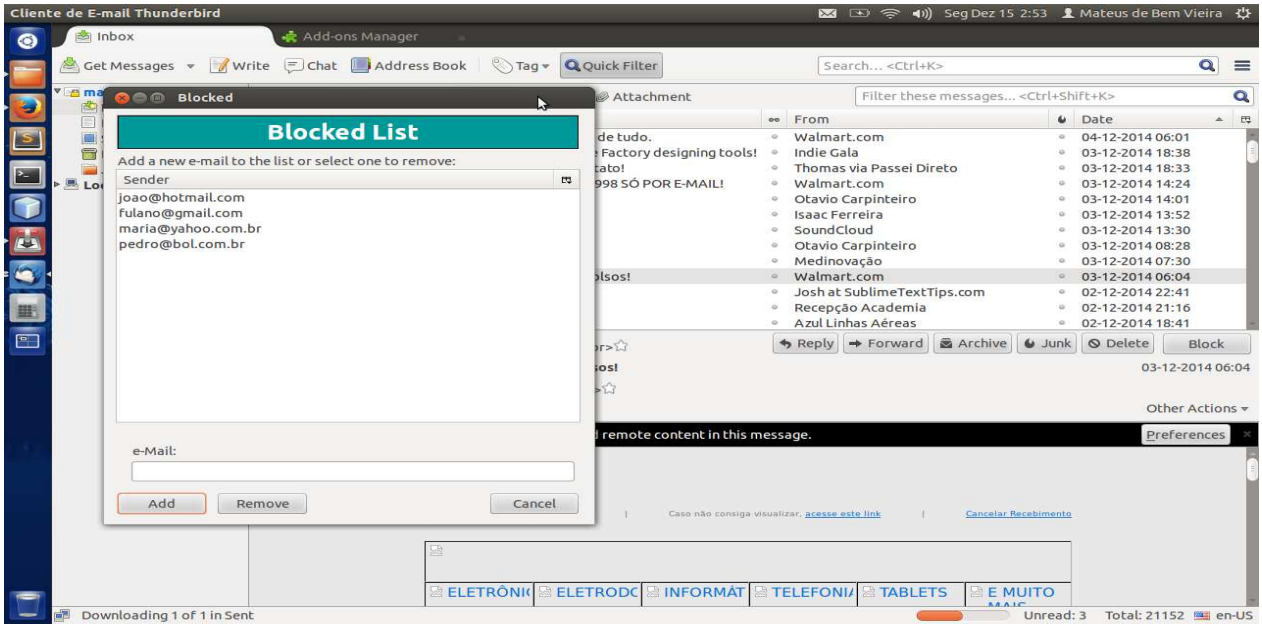

Figura 84: Interface do plug-in - lista de e-mails bloqueados

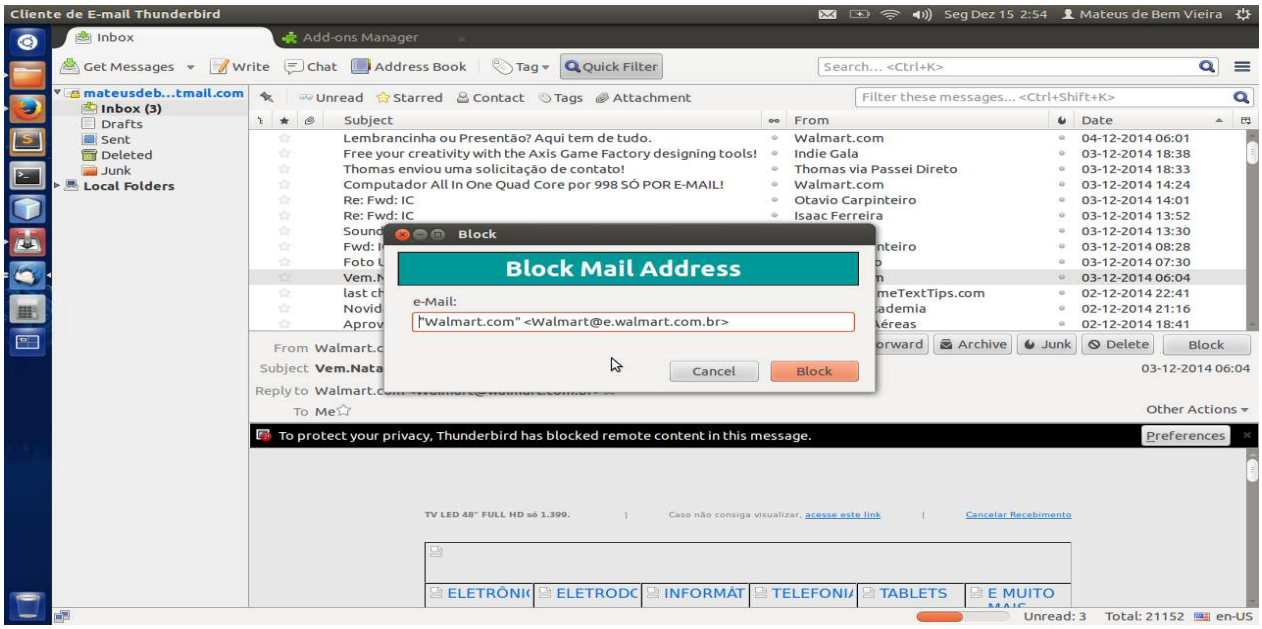

Figura 85: Interface do plug-in - cadastro de e-maisl para bloqueio

6 Conclusão

Este trabalho propõe uma modificação no SMTP para redução de spam. A modificação é implementada em um servidor de e-mail Zimbra. O servidor permite aos usuários marcar, através de um plug-in para o cliente de e-mail Mozilla Thunderbird, quais remetentes (ou domínios) lhes são indesejados.

As mensagens spam são devolvidas aos remetentes, tal como se o usuário não existisse. Assim, não só os recursos computacionais do servidor de destino são preservados como também os do servidor do spammer são onerados.

Foram realizados diversos experimentos para avaliar o custo computacional do servidor modificado em relação ao do servidor original. Quando o host envia e-mails ham, o custo adicional sobre o servidor modificado é insignificante. Quando o host envia e-mails spam, o servidor modificado reduz sensivelmente o uso de sua CPU, bem como a quantidade máxima de e-mails em sua fila *incoming*. A transmissão (Tx) de bytes por sua interface de rede aumenta, devido à devolução imediata dos e-mails spam ao host. Este aumento é, entretanto, compensado pela redução na quantidade de bytes recebidos (Rx) em sua interface de rede.

Com os experimentos, verificou-se, igualmente, que há um custo computacional considerável no host quando este envia e-mails spam. Com o envio de e-mails spam somente, suas taxas médias de uso de CPU, de memória RAM e de recepção (Rx) de bytes por sua interface de rede sofrem um aumento de, respectivamente, 93.48%, 98.59% e de quase 40000% em relação às taxas médias obtidas quando envia e-mails ham somente. No entanto, com o envio de e-mails spam somente, a taxa média de transmissão (Tx) de bytes por sua interface de rede sofre uma redução de 50% em relação à taxa média obtida quando do envio de e-mails ham somente.

6.0.16 Trabalhos futuros

Há algumas possibilidades para prosseguimento deste trabalho. Primeiro, testar outros motores do SGBD MySql para averiguar se possuem melhor desempenho que o InnoDB. Segundo, implementar um serviço para remover entradas expiradas de endereços de e-mail no banco de dados do Zimbra. O prazo de expiração dos endereços pode ser definido pelo administrador. Terceiro, instalação do servidor Zimbra modificado no ambiente de produção da UNIFEI.

Referências

AGRAWAL, N. K. B.; MOLLE, M. Controlling spam emails at the routers. *International Conference on Communications*, v. 3, n. 1, Dezembro 2005.

BUJANG, Y. R. Should we be concerned with spam emails ? a look at its impacts and implications. *Information and Communication Technology for the Muslim World (ICT4M),5th International Conference on*, v. 1, n. 1, Março 2013.

CLAYTON, R. Stopping spam by extrusion detection. *In First Conference on Email and Anti-Spam*, v. 1, n. 1, Julho 2004.

DELL TechCenter. 2014. Disponível em: <`http://tinyurl.com/nb29fdt`>.

GEORGALA, K.; PALIOURAS, G. Spam filtering : an active learning approach using incremental clustering categories and subject descriptors. *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics*, v. 1, n. 23, Junho 2014.

GOODMAN, J. Ip addresses in email clients. *Conference on Email and Anti-Spam*, v. 1, n. 1, Julho 2004.

HARKER, R. Selectively rejecting spam using sendmail. *Proceedings of the Eleventh Systems Administration Conference*, v. 1, n. 1, Outubro 1997.

INTERNET ENGINEERING TASK FORCE. *RFC 5321*. [S.l.], 2008.

INTERNET ENGINEERING TASK FORCE. *RFC 5322*. [S.l.], 2008.

KASPERSKY Lab Spam Study. 2014. Disponível em: <`http://www.kaspersky.com/about/news/spam`>.

KRISHNAMURTHY, B.; BLACKMOND, E. Shred : Spam harassment reduction via economic disincentives background. *Working Paper: http://www.research.att.com/ bala/papers/shred-ext.pdf*, v. 1, n. 1, Março 2010.

MARSONO, M. W. E.-K. M. N.; GEBALI, F. A spam rejection scheme during smtp sessions based on layer-3 e-mail classification. *Journal of Network and Computer Applications*, v. 32, n. 1, Novembro 2009.

MARSONO, M. W. E.-K. M. N.; GEBALI, F. Targeting spam control on middleboxes: Spam detection based on layer-3 e-mail content classification. *Computer Networks*, v. 53, n. 6, Março 2009.

PFLEEGER, S. L.; BLOOM, G. Canning spam : Proposed solutions to unwanted email. *IEEE Security and Privacy*, v. 1, n. 1, Abril 2005.

POSTFIX. *Postfix Architecture Overview*. [S.l.], 2010.

POSTFIX org. 2014. Disponível em: <http://www.postfix.org/>.

RAMACHANDRAN, A.; FEAMSTER, N. Understanding the network-level behavior of spammers. *Iconference on Applications, technologies, architectures, and protocols for computer communications*, v. 36, n. 4, Outubro 2006.

SCHWARTZ PETER ZAITSEV, V. T.-J. D. Z. A. L. J. B. *High Performance Mysql*. Sebastopol, CA: O Reilly, 2012.

TEMPLETOM, B. *Reflections on the 25th Anniversary of Spam*. 2014. Disponível em: <http://www.templetons.com/brad/spam/spam25.html>.

TEXAS Instruments. 2014. Disponível em: <http://tinyurl.com/oslvqco>.

TRAN, M.; ARMITAGE, G. Evaluating the use of spam-triggered tcp / ip rate control to protect smtp servers. *Australian TelecommunicationsNetworks and Applications Conference*, v. 1, n. 1, Dezembro 2004.

VLECK, T. The history of electronic mail. *IEEE Annals of the History of Computing*, v. 34, n. 1, Janeiro 2012.

WANG, C.-C.; CHEN, S.-Y. Using header session messages to anti-spamming. *Computers and Security*, v. 26, n. 5, Dezembro 2007.

ZIMBRA Software. 2014. Disponível em: <http://www.zimbra.com/>.

Anexo A - Requisitos do sistema de arquivos para o Postfix

A fila de e-mail Postfix exige que:

- Ao Mudar o nome de um arquivo em um diretório no mesmo dispositivo não se altere o número de inode do arquivo.
- Que após um arquivo ao ser apagado a instrução fsync () retornar com êxito. Isso deve ocorrer quando um arquivo for renomeado para um diretório no mesmo dispositivo.
- Quando Postfix, em uma máquina virtual , libera um arquivo com fsync (), as informações do arquivo não devem ser armazenadas em cache na memória temporária. Em vez disso, as informações devem ser imediatamente gravadas no disco antes fsync () e retornar à máquina virtual.
- Postfix pode definir o bit de execução em um arquivo de fila.

Além destes requisitos, a entrega do Postfix ao maildir ainda requer:

- Que um arquivo possa ser um hard link entre diretórios diferentes.
- Que um arquivo não se perca, quando o hard link em seu sub diretório é desvinculada do diretório original.
- Que o bit setgid funcione para os arquivos no diretório *command* do Postfix. Isso é necessário para acessar a fila de mensagens com o comando postdrop, e para o acesso protegido no soquetes Unix com os comandos postdrop e postqueue.

Obs.: Sistemas de arquivos modernos como, EXT3 e EXT4 atendem a totalidade dessas exigências, infelizmente o mesmo não ocorre com o sistema de arquivo NTFS.

Apêndices

APÊNDICE A – Mudanças no arquivo smtpd_check.c

smtpd_check.c

```
#include <sys_defs.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <string.h>
#include <ctype.h>
#include <stdarg.h>
#include <netdb.h>
#include <setjmp.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>

/* Inicio dos includes adicionados */

#include <mysql/mysql.h>
#include <stdio.h>
#include <string.h>

/* Fim dos includes adicionados */
.
.
.
/* Inicio da função que checa se o endereço do remetente consta na tabela
   de spam  */
```

```
25
static  int check_spam (char *acount_id,char *endSpam)
{
    //monta a seguinte srting de consulta sql --> "select endereco from
        Spam
    //                                          where account_id = '"
        destinatario"' and endSpam = '"remetente "'"
30  char sql[] = "select endereco from Spam where account_id = '";
    strcat(sql,acount_id);
    strcat(sql,"' and endSpam = '");
    strcat(sql,endSpam);
    strcat(sql,"'");
35
    //conecta ao banco
    MYSQL DBCon; // Vairiavel de conexão
    MYSQL_RES *result; //variável que recebe o resultado
    MYSQL_ROW dados; //variável que recebe os dados
40
    mysql_init(&DBCon);//conexão propriamente dita

    //Conecta com o banco de dados
    mysql_real_connect(&DBCon, "127.0.0.1", "root", "magika", "zimbra", 0,
        NULL, 0);
45
   // Executa a consulta
    mysql_query(&DBCon,sql);

   // Recebe os dados da cunsulta
50 result = mysql_store_result(&DBCon);

   // Se achou no BD
   if (result)
   {
55     // Limpa da memória
       mysql_free_result(result);
       // Fecha a conexão
       mysql_close(&DBCon);
       return 1; //retorna 1 para tratar como SPAM
60


   }
   // Se não achou no BD
   // Limpa da memória
```

```
mysql_free_result(result);
// Fecha a conexão
mysql_close(&DBCon);
return 0; //Retorna 0 RAM
}
/* Fim da função que checa se o endereço do remetente consta na tabela de
    spam*/
.
.
.
static int reject_unknown_address(SMTPD_STATE *state, const char *addr,
                                  const char *reply_name, const char *reply_class
                                      )
{
    const char *myname = "reject_unknown_address";
    const RESOLVE_REPLY *reply;
    const char *domain;
    const char *account_to_check; //adicionado

   account_to_check=state->sender;//adicionado

    if (msg_verbose)
        msg_info("%s: %s", myname, addr);

    /*
     * Resolve the address.
     */
    reply = smtpd_resolve_addr(addr);
    if (reply->flags & RESOLVE_FLAG_FAIL)
        reject_dict_retry(state, addr);

    /*
     * Skip local destinations and non-DNS forms.
     */
    if ((domain = strrchr(CONST_STR(reply->recipient), '@')) == 0)
        return (SMTPD_CHECK_DUNNO);
    domain += 1;
    if (reply->flags & RESOLVE_CLASS_FINAL)
        return (SMTPD_CHECK_DUNNO);
    if (domain[0] == '[' && domain[strlen(domain) - 1] == ']')
        return (SMTPD_CHECK_DUNNO);
    // Linhas  abaixo adicionadas
    if (check_spam(account_to_check, addr) == 1)
```

```
        return (SMTPD_CHECK_DUNNO);

    /*
     * Look up the name in the DNS.
110  */
    return (reject_unknown_mailhost(state, domain, reply_name, reply_class)
        );
```

APÊNDICE B – Scripts de manipulação da base da dados

consulta.sh

```
#!/bin/bash
su - zimbra -c mysql -u zimbra -D zimbra -e 'SELECT * FROM Spam'
```

insere.sh

```
#!/bin/bash
IFS=:
  while read account_id endSpam
do
        echo
        echo "$account_id"
        echo "$endSpam"
su - zimbra -c "mysql -u zimbra -D zimbra -e 'INSERT INTO Spam (
    account_id, endSpam) Values (\"$account_id\",\"$endSpam\")'"
        [ "$?" = "0" ] && echo "Operacao OK" || echo "Operacao: ERRO"
done < $1
```

apaga.sh

```
#!/bin/bash
su - zimbra -c mysql -u zimbra -D zimbra -e 'DELETE FROM Spam'
```

APÊNDICE C – Scripts de gerenciamento do Zimbra

limpamailbox.sh

```
#!/bin/bash

mailbox=$(su - zimbra -c "zmaccts | grep -v never | grep @ | cut -c 1-40")

for box in $mailbox
do
        su - zimbra -c "zmmailbox -z -m $box emptyFolder /Inbox"
done
```

criausuario.sh

```
#!/bin/bash

while read account_id
do
        echo "$account_id"
        #zmprov ca $account_id@dominio.com.br karma#guria displayName
            $account_id

done < users
```

APÊNDICE D – Script de monitoramento

monitor.sh

```
#!/bin/bash
while true      #Pra rodar via cron, comente esta linha
do                      #Pra rodar via cron, comente esta linha
  vivo="$(uptime)"
  data="$(date)"
  carga="$($_CMD uptime |awk -F'average:' '{ print $2}')"

  tprocessos="$($_CMD ps axue | grep -vE "^USER|grep|ps" | wc -l)"

  ramusada="$($_CMD free -mto | grep Mem: | awk '{ print $3 " MB" }')"
  ramlivre="$($_CMD free -mto | grep Mem: | awk '{ print $4 " MB" }')"
  ramtotal="$($_CMD free -mto | grep Mem: | awk '{ print $2 " MB" }')"



  ping -c1  200.131.128.1 > /dev/null
  if [ "$?" != "0" ] ; then
    rede="Falhou"
  else
    rede="Ok"
    rx1=$(cat /sys/class/net/eth0/statistics/rx_packets)
    tx1=$(cat /sys/class/net/eth0/statistics/tx_packets)
    sleep 1
    rx2=$(cat /sys/class/net/eth0/statistics/rx_packets)
    tx2=$(cat /sys/class/net/eth0/statistics/tx_packets)
    rx=$(expr $rx2 - $rx1)
    tx=$(expr $tx2 - $tx1)
    echo "" >> /root/logs/logmail.log
    echo "[inicio]" >> /root/logs/logmail.log
    echo "Data/hora: $data" >> /root/logs/logmail.log
    echo "Conexao com a rede: $rede" >> /root/logs/logmail.log
    echo "TX $tx pkts/s RX $rx pkts/s" >> /root/logs/logmail.log
    echo "Uptime: $vivo" >> /root/logs/logmail.log
```

```
        echo "Carga media: $carga" >> /root/logs/logmail.log
        echo "Numero total de processos: $tprocessos" >> /root/logs/logmail.log
35      echo "RAM Usada: $ramusada, RAM Livre: $ramlivre, RAM Total: $ramtotal"
            >> /root/logs/logmail.log
        echo "--Filas--" >> /root/logs/logmail.log
        qdir=$(/opt/zimbra/postfix/sbin/postconf -h queue_directory)

        incoming=$(/usr/bin/find $qdir/incoming -type f -print | wc -l | awk '{
            print $1}')
40      echo "incoming:$incoming" >> /root/logs/logmail.log

        maildrop=$(/usr/bin/find $qdir/maildrop -type f -print | wc -l | awk '{
            print $1}')
        echo "maildrop:$maildrop" >> /root/logs/logmail.log

45      active=$(/usr/bin/find $qdir/active -type f -print | wc -l | awk '{
            print $1}')
        echo "active:$active" >> /root/logs/logmail.log

        deferred=$(/usr/bin/find $qdir/deferred -type f -print | wc -l | awk '{
            print $1}')
        echo "deferred:$deferred" >> /root/logs/logmail.log
50
        hold=$(/usr/bin/find $qdir/hold -type f -print | wc -l | awk '{print $1
            }')
        echo "hold:$hold" >> /root/logs/logmail.log

        corrupt=$(/usr/bin/find $qdir/corrupt -type f -print | wc -l | awk '{
            print $1}')
55      echo "corrupt:$corrupt" >> /root/logs/logmail.log

        echo "[fim]" >> /root/logs/logmail.log
        echo "" >> /root/logs/logmail.log
    fi
60  done #Pra rodar via cron, comente esta linha
```

APÊNDICE E – Scripts de envios e arquivos auxiliares

montaenvio.sh

```
#!/bin/bash

#-------------------------------------------------------------------#
#$1=primeiro parametro passado no script=aquivo de usuario          #
#$2=segundo parametro passado no script=quantidade de usuario       #
#$3=terceiro parametro passado no script=aquivo de remetente        #
#$4=quarto parametro passado no script=quantidade de remetente      #
#$5=quinto parametro passado no script=quantidade de emails         #
#-------------------------------------------------------------------#

cont3=0
echo > $5-$4-$2.tmp
while true
do

        while read user
        do
                while read sender
                do
                        echo $user $sender >> $5-$4-$2.tmp
                        cont1=$(expr $cont1 + 1)
                        cont3=$(expr $cont3 + 1)

                        if [ $4 == $cont1 ]
                        then
                                cont1=0
                                break
                        fi

                done < $3
```

```
        cont2=$(expr $cont2 + 1)
        cont3=$(expr $cont3 + $cont2)
        if [ $2 == $cont2 ]
        then
                cont2=0
                break
        fi

        done < $1

if [ $(cat $5-$4-$2.tmp | wc -l) -gt $5 ]
        then
                break
        fi
done
more $5-$4-$2.tmp | tr ' ' ':' | head -n $5 > $5-$4-$2.rem
rm -f *.tmp
```

enviaemail.sh

```
#!/bin/bash

host=`hostname -s`
smtp=127.0.0.1
data=`LC_ALL=C date "+%a, %d %b %Y %k:%M:%S %z"`

count=0
echo "" > /root/logs/logmail.log
ssh root@200.131.128.221 'echo "" > /root/logs/logmail.log'
ssh root@200.131.128.222 'echo "" > /root/logs/logmail.log'

enviaMail() {
[ -z $remetente ] && echo nao existe remetente && return 1
[ -z $destinatario ] && echo nao existe destinatario && return 1
echo '#!/bin/bash' > $email
echo -n "(" >> $email
echo "echo HELO $host; sleep 1" >> $email
echo "echo MAIL FROM: $remetente; sleep 1" >> $email
echo "echo RCPT TO: $destinatario; sleep 1" >> $email
[ ! -z "$copia" ] && echo "echo RCPT TO: $copia; sleep 1" >> $email
echo "echo DATA; sleep 1" >> $email
echo "echo User-Agent: Shell Script via Telnet" >> $email
```

```
echo "echo From: $remetente" >> $email
if [ -z "$assunto" ]; then
     echo "echo Subject: \"sem assunto\"" >> $email
else
     echo "echo Subject: $assunto" >> $email
fi
echo "echo Date: $data" >> $email
echo "echo To: $destinatario" >> $email
echo "echo \"\"" >> $email
echo "echo \".\"; sleep 1" >> $email
echo "echo \"\"" >> $email
echo "echo QUIT" >> $email
echo ") | telnet $smtp 25" >> $email
chmod +x $email
./$email >>~/logs/envio$vol.log 2>>~/logs/erros$vol.log
rm -f $email
}
for arquivo in $(cat catalogo)
do
   vol=$arquivo
        for linha in $(cat $arquivo)
        do
                destinatario=$(echo $linha | cut -d: -f1)@emailchange.
                    unifei.edu.br
                copia=$(echo $linha | cut -d: -f1)@emailnochange.unifei.edu
                    .br
                remetente=$(echo $linha | cut -d: -f2)
                count=$(($count+1))
                email=email$count.sh
                assunto=teste$count
                enviaMail
        done
   cp /root/logs/logmail.log /root/logs/logmail$vol.log
   ssh root@200.131.128.222 cp /root/logs/logmail.log /root/logs/
       logmail$vol.log
   ssh root@200.131.128.221 cp /root/logs/logmail.log /root/logs/
       logmail$vol.log
   echo "" > /root/logs/logmail.log
   ssh root@200.131.128.221 'echo "" > /root/logs/logmail.log'
   ssh root@200.131.128.222 'echo "" > /root/logs/logmail.log'
done
```

Exemplo de script gerado pelo script enviaemail.sh

```
#!/bin/bash
(echo HELO localhost; sleep 1
echo MAIL FROM: blueoceancqbu@yahoo.se; sleep 1
echo RCPT TO: user1@emailchange.unifei.edu.br; sleep 1
echo RCPT TO: user1@emailnochange.unifei.edu.br; sleep 1
echo DATA; sleep 1
echo User-Agent: Shell Script via Telnet
echo From: blueoceancqbu@yahoo.se
echo Subject: teste68202
echo Date: Fri, 19 Sep 2014 14:23:33 -0300
echo To: user1@emailchange.unifei.edu.br
echo ""
echo "."; sleep 1
echo ""
echo QUIT
) | telnet 127.0.0.1 25
```

Exemplo do arquivo catalogo

$100 - 30 - 1.rem$

$1000 - 30 - 1.rem$

$10000 - 30 - 1.rem$

$100000 - 30 - 1.rem$

Exemplo do arquivo de envio

$user1\ a\_atkinsau@optonline.net$

$user1 : mae_nongjon@yahoo.com$

$user1 : aida_frye_ik@moen.com$

$user1 : dextersolas@hotmail.com$

$user1 : AlbertoWhalen@0451.com$

$user2 : a_atkinsau@optonline.net$

$user2 : mae_nongjon@yahoo.com$

$user2 : aida_frye_ik@moen.com$

$user2 : dextersolas@hotmail.com$

$user2 : AlbertoWhalen@0451.com$

APÊNDICE F – Scripts de tratamento de log e estatísticas

tratalog.sh

```
#!/bin/bash

cp mailbox.log mailbox.log.bkp
for arq in *.gz
do
        gunzip $arq
done

listalog=$(ls -t mailbox.log.20* )
more $listalog | grep "Adding Message" >> mailbox.log

head -n 100 mailbox.log > mailbox.100-$1.log
head -n 1100 mailbox.log | tail -n 1000 > mailbox.1000-$1.log
head -n 11100 mailbox.log | tail -n 10000 > mailbox.10000-$1.log
head -n 111100 mailbox.log | tail -n 100000 > mailbox.100000-$1.log
```

geraestatistica.sh

```
#!/bin/bash

for arq in *.log
do
cat $arq | grep TX | cut -d " " -f2 > redetx.tmp
cat $arq | grep RX | cut -d " " -f5 > rederx.tmp
cat $arq | grep "Carga media" | cut -d " " -f5  | tr -d "," | cut -d . -f2
   > cargamedia.tmp
cat $arq | grep "processos" | cut -d " " -f5 > processos.tmp
cat $arq | grep "RAM" | cut -d " " -f3 > ram.tmp
cat $arq | grep "incoming" | cut -d : -f2 > incoming.tmp
cat $arq | grep "maildrop" | cut -d : -f2 > maildrop.tmp
```

```
cat $arq | grep "active" | cut -d : -f2 > active.tmp
cat $arq | grep "deferred" | cut -d : -f2 > deferred.tmp
cat $arq | grep "hold" | cut -d : -f2 > hold.tmp
cat $arq | grep "corrupt" | cut -d : -f2 > corrupt.tmp

echo -e "tx\trx\tc.media\tproc\tram\tinc.\tmaild\tact\tdef\thold\tcorrupt"
    > total-$arq.txt
echo -e "tx,rx,c.media,proc,ram,inc,maild,act,def,hold,corrupt" > total-
    $arq.csv

more total-$arq.txt > media-$arq.txt
more total-$arq.csv > media-$arq.csv

paste redetx.tmp rederx.tmp cargamedia.tmp processos.tmp ram.tmp incoming.
    tmp maildrop.tmp active.tmp deferred.tmp hold.tmp corrupt.tmp >> total-
    $arq.txt
paste -d , redetx.tmp rederx.tmp cargamedia.tmp processos.tmp ram.tmp
    incoming.tmp maildrop.tmp active.tmp deferred.tmp hold.tmp corrupt.tmp
    >> total-$arq.csv

for tmp in *.tmp
do
   soma=0
   media=0
   tam=$(more $tmp | wc -l)

   while read linha
   do
        soma=$( echo "$soma + $linha" | bc )
   done < $tmp

   media=$( echo "$soma / $tam" | bc )
   echo $media > $tmp.med
done

paste redetx.tmp.med rederx.tmp.med cargamedia.tmp.med processos.tmp.med
    ram.tmp.med incoming.tmp.med maildrop.tmp.med active.tmp.med deferred.
    tmp.med hold.tmp.med corrupt.tmp.med >> media-$arq.txt
paste -d , redetx.tmp.med rederx.tmp.med cargamedia.tmp.med processos.tmp.
    med ram.tmp.med incoming.tmp.med maildrop.tmp.med active.tmp.med
    deferred.tmp.med hold.tmp.med corrupt.tmp.med >> media-$arq.csv

rm -f *.med
```

```
rm -f *.tmp

done

#echo "----------TOTAL----------"
#more total-$arq.txt
#echo "----------MEDIA----------"
#more media-$arq.txt
#echo ""
#echo "----------CSV----------"
#echo "----------TOTAl CSV----------"
#more total-$arq.csv
#echo "----------MEDIA CSV----------"
#more media-$arq.csv
#echo ""
#ls -l *$arq.*
```

verificaatraso.sh

```
#!/bin/bash

cat $1 | grep user | cut -d" " -f2 | cut -d: -f3 > f1.tmp
cat $2 | grep user | cut -d" " -f2 | cut -d: -f3 > f2.tmp
paste f1.tmp f2.tmp > f.tmp
n1=$(echo $1 | cut -dx -f2 | cut -d. -f1)
n2=$(echo $2 | cut -dx -f2 | cut -d. -f1)
n=$n1$n2
echo nc,c,dif > diffila$n.csv
while read linha
do

    valor1=$(echo $linha | cut -d" " -f1 | tr ',' '.')
    valor2=$(echo $linha | cut -d" " -f2 | tr ',' '.')
    atrazo=$(echo "$valor1 - $valor2" | bc)
    echo $valor1,$valor2,$atrazo >> diffila$n.csv
    echo $atrazo >> atrazo.tmp
done < f.tmp

soma=0
media=0
tam=$(more atrazo.tmp | wc -l)

while read linha
```

```
25 do
       soma=$( echo "$soma + $linha" | bc )
   done < atrazo.tmp

   media=$( echo "$soma / $tam" | bc )
30 echo $media > med$n.med

   rm -f *.tmp
```