

REDES DE COMPUTADORES: IMPLEMENTAÇÃO E CONFIGURAÇÃO DE UM SERVIDOR ANTISPAM

COMPUTER NETWORKS: IMPLEMENTING AND CONFIGURING AN ANTI-SPAM SERVER

João Paulo Ribeiro Bastos, Matheus Soares dos Santos, Marco Antonio Piloto

BASTOS, João Paulo Ribeiro; SANTOS, Matheus Soares dos; PILOTO, Marco Antonio. Redes de Computadores: implementação e configuração de um servidor antispam. Revista Tecnológica da FATEC-PR, Edição Especial, p. 46-29, jan/dez, 2019.

RESUMO

Este artigo apresenta importância da implementação de um servidor AntiSpam, ou seja, iremos mostrar desde a instalação da distribuição Linux (Debian), instalação dos pacotes que fazem a função do AntiSpam e sua configuração. Redes de Computadores estão em diversos contextos: na universidade, no trabalho, no lazer, na segurança, entre outros meios de vivência, permitindo que diversos serviços e recursos possam ser compartilhados, otimizando a comunicação e permitindo maior interação entre os usuários. Mas, para projetar e desenvolver Sistemas de Informação são necessários conhecimentos sobre a performance das Redes de Computadores e a sua relação com os Sistemas. Neste contexto, este livro busca proporcionar para o estudante uma visão de como as Redes funcionam, como elas auxiliam as Organizações a se inovarem cada vez mais, de forma que o estudante possa compreender e utilizar os termos técnicos das Redes de Computadores, enriquecendo sua cultura profissional para cooperar em equipes multidisciplinares, por meio do conhecimento das principais tecnologias, protocolos e fundamentos de Redes de Computadores. As Redes de Computadores surgiram no período da década 1970, e logo na seguinte dois outros importantes modelos de protocolos de interconexão de Redes apareceram: o Modelo OSI e o Modelo TCP/IP. Conhecer e compreender os principais fundamentos desses protocolos de comunicação, oportuniza aos estudantes a construção de uma base de conhecimentos sólida para avançar na compreensão e resolução de problemas na área.

Palavras chave: Redes de Computadores. Anti Spam. Desenvolvimento e Modelos.

ABSTRACT

This article presents the importance of implementing an AntiSpam server, that is, we will show you from the installation of the Linux distribution (Debian), installation of the packages that make AntiSpam function and its configuration. Computer networks are in different contexts: at the university, at work, at leisure, in security, among other means of living, allowing different services and resources to be shared, optimizing communication and allowing greater interaction between users. However, in order to design and develop Information Systems, knowledge about the performance of Computer Networks and their relationship with Systems is required. In this context, this book seeks to provide students with a view of how networks work, how they help organizations to innovate more and more, so that students can understand and use the technical terms of computer networks, enriching their culture. professional to cooperate in multidisciplinary teams, through knowledge of the main technologies, protocols and

fundamentals of Computer Networks. Computer Networks appeared in the 1970s, and in the following two other important models of Network interconnection protocols appeared: the OSI Model and the TCP / IP Model. Knowing and understanding the main fundamentals of these communication protocols, gives students the opportunity to build a solid knowledge base to advance in the understanding and resolution of problems in the area.

Keywords: Computer Networks. Anti-Spam. Development and Models.

1 INTRODUÇÃO

Com o advento da internet e a popularização das ferramentas diversos meios foram atualizados dentre eles o tão conhecido e utilizado meio de comunicação evoluiu para e-mails, diversas empresas começaram a oferecer serviços por e-mails e dentre estas empresas algumas pessoas mais ousadas começaram a utilizar este meio para envio de propagandas falsas com o fim de enganar outras pessoas.

Este processo de e-mails em massa afim de enganar pessoas foi denominado de SPAM. Nos anos 70 o então famoso grupo de humor britânico *Monty Python* conhecido por seus esquetes apresentou o SPAM foi tema central da cena.

Inspirados pela cena então os primeiros programadores dos anos 80 chamaram o ato de e-mails em massa de SPAM. Ao passar dos anos diversos filtros de SPAM foram criados e disto se tratará este projeto.

Devido à grande necessidade de interoperabilidade entre produtos e serviços heterogêneos, o estudo e a compreensão dos Modelos de Referência OSI e TCP/IP são de essencial importância para os estudantes de Tecnologia da Informação (TI). Isto porque eles fornecem diretrizes gerais para projeção, implementação e manutenção dos protocolos de Rede de Computadores.

A Internet é um grande conjunto de Redes de Computadores interligados pelo mundo de forma integrada, viabilizando a conectividade independente do tipo de máquina que seja utilizada. Para manter essa multicompatibilidade da Rede, é utilizado um conjunto de protocolos e de serviços em comum, permitindo que os usuários a ela conectados usufruam de serviços de informação em alcance mundial.

1.1 OBJETIVO GERAL

Apresentaremos neste conteúdo a instalação configuração de um servidor AntiSpam. Apesar da utilização crescente de outras formas de comunicação como redes sociais e aplicativos de mensagens instantâneas, e-mails ainda são bastante usados, principalmente nos meios empresariais, para comunicação interna. Segundo um relatório de 2010, nove dentre dez e-mails recebidos são considerados lixo eletrônico (spam).

1.2 OBJETIVOS ESPECÍFICOS

- a) Construção de Redes para controle dos documentos;
- b) Adaptação e Configuração de Redes;
- c) Aplicar Sistema Operacional nas Rotinas;
- d) Criação de Modelos para AntiSpam.

2 JUSTIFICATIVA

Como abordado na introdução sem um sistema de filtro de e-mails uma empresa está correndo risco de sofrer com sequestro ou perda de dados. E não só as empresas podem sofrer com este mal as pessoas físicas também podem sofrer com sequestro ou perda de dados por falta de um sistema AntiSpam adequado ao dia a dia.

Do lado do servidor, a máquina precisa ser poderosa o suficiente para atender a todas as requisições de todos os clientes. O problema é que construir uma única máquina capaz de fazer isso é impossível devido aos altos custos. Por isso, a melhor opção é alocar várias máquinas que, juntas, comportam-se como uma só. Esse é o conceito de um parque de servidores. Todos os elementos de um parque de servidores estão alocados em um ou mais espaços físicos denominados Centrais de Processamento de Dados (Data Centers).

Planejamento prévio é uma recomendação válida e pertinente para todo projeto. Afinal, muitos erros decorrem diretamente do fato de essa etapa ter sido negligenciada ou feita inadequadamente. Ela busca garantir, em linhas gerais, que se saiba claramente onde se quer chegar e o que precisará ser executado para esse fim.

No desenvolvimento de softwares, não é diferente. Antes de iniciar-se o processo de codificação, é recomendada a seguinte prática: a criação de um modelo geral que descreva como esse sistema será distribuído em componentes (ou módulos) e suas conexões e relações. Mas essa responsabilidade compete a qual profissional? E ele fará uso de qual principal instrumento de planejamento? Quais são as metodologias mais comuns adotadas?

Esses e outras perguntas serão respondidas ao longo desta disciplina, que também ensinará de que forma é possível executar, com propriedade, esse importante papel em projetos de desenvolvimento de software

3 REVISÃO BIBLIOGRÁFICA

3.1 DESENVOLVIMENTO DOS PROJETOS DE REDES

O desenvolvimento deste projeto parte primeiramente em baixar a imagem do sistema que utilizaremos para criar os filtros AntiSpam, iremos acessar o site do Debian Servidor lá

iremos acessar a parte de versões anteriores a versão que iremos utilizar neste projeto é a versão Servidor Debian 8 'Jessie', nesta opção podemos acessar toda documentação do sistema operacional e baixar sua imagem.

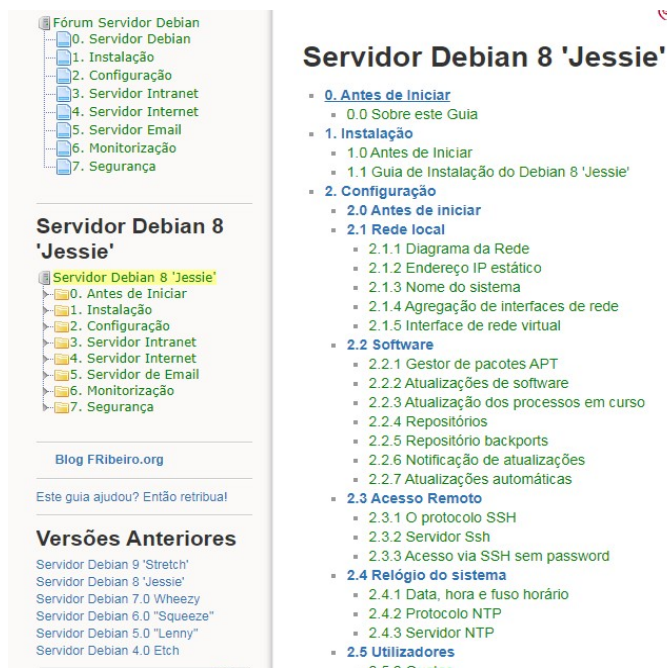


Figura 1: versão de imagem Debian
Fonte: autor, 2018.

Após baixar a imagem iremos a emular no sistema Windows para isto iremos utilizar um programa chamado Oracle VirtualBox iremos baixar do site da ORACLE



Figura 2: download VM Oracle
Fonte: itigic.com

Após baixar e instalar a ferramenta de virtualização precisaremos configurar a ferramenta utilizaremos as seguintes configurações na máquina virtual:

TAMANHO DE MEMORIA RAM: 1024 MB (1GB)

HDD VDI DINAMICAMENTE ALOCADO: 8GB

AUDIO: Windows DirectSound controlador ICH AC97

REDE: INTEL PRO/1000MT DESKTOP (NAT)



Figura 3: Configuração da VM
Fonte: Autor, 2018.

3.2 INSTALAÇÃO DO SISTEMA OPERACIONAL

Assim que a máquina virtual foi criada iremos instalar o sistema operacional para que possamos configurar os filtros AntiSpam, ao iniciar a máquina o programa irá pedir a localização da iso de boot.

No menu de instalação iremos selecionar a opção Install, depois selecionar o idioma que será aplicada no sistema operacional.

Após escolher a linguagem o sistema operacional será instalado, quando terminar a instalação vamos dar ao nome ao equipamento e a opção que não será utilizada aqui e colocar a máquina em um domínio.

Ao terminar a configuração de nome domínio irá começar a configuração do usuário root (administrador) e da conta não administrativa.

Para finalizar o sistema configurar as opções de data e hora e particionar o disco, ao finalizar a instalação a máquina começa a reiniciar e iniciar o sistema.

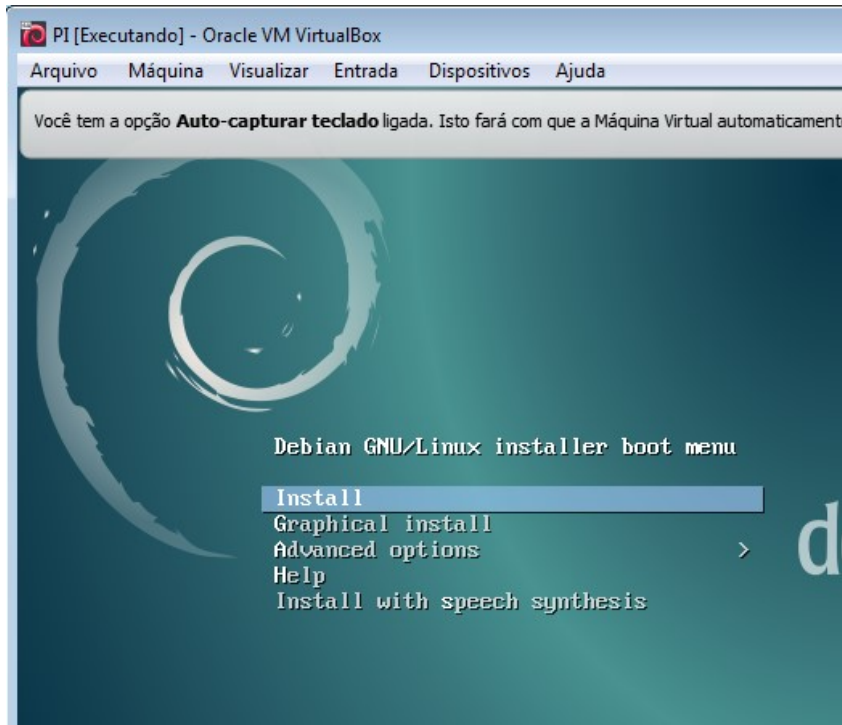


Figura 4: início da instalação
 Fonte: Autor, 2018.

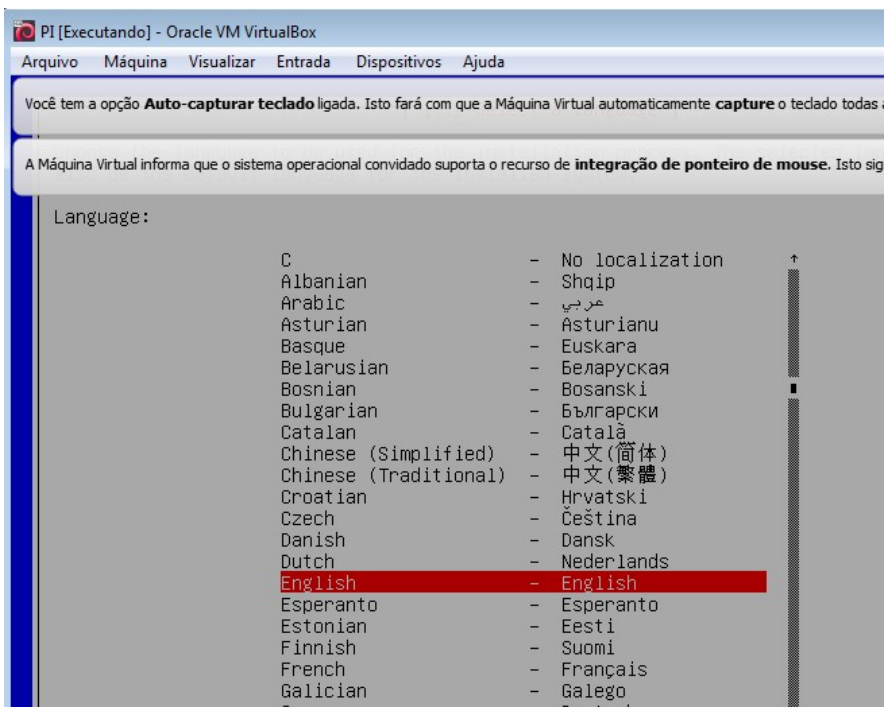


Figura 5: escolha do idioma
 Fonte: Autor, 2018.

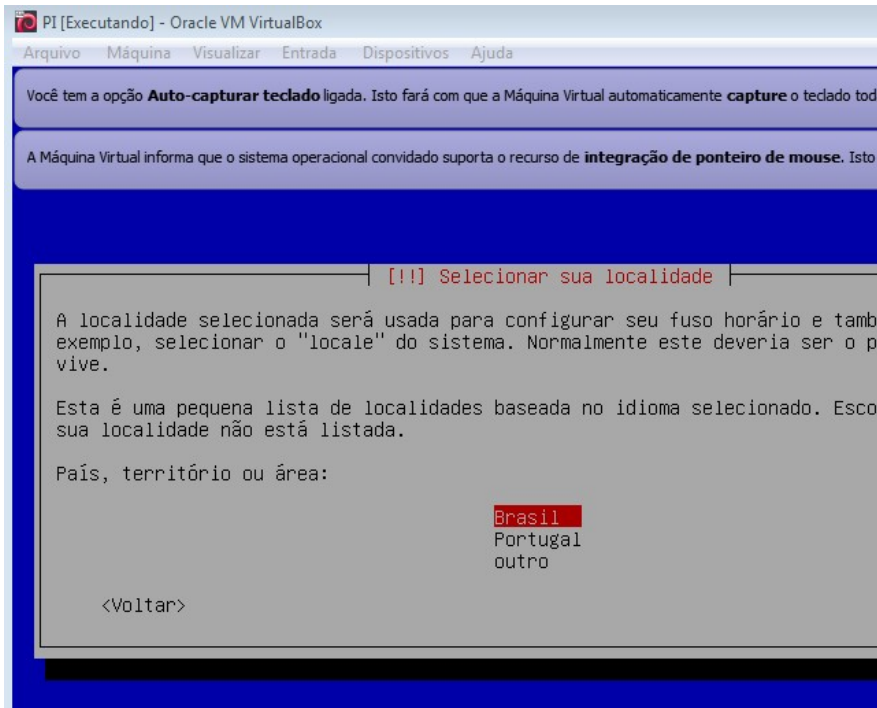


Figura 6: escolha de localidade
 Fonte: Autor, 2018.

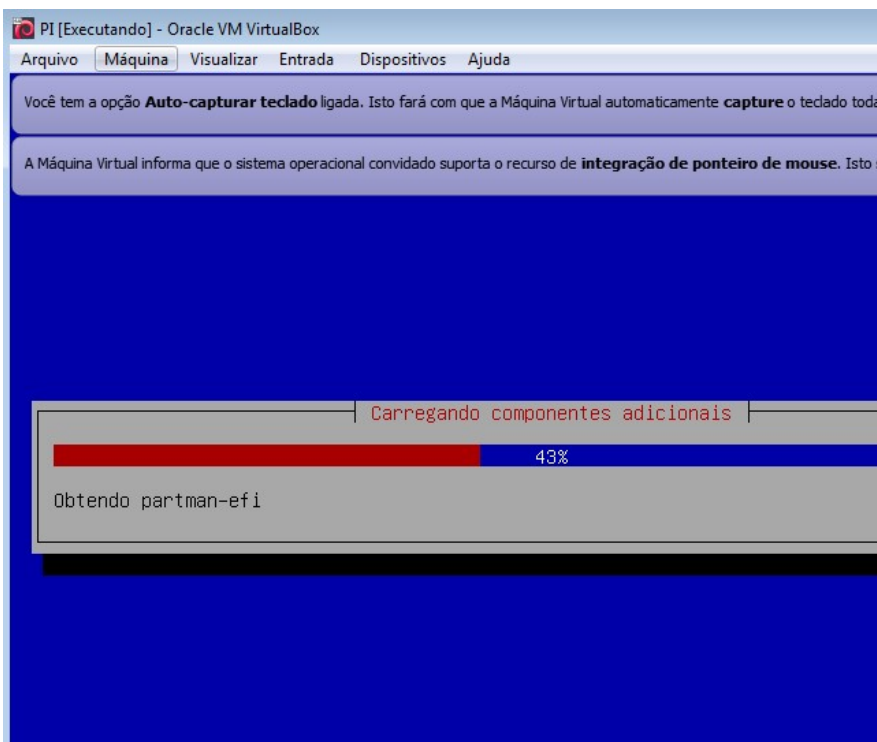


Figura 7: instalação do sistema OP
 Fonte: Autor, 2018.

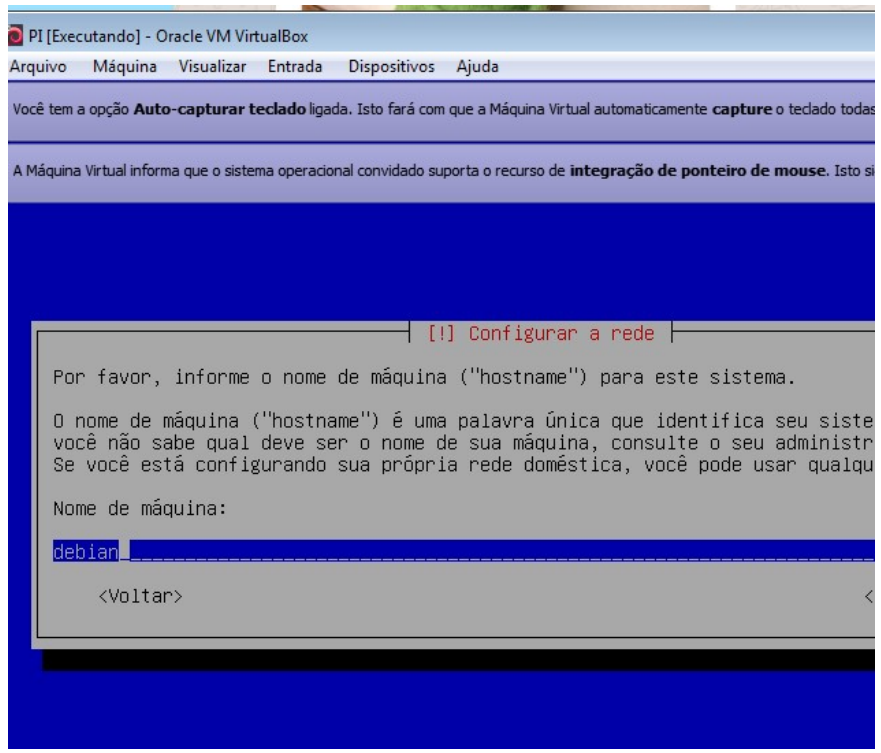


Figura 8: Definir nome do host
 Fonte: Autor, 2018.

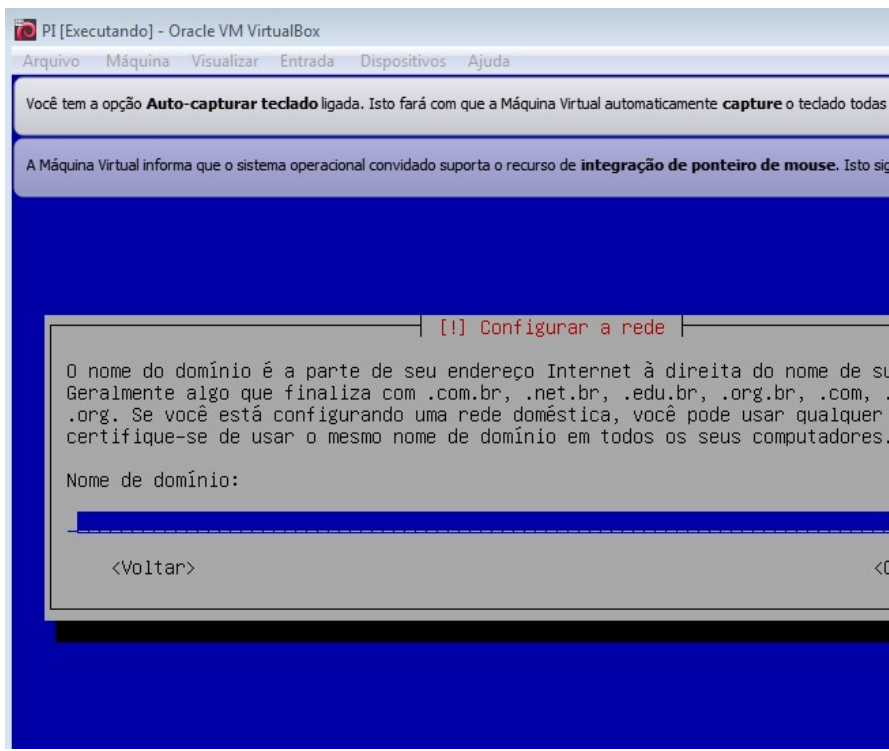


Figura 9: Definir o nome de domínio
 Fonte: Autor, 2018.

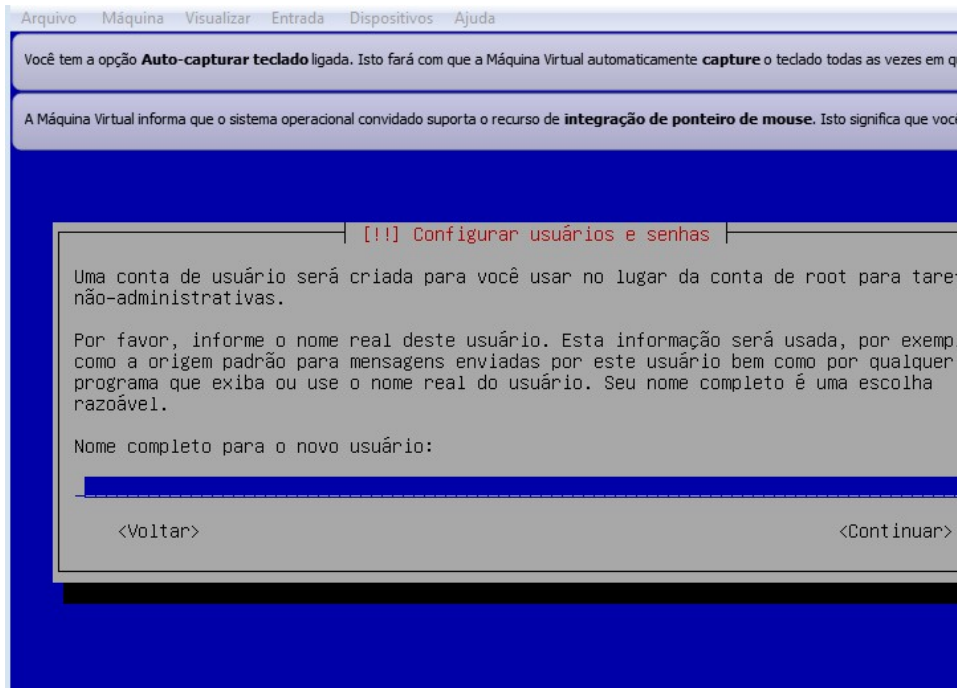


Figura 10: Nome de usuário do sistema
 Fonte: Autor, 2018.

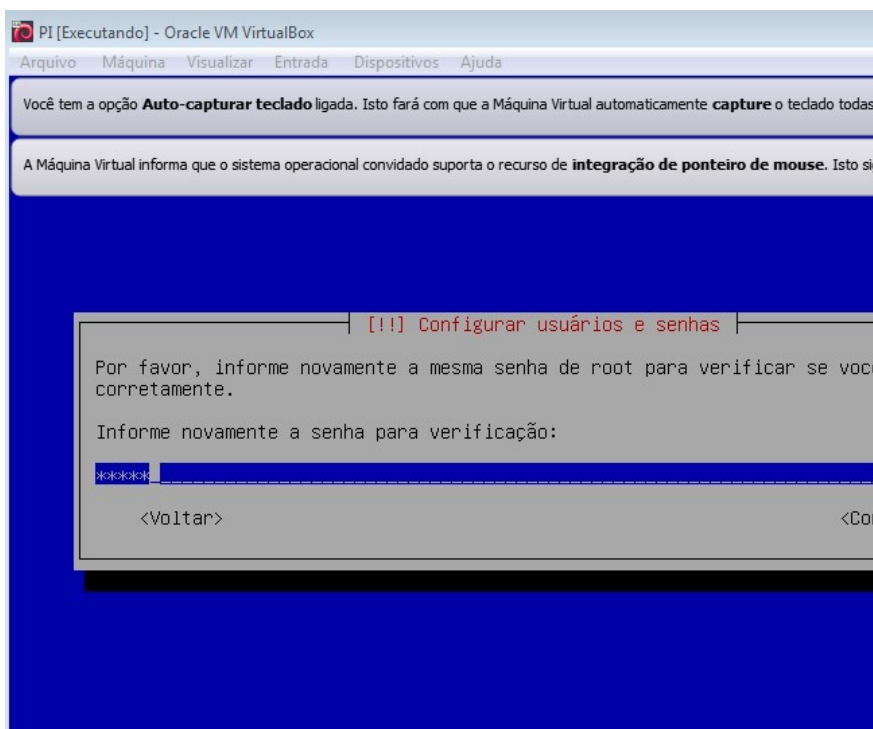


Figura 11: definir senha de root
 Fonte: Autor, 2018.

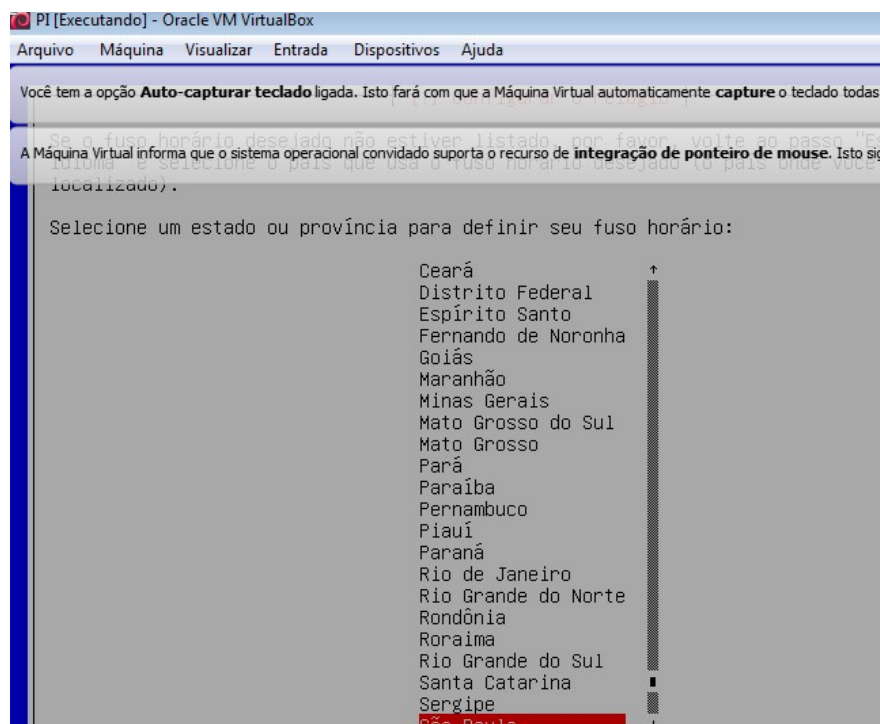


Figura 12: Definições de data e hora
 Fonte: Autor, 2018.

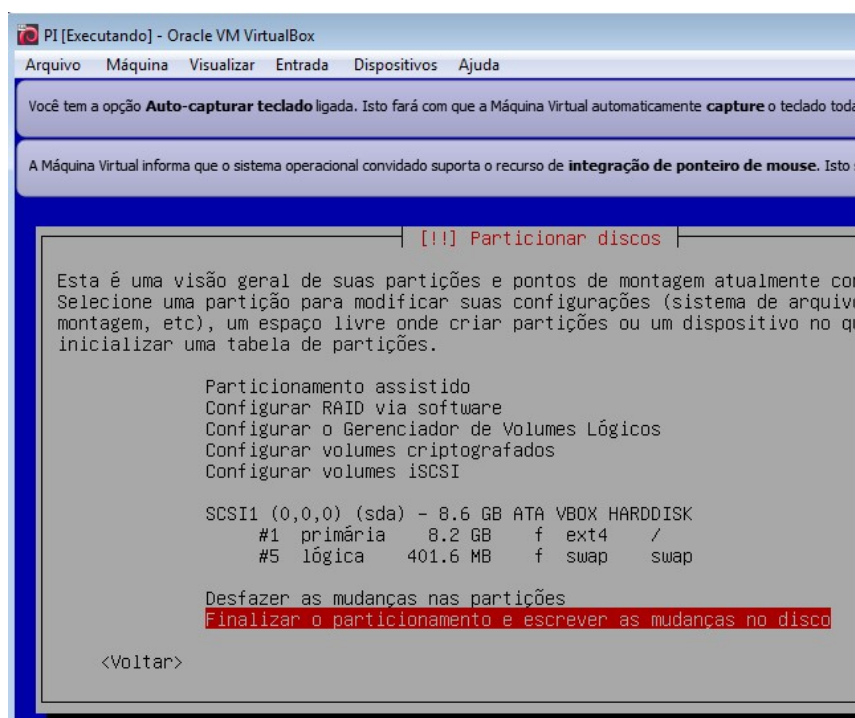


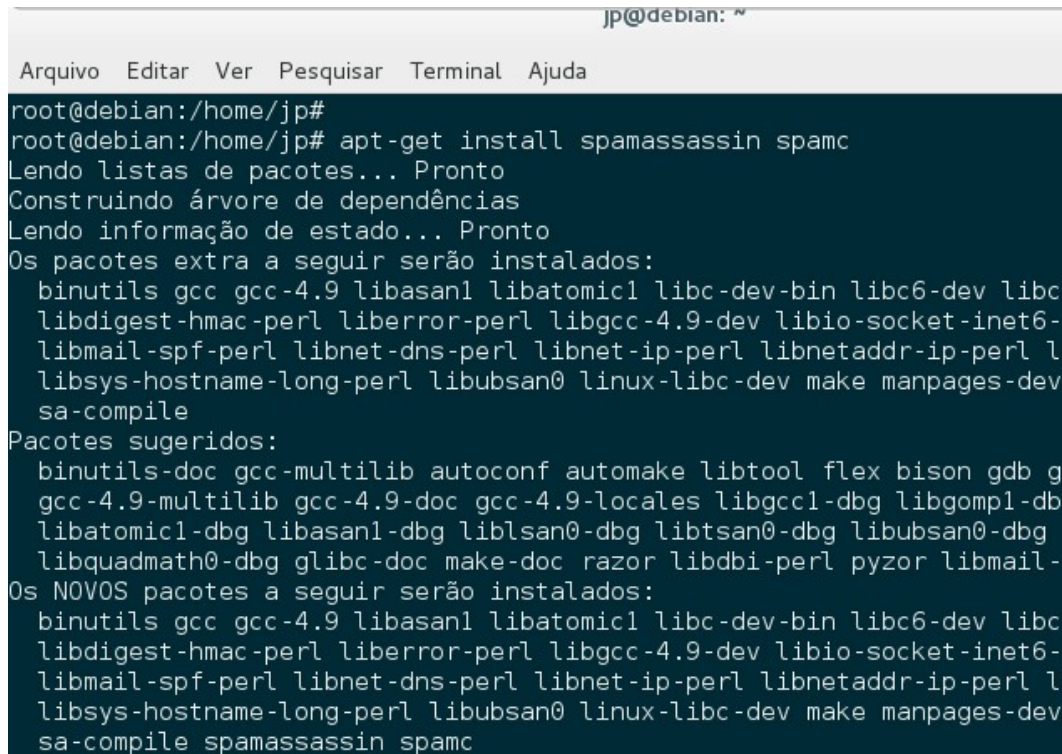
Figura 13: partição de disco
 Fonte: Autor, 2018.

3.3 INSTALANDO A FERRAMENTA DE FILTRO

Utilizaremos a ferramenta *spamassassin* que é um programa *AntiSpamopensource* que permite a criação de filtros e bloquear e-mails não desejáveis.

Para instalar a ferramenta iremos utilizar o comando:

```
apt-get install spamassassin spamc
```



```
jp@debian: ~  
Arquivo Editar Ver Pesquisar Terminal Ajuda  
root@debian:/home/jp#  
root@debian:/home/jp# apt-get install spamassassin spamc  
Lendo listas de pacotes... Pronto  
Construindo árvore de dependências  
Lendo informação de estado... Pronto  
Os pacotes extra a seguir serão instalados:  
binutils gcc gcc-4.9 libasan1 libatomic1 libc-dev-bin libc6-dev libc  
libdigest-hmac-perl liberror-perl libgcc-4.9-dev libio-socket-inet6-  
libmail-spf-perl libnet-dns-perl libnet-ip-perl libnetaddr-ip-perl l  
libsys-hostname-long-perl libubsan0 linux-libc-dev make manpages-dev  
sa-compile  
Pacotes sugeridos:  
binutils-doc gcc-multilib autoconf automake libtool flex bison gdb g  
gcc-4.9-multilib gcc-4.9-doc gcc-4.9-locales libgcc1-dbg libgomp1-db  
libatomic1-dbg libasan1-dbg liblsan0-dbg libtsan0-dbg libubsan0-dbg  
libquadmath0-dbg glibc-doc make-doc razor libdbi-perl pyzor libmail-  
Os NOVOS pacotes a seguir serão instalados:  
binutils gcc gcc-4.9 libasan1 libatomic1 libc-dev-bin libc6-dev libc  
libdigest-hmac-perl liberror-perl libgcc-4.9-dev libio-socket-inet6-  
libmail-spf-perl libnet-dns-perl libnet-ip-perl libnetaddr-ip-perl l  
libsys-hostname-long-perl libubsan0 linux-libc-dev make manpages-dev  
sa-compile spamassassin spamc
```

Figura 14: instalação do filtro de Spam
Fonte: Autor, 2018.

Utilizaremos também o plugin razor pyzor sua instalação será efetuada através do comando

```
apt-get install razor pyzor
```

Após a instalação da ferramenta seu local de instalação fica em

```
/etc/mail/spamassassin
```

3.4 CONFIGURANDO SPAMASSASSIN

Para configurar o spamassin primeiro vamos acessar o documento de configuração que fica localizado dentro da pasta etc vamos abrir o documento com o comando pico

```
Cd /etc/mail/spamassassin/
```

pico local.cf

Dentro do documento de configuração vamos fazer as modificações, podemos usar o símbolo '#' para fazer citações então tudo que vem após *add **SPAM** to the subject* header spam e-mails o spamassassin vai ler o assunto do e-mail e se conter as palavras adicionadas o ele começara a recusar. No nosso servidor configurei as palavras EMAGRECIMENTO e emagrecimento. Abaixo temos a opção de *White* e *blacklist*, tudo que for incluído após a citação *whitelist* irá passar tudo que vem, após *blacklist* será bloqueado (*segundo a biblioteca da ferramenta tudo que é bloqueado ou liberado desta forma *@dominio está sendo aplicado diretamente no domínio ou seja, se for inserido no documento a linha whitelist_from *@gmail.com será liberado tudo que vem do gmail*).

```
# This is the right place to customize your installation of SpamAssassin
#
# See 'perldoc Mail::SpamAssassin::Conf' for details of what can be
# tweaked.
#
# Only a small subset of options are listed below
#
#####

Add *****SPAM***** to the Subject header of spam e-mails

EMAGRECIMENTO
emagrecimento

##### WHITELIST
whitelist_from *@gmail.com

##### BLACKLIST
```

Figura 15: configuração do spamassassin

Fonte: Autor, 2018.

Após a configuração precisamos integrar as duas ferramentas o postfix e o spamassassin.

Um dos erros mais comuns e fonte de diversos problemas é a criação de um distanciamento entre arquitetos de software e engenheiros de software e programadores, assim como ocorre, muitas vezes, na construção civil. Esse fator pode ter origem na coordenação geral de uma empresa ou projeto, mas também pode ter suas raízes nos próprios arquitetos e engenheiros.

Ser responsável pelas definições gerais de um software não significa ser dono da razão nem possuir qualquer outro poder sobrenatural. Ao contrário: exige grande responsabilidade, além de capacidade de ouvir e avaliar diferentes pontos de vista, incluindo o cliente e os engenheiros de software.

Com estes últimos, deve-se formar uma perspectiva de equipe que compartilha um objetivo comum. Por essa razão, as preocupações e questionamentos dos stakeholders

envolvidos devem ser sempre consideradas e avaliadas.

Outro erro frequente diz respeito à criação de elefantes brancos, uma metáfora que se refere à idealização de uma arquitetura desproporcionalmente grande e complexa em relação às reais necessidades do projeto. O resultado certamente será um grande desperdício de recursos, uma vez que seu desenvolvimento terá dificuldades desnecessárias e não agregarão benefícios reais ao cliente. Ademais, há risco de, eventualmente, ser necessário um retorno à fase de concepção, para correções, ou mesmo de haver dificuldades posteriores de manutenção e atualização.

Ao modelar e fazer a definição das funções de cada um dos componentes da Arquitetura de Software, é importante buscar atender às principais características de coesão e acoplamento, uma vez que, dessa observância, tende a emergir um projeto robusto e que não gerará dificuldades e conflitos na fase de desenvolvimento e codificação.

O primeiro conceito, de coesão, guarda sentido, primeiramente, com as funções atribuídas a um dado componente. A ideia por ela expressa é de que essas funções devem ser coesas e relacionadas. É de se imaginar que um módulo de gestão de folha de pagamentos também cuide, eventualmente, dos controles de apontamentos e bancos de horas. Porém seria pouco razoável incluir no mesmo componente o processamento de balanço contábil ou controle de caixa.

Em termos de acoplamento, tem-se a ideia de um encaixe adequado entre os múltiplos componentes, ou seja, eles se completam e possuem uma boa interdependência. Em outras palavras, não extrapolam suas funções nem se sobrepõem uns aos outros.

Somando ambos conceitos, podemos fazer uma analogia com um quebra-cabeças: cada peça tem a forma exata do local onde deve ser encaixada, não sendo maior nem menor, sem sobreposições, ou seja, é o acoplamento perfeito. A imagem de sua face envolve somente elementos daquela parte da imagem total, pixel a pixel, próximos e correlacionados: é a coesão.

A atividade do arquiteto de *software* está presente em todo e qualquer projeto de desenvolvimento de sistemas, independentemente de o cargo específico existir formalmente ou não. Afinal, elaborar o design geral é uma atividade que precisará, de algum modo, ser feita, explícita ou implicitamente, planejada ou resultante, detalhada ou simplificada.

Todavia, em cenários mais complexos ou com um volume grande de projetos concomitantes, a tendência é que se adote a contratação de um profissional especializado, o *Arquiteto de Software*.

Nesse contexto, esse profissional – usualmente o único (ou um dos poucos) que exercem essa função numa organização – ficará responsável exclusivamente pelo desenvolvimento das arquiteturas de software dos projetos. Em outras palavras, todos os sistemas passarão primeiro por suas mãos, cabendo a esse profissional a responsabilidade de

desenvolver modelos adequados e que permitam que a continuidade do desenvolvimento e codificação se deem de modo fluido e eficiente.

4 METODOLOGIA UTILIZADA NO DESENVOLVIMENTO

Este trabalho foi desenvolvido por meio de uma pesquisa bibliográfica e a aplicação de uma teoria na prática, seguindo os passos e como foram desenvolvidos conforme destacados a seguir:

- a) Seleção e o estudo da bibliografia;
- b) Levantamento de ferramentas para apoiar no desenvolvimento profissional dos colaboradores em suas habilidades e competências;
- c) Análise comparativa entre a teoria e a prática utilizada no caso de estudo;
- d) Conclusões e considerações.

5 DESENVOLVIMENTO

Em seu papel, o arquiteto precisa lidar com diversos fatores humanos e sociais dentro das organizações. Primeiramente, há a relação com engenheiros de software, que pode ser delicada em virtude de haver algumas áreas de penumbra, ou seja, de sobreposição entre as funções. Se isso não for bem administrado e amenizado por meio de um bom relacionamento, pode tornar-se uma grande e persistente fonte de conflitos.

Não menos importantes são as etapas seguintes de engenharia e programação, que não podem receber um grande elefante branco, que seria de difícil desenvolvimento, integração, manutenção e atualização. Entregar a essa equipe um design mal elaborado será fonte de conflitos, atrasos e desperdício de recursos.

Além disso, o desenvolvimento da Arquitetura de Software requer do profissional mais do que simplesmente conhecimento técnico e experiência. É preciso que haja uma boa comunicação e capacidade de abstração, uma vez que o bom desempenho de seu trabalho dependerá da adequada captação das necessidades e desejos dos diversos stakeholders envolvidos no projeto, além da habilidade de negociar expectativas por vezes conflitantes.

Assim, é essencial que o arquiteto de *software* conheça seu contexto de atuação. O cargo de arquiteto de *software* é mais frequente em empresas de grande porte, com muitos projetos simultâneos e/ou poucos projetos, porém de alta complexidade. E há uma razão para isso.

Primeiramente, é preciso levar em conta que sua atuação é concentrada nos estágios iniciais de um software, com pouca participação – apesar de existente – no restante do processo de desenvolvimento. Afinal, se esse profissional se envolver permanentemente nas etapas seguintes, talvez acabe fazendo mais sentido seu cargo ser de engenheiro de software.

Assim, se não houver uma demanda significativa, em quantidade ou complexidade, é difícil justificar a presença de um colaborador exclusivo para a função de arquiteto de *software*. O quadro a seguir apresenta, didaticamente, a probabilidade de atuação de um profissional dedicado exclusivamente à função de arquiteto de *software* em uma empresa, com base na complexidade e quantidade de projetos.

Quanto mais complexo for o contexto de desenvolvimento de software, mais importante é o papel do arquiteto de *software*. Isso será ilustrado em um exemplo bastante incomum, visando contemplar melhor a aplicação dos conceitos estudados.

Àqueles profissionais que desejam desenvolver essa função, é necessário não se assustar diante de áreas complexas e desconhecidas como a que ilustraremos a seguir. Ao contrário: essa situação deve ser familiar no sentido de que será o arquiteto de software que irá traduzi-la num modelo bem construído, robusto e compreensível para todos os envolvidos no projeto.

6 CONCLUSÕES E RECOMENDAÇÕES

Concluimos que nos dias de hoje o e-mail é uma ferramenta crucial no dia a dia de uma pessoa jurídica ou física seja para receber informações, contas a pagar.

Esta ferramenta é muito utilizada como auxílio no trabalho ou como meio de inscrição em uma rede social hoje podemos dizer que sem um e-mail você não utiliza a internet.

Sendo assim uma ferramenta de filtragem de spam é crucial, pois com esta ferramenta podemos separar o que é uma notícia falsa de uma notícia verdadeira, podemos separar o que é um e-mail que quer te fazer instalar um arquivo malicioso com fins de roubar suas informações a de um e-mail que vai te levar a uma entrevista de emprego, por fim podemos ressaltar a importância de um AntiSpam de e-mail.

A Arquitetura de Software é um modelo geral que explica como o sistema irá funcionar e suas principais funcionalidades. Assim, precisa representar quais módulos ou componentes farão quais papéis e como se conectam. A definição adequada dessas questões tornará mais fluidas e eficientes as fases posteriores de desenvolvimento e codificação.

Além dessas questões, a seguir serão estudadas as características que devem estar presentes no processo de desenvolvimento e fatores gerais que devem ser considerados na Arquitetura de Software. Esse estudo é importante porque, embora o ato de esboçar uma Arquitetura de Software possa parecer, inicialmente, uma tarefa até simples, a realidade dos projetos costuma ser bastante desafiadora.

REFERENCIAS

BASS, L.; CLEMENTES, P.; KAZMAN, R. **Software Architecture in Practice**. 2. ed. Boston: Addison-Wesley Professional, 2003.

FOWLER, M. **Padrões de Arquitetura de Aplicações Corporativas**. Porto Alegre: Bookman, 2006.

FUNDAÇÃO Oscar Niemeyer. **Museu Oscar Niemeyer**. Disponível em: <www.niemeyer.org.br/obra/pro513>. Acesso em: 17 abr 2019.

KOCH, R. **Os Segredos Para Conseguir Mais Com Menos Nos Negócios e na Vida**. Belo Horizonte: Gutenberg, 2015.

LARMAN, C. **Utilizando UML e Padrões: uma introdução à análise e ao projeto orientado a objetos**. 3. ed. Porto Alegre: Bookman, 2007.

WEISZFLOG, W. **Michaelis Moderno Dicionário da Língua Portuguesa**. 2004. São Paulo: Moderna. Disponível em: <michaelis.uol.com.br>. Acesso em: 17 abr 2019.