Linux:
Administração de Redes

Alexandre Folle de Menezes
Gleydson Mazioli da Silva
Nicolai Langfeldt
Vladimir Vuksan

Tradução Oficial Alfamídia
Alexandre Folle de Menezes

Versão 1.3, agosto/2002.

As marcas registradas utilizadas no decorrer deste livro são usadas unicamente para fins didáticos, sendo estas propriedade de suas respectivas companhias.

A Alfamídia não assume qualquer responsabilidade por erros ou omissões, ou por danos resultantes do uso das informações contidas neste livro.

Os capítulos 3, 5, 6, 7, e 12 são adaptações do texto:

Guia Foca GNU/Linux
Copyleft © 1999-2002 - Gleydson Mazioli da Silva.
Fonte: http://focalinux.cipsga.org.br

O capítulo 4 é uma adaptação do texto:

DHCP mini-HOWTO
Copyright © 1999-2002 – Vladimir Vuksan <vuksan@veus.hr>.
Fonte: http://br.tldp.org/documentos/comofazer/html/ DHCP/DHCP.pt_BR.html

---

O capítulo 8 é uma adaptação do texto:

DNS HOWTO
Copyright © 1999-2002 – Nicolai Langfeldt <janl@math.uio.no>.
Fonte:
http://br.tldp.org/documentos/comofazer/html/DNS -HOWTO/DNS-HOWTO.pt_BR.html

---

O capítulo 9 é uma adaptação do texto:

NFS HOWTO
Copyright © 1999-2002 – Nicolai Langfeldt <janl@math.uio.no>.
Fonte:
http://br.tldp.org/documentos/comofazer/html/nfs.howto/nfs.howto.pt_BR.html

---

O capítulo 13 é uma adaptação do texto:

Linux: Perguntas e Respostas
Copyleft © 1999-2002 - Unicamp.
Fonte: http://ftp.unicamp.br/pub/linuxdocs/conectiva/pr/pr.html

---

Os demais capítulos foram escritos por:

Copyright © 2002 por Alexandre Folle de Menezes <menezes@alfamidia.com.br>

Esta apostila é uma coletânea de textos com licença GNU ou livres encontrados na Internet, conforme referências acima.

Este material foi totalmente montado com fins didáticos, sem objetivo comercial. Foram acrescentados exemplos e exercícios desenvolvidos pela Alfamídia Ltda.

CONTEÚDO

1 REDES DE COMPUTADORES

1.1 O que é uma rede

Rede é a conexão de duas ou mais máquinas com o objetivo de compartilhar recursos entre uma máquina e outra. Os recursos podem ser:

- Compartilhamento do conteúdo de seu disco rígido (ou parte dele) com outros usuários. Os outros usuários poderão acessar o disco como se estivesse instalado na própria máquina. Também chamado de servidor de arquivos.
- Compartilhamento de uma impressora com outros usuários. Os outros usuários poderão enviar seus trabalhos para uma impressora da rede. Também chamado de servidor de impressão.
- Compartilhamento de acesso a Internet. Outros usuários poderão navegar na Internet, pegar seus e-mails, ler notícias, bate-papo no IRC, ICQ através do servidor de acesso Internet. Também chamado de servidor *Proxy*.
- Servidor de Internet/Intranet. Outros usuários poderão navegar nas páginas Internet localizadas em seu computador, pegar e-mails, usar um servidor de IRC para bate-papo na rede, servidor de ICQ, etc.

Com os itens acima funcionando é possível criar permissões de acesso da rede, definindo quem terá ou não permissão para acessar cada compartilhamento ou serviço existente na máquina (www, ftp, irc, icq, etc), e registrando/avisando sobre eventuais tentativas de violar a segurança do sistema, *firewalls*, etc.

Alguns termos muito usados quando se fala sobre redes de computadores:

- ***host*** (máquina): um computador que faz parte da rede;
- ***localhost*** (máquina local): é o computador no qual o usuário está trabalhando;
- ***remote host*** (máquina remota): qualquer outro computador que faça parte da rede. Geralmente o usuário não possui acesso físico à esta máquina;
- **servidor**: host que disponibiliza algum recurso pela rede;
- **cliente**: máquina que utiliza os recursos utilizados pelo servidor;

Entre outras ilimitadas possibilidades que dependem do conhecimento do indivíduo no ambiente GNU/Linux, já que ele permite muita flexibilidade para fazer qualquer coisa funcionar em rede.

A comunicação entre computadores em uma rede é feita através do *Protocolo de Rede*.

1.2 Protocolo de Rede

O protocolo de rede é a linguagem usada para a comunicação entre um computador e outro. Existem vários tipos de protocolos usados para a

comunicação de dados, alguns são projetados para pequenas redes locais (como é o caso do NetBIOS) outros para redes mundiais (como o TCP/IP, que possui características de roteamento).

Dentre os protocolos, o que mais se destaca atualmente é o TCP/IP, devido ao seu projeto, velocidade e capacidade de roteamento.

1.3 Internet

O termo *internet* é usado para definir um conjunto de redes de computadores, interligadas entre si. As redes podem ser homogêneas (mesma arquitetura de máquinas, sistemas operacionais e protocolos) ou heterogêneas (diversas plataformas diferentes interligadas). O objetivo é permitir que os dados sejam trocados de forma transparente, sem que um nó saiba detalhes sobre os demais.

A Internet (com "i" maiúsculo), que é uma grande rede mundial de computadores, é um exemplo extremo do conceito de internet (com "i" minúsculo).

No final dos anos 60, o Departamento de Defesa americano iniciou uma parceria com universidades para pesquisar novas tecnologias de comunicação de dados. Os participantes fizeram a ARPANET, que foi a primeira rede de troca de pacotes da história. O experimento foi um sucesso, e a rede logo se espalhou em instituições militares e universidades por todo país.

Mas os primeiros protocolos da ARPANET eram lentos e sujeitos a travamentos freqüentes da rede. Foi necessária a criação de um novo conjunto de protocolos, e no início da década de 80 a ARPANET se converteu para o TCP/IP.

O governo americano passou a exigir que todas as suas redes usassem o TCP/IP, o que encorajou a adoção do protocolo por vários fabricantes de computadores.

A facilidade de operação do TCP/IP permitiu que a ARPANET crescesse rapidamente, transformando-se no que hoje conhecemos por Internet.

1.4 O Modelo OSI

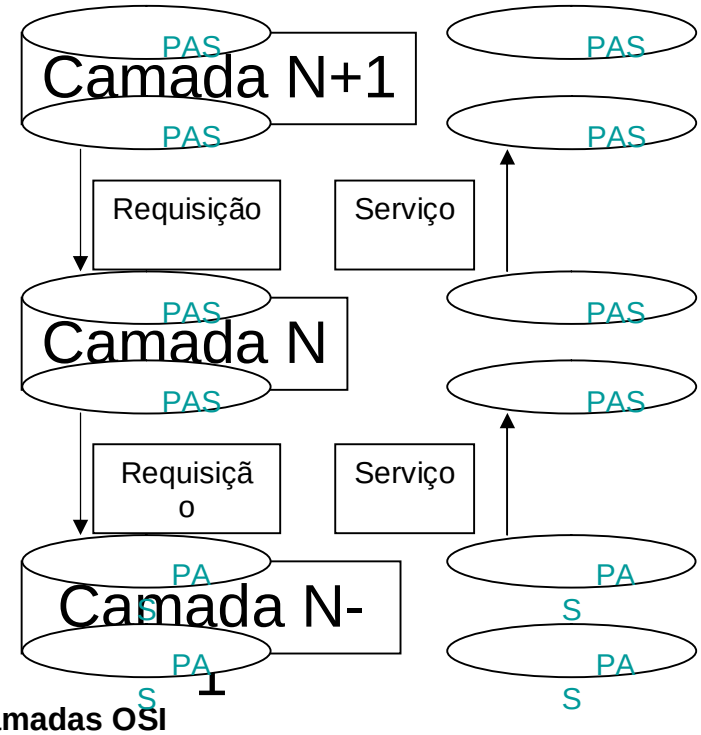
Para satisfazer requerimentos de clientes para a capacidade de computação remota, fabricantes de computadores de grande porte desenvolveram uma variedade de arquiteturas de redes.

Algumas destas arquiteturas definem o inter-relacionamento de fornecedores de *hardware* e *software*, em particular, para permitir o fluxo de comunicações através da rede para fabricantes de computadores em geral. Com a finalidade de padronizar o desenvolvimento de produtos para redes de comunicação de dados, foi elaborado um modelo aberto que teve como referência o *OSI - Open System Interconnection* pela ISO (*International Organization for Standardization*).

As principais características do Modelo OSI são:

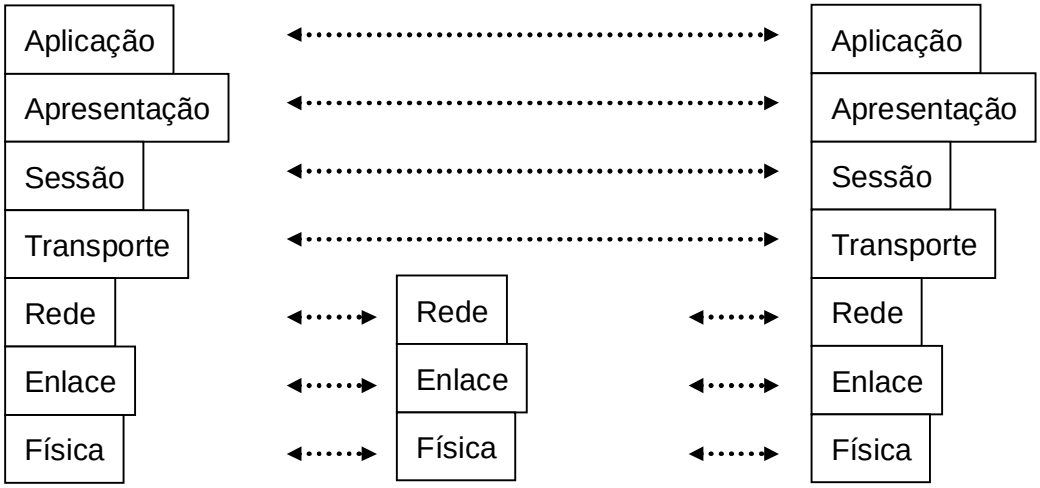
- Divisão em Camadas: as diferentes funcionalidades são fornecidas por diferentes protocolos, organizados em camadas;

- Pontos de Acesso de Serviços: as várias camadas se comunicam com as demais por uma interface padronizada, possibilitando a substituição do protocolo de uma das camadas por outro equivalente.



1.4.1 As camadas OSI

O Modelo OSI é dividido em 7 camadas, sendo que nem todas as entidades da rede precisam implementar todas elas.



1.4.2 A camada Física

A camada física lida com a transmissão pura de bits através de um canal de comunicação. As questões de projeto são concernentes à garantia de que, quando um lado transmite um bit 1, este seja recebido como bit 1 do outro lado.

As questões típicas aqui são quantos volts devem ser usados para representar um 1 e quantos um 0, quantos microssegundos dura um bit, se a transmissão pode ocorrer simultaneamente em ambos os sentidos, como a

conexão inicial é estabelecida e desfeita, quantos pinos o conector de rede possui e para que servem.

As questões de projeto lidam majoritariamente com interfaces mecânicas, elétricas e procedimentais, e com o meio físico de transmissão que está abaixo da camada física.

O projeto da camada física pode ser considerado como responsabilidade do engenheiro eletrônico.

1.4.3 A camada de Enlace

A tarefa principal da camada de enlace de dados é pegar a facilidade de transmissão de dados brutos e transformá-la em uma linha que pareça à camada de rede ser livre de erros de transmissão. É tarefa dessa camada resolver os problemas causados por quadros danificados, perdidos ou duplicados.

Ela realiza essa tarefa fazendo com que o transmissor fragmente os dados de entrada em quadros, transmita-os seqüencialmente e processe os quadros de confirmação mandados de volta pelo receptor.

Dado que a camada física meramente aceita uma seqüência de bits sem se importar com o significado ou a estrutura, cabe à camada de enlace de dados criar e reconhecer os limites dos quadros. Isto pode ser conseguido anexando-se padrões de bits especiais ao começo e fim do quadro.

A camada de enlace pode oferecer várias classes diferentes de serviço à camada de rede, cada qual com qualidade e preço diferentes.

É também tarefa desta camada impedir que um transmissor rápido afogue um receptor lento com dados, através de algum mecanismo regulador de tráfego.

1.4.4 A camada de Rede

A camada de rede se preocupa com o controle da operação da sub-rede, determinando como os pacotes são roteados da origem para o destino. As rotas podem ser baseadas em tabelas estáticas ou altamente dinâmicas, mudando para cada pacote.

É também função desta camada o controle de congestionamentos de pacotes.

A contabilidade do uso da rede é outra das funções da camada de rede.

1.4.5 A camada de Transporte

A função básica da camada de transporte é aceitar dados da camada de sessão, dividi-los se necessário em unidades menores, passá-las à cama de rede e garantir que os pedaços cheguem corretamente ao outro lado.

Em condições normais, a camada de transporte cria uma conexão de rede distinta para cada conexão de transporte exigida pela camada de sessão. Mas devido a requerimentos de banda e limitações de custo, pode ser necessário multiplexar várias conexões de transporte em uma conexão de rede, ou vice-versa.

A camada de transporte é uma verdadeira camada fim-a-fim. Ela é responsável pelo estabelecimento, manutenção e encerramento das conexões fim-a-fim.

1.4.6 A camada de Sessão

A camada de sessão permite a usuários em máquinas diferentes estabelecerem sessões entre eles. Uma sessão de transporte pode ser usada para permitir ao usuário logar-se remotamente, ou para transferir um arquivo entre duas máquinas.

Um dos serviços desta camada é gerenciar o controle de diálogos, acompanhando de quem é a vez de transmitir.

Um outro serviço da camada de sessão é a sincronização, fornecendo meios de inserir *checkpoints* no fluxo de dados, de forma que, depois de uma falha, somente os dados após o último *checkpoint* precisam ser repetidos.

1.4.7 A camada de Apresentação

Ao contrário de todas as camadas inferiores, que se interessavam em trocar bits confiavelmente, a camada de apresentação se relaciona com a sintaxe e a semântica da informação transmitida.

A camada de apresentação também pode realizar compressão de dados para reduzir o tráfego, ou criptografia para garantir a privacidade.

1.4.8 A camada de Aplicação

A camada de aplicação dentro do processo de comunicação faz interface com a aplicação. Ou seja, baseado em solicitações de uma aplicação de rede, esta camada seleciona serviços a serem fornecidos por camadas mais baixas.

Esta camada deve providenciar todos os serviços diretamente relacionados aos usuários. Alguns destes serviços são:

- identificação da intenção das partes envolvidas na comunicação e sua disponibilidade e autenticidade
- estabelecimento de autoridade para comunicar-se
- acordo sobre o mecanismo de privacidade
- determinação da metodologia de alocação de custo
- determinação de recursos necessários para prover uma qualidade de serviços aceitável
- sincronização de cooperação para aplicações
- seleção da disciplina de diálogo
- responsabilidade da recuperação de erros de estabelecimento
- acordo na validação de dados
- transferência de informações

Exercícios

1. Qual a diferença entre *internet* e *Internet*?

2. Qual a diferença entre cliente e servidor?

3. O que é um protocolo de rede?

4. Qual da opções abaixo não é uma camada do modelo OSI?
   a) Enlace
   b) Transporte
   c) Sessão
   d) Roteamento
   e) Rede

2 TCP/IP

O TCP/IP é um protocolo que conecta redes LAN ou WAN, homogêneas ou heterogêneas. Ele pode fazer tanto a comunicação entre hosts ponto-a-ponto, como a comunicação entre cliente e servidor.

Existem dois tipos de interação entre aplicações.

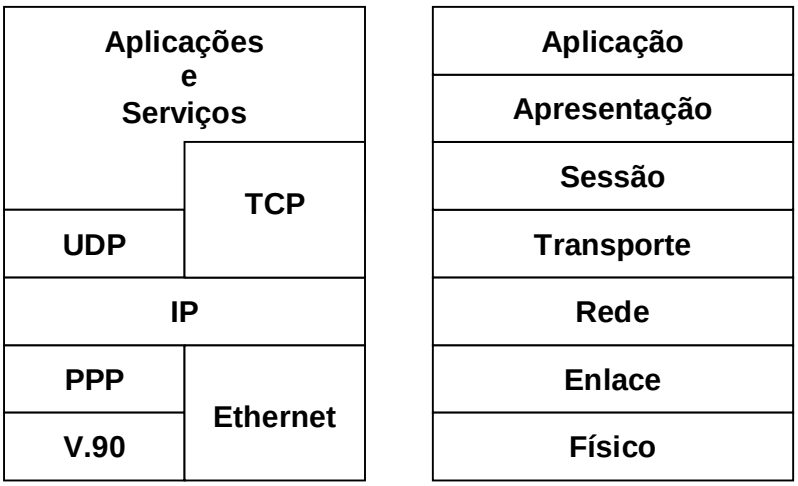
Comunicação Orientada à Conexão é apropriada quando as aplicações precisam de uma troca contínua de dados.

Em contraste, a Comunicação Não Orientada à Conexão é apropriada quando as aplicações trocam mensagens isoladas, geralmente com quantidades pequenas de dados.

Para conseguir uma troca de dados confiável entre os nós, são necessários vários procedimentos: empacotamento dos dados, determinação do caminho a ser seguido pelos pacotes, transmissão dos dados, adaptação da taxa de transmissão de acordo com a capacidade do destino de receber dados, gerenciamento de erros e retransmissão de dados.

Isso resulta em uma implementação complicada. Para facilitar essa tarefa, pode se usar uma implementação modular, agrupando as tarefas relacionadas em camadas distintas.

Para o TCP/IP foi adotado um modelo de comunicação dividido em camadas, que depois viria a influenciar o modelo OSI que hoje é considerado padrão.



Cada camada acrescenta informações de controle ao pacote de dados e transmite seu pacote para a camada inferior, que também adiciona seus dados.

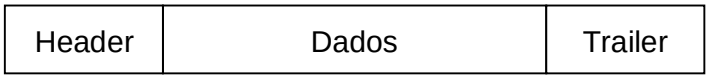
2.1 A camada Física

A camada física trabalha com a mídia física, conectores, e os sinais que representam 0 e 1. Por exemplo, um modem padrão V.90 ou V.92 implementam as funções da camada física.

2.2 A camada de Enlace de Dados

A camada de enlace de dados usa a capacidade de transmissão de dados brutos da camada física e transforma-a em uma linha que pareça à camada de rede ser livre de erros de transmissão. O protocolo PPP é um exemplo de camada de enlace de dados.

Na camada de Enlace de Dados, os dados são organizados em *quadros.*



Cada quadro tem um cabeçalho que inclui o endereço e controla a informação e um trailer que é usado para detecção de erros.

Um cabeçalho de quadro de LAN contém a os endereços “físicos” de origem e destino, que identificam as placas de rede.

Note que o Enlace pode ser uma rede local ou uma conexão Ponto-a-Ponto.

2.3 A camada de Rede

O IP (*Internet Protocol*) executa as funções da camada de rede. O IP roteia os dados entre sistemas. Os dados podem atravessar um único enlace ou podem ser retransmitidos por vários enlaces pela internet. As unidades de dados são chamadas *datagramas*.

Os datagramas têm um cabeçalho IP que contém endereçamento de rede. Os roteadores examinam o endereço de destino no cabeçalho IP para direcionar os datagramas para seu destino.

A camada IP é dita Não-Orientada à Conexão porque cada datagrama é roteado independentemente e o IP não garante a entrega confiável ou em seqüência dos datagramas.

2.4 Camada de Transporte

2.4.1 TCP

O TCP (*Transmission Control Protocol*) proporciona conexões de dados confiáveis para as aplicações. O TCP conta com mecanismos que garantem que os dados são entregues às suas aplicações locais:

- Íntegros
- Em seqüência
- Completos
- Sem duplicatas

Os mecanismos básicos que o TCP usa para conseguir isso são:

- Numeração dos segmentos

- Estabelecimento de *Timeout*
- Retransmissão dos segmentos

O lado que recebe os dados deve colocá-los em na seqüência correta, descartando duplicatas e confirmando o recebimento dos mesmos.

O TCP é implementado apenas nos hosts. O TCP é um protocolo full-duplex, ou seja, ambos os lados podem enviar dados ao mesmo tempo.

O TCP adiciona um cabeçalho ao pacote de dados da aplicação, formando um *segmento*.

O TCP passa os segmentos ao IP, que então roteia os mesmos até seu destino. O TCP aceita segmentos do IP, determina qual aplicação é o destino, e passa os dados para a aplicação apropriada.

2.4.2 UDP

O UDP não faz nenhuma garantia quanto à entrega dos dados, e é dever da aplicação trocar informações que confirmem a chegada dos dados. O UDP é implementado apenas nos hosts.

Com o UDP (*User Datagram Protocol*), uma aplicação manda uma mensagem isolada para outra aplicação. O UDP adiciona um cabeçalho, formando um *datagrama UDP*.

O UDP passa os segmentos ao IP, que então roteia os mesmos até seu destino. O UDP aceita segmentos do IP, determina qual aplicação é o destino, e passa os dados para a aplicação apropriada.

2.4.3 ICMP

O IP tem um projeto simples e elegante. Em condições normais, o IP faz um uso muito eficiente da memória e recursos de transmissão.

Como IP provê um serviço de expedição de datagramas sem conexão e não confiável, e além disso um datagrama viaja de um *gateway* a outro até alcançar um *gateway* que possa expedi-lo diretamente aa estação destino; é necessário um mecanismo que emita informações de controle e de erros quando acontecerem problemas na rede. Alguns dos problemas típicos que podem acontecer são:

- Um *gateway* não pode expedir ou rotear um datagrama;
- Um *gateway* detecta uma condição não usual, tal como congestionamento.

O mecanismo de controle que emite mensagens quando acontece algum erro é a função principal do protocolo ICMP. O ICMP permite aos *gateways* enviar mensagens de erros ou de controle a outros *gateways* ou *hosts*. ICMP provê comunicação entre o software de IP numa máquina e o software de IP numa outra máquina.

Tabela 2-1 Mensagens ICMP

Tipo	Mensagem
0	*Echo Reply*
3	*Destination Unreachable*

4	*Source Quench*
5	*Redirect*
8	*Echo*
11	*Time Exceeded*
12	*Parameter Problem*
13	*Time Stamp*
14	*Time Stamp Reply*

ICMP somente reporta condições de erros à fonte original. A fonte deve relatar os erros aos programas de aplicação individuais e tomar ação para corrigir o problema. Uma das mensagens que o ICMP pode enviar é: *Destination Unreachable*, o qual, por sua vez pode ser dos seguintes tipos:

- *Network Unreachable* (rede não alcançável)
- *Host Unreachable* (máquina não alcançável)
- *Port Unreachable* (porta não alcançável)
- *Destination Host Unknown* (máquina destino desconhecido)
- *Destination Network Unknown* (rede destino desconhecida)

2.5 Montagem dos pacotes

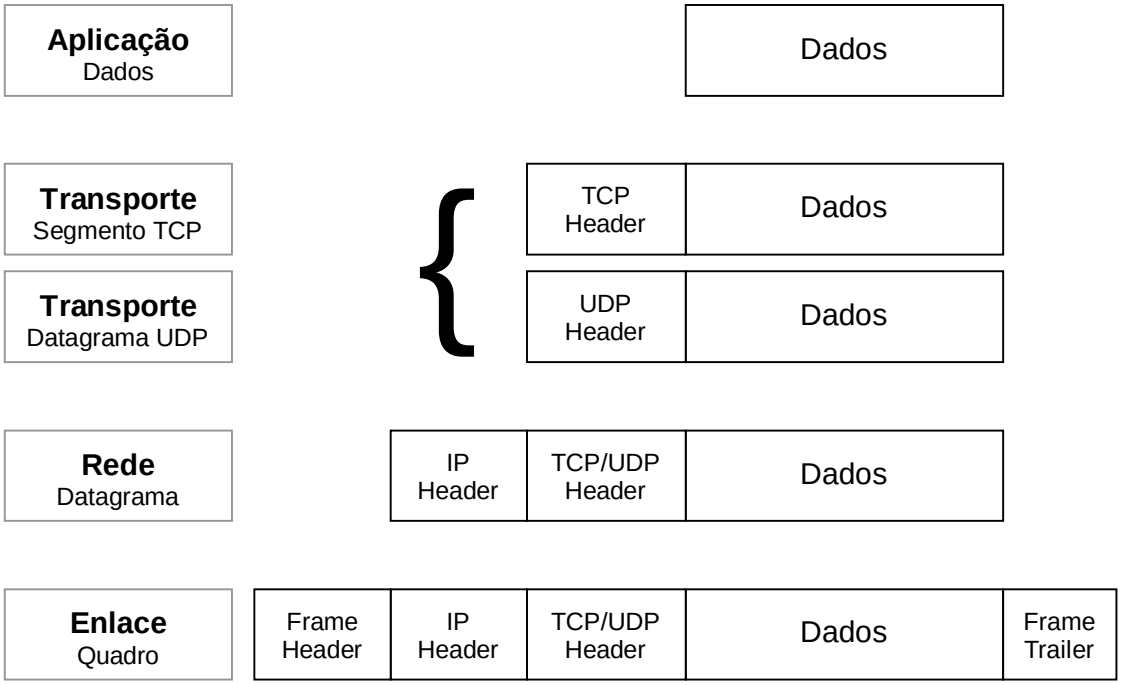
Na maioria das redes, a informação é dividida em várias partes, chamadas de pacotes, por duas razões: compartilhamento de recursos e detecção/correção de erros.

Obviamente não é justo que um único usuário da domine os recursos da rede por muito tempo. Com a divisão da informação em pacotes, cada um dos pacotes pode ser enviado/recebido individualmente, permitindo assim que outros pacotes possam trafegar pela rede. Isso possibilita um compartilhamento justo dos recursos.

Na maioria dos casos, os dispositivos são conectados através de cabos. Em alguns casos, usa-se ondas de rádio ou mesmo luz infravermelha. Ambas as formas de conexão física estão sujeitas a interferências, que podem corromper os dados que trafegam na rede. Grande parte do trabalho complexo em redes é detectar e solucionar os erros no tráfego dos dados.

A maioria das técnicas de detecção e correção de erros é baseada no uso de *checksums*. Quando a informação é enviada, é anexado em seu final um número indicando a soma de todos os bytes da mesma. Na recepção, esse número é comparado com a soma dos dados recebidos. Se houver diferença, a informação está corrompida e deve ser retransmitida.

Caso o bloco de dados seja muito grande, o reenvio vai tomar muito tempo, degradando a performance da rede. Para minimizar este problema, divide-se a informação em pacotes. Se houver algum erro, basta retransmitir apenas os pacotes corrompidos.



2.6 Endereçamento

Em uma rede, o endereço de um dispositivo é uma forma de identificar esse dispositivo como sendo único. Normalmente, os endereços de rede possuem um formato padronizado e bem definido.

2.6.1 Endereçamento de Enlace (MAC)

Os endereços MAC (*Media Access Control*) são atribuídos aos adaptadores de rede durante sua fabricação, sendo que cada adaptador tem um endereço que o identifica como único. Cada fabricante tem um código que o diferencia dos demais.

Os endereços MAC são escritos no seguinte formato:

```
00-c0-49-3f-c6-0c
```

Os primeiros bytes contêm o código do fabricante, os demais contêm o modelo e número serial do adaptador de rede.

2.6.2 Endereçamento de Rede (IP)

Em redes roteadas, o endereço é composto de pelo menos dois números: o da rede e o do nó. Se dois dispositivos possuírem endereços com o mesmo número de rede, então eles estão localizados na mesma rede. Do contrário, estão em redes distintas, mas unidas através de um roteador. O número que irá diferencia-los dentro desta rede é o número do nó.

Os endereços IP são números que identificam seu computador em uma rede TCP/IP. Inicialmente você pode imaginar o IP como um número de

telefone. O IP é composto por quatro bytes e a convenção de escrita dos números é chamada de "notação decimal pontuada". Por convenção, cada interface (placa usada p/ rede) do computador ou roteador tem um endereço IP. Também é permitido que o mesmo endereço IP seja usado em mais de uma interface de uma mesma máquina, mas normalmente cada interface tem seu próprio endereço IP.

As Redes do Protocolo Internet são seqüências contínuas de endereços IP. Todos os endereços dentro da rede têm um número de dígitos dentro dos endereços em comum. A porção dos endereços que são comuns entre todos os endereços de uma rede é chamada de *porção da rede*. O conjunto dos dígitos restantes é chamado de *porção dos hosts*. O número de bits que são compartilhados por todos os endereços dentro da rede é chamado de *netmask* (máscara da rede) e o papel da *netmask* é determinar quais endereços pertencem ou não a rede. Por exemplo, considere o seguinte:

Tabela 2-2 Formação de Endereços IP

Endereço do Host	`192.168.110.23`
Máscara da Rede	`255.255.255.0`
Porção da Rede	`192.168.110.`
Porção do Host	`.23`
Endereço da Rede	`192.168.110.0`
Endereço Broadcast	`192.168.110.255`

Qualquer endereço que é finalizado em zero em sua *netmask* revelará o *endereço da rede* a que pertence. O endereço de rede é então sempre o menor endereço numérico dentro da escalas de endereços da rede e sempre possui a *porção host* dos endereços codificada como zeros.

O endereço de *broadcast* é um endereço especial que cada computador em uma rede "escuta" em adição a seu próprio endereço. Este é um endereço onde os datagramas enviados são recebidos por todos os computadores da rede. Certos tipos de dados, como informações de roteamento e mensagens de alerta, são transmitidos para o endereço *broadcast*, assim todo computador na rede pode recebê-las simultaneamente.

Existem dois padrões normalmente usados para especificar o endereço de *broadcast*. O mais amplamente aceito é para usar o endereço mais alto da rede como endereço broadcast. No exemplo acima este seria 192.168.110.255. Por algumas razões outros sites têm adotado a convenção de usar o endereço de rede como o endereço broadcast. Na prática não importa muito se usar este endereço, mas você deve ter certeza que todo computador na rede esteja configurado para escutar o mesmo *endereço broadcast*.

2.6.2.1 Classes de Rede IP

Por razões administrativas após pouco tempo no desenvolvimento do protocolo IP alguns grupos arbitrários de endereços foram formados em redes

e estas redes foram agrupadas no que foram chamadas de *classes*. Estas classes armazenam um tamanho padrão de redes que podem ser usadas. As faixas alocadas são:

Tabela 2-3 Classes de Endereçamento

Classe	Bits Mais Significativos	Máscara de Rede	Endereços Possíveis na Rede
A	00000	255.0.0.0	0.0.0.0 – 127.255.255.255
B	10000	255.255.0.0	128.0.0.0 - 191.255.255.255
C	11000	255.255.255.0	192.0.0.0 - 223.255.255.255
D	11100	240.0.0.0	224.0.0.0 - 239.255.255.255
E	11110		240.0.0.0 – 255.255.255.255

2.6.2.2 O endereço de *loopback*

O endereço de *loopback* é um endereço especial que permite fazer conexões com você mesmo. Todos os computadores que usam o protocolo TCP/IP utilizam este endereço, e existem várias razões porque precisa fazer isto, por exemplo você pode testar vários programas de rede sem interferir com ninguém em sua rede. Por convenção, os endereços IP 127.0.0.1 a 127.255.255.254 foram escolhidos especificamente para a *loopback*, assim se abrir uma conexão para 127.0.0.1, abrirá uma conexão para o próprio computador local.

2.6.2.3 Endereços reservados para uso em Redes Privadas

Se você estiver construindo uma rede privada que nunca será conectada a Internet, então você pode escolher qualquer endereço que quiser. No entanto, para sua segurança e padronização, existem alguns endereços IP's que foram reservados especificamente para este propósito. Eles estão especificados no RFC1597 e são os seguintes:

Tabela 2-4 Endereços IP Reservados

ENDEREÇOS RESERVADOS PARA REDES PRIVADAS		
Classe	**Máscara de Rede**	**Endereço da Rede**
A	255.0.0.0	10.0.0.0 - 10.255.255.255
B	255.255.0.0	172.16.0.0 - 172.31.255.255
C	255.255.255.0	192.168.0.0 - 192.168.255.255

Você deve decidir primeiro qual será a largura de sua rede e então escolher a classe de rede que será usada.

2.6.2.4 Referência rápida de máscara de redes

A tabela abaixo faz referência às máscaras de rede mais comuns e a quantidade de máquinas máximas que ela atinge. Note que a especificação da máscara tem influência direta na classe de rede usada:

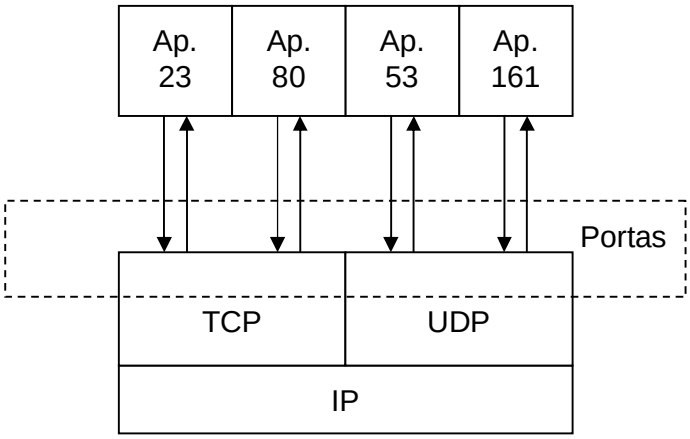
Máscara (octal)	Máscara (32 bits)	Número de Máquinas
Classe A:		
/8	/255.0.0.0	16,777,214
/9	/255.128.0.0	8,388,606
/10	/255.192.0.0	4,194,302
/11	/255.224.0.0	2,197,150
/12	/255.240.0.0	1,048,574
/13	/255.148.0.0	524,286
/14	/255.252.0.0	262,142
/15	/255.254.0.0	131,070
Classe B:		
/16	/255.255.0.0	65,534
/17	/255.255.128.0	32,766
/18	/255.255.192.0	16,382
/19	/255.255.224.0	8,190
/20	/255.255.240.0	4,094
/21	/255.255.248.0	2,046
/22	/255.255.252.0	1,022
/23	/255.255.254.0	510
Classe C:		
/24	/255.255.255.0	254
/25	/255.255.255.128	126
/26	/255.255.255.192	62
/27	/255.255.255.224	30
/28	/255.255.255.240	14
/29	/255.255.255.248	6
/30	/255.255.255.252	2
/32	/255.255.255.255	1

Qualquer outra máscara fora desta tabela (principalmente para a classe A), deverá ser redimensionada com uma calculadora de IP para chegar a um número aproximado de redes/máquinas aproximados que deseja.

2.6.3 Endereçamento de Sessão (Portas)

Todo processo que deseje estabelecer comunicação com outro processo deve se identificar de alguma forma. O TCP/IP implementa essa comunicação através do uso do conceito de portas (ou *ports*).

A porta é um número de 16 bits que identifica processos (ou serviços de rede). O número da porta de origem e o número da porta de destino estão incluídos no cabeçalho de cada segmento TCP ou pacote UDP.



Um *socket* é uma combinação de um endereço IP com um número de porta, e identifica um processo como único na rede.

2.7 Roteamento

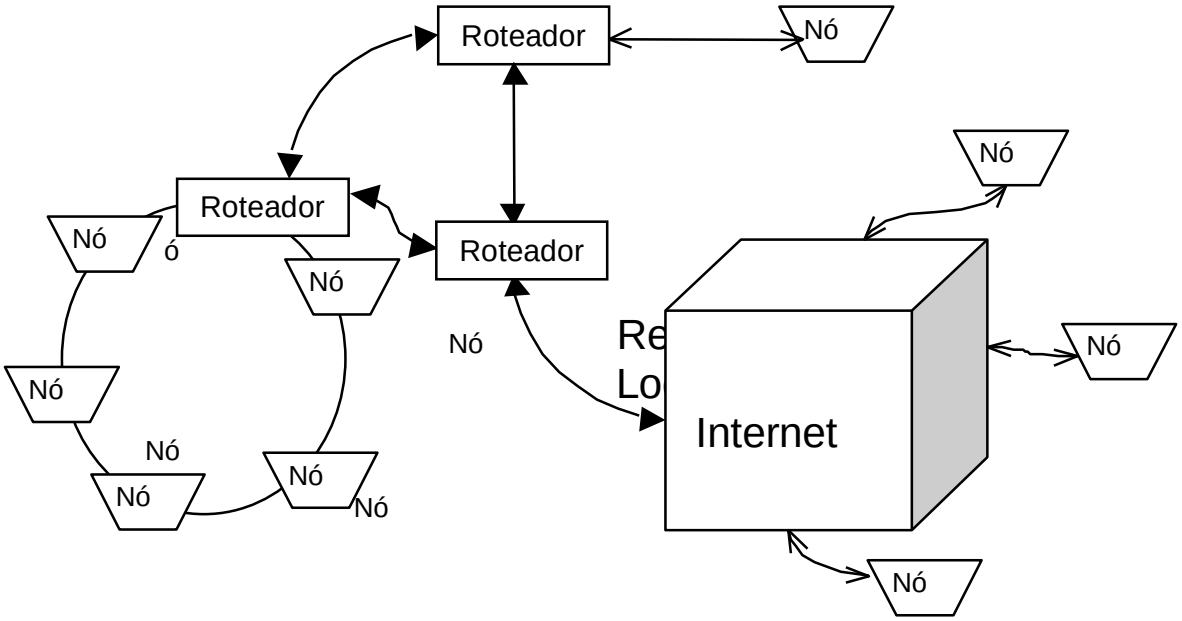
O TCP/IP pode ser usado em redes locais e para interligação de redes. As diversas redes locais conversam através dos roteadores. Pode haver mais de um caminho entre dois pontos.

As redes isoladas são conectadas por meio de Roteadores IP. Roteadores modernos são equipados com vários *slots* que podem receber diferentes tipos de adaptadores de rede: Ethernet, Token-Ring, FDDI, PPP, etc.

O software de IP roda nos *hosts* e nos roteadores.

- Se o destino está no mesmo enlace, manda-se o datagrama diretamente para ele;
- Se o destino não está no mesmo enlace, manda-se o pacote para o roteador local;
- Se o destino não estiver no mesmo enlace do roteador, este irá repassá-lo ao próximo roteador. Este processo continua até que o destino seja atingido.

Pode ser útil ilustrar isto com um exemplo. Imagine um simples roteador de escritório, ele pode ter um link intermitente com a Internet, um número de segmentos Ethernet alimentando as estações de trabalho e outro link PPP intermitente fora de outro escritório. Quando o roteador recebe um datagrama de qualquer de suas conexões de rede, o mecanismo que usa determina qual a



próxima interface deve enviar o datagrama. Computadores simples também precisam rotear, todos os computadores na Internet têm dois dispositivos de rede. Um é a interface *loopback* (explicada acima), o outro é usado para falar com o resto da rede, talvez uma interface Ethernet, talvez uma interface serial PPP ou SLIP.

Cada computador mantém uma lista de regras especiais de roteamento, chamada *tabela de roteamento*. Esta tabela contém colunas que tipicamente contém no mínimo três campos, o primeiro é o *endereço de destino*, o segundo é o *nome da interface* que o datagrama deve ser roteado e o terceiro é opcionalmente o *endereço IP* da outra máquina que levará o datagrama em seu próximo passo através da rede. No GNU/Linux você pode ver a tabela de roteamento usando um dos seguintes comandos:

```
cat /proc/net/route
route -n
netstat -r
```

O processo de roteamento é muito simples: um datagrama (pacote IP) é recebido, o endereço de destino (para quem ele é) é examinado e comparado com cada item da tabela de roteamento. O item que mais corresponder com o endereço é selecionado e o datagrama é direcionado a interface especificada.

Se o campo *gateway* estiver preenchido, então o datagrama é direcionado para aquele computador pela interface especificada, caso contrário o endereço de destino é assumido sendo uma rede suportada pela interface.

Em uma LAN pequena, as tabelas de roteamento podem ser feitas manualmente. Em redes maiores, os roteadores mantém suas tabelas atualizadas trocando informações entre si. Roteadores podem descobrir eventos como:

- Uma nova rede adicionada a internet;
- Um caminho foi interrompido, e um destino não pode ser atingido;
- Uma nova rota foi estabelecida para um destino.

Não existe um padrão único para troca de informação entre roteadores. A liberdade de escolha do protocolo estimulou a competição e levou a grandes melhorias nos protocolos.

Os protocolos de roteamento mais usados são o RIP e o OSPF.

2.7.1 Reempacotamento

Existe um evento olímpico onde um competidor nada uma parte do percurso, pega uma bicicleta e pedala outra parte, e corre uma terceira etapa. O IP funciona da mesma maneira. O datagrama foi projetado para poder ser mudado de uma mídia para outra até chegar ao seu destino.

Antes de um datagrama ser transmitido por um enlace, ele é empacotado em um quadro apropriado para o enlace. Quando um roteador recebe o quadro:

- O roteador desempacota o quadro e extrai o datagrama
- O roteador analisa o endereço de destino e descobre a mídia do próximo trecho
- O roteador reempacota o datagrama em um novo quadro, apropriado para o próximo laço

Exercícios

1. Em que nível de TCP/IP rodam o telnet e o ftp?

a) Físico
b) Sessão
c) Aplicação
d) Transporte
e) Enlace

2. O que significa a sigla MAC?

a) Media Assynchronous Connection
b) Master Assynchronous Connection
c) Media Access Connection
d) Media Access Control
e) Master Access Control

3. Que endereço de rede não é roteado na internet?
   a. 128.9.0.0
   b. 10.0.0.0
   c. 191.168.72.0
   d. 171.20.20.0
   e. 8.0.0.0

4. Qual das seguintes aplicações não usa o protocolo UDP?
   a. TFTP
   b. DNS
   c. RPC
   d. FTP
   e. SNMP

5. São máscaras padrão de redes classe A, B e C:
   a. 0.0.0.255, 0.0.255.255, 0.255.255.255
   b. 0.0.0.0, 0.0.0.255, 0.0.255.255
   c. 255.0.0.0, 255.255.0.0, 255.255.255.0
   d. 0.0.0.0, 255.0.0.0, 255.255.0.0
   e. 255.255.255.0, 255.255.0.0, 255.0.0.0

3 CONFIGURAÇÃO DO TCP/IP NO LINUX

Adaptado do Guia Foca GNU/Linux Intermediário – Capítulo 15.

3.1 Instalando uma máquina em uma rede existente

Se você quiser instalar uma máquina GNU/Linux em uma rede TCP/IP existente então você deve contatar qualquer um dos administradores da sua rede e perguntar o seguinte:

- Endereço IP de sua máquina
- Nome da Máquina
- Endereço IP da rede
- Endereço IP de broadcast
- Máscara da Rede IP
- Endereço do Roteador
- Endereço do Servidor de Nomes (DNS)

Você deve então configurar seu dispositivo de rede GNU/Linux com estes detalhes. Você não pode simplesmente escolhê-los e esperar que sua configuração funcione.

3.2 Configuração da interface Ethernet

As interfaces de rede no GNU/Linux estão localizadas no diretório /dev e a maioria é criada dinamicamente pelos softwares quando são requisitadas. Este é o caso das interfaces ppp e plip que são criadas dinamicamente pelos softwares.

Abaixo a identificação de algumas interfaces de rede no GNU/Linux (o "?" significa um número que identifica as interfaces seqüencialmente, iniciando em 0):

- eth? - Placa de rede Ethernet e WaveLan.
- ppp? - Interface de rede PPP (protocolo ponto a ponto).
- slip? - Interface de rede serial
- plip? - Interface de porta paralela
- arc?e, arc?s - Interfaces Arcnet
- sl?, ax? - Interfaces de rede AX25 (respectivamente para kernels 2.0.xx e 2.2.xx.
- fddi? - Interfaces de rede FDDI.
- dlci??, sdla? - Interfaces Frame Relay, respectivamente para para dispositivos de encapsulamento DLCI e FRAD.
- tr? – Interface Token Ring
- eql - Balanceador de tráfego para múltiplas linhas

3.2.1 A interface loopback

A interface *loopback* é um tipo especial de interface que permite fazer conexões com você mesmo. Todos os computadores que usam o protocolo TCP/IP utilizam esta interface e existem várias razões porque precisa fazer isto, por exemplo, você pode testar vários programas de rede sem interferir com ninguém em sua rede. Por convenção, o endereço IP 127.0.0.1 foi escolhido especificamente para a *loopback*, assim se abrir uma conexão `telnet` para 127.0.0.1, abrirá uma conexão para o próprio computador local.

A configuração da interface *loopback* é simples e você deve ter certeza que fez isto (mas note que esta tarefa é normalmente feita pelos scripts de inicialização existentes em sua distribuição).

```
[root@gauss:~] # ifconfig lo 127.0.0.1
```

Caso a interface *loopback* não esteja configurada, você poderá ter problemas quando tentar qualquer tipo de conexão com as interfaces locais, tendo problemas até mesmo com o comando `ping`.

3.2.2 Configurando uma interface com o `ifconfig`

Depois de configurada fisicamente, a interface precisa receber um endereço IP para ser identificada na rede e se comunicar com outros computadores, além de outros parâmetros como o endereço de *broadcast* e a *máscara de rede*. O comando usado para fazer isso é o `ifconfig` (interface configure).

Para configurar a interface de rede Ethernet (eth0) com o endereço 192.168.1.1, máscara de rede 255.255.255.0, podemos usar o comando:

```
[root@gauss:~] # ifconfig eth0 192.168.1.1 netmask 255.255.255.0 up
```

O comando acima ativa a interface de rede. A palavra up pode ser omitida, pois a ativação da interface de rede é o padrão. Para desativar a mesma interface de rede, basta usar usar o comando:

```
[root@gauss:~] # ifconfig eth0 down
```

Digitando `ifconfig` são mostradas todas as interfaces ativas no momento, pacotes enviados, recebidos e colisões de datagramas. Para mostrar a configuração somente da interface eth0, use o comando: `ifconfig eth0`

3.2.3 Configurando uma interface durante o boot

As interfaces de rede podem ser configuradas automaticamente durante o boot. Para isso existem os arquivos `/etc/sysconfig/network-scripts/ifcfg-*`. Vamos ver por exemplo o `ifcfg-eth0`:

```
DEVICE=eth0
BOOTPROTO=
```

```
ONBOOT=yes
IPADDR=192.168.72.100
NETMASK=255.255.255.0
BROADCAST=192.168.72.255
NETWORK=192.168.72.0
USERCTL=no
IPXNETNUM_802_2=""
IPXPRIMARY_802_2="no"
IPXACTIVE_802_2="no"
IPXNETNUM_802_3=""
IPXPRIMARY_802_3="no"
IPXACTIVE_802_3="no"
IPXNETNUM_ETHERII=""
IPXPRIMARY_ETHERII="no"
IPXACTIVE_ETHERII="no"
IPXNETNUM_SNAP=""
IPXPRIMARY_SNAP="no"
IPXACTIVE_SNAP="no"
```

Caso a interface fosse configurada por DHCP e não com um número fixo, teríamos o seguinte:

```
DEVICE=eth0
BOOTPROTO=dhcp
DHCP_HOSTNAME=gauss
ONBOOT=yes
IPADDR=
NETMASK=
BROADCAST=
NETWORK=
USERCTL=no
IPXNETNUM_802_2=""
IPXPRIMARY_802_2="no"
IPXACTIVE_802_2="no"
IPXNETNUM_802_3=""
IPXPRIMARY_802_3="no"
IPXACTIVE_802_3="no"
IPXNETNUM_ETHERII=""
IPXPRIMARY_ETHERII="no"
IPXACTIVE_ETHERII="no"
IPXNETNUM_SNAP=""
IPXPRIMARY_SNAP="no"
IPXACTIVE_SNAP="no"
```

3.2.4 Definindo diversos endereços IP para a mesma interface

Este interessante e útil recurso do GNU/Linux é conhecido como *IP Aliasing* e permite fazer nossa interface de rede local responder por diversos

endereços IP diferentes, mesmo sendo de classes de rede diferentes. Para usuários externos, a impressão é que a rede tem "muitas" máquinas, quando na realidade apenas uma responde por todos estes endereços virtuais. Outra aplicação útil é durante a transição de uma faixa de endereços na rede de `192.168.0.*` para `192.168.1.*`, por exemplo, isto poderá ser feito sem gerar transtornos na rede e sem parar o servidor.

Este recurso é a base para a construção de Máquinas virtuais baseadas em endereço IP (usado por *daemons* como Apache, proftpd e outros). A configuração deste recurso é simples, supondo que temos a interface `eth0` com o endereço atual `192.168.1.1`:

1. Compile seu kernel com o suporte a *IP Aliasing* (embutido ou como módulo).
2. Digite `ifconfig eth0:0 192.168.1.10` - Isto cria um apelido para a placa de rede, chamado `eth0:0`, que responderá as requisições para o endereço `192.168.1.10`.
3. Execute o comando `ifconfig` (sem parâmetros) para verificar se a nova interface foi ativada. Você deverá ver algo como:

```
eth0    Encapsulamento do Link: Ethernet  Endereço de HW 00:80:AE:B3:AA:AA
          inet end.: 192.168.1.1  Bcast:192.168.1.255  Masc:255.255.255.0
          UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1
          RX packets:979 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1228 errors:0 dropped:0 overruns:0 carrier:0
          colisões:1 txqueuelen:100
          RX bytes:71516 (69.8 Kb)  TX bytes:1146031 (1.0 Mb)
          IRQ:10 Endereço de E/S:0x300

eth0:0  Encapsulamento do Link: Ethernet  Endereço de HW 00:80:AE:B3:AA:AA
          inet end.: 192.168.1.10  Bcast:192.168.1.255  Masc:255.255.255.0
          UP BROADCASTRUNNING MULTICAST  MTU:1500  Métrica:1
          IRQ:10 Endereço de E/S:0x300
```

Note que o endereço de Hardware é o mesmo para ambos os dispositivos (porque a mesma placa de rede está respondendo endereços diferentes).

4. Se necessário ajuste as rotas ou gateway com o comando route (veja seção 3.3).

Podem ser especificados quantos apelidos você quiser e para qualquer faixa de endereços que desejar:

```
ifconfig eth0:1 192.168.1.11
ifconfig eth0:2 10.0.0.1
ifconfig eth0:3 200.132.12.10
```

Para desativar uma interface virtual, utilize o comando:

```
ifconfig eth0:1 down.
```

ATENÇÃO: Quando você desativa uma interface física (eth0), todas as interfaces virtuais também são desativadas.

3.3 Configurando uma rota no Linux

A configuração da rota é feita através da ferramenta route. Para adicionar uma rota para a rede 192.168.1.0 acessível através da interface eth0 basta digitar o comando:

```
[root@gauss:~] # route add  -net 192.168.1.0 eth0
```

Para apagar a rota acima da *tabela de roteamento*, basta substituir a palavra add por del. A palavra net quer dizer que 192.168.1.0 é um endereço de rede (lembra-se das explicações em Endereço IP, Section 4.3?)) para especificar uma máquina de destino, basta usar a palavra -host. Endereços de máquina de destino são muito usadas em conexões de rede apenas entre dois pontos (como ppp, plip, slip). Por padrão, a interface é especificada como último argumento. Caso a interface precise especifica-la em outro lugar, ela deverá ser precedida da opção -dev.

Para adicionar uma rota padrão para um endereço que não se encontre na tabela de roteamento, utiliza-se o *gateway padrão da rede*. Através do gateway padrão é possível especificar um computador (normalmente outro gateway) que os pacotes de rede serão enviados caso o endereço não confira com os da tabela de roteamento. Para especificar o computador 192.168.1.1 como *gateway padrão* usamos:

```
[root@gauss:~] # route add default gw 192.168.1.1 eth0
```

O *gateway padrão* pode ser visualizado através do comando route -n e verificando o campo gateway. A opção gw acima, especifica que o próximo argumento é um endereço IP (de uma rede já acessível através das tabelas de roteamento).

O computador *gateway* está conectado a duas ou mais redes ao mesmo tempo. Quando seus dados precisam ser enviados para computadores fora da rede, eles são enviados através do computador *gateway* e o *gateway* os encaminham ao endereço de destino. Desta forma, a resposta do servidor também é enviada através do *gateway* para seu computador (é o caso de uma típica conexão com a Internet).

A nossa configuração ficaria assim:

```
route add -net 192.168.1.0 eth0
route add default gw 192.168.1.1 eth0
```

Para mais detalhes, veja a página de manual do route ou o *NET3-4-HOWTO*.

3.4 Hostname

Todas as máquinas que integram uma rede TCP/IP devem ter um nome pelo qual são conhecidas pelas outras máquinas da rede. Esse nome é chamado de *hostname*. O comando utilizado para se configurar o *hostname* é o `hostname`.

```
[root@gauss:~] # hostname gauss.alfamidia
```

Para verificar o nome da máquina, usa-se o comando `hostname` sem parâmetros:

```
[root@gauss:~] # hostname
gauss.alfamidia
```

O hostname também pode ser configurado automaticamente durante o boot, através do arquivo /etc/sysconfig/network. Vejamos um exemplo:

```
NETWORKING=yes
HOSTNAME="gauss.alfamidia"
NISDOMAIN=""
GATEWAY=
GATEWAYDEV=
```

Com esse arquivo, o nome da estação vai ser inicializado automaticamente durante o boot.

3.5 Arquivos de configuração

3.5.1 O arquivo `/etc/hosts`

Este arquivo contém uma relação entre o endereço IP e o nome de computadores. A inclusão de um computador neste arquivo dispensa a consulta de um servidor de nomes para obter um endereço IP, sendo muito útil para máquinas que são acessadas freqüentemente. A desvantagem de fazer isto é que você mesmo precisará manter este arquivo atualizado e se o endereço IP de algum computador for modificado, esta alteração deverá ser feita em cada um dos arquivos `hosts` das máquinas da rede. Em um sistema bem gerenciado, os únicos endereços de computadores que aparecerão neste arquivo serão da interface loopback e os nomes de computadores.

```
# /etc/hosts
127.0.0.1       localhost loopback
192.168.0.1     this.host.name
```

Você pode especificar mais que um nome de computador por linha como demonstrada pela primeira linha, a que identifica a interface loopback.

3.5.2 O arquivo `/etc/networks`

O arquivo `/etc/networks` tem uma função similar ao arquivo `/etc/hosts`. Da mesma forma que cada máquina da rede pode ter um nome, as próprias redes também podem ter nomes. Esse arquivo serve para fazer a relação entre nomes de redes e número IP.

Ele contém um banco de dados simples de nomes de redes contra endereços de redes. Seu formato consiste em dois campos por linha e seus campos são identificados como:

```
Nome_da_Rede        Endereço_IP_da_Rede
```

Abaixo um exemplo de como se parece este arquivo:

```
loopnet     127.0.0.0
localnet    192.168.1.0
amprnet     44.0.0.0
```

Quando usar comandos como `route`, se um destino é uma rede e esta rede se encontra no arquivo `/etc/networks`, então o comando `route` mostrará o *nome da rede* ao invés de seu endereço.

3.5.3 O arquivo `/etc/host.conf`

Esse arquivo serve para configurar o sistema de resolução de nomes, onde é possível configurar alguns itens que controlam o comportamento do resolvedor de nomes. A principal opção desse arquivo é a order, que determina a ordem em que os serviços serão analisados. As opções válidas são:

- `bind`: consulta um servidor DNS;
- `hosts`: consulta o arquivo /etc/hosts;
- `nis`: consulta o NIS;

O formato deste arquivo é descrito em detalhes na página de manual resolv+. Em quase todas as situações, o exemplo seguinte funcionará:

```
order hosts,bind
multi on
```

Este arquivo de configuração diz ao resolvedor de nomes para checar o arquivo `/etc/hosts` (parâmetro `hosts`) antes de tentar verificar um *servidor de nomes* (parâmetro `bind`) e retornar um endereço IP válido para o computador procurado. `multi on` retornará todos os endereços IP resolvidos no arquivo /etc/hosts ao invés do primeiro.

Os seguintes parâmetros podem ser adicionados para evitar ataques de IP spoofing:

```
nospoof on
spoofalert on
```

O parâmetro `nospoof on` ativa a resolução reversa do nome (para checar se o endereço pertence realmente àquele nome) e o `spoofalert on` registra falhas desta operação no `syslog`.

3.5.4 O arquivo `/etc/resolv.conf`

Esse arquivo informa como vai ser a consulta ao servidor DNS. O `/etc/resolv.conf` é o arquivo de configuração principal do código do resolvedor de nomes. Seu formato é um arquivo texto simples com um parâmetro por linha, e os endereços de servidores DNS externos são especificados nele. Existem três palavras chaves normalmente usadas que são:

- `domain` Especifica o nome do domínio padrão, para que não seja necessário digitar todo o domínio para se acessar as máquinas da rede local..
- `search` Especifica uma lista de nomes de domínio alternativos ao procurar por um computador, separados por espaços. A linha `search` pode conter no máximo 6 domínios ou 256 caracteres.
- `nameserver` Especifica o endereço IP de um servidor DNS para resolução de nomes. Podem ser especificados até três servidores.

Como exemplo, o `/etc/resolv.conf` se parece com isto:

```
domain maths.wu.edu.au
search maths.wu.edu.au wu.edu.au
nameserver 192.168.10.1
nameserver 192.168.12.1
```

Este exemplo especifica que o nome de domínio a adicionar ao nome não qualificado (Ex. *hostnames* sem o domínio) é `maths.wu.edu.au` e que se o computador não for encontrado naquele domínio então a procura segue para o domínio `wu.edu.au` diretamente. Duas linhas de nomes de servidores foram especificadas, cada uma pode ser chamada pelo código resolvedor de nomes para resolver o nome.

3.6 Outros arquivos de configuração relacionados com a rede

3.6.1 O arquivo `/etc/services`

O arquivo `/etc/services` é um banco de dados simples que associa um nome amigável a humanos a uma porta de serviço amigável a máquinas. É um arquivo texto de formato muito simples, cada linha representa um item no banco de dados. Cada item é dividido em três campos separados por qualquer número de espaços em branco (tab ou espaços). Os campos são:

```
nome      porta/protocolo        apelidos     # comentário
```

Uma palavra simples que representa o nome do serviço sendo descrito.

porta/protocolo

Este campo é dividido em dois sub-campos.

- `porta`: Um número que especifica o número da porta em que o serviço estará disponível. Muitos dos serviços comuns têm designados um número de serviço. Estes estão descritos no RFC-1340.
- `protocolo`: Este sub-campo pode ser ajustado para *tcp* ou *udp*. É importante notar que o item *18/tcp* é muito diferente do item *18/udp* e que não existe razão técnica porque o mesmo serviço precisa existir em ambos. Normalmente o senso comum prevalece e que somente se um serviço esta disponível em ambos os protocolos *tcp* e *udp*, você precisará especificar ambos.
- `apelidos`: Outros nomes podem ser usados para se referir a entrada deste serviço.
- `comentário`: Qualquer texto aparecendo em uma linha após um caractere "#" é ignorado e tratado como comentário.

3.6.2 O arquivo `/etc/protocols`

O arquivo /etc/protocols é um banco de dados que mapeia números de identificação de protocolos novamente em nomes de protocolos. Isto é usado por programadores para permiti-los especificar protocolos por nomes em seus programas e também por alguns programas tal como *tcpdump* permitindo-os mostrar *nomes* ao invés de *números* em sua saída. A sintaxe geral deste arquivo é:

```
nomeprotocolo  número  apelidos
```

4 DHCP

Adaptado do "DHCP mini-HOWTO", por Vladimir Vuksan.

4.1 Protocolo DHCP

DHCP significa Protocolo de Configuração Dinâmica de Máquinas. É usado para controlar parâmetros de controle de rede vitais para as máquinas (ao ser executado em clientes) com a ajuda de um servidor. O DHCP mantém compatibilidade reversa com o BOOTP. Para maiores informações veja a RFC 2131 (antiga RFC 1531) e outras (veja a seção de Recursos Internet no final deste documento). Pode-se verificar ainda o FAQ sobre DHCP.

Este mini-HOWTO aborda tanto o servidor DHCP como o cliente DHCP. Muitos usuários precisam do cliente, que é usado nas estações de trabalho, para obter as informações de rede de um servidor remoto. O servidor é usado pelos administradores de sistemas para distribuir informações de rede para os clientes. Então caso o leitor seja somente um usuário normal de rede, necessitará somente do programa cliente.

4.2 Configuração do Cliente DHCP

Atualmente, existem três clientes DHCP para Linux: o `dhcpcd`, `pump` and `dhclient`. Este mini-HOWTO trata principalmente do dhcpcd.

4.2.1 Obtendo o Programa Cliente (dhcpcd)

Independente da distribuição que se esteja utilizando será necessário obter um programa cliente DHCP para Linux. O pacote necessário é chamado dhcpcd e a sua versão atual é 0.70. Pode-se obter uma descrição do pacote em aqui.

Dependendo da sua distribuição, você pode ter que baixar o daemon cliente DHCP. Se você não quer compilá-lo a partir do código fonte, o pacote que você precisa chama-se dhcpcd, e a versão atual é a 1.3.18. Ele é mantido por Sergei Viznyuk, e atualmente já vem como pacote na maioria das distribuições.

O código fonte do dhcpcd pode ser obtido nos seguintes locais:

ftp://ftp.phystech.com/pub/ (Site Principal)

http://www.cps.msu.edu/~dunham/out/

Depois siga as instruções adiante, devem ser as mesmas.

4.2.2 Slackware

Pode-se obter a última versão do DHCPcd a partir de qualquer espelho do Metalab ou nos seguintes endereços:

ftp://metalab.unc.edu/pub/Linux/system/network/daemons

ftp://ftp.phystech.com/pub/ (Site Principal)

Transfira a última versão do programa `dhcpcd.tar.gz`

- Descompacte-a através do comando:
  ```
  tar -zxvf dhcpcd-1.3.18pl1.tar.gz
  ```

- vá para o diretório gerado e compile o dhcpcd, através dos seguintes comandos:
  ```
  cd dhcpcd-1.3.18pl1
  make
  ```

- Instale-o, (você deve ser o superusuário para fazer isso), através do comando:
  ```
  make install
  ```

Este procedimento irá criar o diretório `/etc/dhcpc` onde o DHCPcd irá armazenar as informações DHCP e o arquivo `dhcpcd` será copiado em `/usr/sbin`.

Para que o sistema utilize o DHCP durante a inicialização, execute os seguintes comandos:

```
cd /etc/rc.d
mv rc.inet1 rc.inet1.OLD
```

Isso moverá o antigo script de inicialização da `rede para rc.inet1.OLD`. Agora será necessário criar o novo script `rc.inet1`. O código a seguir é tudo o que é preciso:

```
#!/bin/sh
#
# rc.inet1      Este script inicializa o sistema INET básico

HOSTNAME=`cat /etc/HOSTNAME` # este procedimento provavelmente não é
                             # necessário, mas será mantido mesmo assim

# Anexar os dispositivos de rede locais
/sbin/ifconfig lo 127.0.0.1
/sbin/route add -net 127.0.0.0 netmask 255.0.0.0 lo

# Caso se tenha uma conexão Ethernet, as linhas a seguir devem  ser
# utilizadas para configurar a interface eth0.  Caso se esteja usando a
# interface local ou SLIP, não inclua o restante das linhas no arquivo.

/usr/sbin/dhcpcd
```

Salve o programa e reinicialize o computador.

Ao finalizar vá para a seção 4.2.6.

4.2.3 Red Hat 6.x e Mandrake 6.x

A configuração do DHCPcd sobre estas distribuições de GNU/Linux é muito fácil. Tudo o que se precisa fazer é inicializar o Painel de Controle digitando:

```
control-panel
```

Depois disso, selecione "Configuração de Rede". Na interface Ethernet desejada (normalmente eth0) defina DHCP como o protocolo de configuração da interface. Os demais parâmetros devem ser deixados em branco.

Ao finalizar vá para a seção 4.2.6.

Note que o RedHat 6.x por padrão instala um cliente DHCP chamado `pump` no lugar do `dhcpcd` mencionado anteriormente. O CD-ROM inclui um pacote do `dhcpcd`, então se não for possível configurar o `pump` pode ser tentado o `dhcpcd`. Depois de instalar o dhcpcd (ex: `rpm -i dhcpcd-1.3.17pl2-1.i386.rpm`), é preciso fazer algumas alterações.

4.2.4 Red Hat 5.x e Conectiva GNU/Linux 3.x

A configuração do DHCPcd sobre estas distribuições de GNU/Linux é muito fácil. Tudo o que se precisa fazer é inicializar o Painel de Controle digitando:

```
control-panel
```

Depois disso, selecione "Configuração de Rede". Na interface Ethernet desejada (normalmente eth0) defina DHCP como o protocolo de configuração da interface. Os demais parâmetros devem ser deixados em branco.

Ao finalizar vá para a seção 4.2.6.

4.2.5 Debian

Há um pacote Debian do DHCPcd em:

http://ftp.debian.org/debian/dists/slink/main/binary -i386/net/

Ou pode-se seguir as instruções de instalação do Slackware.

Para desempacotar um pacote do tipo deb deve-se executar o seguinte comando:

```
dpkg -i /where/ever/your/debian/packages/are/dhcpcd*deb
```

Aparentemente não há necessidade de qualquer configuração para o DHCPcd conforme o descrito a seguir:

De: Heiko Schlittermann (heiko@os.inf.tu-dresden.de)

O pacote dhcpcd instala o script de inicialização usual para pacotes Debian em `/etc/init.d/nome_pacote`, neste caso como `/etc/init.d/dhcpcd`, e estabelece os links dos diversos diretórios `/etc/rc?.d/`.

O conteúdo dos diretórios `/etc/rc?.d/` será então executado durante a inicialização do sistema.

Caso o sistema não seja reinicializado após a instalação, o daemon deve se iniciado manualmente:

```
/etc/init.d/dhcpcd start
```

Ao finalizar vá para a seção 4.2.6.

4.2.6 Executando o DHCPCD

Após a reinicialização da máquina, a interface de rede deve ser configurada. Digite:

```
ifconfig
```

Deve-se obter algo como:

```
lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Bcast:127.255.255.255  Mask:255.0.0.0
        UP BROADCAST LOOPBACK RUNNING  MTU:3584  Metric:1
        RX packets:302 errors:0 dropped:0 overruns:0 frame:0
```

```
        TX packets:302 errors:0 dropped:0 overruns:0 carrier:0 coll:0

eth0    Link encap:Ethernet  HWaddr 00:20:AF:EE:05:45
        inet addr:24.128.53.102  Bcast:24.128.53.255  Mask:255.255.254 .0
        ^^^^^^^^^^^^^^^^^^^^^^^
        UP BROADCAST NOTRAILERS RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:24783 errors:1 dropped:1 overruns:0 frame:1
        TX packets:11598 errors:0 dropped:0 overruns:0 carrier:0 coll:96
        Interrupt:10 Base address:0x300
```

Caso se tenha algum número normal no campo `inet addr`, então a máquina estará configurada. Caso se obtenha `0.0.0.0` não se desespere, trata-se de uma configuração temporária antes que o dhcpcd obtenha o endereço IP. Se mesmo depois de alguns minutos se visualizar somente o endereço `0.0.0.0`, por favor verifique a seção 4.2.8. O DHCPcd é um daemon que seguirá rodando enquanto o computador estiver ligado. A cada três horas ele irá contatar o servidor DHCP e tentará renovar a reserva do endereço IP. Todas as mensagens serão registradas no arquivo do syslog (no Slackware em `/var/adm/syslog`, RedHat/Conectiva/Debian em `/var/log/syslog`).

Um detalhe final: É necessário especificar os servidores de nomes. Há duas formas de fazê-lo, pode-se solicitar ao provedor para informar os endereços do servidor de nomes e colocá-los em `/etc/resolv.conf` ou o DHCPcd obterá uma lista do servidor DHCP e construirá um `resolv.conf` em `/etc/dhcpc`.

Decidi usar o `resolv.conf` gerados pelo DHCPcd da seguinte forma:

Gere uma cópia de segurança do `/etc/resolv.conf` da seguinte forma:

```
mv /etc/resolv.conf /etc/resolv.conf.OLD
```

Caso o diretório `/etc/dhcpc` não exista, ele deve ser criado da seguinte forma:

```
mkdir /etc/dhcpc
```

Crie um link simbólico de `/etc/dhcpc/resolv.conf` para `/etc/resolv.conf`, através do seguinte comando:

```
ln -s /etc/dhcpc/resolv.conf /etc/resolv.conf
```

Caso isso não funcione, pode-se tentar o ajuste sugerido por nothing@cc.gatech.edu, com uma pequena correção por Henrik Stoerner.

O último passo foi executado somente porque meu dhcpcd não cria o arquivo `/etc/dhcpc/resolv.conf`. Efetuei as seguintes mudanças no arquivo `/etc/sysconfig/network-scripts/ifup` (que são mudanças feias, mas funcionam):

```
elif [ "$BOOTPROTO" = dhcp  -a "$ISALIAS" = no ]; then
    echo -n "Usando DHCP para ${DEVICE}...  "
    /sbin/dhcpcd -c /etc/sysconfig/network-scripts/ifdhcpc-done ${DEVICE}
    echo "echo \$$ > /var/run/dhcp-wait-${DEVICE}.pid; exec sleep 30" |sh

    if [ -f /var/run/dhcp-wait-${DEVICE}.pid ]; then
        echo "falhou."
        exit 1
```

Foi mudada para:

```
elif [ "$BOOTPROTO" = dhcp  -a "$ISALIAS" = no ]; then
    echo -n "Usando DHCP para ${DEVICE}...  "
    /sbin/dhcpcd
    echo "echo \$$ > /var/run/dhcp-wait-${DEVICE}.pid; exec sleep 30" |sh

    if [ ! -f /var/run/dhcp-wait-${DEVICE}.pid ]; then
        ^^^^^^
        echo "falhou."
        exit 1
```

Note o "`!`" (ponto de exclamação) em `if [ ! -f /var/run/dhcp-wait-${DEVICE}.pid ];`

Agora sente-se e aproveite. :-).

4.2.7 Inicializando e encerrando o dhcpcd

Caso se necessite de conectividade de rede somente ocasionalmente, pode-se inicializar o dhcpcd a partir da linha de comando (deve-se reinicializar para se executar este passo) com:

```
/usr/sbin/dhcpcd
```

Quando se quiser finalizar as funcionalidades de rede, digite:

```
/usr/sbin/dhcpcd -k
```

4.2.8 Solução de Problemas

Caso se tenha seguido todos os passos descritos anteriormente e não foi possível acessar a rede, há algumas causas prováveis:

4.2.8.1 A placa de rede não está configurada adequadamente

Durante o processo de inicialização o GNU/Linux irá tentar localizar a placa de rede e deverá algo parecido com isso:

```
eth0: 3c509 at 0x300 tag 1, 10baseT port, address 00 20 af ee 11 11, IRQ 10.
3c509.c:1.07 6/15/95 becker@cesdis.gsfc.nasa.gov
```

Se uma mensagem como esta não aparecer, a placa Ethernet pode não ter sido reconhecida pelo Linux. Caso se utilize uma placa genérica (como um clone NE2000), procure por um disco com utilitários DOS que possam ser usados para configurar a placa. Tente diferentes IRQs até que o GNU/Linux reconheça a placa (IRQs 9, 10, 12 são normalmente boas alternativas).

4.2.8.2 O servidor DHCP suporta RFC 1541/O servidor é um Windows NT

Tente executar o dhcpcd digitando:

```
dhcpcd -r
```

Use o comando `ifconfig` para verificar se a interface de rede está configurada (espere alguns segundos pelo processo de configuração, pois inicialmente o endereço apresentado será `inet.addr=0.0.0.0`)

Caso o problema seja resolvido desta forma, deve-se adicionar o indicador "`-r`" aos programas de inicialização, ou seja além de usar o `/sbin/dhcpcd` deve-se comandar `/sbin/dhcpcd -r`

4.2.8.3 Durante a inicialização se obtém a mensagem "Usando DHCP para eth0 ... falhou" mas o sistema funciona perfeitamente.

Aparentemente se está usando RedHat e as instruções não foram seguidas adequadamente :-). Você está esquecendo do ponto de exclamação em um dos comandos, vá até a seção 4.2.6 e verifique como corrigir o problema.

4.2.8.4 A rede funciona por alguns minutos e subitamente para de responder.

Há algumas notícias de que o programa gated (servidor de caminho padrão) gera erros em máquinas GNU/Linux conforme o descrito acima. Verifique se o programa gated está sendo executado através do seguinte comando:

```
ps -auxww | grep gate
```

Caso ele esteja sendo executado, tente removê-lo com o gerenciador RPM ou removendo a entrada em `/etc/rc.d/`

4.2.8.5 A placa Ethernet é reconhecida durante a inicialização do sistema mas obtém-se a mensagem "NO DHCPOFFER" nos arquivos de registros de ocorrências. Isso ocorre também com a placa PCMCIA.

É preciso ter certeza de que a porta 10BaseT (RJ45) da placa de rede está ativada. A melhor forma de verificar isso é checar qual o conector que está configurado na inicialização, ou seja:

```
eth0: 3c509 at 0x300 tag 1, 10baseT port, address  00 20 af ee 11 11, IRQ 10.
                            ^^^^^^^^^^^^
3c509.c:1.07 6/15/95 becker@cesdis.gsfc.nasa.gov
```

Tenho recebido notícias de usuários de portáteis que têm tido este tipo de problema com os utilitários PCMCIA (especificamente o `ifport`) que deve ser configurado para um conector de tipo 10Base2 (thinnet). Deve-se ter certeza de estar usando o padrão 10BaseT para a conexão. Caso não tenha, deve-se reconfigurar a placa e reinicializar o computador.

4.2.8.6 Meu cliente faz as requisições mas não recebe resposta (Contribuição de Peter Amstutz).

Em alguns sistemas, é necessário incluir algum hostname para sua máquina como parte da requisição. Com o dhcpcd, use `dhcpcd –h algumhost`.

4.2.8.7 Segui todos os passos mas ainda não consigo conectar a máquina à rede.

O modem a cabo normalmente irá guardar o endereço Ethernet da placa de rede, então ao se mudar o computador ou se tentar utilizar uma outra placa, será necessário ensinar o modem a reconhecer o novo hardware. Normalmente basta desligar o modem e religá-lo enquanto o computador estiver ligado ou então chamar o suporte técnico e avisá-lo de que a placa de rede foi alterada.

Existem regras de firewall (regras `ipfwadm`) que desabilita o tráfego nas portas 67 e 68, usadas pelo DHCP para distribuir informações de configuração. Verifique as regras do firewall cuidadosamente.

4.3 Configuração do Servidor DHCP

4.3.1 Servidor DHCP para UNIX

Há diversos servidores DHCP disponíveis para sistema do tipo Unix, comerciais ou de livre distribuição. Um dos mais populares servidores DHCP de livre distribuição é o DHCPd de Paul Vixie/ISC. A versão atual é 1.0 (sugerida por muitos usuários), mas a 2.0 encontra-se em estágio beta. Ela pode ser obtida em:

ftp://ftp.isc.org/isc/dhcp/

Descompacte-a, vá até o diretório da distribuição e digite:

```
./configure
```

Levará algum tempo para se configurar todos os parâmetros. Após isto estar finalizado, digite:

```
make
```

e

```
make install
```

4.3.2 Configuração de Rede

Ao finalizar a instalação digite ifconfig -a. Deve-se obter um resultado similar a:

```
eth0      Link encap:10Mbps Ethernet  HWaddr 00:C0:4F:D3:C4:62
          inet addr:183.217.19.43  Bcast:183.217.19.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2875542 errors:0 dropp ed:0 overruns:0
          TX packets:218647 errors:0 dropped:0 overruns:0
          Interrupt:11 Base address:0x210
```

Caso o parâmetro MULTICAST não esteja presente, deve-se reconfigurar o kernel para sua adição. Em muitos sistemas isso não será necessário.

O próximo passo será adicionar a rota 255.255.255.255. Apresentamos um extrato retirado ao arquivo README do DHCPd:

"Para que o dhcpd funcione perfeitamente, escolha alguns clientes DHCP (por exemplo Windows 9x), que sejam capazes de enviar pacotes com um endereço IP de destino 255.255.255.255. Infelizmente o GNU/Linux teima em mudar 255.255.255.255 no endereço de divulgação da subrede local (neste caso 192.5.5.223). Isso cria uma violação do protocolo DHCP e enquanto muitos clientes DHCP não avisam do problema, outros (como os clientes DHCP Microsoft) o fazem. Clientes com este tipo de problema não visualizam a mensagem DHCPOFFER enviada pelo servidor."

Digite:

```
route add -host 255.255.255.255 dev eth0
```

caso se obtenha a mensagem

```
"255.255.255.255: máquina desconhecida"
```

deve-se tentar adicionar a seguinte entrada ao arquivo /etc/hosts:

```
255.255.255.255 all-ones
```

tente então:

```
route add -host all-ones dev eth0
```

ou

```
route add -net 255.255.255.0 dev eth0
```

eth0 é obviamente o nome do dispositivo de rede que está sendo usado. Caso seja diferente, faça as devidas alterações.

4.3.3 Opções do DHCPd

Agora é necessário configurar o DHCPd. Para se fazer isso deve-se criar ou editar o arquivo /etc/dhcpd.conf.

Comumente se deseja definir endereços IP de forma aleatória. Isso pode ser feito da seguinte forma:

```
default-lease-time 600;
max-lease-time 7200;
option subnet-mask 255.255.255.0;
option broadcast-address 192.168.1.255;
option routers 192.168.1.254;
option domain-name-servers 192.168.1.1, 192.168.1.2;
option domain-name "dominio.org.br";

subnet 192.168.1.0 netmask 255.255.255.0 {
   range 192.168.1.10 192.168.1.100;
   range 192.168.1.150 192.168.1.200;
}
```

Isso fará com que o servidor DHCP forneça ao cliente um endereço IP na faixa 192.168.1.10-192.168.1.100 ou 192.168.1.150-192.168.1.200. Ele liberará um endereço por 600 segundos caso o cliente não defina um tempo específico de utilização de endereço. De qualquer forma o tempo máximo permitido será de 7.2 segundos. O servidor irá "avisar" ao cliente que ele pode usar 255.255.255.0 como máscara de subrede, 192.168.1.255 como endereço de distribuição, 192.168.1.254 como roteador ou caminho padrão, 192.168.1.1 e 192.168.1.2 como servidores DNS.

Pode-se ainda definir endereços IP específicos baseados nos endereços Ethernet dos clientes, como por exemplo:

```
host conec {
   hardware ethernet 08:00:2b:4c:59:23;
   fixed-address 192.168.1.222;
}
```

Este procedimento irá definir o endereço 192.168.1.222 para o cliente com endereço Ethernet igual a 08:00:2b:4c:59:23.

Pode-se misturar os procedimentos, definindo-se certos clientes com endereços IP estáticos (por exemplo servidores) e outros com endereços dinâmicos (como por exemplo portáteis). Há diversas opções como: endereços de servidores Windows, servidores de data e horário, etc... Caso se necessite alguma destas opções por favor verifique a página de manual on line do dhcpd.conf.

4.3.4 Inicializando o Servidor

Podemos agora acionar o servidor DHCP. Basta simplesmente digitar ou incluir nos programas de inicialização do sistema:

```
/usr/sbin/dhcpd
```

Caso se deseje verificar se tudo está funcionando perfeitamente, deve-se acionar inicialmente o modo de depuração e colocar o servidor em primeiro plano. Isso pode ser feito através do comando:

```
/usr/sbin/dhcpd -d -f
```

Inicialize algum dos clientes e verifique a console do servidor. Deverão ser apresentadas diversas mensagens de depuração.

Exercícios

1. Qual a utilidade de um servidor DHCP?

2. Quais os clientes DHCP mais utilizados no GNU/Linux?

5 COMANDOS DE REDE

Adaptado do Guia Foca GNU/Linux Intermediário – Capítulo 11.

Este capítulo traz alguns comandos úteis para uso em rede e ambientes multiusuário.

5.1 O comando `who`

Mostra quem está atualmente conectado no computador. Este comando lista os nomes de usuários que estão conectados em seu computador, o terminal e data da conexão.

who [*opções*]

onde:

opções

`-H, --heading`

Mostra o cabeçalho das colunas.

`-i, -u, --idle`

Mostra o tempo que o usuário está parado em Horas:Minutos.

`-m, i am`

Mostra o nome do computador e usuário associado ao nome. É equivalente a digitar who i am ou who am i.

`-q, --count`

Mostra o total de usuários conectados aos terminais.

`-T, -w, --mesg`

Mostra se o usuário pode receber mensagens via talk (conversação).

\+ O usuário recebe mensagens via talk

\- O usuário não recebe mensagens via talk.

? Não foi possível determinar o dispositivo de terminal onde o usuário está conectado.

5.2 O comando `telnet`

Permite acesso a um computador remoto. É mostrada uma tela de acesso correspondente ao computador local onde deve ser feita a autenticação do usuário para entrar no sistema. Muito útil, mas deve ser tomado cuidados ao disponibilizar este serviço para evitar riscos de segurança.

telnet [*opções*][*ip/dns*][*porta*]

onde:

ip/dns

Endereço IP do computador de destino ou nome DNS.

porta

Porta onde será feita a conexão. Por padrão, a conexão é feita na porta *23*.

opções

`-8`

Requisita uma operação binária de 8 bits. Isto força a operação em modo binário para envio e recebimento. Por padrão, telnet não usa 8 bits.

`-a`

Tenta um login automático, enviando o nome do usuário lido da variável de ambiente USER.

`-d`

Ativa o modo de debug.

`-r`

Ativa a emulação de rlogin.

`-l [usuário]`

Faz a conexão usando [usuário] como nome de usuário.

Exemplo: telnet 192.168.1.1, telnet 192.168.1.1 23.

5.3 O comando `finger`

Mostra detalhes sobre os usuários de um sistema. Algumas versões do finger possuem bugs e podem significar um risco para a segurança do sistema. É recomendado desativar este serviço na máquina local.

finger [*usuário*] [*usuário@host*]

Onde:

usuário

Nome do usuário que deseja obter detalhes do sistema. Se não for digitado o nome de usuário, o sistema mostra detalhes de todos os usuários conectados no momento.

usuário@host

Nome do usuário e endereço do computador que deseja obter detalhes.

`-l`

Mostra os detalhes de todos os usuários conectados no momento. Entre os detalhes, estão incluídos o *nome do interpretador de comandos* (shell) do usuário, *diretório home*, *nome do usuário*, *endereço*, etc.

`-p`

Não exibe o conteúdo dos arquivos .plan e .project

Se for usado sem parâmetros, mostra os dados de todos os usuários conectados atualmente ao seu sistema.

Exemplo: finger, finger root.

5.4 O comando `ftp`

Permite a transferência de arquivos do computador remoto/local e vice versa. O file transfer protocol é o sistema de transmissão de arquivos mais usado na Internet. É requerida a autenticação do usuário para que seja permitida a conexão. Muitos servidores ftp disponibilizam acesso anônimo aos usuários, com acesso restrito.

Uma vez conectado a um servidor ftp, você pode usar a maioria dos comandos do GNU/Linux para operá-lo.

ftp [*ip/dns*]

Abaixo alguns dos comandos mais usados no FTP:

`ls`

Lista arquivos do diretório atual.

`cd [diretório]`

Entra em um diretório.

`get [arquivo]`

Copia um arquivo do servidor ftp para o computador local. O arquivo é gravado, por padrão, no diretório onde o program ftp foi executado.

`hash [on/off]`

Por padrão esta opção está desligada. Quando ligada, faz com que o caracter "#" seja impresso na tela indicando o progresso do download.

`mget [arquivos]`

Semelhante ao get, mas pode copiar diversos arquivos e permite o uso de curingas.

`send [arquivo]`

Envia um arquivo para o diretório atual do servidor FTP (você precisa de uma conta com acesso a gravação para fazer isto).

`prompt [on/off]`

Ativa ou desativa a pergunta para a cópia de arquivo. Se estiver como off assume sim para qualquer pergunta.

Exemplo: `ftp ftp.br.debian.org`

5.5 O comando `whoami`

Mostra o nome que usou para se conectar ao sistema. É útil quando você usa várias contas e não sabe com qual nome entrou no sistema :-)

5.6 O comando `dnsdomainname`

Mostra o nome do domínio de seu sistema.

5.7 O comando `hostname`

Mostra ou muda o nome de seu computador na rede.

5.8 O comando `talk`

Inicia conversa com outro usuário em uma rede local ou Internet. Talk é um programa de conversação em tempo real onde uma pessoa vê o que a outra escreve.

talk [*usuário*] [*tty*]

ou

talk [*usuário@host*]

Onde:

usuário

Nome de login do usuário que deseja iniciar a conversação. Este nome pode ser obtido com o comando who (veja who, Seção 9.1).

tty

O nome de terminal onde o usuário está conectado, para iniciar uma conexão local.

usuário@host

Se o usuário que deseja conversar estiver conectado em um computador remoto, você deve usar o nome do usuário@hosname do computador.

Após o talk ser iniciado, ele verificará se o usuário pode receber mensagens, em caso positivo, ele enviará uma mensagem ao usuário dizendo como responder ao seu pedido de conversa. Veja em 5.1 O comando `who`.

Você deve autorizar o recebimento de talks de outros usuários para que eles possam se comunicar com você , para detalhes veja o comando O comando `mesg` a seguir.

5.9 O comando `mesg`

Permite ou não o recebimentos de requisições de talk de outros usuários.

mesg [*y/n*]

Onde: *y* permite que você receba "talks" de outros usuários.

Digite mesg para saber se você pode ou não receber "talks" de outros usuários. Caso a resposta seja "n" você poderá enviar um talk para alguém mas o seu sistema se recusará em receber talks de outras pessoas.

É interessante colocar o comando mesg y em seu arquivo de inicialização .bash_profile para permitir o recebimento de "talks" toda vez que entrar no sistema.

5.10 O comando `ping`

Verifica se um computador está disponível na rede. Este comando é muito utilizado por alguns programas de conexão e administradores para verificar se uma determinada máquina está conectada na rede e também para verificar o tempo de resposta de cada máquina da rede. O ping envia pacotes ICMS ECHO_REQUEST para um computador, este quando recebe o pacote envia uma resposta ao endereço de origem avisando que está disponível na rede.

ping [*opções*][*IP/DNS*]

onde:

IP/dns

Endereço IP ou nome DNS do endereço.

opções

-c [num]

Envia *num* pacotes ao computador de destino.

-f

Flood ping. Envia novos pacotes antes de receber a resposta do pacote anterior. Para cada requisição enviada, um "." é mostrado na tela e para cada resposta recebida, um backspace é mostrado. Somente o usuário root pode utilizar esta opção e pode te auxiliar muito na detecção de erros de transmissão de pacotes em interfaces das máquinas em sua rede.

-i [seg]

Aguarda [seg] segundos antes de enviar cada pacote.

-q

Não mostra as requisições enquanto são enviadas, somente mostra as linhas de sumário no inicio e término do programa.

-s [tamanho]

Especifica o tamanho do pacote que será enviado.

-v, --verbose

Saída detalhada, tanto os pacotes enviados como recebidos são listados.

Exemplo: ping 192.168.1.1, ping www.br.debian.org.

5.11 O comando `rlogin`

Executa um login em uma máquina local ou remota.

rlogin [*opções*] [*IP/DNS*]

onde:

IP/DNS

Endereço IP ou DNS do computador que será acessado.

opções

`-l [nome]`

Entra com o user id [nome] no sistema.

rlogin é usado para executar comandos interativamente no computador de destino (como se você estivesse sentado diante dele, muito semelhante ao telnet). Para executar comandos não interativamente veja rsh, Seção 11.11.

5.12 O comando `rsh`

Executa um comando em um computador local ou remoto.

rsh [*opções*] [*IP/DNS*] [*comando*]

Onde:

IP/DNS

Endereço IP ou nome DNS do computador.

comando

Comando que será executado no computador local/remoto.

opções

-l [nome]

Entra no sistema usando o login [nome].

rsh é usado somente para executar comandos. Para usar um shell interativo veja **Error! Reference source not found.** e O comando `rlogin`.

5.13 O comando `w`

Mostra quem está conectado no sistema e o que cada um está fazendo.

w [*opções*] [*usuário*]

onde:

usuário

Nome do usuário que deseja ver os detalhes. Se o usuário não for digitado, o comando w mostra detalhes de todos os usuários conectados no sistema.

opções

`-h`

Não mostra o cabeçalho

`-u`

Ignora os nomes de usuários enquanto verifica os processo atuais e tempos de CPU.

`-f`

Mostra ou oculta o campo *FROM* na listagem.

5.14 O comando `traceroute`

Mostra o caminho percorrido por um pacote para chegar ao seu destino. Este comando mostra na tela o caminho percorrido entre os Gateways da rede e o tempo gasto de retransmissão. Este comando é útil para encontrar computadores defeituosos na rede caso o pacote não esteja chegando ao seu destino.

traceroute [*opções*] [*host/IP de destino*]

Onde:

host/IP destino

É o endereço para onde o pacote será enviado (por exemplo, www.debian.org). Caso o tamanho do pacote não seja especificado, é enviado um pacote de 38 bytes.

opções

`-l`

Mostra o tempo de vida do pacote (ttl)

`-m [num]`

Ajusta a quantidade máximas de ttl dos pacotes. O padrão é 30.

`-n`

Mostra os endereços numericamente ao invés de usar resolução DNS.

`-p [porta]`

Ajusta a porta que será usada para o teste. A porta padrão é 33434.

`-r`

Pula as tabelas de roteamento e envia o pacote diretamente ao computador conectado a rede.

`-s [end]`

Usa o endereço IP/DNS [end] como endereço de origem para computadores com múltiplos endereços IPs ou nomes.

`-v`

Mostra mais detalhes sobre o resultado do traceroute.

`-w [num]`

Configura o tempo máximo que aguardará por uma resposta. O padrão é 3 segundos.

Exemplos: traceroute www.debian.org, traceroute www.linux.org.

5.15 O comando `netstat`

Mostra conexões de rede, tabela de roteamento, estatísticas de interfaces, conexões masquerade, e mensagens.

netstat [*opções*]

Onde:

opções

`-i [interface]`

Mostra estatísticas da interface [interface].

`-M, --masquerade`

Se especificado, também lista conexões masquerade.

`-n, --numeric`

Usa endereços numéricos ao invés de tentar resolver nomes de hosts, usuários e portas.

`-c, --continuos`

Mostra a listagem a cada segundo até que a CTRL+C seja pressionado.

Se não for especificada nenhuma opção, os detalhes das conexões atuais serão mostrados.

5.16 O comando `wall`

Envia uma mensagem a todos os usuários do sistema. Este comando faz a leitura de um arquivo ou entrada padrão e escreve o resultado em todos os terminais onde existem usuários conectados. Somente o usuário root pode utilizar este comando.

wall [*arquivo*]

Exemplos: wall /tmp/mensagem.txt, echo Teste de mensagem enviada a todos os usuários conectados ao sistema wall.

6 O SSH

Adaptado do Guia Foca GNU/Linux Intermediário – Capítulo 14.

O serviço de ssh permite fazer o acesso remoto ao console de sua máquina, em outras palavras, você poderá acessar sua máquina como se estivesse conectado localmente ao seu console (substituindo o rlogin e rsh). A principal diferença com relação ao serviço telnet padrão, rlogin e rsh é que toda a comunicação entre cliente/servidor é feita de forma encriptada usando chaves públicas/privadas RSA para criptografia garantindo uma transferência segura de dados.

A velocidade do console remoto conectado via Internet é excelente (melhor que a obtida pelo telnet e serviços r*) dando a impressão de uma conexão em tempo real (mesmo em links discados de 9.600 KB/s), a compactação dos dados também pode ser ativada para elevar ainda mais a velocidade entre cliente-servidor ssh. Além do serviço de acesso remoto, o scp possibilita a transferência/recepção segura de arquivos (substituindo o rcp).

Em conexões sem criptografia (rsh, rlogin) os dados trafegam de forma desprotegida e caso exista algum sniffer instalado em sua rota com a máquina destino, todo o que fizer poderá ser capturado (incluindo senhas).

6.1 Versão

É assumido que esteja usando a versão 2.0 do ssh. As explicações contidas aqui podem funcionar para versões posteriores, mas é recomendável que leia a documentação sobre modificações no programa (changelog) em busca de mudanças que alterem o sentido das explicações fornecidas aqui.

6.2 História

O openSSH (explicado neste capítulo) é baseado na última versão livre do implementação de Tatu Ylonen com todos os algoritmos patenteados (para bibliotecas externas) removidos, todos as falhas de segurança corrigidas, novas características e muitas outras melhorias. O openSSH foi criado por Aaron Campbell, Bob Beck, Markus Friedl, Niels Provos, Theo de Raadt e Dug Song.

6.3 Contribuindo

A Homepage principal do OpenSSH é http://www.unixuser.org/~haruyama/security/openssh/index.html. Falhas, correções e sugestões podem ser enviadas para a lista de discussão openssh-unix-dev@mindrot.org (aberta a postagens de usuários não inscritos).

6.4 Características

Abaixo as principais características do serviço ssh (Openssh).

- Conexão de dados criptografada entre cliente/servidor.
- Cópia de arquivos usando conexão criptografada.

- Suporte a ftp criptografado (sftp).
- Suporte a compactação de dados entre cliente/servidor.
- Controle de acesso das interfaces servidas pelo servidor ssh.
- Suporte a controle de acesso tcp wrappers.
- Autenticação usando um par de chaves pública/privada RSA ou DSA.
- Algoritmo de criptografia livre de patentes.
- Suporte a PAM.
- Suporte a caracteres ANSI (cores e códigos de escape especiais no console).

6.5 Ficha técnica

Pacote: ssh

Utilitários:

- ssh - Cliente ssh (console remoto).
- slogin - Link simbólico para o programa ssh.
- sshd - Servidor de shell seguro ssh.
- scp - Programa para transferência de arquivos entre cliente/servidor
- ssh-keygen - Gera chaves de autenticação para o ssh
- sftp - Cliente ftp com suporte a comunicação segura.
- sftp-server - Servidor ftp com suporte a comunicação segura.
- ssh-add - Adiciona chaves de autenticação DSA ou RSA ao programa de autenticação.
- ssh-agent - Agente de autenticação, sua função é armazenar a chave privada para autenticação via chave pública (DSA ou RSA).
- ssh-keyscan - Scaneia por chaves públicas de autenticação de hosts especificados. O principal objetivo é ajudar na construção do arquivo local know_hosts.
- ssh-copy-id - Usado para instalação do arquivo identity.pub em uma máquina remota.

Arquivos de configuração:

- /etc/ssh/sshd_config - Arquivo de configuração do servidor ssh.
- /etc/ssh/ssh_config - Arquivo de configuração do cliente ssh.
- ~/.ssh/config - Arquivo de configuração pessoal do cliente ssh.

6.6 Servidor ssh

6.6.1 Requerimentos de Hardware

É recomendado no mínimo 6MB de memória RAM para a execução do serviço ssh mais o kernel do Linux. Este limite deve ser redimensionado para servidores de acesso dedicado, uma quantidade de 64MB deve ser confortável para centenas de usuários conectados simultaneamente (o que raramente acontece).

O ssh que acompanha a distribuição Debian vem com o suporte a tcp wrappers compilado por padrão.

6.6.2 Arquivos de log criados pelo servidor ssh

Detalhes sobre a execução do servidor sshd (como inicio, autenticação e término) são enviadas ao syslog do sistema. A *prioridade* e *nível* são definidos no arquivo de configuração /etc/ssh/sshd_config (veja 6.8.5 Exemplo de sshd_config com explicações das diretivas).

6.6.3 Instalação do servidor openSSH

```
apt-get install ssh
```

Por padrão o servidor sshd é instalado como daemon, também é possível executa-lo via inetd mas isto não é aconselhável porque o servidor gera uma chave aleatória de seção toda vez que é iniciado, isto podendo levar vários segundos (quando é usada a versão 1 do protocolo ssh, veja 6.8.4 Diferenças nas versões do protocolo).

6.6.4 Iniciando o servidor/reiniciando/recarregando a configuração

O arquivo que controla o funcionamento do daemon do ssh é controlado pelo arquivo `/etc/init.d/ssh`.

A execução do ssh através de inetd é automática quando é feita uma requisição para a porta *22*.

6.6.5 Opções de linha de comando

Opções de linha de comando do servidor sshd:

- -b bits - Especifica o número de bits da chave do servidor (768 por padrão).
- -d - Modo de depuração - O servidor envia detalhes sobre seu funcionamento aos logs do sistema e não é executado em segundo plano. Ele também responderá conexões pelo mesmo processo. Podem ser usadas no máximo 3 opções *-d* para aumentar os detalhes de depuração.
- -f arquivo_configuração Indica um arquivo de configuração alternativo (por padrão é usado /etc/ssh/sshd_config). O ssh pode ser configurado através de opções de linha de comando, mas requer um arquivo de configuração para ser executado. Opções de linha de comando substituem as especificadas no arquivo de configuração.
- -g segundos - Especifica o tempo máximo para a digitação de senha de acesso. Após o tempo especificado o servidor encerra a conexão. O valor padrão é 600 segundos e 0 desativa este recurso.
- -h arquivo_chave - Diz qual arquivo contém a chave privada local. O padrão é /etc/ssh/ssh_host_key e somente o usuário root deve ter permissões de leitura neste arquivo. Será necessário especificar esta opção caso o sshd não esteja sendo executado como usuário root.

É possível ter múltiplos arquivos de chaves para os protocolos 1 e 2 do ssh.

- -i - Indica que o servidor sshd será executado pelo inetd. Isto não é aconselhável porque o servidor gerará a chave aleatória de seção toda vez que for iniciado e isto pode levar alguns segundos. Esta opção pode se tornar viável com o uso do protocolo 2 ou criando chaves pequenas como 512 bytes (no ssh 1), mas a segurança criptográfica também será diminuída. Veja as diferenças entre os dois protocolos em 6.8.4 Diferenças nas versões do protocolo.
- -k segundos - Especifica a freqüência da geração de novas chaves do daemon sshd. O valor padrão é 3600 segundos e 0 desativa este recurso.

ATENÇÃO: NÃO desative este recurso!!! Esta opção traz a segurança que uma nova chave gerada de servidor será gerada constantemente (esta chave é enviada junto com a chave pública quando o cliente conecta e fica residente na memória volátil), assim mesmo que um cracker consiga obtê-la interceptando as conexões, será praticamente impossível tentar qualquer coisa. Valores menores tendem a aumentar ainda mais a segurança.

- -p porta - Especifica a porta que o daemon sshd atenderá as requisições. Por padrão é usada a porta 22.
- -q - Nenhuma mensagem será enviada ao syslog do sistema.
- -u tam - Especifica o tamanho do campo de nome do computador que será armazenado no arquivo utmp. A opção *u0* faz somente endereços IP serem gravados.
- -D - Quando usada não faz o sshd iniciar em segundo plano.
- -V versão_cliente - Assume que o cliente possui a versão ssh especificada (1 ou 2) e não faz os testes de identificação de protocolo.
- -4 - Força o uso do protocolo IP tradicional (IPv4).
- -6 - Força o uso da nova geração do protocolo IP (IPv6).

A maioria das opções são realmente úteis para modificar o comportamento do servidor ssh sem mexer em seu arquivo de configuração (para fins de testes) ou para executar um servidor ssh pessoal, que deverá ter arquivos de configuração específicos.

6.7 Cliente ssh

Esta é a ferramenta usada para seções de console remotos. O arquivo de configuração de usuários é `~/.ssh/config`, e o arquivo global `/etc/ssh/ssh_config`. Para conectar a um servidor ssh remoto:

```
ssh ip/nome_do_servidor_ssh
```

O uso da opção *-C* é recomendado para ativar o modo de compactação dos dados (útil em conexões lentas). A opção *-l usuário* pode ser usada para alterar a identificação de usuário (quando não é usada, o login local é usado como nome de usuário remoto). Uma porta alternativa pode ser especificada usando a opção *-p porta* (a 22 é usada por padrão).

Na primeira conexão, a chave pública do servidor remoto será gravada em ~/.ssh/know_hosts ou ~/.ssh/know_hosts2 (dependendo da versão do servidor ssh remoto, veja 6.8.4 Diferenças nas versões do protocolo), e verificada a cada conexão como checagem de segurança para se certificar que o servidor não foi alvo de qualquer ataque ou modificação não autorizada das chaves. Por padrão, o cliente utilizará o protocolo ssh versão 1, a opção *-2* permite usar o protocolo versão 2.

Variáveis de ambiente personalizadas para o ssh poderão ser definidas no arquivo ~/.ssh/environment. Comandos que serão executados somente na conexão ssh em ~/.ssh/rc e /etc/ssh/sshrc caso contrário será executado o xauth por padrão.

OBS: Para utilizar autenticação Rhosts/Rhosts+RSA (arquivos ~/.rhosts/~/.shosts) o programa ssh deverá ter permissões SUID root e conectará usando portas baixas (menores que 1024).

```
Exemplos:
# Conecta-se ao servidor remoto usando o login do usuário atual
ssh ftp.sshserver.org

# Conecta-se ao servidor remoto usando o login john (via ssh versão 2)
ssh -2 ftp.sshserver.org -l john

# Conecta-se ao servidor remoto usando compactação e o login john
ssh ftp.sshserver.org -C -l john

# Conecta-se ao servidor remoto usando compactação, o login john,
# ativa o redirecionamento do agente de autenticação ( -A) e redirecionamento
# de conexões X11 ( -X). Veja a próxima seção para entender como o
# suporte a redirecionamento de conexões do X funciona.
ssh ftp.sshserver.org -C -A -X -l john
```

6.7.1 Redirecionamento de conexões do X

O redirecionamento de conexões do X Window poderá ser habilitado em ~/.ssh/config ou /etc/ssh/ssh_config ou usando as opções *-A -X* na linha de comando do ssh (as opções *-a* e *-x* desativam as opções acima respectivamente). Uma variável $DISPLAY é criada automaticamente para fazer o redirecionamento ao servidor X local.

Ao executar um aplicativo remoto, a conexão é redirecionada a um DISPLAY proxy criado pelo ssh (a partir de :10, por padrão) que faz a conexão com o display real do X (:0), ou seja, ele pulará os métodos de autenticação xhost e cookies. Por medidas de segurança é recomendável habilitar o redirecionamento individualmente somente se você confia no administrador do sistema remoto.

```
# Exemplo de configuração do ssh_config

# Permite Redirecionamento de conexões para o próprio computador (nomes de
# máquinas podem ser especificadas).
Host 127.0.0.1
    ForwardAgent yes
    ForwardX11 yes
```

```
    # Opções específicas do cliente para conexões realizadas a 192.168.1.4
usando
    # somente o protocolo 2
    Host 192.168.1.4
       # As 2 linhas abaixo ativam o redirecionamento de conexões do X
       ForwardAgent yes
       ForwardX11 yes
       PasswordAuthentication yes
       Port 22
       Protocol 2
       Cipher blowfish

    # Opções específicas do cliente para conexões realizadas a 192.168.1.5
usando
    # somente o protocolo 1
    Host 192.168.1.5
       # As 2 linhas abaixo desativam o redirecionamento de conexões do X
       ForwardAgent no
       ForwardX11 no
       PasswordAuthentication yes
       Port 22
       Protocol 1
       Cipher blowfish

    #   CheckHostIP yes
    #   RhostsAuthentication no
    #   RhostsRSAAuthentication yes
    #   RSAAuthentication yes
    #   FallBackToRsh no
    #   UseRsh no
    #   BatchMode no
    #   StrictHostKeyChecking yes
    #   IdentityFile ~/.ssh/identity
    #   IdentityFile ~/.ssh/id_dsa
    #   IdentityFile ~/.ssh/id_rsa1
    #   IdentityFile ~/.ssh/id_rsa2
    #   EscapeChar ~
```

6.7.2 scp

Permite a cópia de arquivos entre o cliente/servidor ssh. A sintaxe usada por este comando é a seguinte:

scp [*origem*] [*destino*]

Os parâmetros de *origem* e *destino* são semelhantes ao do comando cp mas possui um formato especial quando é especificado uma máquina remota:

- Um caminho padrão - Quando for especificado um arquivo local. Por exemplo: /usr/src/arquivo.tar.gz.

- usuario@host_remoto:/diretório/arquivo - Quando desejar copiar o arquivo de/para um servidor remoto usando sua conta de usuário. Por exemplo: gleydson@ftp.debian.org:~/arqs.

A opção *-C* é recomendável para aumentar a taxa de transferência de dados usando compactação. Caso a porta remota do servidor sshd seja diferente de 22, a opção *-P porta* deverá ser especificada (é "P" maiúscula mesmo, pois a *-p* é usada para preservar permissões/data/horas dos arquivos transferidos).

```
Exemplos:
# Para copiar um arquivo local chamado /pub/teste/script.sh para
# meu diretório pessoal em ftp.sshserver.org
scp -C /pub/teste/script.sh gleydson@ftp.sshserver.org:~/

# Para fazer a operação inversa a acima (copiando do servidor remoto pa ra o
local)
# é só inverter os parâmetros origem/destino:
scp -C gleydson@ftp.sshserver.org:~/script.sh /pub/teste

# Para copiar o arquivo local chamado /pub/teste/script.sh para
# o diretório /scripts dentro do meu diretório pessoal  em ftp.sshserver.org
# com o nome teste.sh
scp -C /pub/teste/script.sh gleydson@ftp.sshserver.org:~/scripts/teste.sh

# O exemplo abaixo faz a transferência de arquivos entre 2 computadores
remotos:
# O arquivo teste.sh é lido do servi dor server1.ssh.org e copiado para
# server2.ssh.org (ambos usando o login gleydson)
scp -C gleydson@server1.ssh.org:~/teste.sh gleydson@server2.ssh.org:~/
```

6.7.3 sftp

Permite realizar transferência de arquivos seguras através do protocolo ssh. A conexão e transferências são realizadas através da porta 22 (ainda não é possível modificar a porta padrão). A sintaxe para uso deste comando é a seguinte:

sftp usuario@host_remoto

Compactação pode ser especificada através da opção *-C*. Um arquivo contendo os comandos usados na seção sftp poderá se especificado através da opção *-b arquivo* para automatizar tarefas.

OBS1: Para desativar o servidor sftp, remova a linha SubSystem sftp /usr/lib/sftp-server (que inicializa o sub-sistema ftp) do arquivo /etc/ssh/sshd_config e reinicie o servidor sshd.

OBS2: O suporte ao programa sftp somente está disponível ao protocolo ssh versão 2 e superiores.

OBS3: Algumas opções comuns do cliente ftp padrão (como *mget*) ainda não estão disponíveis ao sftp. Veja a página de manual para detalhe sobre as opções disponíveis.

6.8 14.3 Servidor ssh

6.8.1 sshd

Este é o daemon de controle da conexão encriptada via protocolo ssh, transferência de arquivos e shell interativo. As opções de linha de comando estão disponíveis em 6.6.5 Opções de linha de comando. Seu arquivo de configuração principal é /etc/ssh/sshd_config, um exemplo e descrição das opções deste arquivo é encontrada em 6.8.5 Exemplo de sshd_config com explicações das diretivas.

OBS1: É recomendável que o arquivo `/etc/ssh/sshd_config` seja lido somente pelo dono/grupo, por conter detalhes de acesso de usuários, grupos e intervalo entre a geração de chave de seção.

OBS2: Se estiver ocorrendo falhas no acesso ao servidor ssh, verifique as permissões nos arquivos `/etc/hosts.allow` e `/etc/hosts.deny` (o nome do serviço é sshd). Mesmo operando como daemon, o servidor utiliza estes arquivos para fazer um controle de acesso adicional.

6.8.2 Controle de acesso

É definido pelas opções `ListenAddress`, `AllowUsers`, `DenyUsers`, `AllowGroups`, `DenyGroups` e `PermitRootLogin` do arquivo de configuração `sshd_config` (veja 6.8.5 Exemplo de sshd_config com explicações das diretivas) e via tcpd (arquivos `hosts.allow` e `hosts.deny`).

6.8.3 Usando autenticação RSA - chave pública/privada

Este método de autenticação permite a criação de um par de chaves: uma pública (que será distribuído nas máquinas que você conecta) e outra privada (que ficará em seu diretório pessoal). A encriptação e decriptação são feitas usando chaves separadas e não é possível conseguir a chave de decriptação usando a chave de encriptação. É possível inclusive gerar uma chave sem senha para entrar diretamente em um sistema remoto (este esquema é um pouco mais seguro que os arquivos `~/.rhosts` e `~/.shosts`), mas deverá ser levado em consideração a possibilidade de acesso físico ao seu diretório pessoal, qualquer um que tenha posse de sua chave privada poderá ter acesso ao sistema remoto.

Siga os seguintes passos para se autenticar usando RSA 1 - usada na versão 1 do ssh:

1. Gere um par de chaves pública/privada usando o comando:

```
ssh-keygen
```

Um par de chaves RSA versão 1 será gerado com o tamanho de 1024 bits por padrão que garante uma boa segurança/velocidade e salvas no diretório `~/.ssh` com o nome identity e identity.pub. Para alterar o tamanho da chave use a opção *-b tamanho*. Depois de gerar a chave, o ssh-keygen pedirá uma frase-senha (é recomendável ter um tamanho maior que 10 caracteres e podem ser incluídos espaços). Se não quiser digitar uma senha para acesso

ao sistema remoto, tecle <Enter> quando perguntado. Mude as permissões do diretório `~/.ssh` para 750.

A opção *-f* especifica o diretório e nome das chaves. A chave pública terá a extensão .pub adicionada ao nome especificado.

ATENÇÃO Nunca distribua sua chave privada, nem armazene-a em servidores de acesso públicos ou outros métodos que permitem outros terem acesso a ela. Se precisar de uma cópia de segurança, faça em disquetes e guarde-a em um lugar seguro.

2. Instale a chave pública no servidor remoto que deseja se conectar, por exemplo, www.sshserver.org:

```
ssh-copy-id -i ~/.ssh/identity gleydson@www.sshserver.org
```

A função do utilitário acima é entrar no sistema remoto e adicionar a chave pública local ~/.ssh/identity.pub no arquivo /home/gleydson/.ssh/authorized_keys do sistema remoto www.sshserver.org. O mesmo processo poderá ser feito manualmente usando os métodos tradicionais (ssh/scp). Caso o arquivo remoto /home/gleydson/.ssh/authorized_keys não existe, ele será criado. Seu formato é idêntico ao ~/.ssh/know_hosts e contém uma chave pública por linha.

3. Agora utilize o ssh para entrar no sistema remoto usando o método de chave pública/privada. Entre com a senha que usou para gerar o par de chaves público/privado (ele entrará diretamente caso não tenha digitado uma senha).

Para autenticar em uma versão 2 do ssh (usando chave RSA 2 ou DSA):

1. Gere um par de chaves pública/privada usando o comando:

```
ssh-keygen -t rsa -f ~/.ssh/id_rsa
```

ou

```
ssh-keygen -t dsa -f ~/.ssh/id_rsa
```

Um par de chaves RSA 2/DSA será gerado. Para alterar o tamanho da chave use a opção *-b tamanho*. Depois de gerar a chave, o ssh-keygen pedirá uma frase-senha (é recomendável ter um tamanho maior que 10 caracteres e podem ser incluídos espaços). Se não quiser digitar uma senha para acesso ao sistema remoto, tecle <Enter> quando perguntado. Mude as permissões do `diretório ~/.ssh` para 750.

ATENÇÃO Nunca distribua sua chave privada, nem armazene-a em servidores de acesso públicos ou outros métodos que permitem outros terem acesso a ela. Se precisar de uma cópia de segurança, faça em disquetes e guarde-a em um lugar seguro.

2. Instale a chave pública no servidor remoto que deseja se conectar copiando o arquivo com:

```
scp ~/.ssh/id_rsa.pub usuario@servidorremoto:~/.ssh/authorized_keys2
```

Caso o arquivo remoto /home/gleydson/.ssh/authorized_keys2 não existe, ele será criado. Seu formato é idêntico ao ~/.ssh/know_hosts2 e contém uma chave pública por linha.

3. Agora utilize o ssh para entrar no sistema remoto usando o método de chave pública/privada. Entre com a senha que usou para gerar o par de chaves público/privado (ele entrará diretamente caso não tenha digitado uma senha).

O tipo de chave criada por padrão é a *rsa1* (compatível com as versões 1 e 2 do ssh). A opção *-t [chave]* poderá ser usada (ao gerar a chave) para selecionar o método de criptografia:

- rsa1 - Cria uma chave rsa compatível com a versão 1 e 2 do ssh (esta é a padrão).
- *rsa* - Cria uma chave rsa compatível somente com a versão 2 do ssh.
- *dsa* - Cria uma chave dsa compatível somente com a versão 2 do ssh.

Para trocar a senha utilize o comando:

```
ssh-keygen -p -t rsa -f ~/.ssh/identity –
```

Será pedida sua senha antiga e a nova (no mesmo estilo do passwd). Opcionalmente você pode utilizar a sintaxe:

```
ssh-keygen -p -f ~/.ssh/identity -P senha_antiga -N senha_nova
```

Que troca a senha em um único comando (útil para ser usado em scripts junto com a opção *-q* para evitar a exibição de mensagens de saída do ssh-keygen).

6.8.4 Diferenças nas versões do protocolo

Retirada da página de manual do sshd:

Protocolo SSH versão 1

Cada servidor possui uma chave RSA específica (1024 bits por padrão) usada para identifica-lo. Quando o sshd inicia, ele gera uma chave RSA do servidor (768 bits por padrão, valor definido por ServerKeyBits) que é recriada a cada hora (modificado por KeyRegenerationInterval no sshd_config) e permanece sempre residente na RAM.

Quando um cliente se conecta o sshd responde com sua chave pública da máquina e chaves do servidor. O cliente ssh compara a chave RSA com seu banco de dados (em ~/.ssh/know_hosts) para verificar se não foi modificada.

Estando tudo OK, o cliente gera um número aleatório de 256 bits, o encripta usando ambas as chaves de máquina e chave do servidor e envia este número ao servidor. Ambos os lados então usam este número aleatório como chave de seção que é usado para encriptar todas as comunicações seguintes na seção.

O resto da seção usa um método de embaralhamento de dados convencional, atualmente Blowfish ou 3DES (usado como padrão). O cliente

seleciona o algoritmo de criptografia que será usado de um destes oferecidos pelo servidor. Após isto o servidor e cliente entram em um diálogo de autenticação. O cliente tenta se autenticar usando um dos seguintes métodos de autenticação:

- ~/.rhosts ou ~/.shosts (normalmente desativada).
- ~/.rhosts ou ~/.shosts combinado com autenticação RSA (normalmente desativada).
- Autenticação RSA por resposta de desafio.
- Autenticação baseada em senha.

A autenticação usando Rhosts normalmente é desativada por ser muito insegura mas pode ser ativada no arquivo de configuração do servidor se realmente necessário. A segurança do sistema não é melhorada a não ser que os serviços rshd, rlogind, rexecd e rexd estejam desativados (assim, o rlogin e rsh serão completamente desativados na máquina).

Protocolo SSH versão 2

A versão 2 funciona de forma parecida com a 1: Cada máquina possui uma chave DSA específica usada para se identificar. A diferença é que quando o sshd inicia, ele não gera uma chave de servidor. A segurança de redirecionamento é oferecida através da concordância do uso de uma chave Diffie-Hellman. Esta concordância de chave resulta em uma seção com chave compartilhada. O resto da seção é encriptada usando um algoritmo simétrico, como Blowfish, 3DES, CAST128, Arcfour, 128 bit AES, ou 256 bit AES.

O cliente que seleciona o algoritmo de criptografia que será usado entre os oferecidos pelo servidor. Adicionalmente a integridade da seção é oferecida através de um código de autenticação de mensagem criptográfica (hmac-sha1 ou hmac-md5). A versão 2 do protocolo oferece um método de autenticação baseado em chave pública (PubkeyAuthentication) e o método de autenticação convencional usando senhas.

6.8.5 Exemplo de sshd_config com explicações das diretivas

Abaixo segue um exemplo deste arquivo que poderá ser adaptado ao seu sistema. O objetivo é ser ao mesmo tempo útil para sua configuração e didático:

```
# Modelo personalizado para o guia Foca GNU/Linux baseado na configuração
# original do FreeBSD.
# Autor: Gleydson Mazioli da Silva
# Data: 20/09/2001.

# Porta padrão usada pelo servidor sshd.  Múltiplas portas  podem ser
# especificadas separadas por espaços.
Port 22

# Especifica o endereço IP das interfaces de rede que o servidor sshd
# servirá requisições.  Múltiplos endereços podem ser especificados
# separados por espaços.  A opção Port deve vir antes desta opção
ListenAddress 0.0.0.0

# Protocolos aceitos pelo servidor, primeiro será verificado se o cliente é
```

```
# compatível com a versão 2 e depois a versão 1.  Caso seja especificado
# somente a versão 2 e o cliente seja versão 1, a  conexão será descartada.
# Quando não é especificada, o protocolo ssh 1 é usado como padrão.
Protocol 2,1

# As 4 opções abaixo controlam o acesso de usuários/grupos no sistema.
# Por padrão o acesso a todos é garantido (exceto o acesso root se
# PermitRootLogin for "no").  AllowUsers e AllowGroups definem uma lista
# de usuários/grupos que poderão ter acesso ao sistema.  Os coringas
# "*" e "?" podem ser especificados.  Note que somente NOMES são válidos,
# UID e GID não podem ser especificados.
#
# As diretivas Allow são processadas primeiro e depois Deny.  O método que
# estas diretivas são processadas é idêntico a diretiva
# "Order mutual-failure" do controle de acesso do Apache:
# O usuário deverá TER acesso via AllowUsers e AllowGroups e NÃO ser bloqueado
# por DenyUsers e DenyGroups para ter acesso ao sistema.  Se uma das diretivas
# não for especificada, "*" é assumido como padrão.
# Estas permissões são checadas após a autenticação do usuário, porque
# dados a ele pelo /etc/passwd e PAM são obtidos após o processo de
# autenticação.
#AllowUsers gleydson teste?
#DenyUsers root adm
#AllowGroups users
#DenyGroups root adm bin

# Permite (yes) ou não (no) o login do usuário root
PermitRootLogin no

# Chaves privadas do servidor (as chaves públicas possuem um ".pub" adicionado
# no final do arquivo.
HostKey /etc/ssh/ssh_host_key
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

# Tamanho da chave.  768 bits é o padrão
ServerKeyBits 768

# Tempo máximo para login no sistema antes da conexão ser fechada
LoginGraceTime 600

# Tempo para geração de nova chave do servidor (segundos).  O padrão é
# 3600 segundos (1 hora).
KeyRegenerationInterval 3600

# Ignora os arquivos ~/.rhosts e ~/.shosts
IgnoreRhosts yes

# Ignora (yes) ou não (no) os arquivos ~/.ssh/known_hosts quando for usado
```

```
# para a opção RhostsRSAAuthentication.  Se você não confia neste mecanismo
# ajuste esta opção para yes.
IgnoreUserKnownHosts no

# Checa por permissões de dono dos arquivos e diretório de usuá rio antes de
# fazer o login.  É muito recomendável para evitar riscos de segurança
# com arquivos lidos por todos os usuários.
StrictModes yes

# Permite (yes) ou não (no) o redirecionamento de conexões X11.  A segurança
# do sistema não é aumentada com a  desativação desta opção, outros métodos
# de redirecionamento podem ser usados
X11Forwarding yes

# Especifica o número do primeiro display que será usado para o redirecionamento
# X11 do ssh.  Por padrão é usado o display 10 como inicial para evitar conf lito
# com display X locais
X11DisplayOffset 10

# Mostra (yes) ou não (no) a mensagem em /etc/motd no login.  O padrão é "no".
PrintMotd no

# Mostra (yes) ou não (no) a mensagem de último login do usuário.  O padrão é
"no".
PrintLastLog no

# Permite (yes) ou não (no) o envio de pacotes keepalive (para verificar se o
# cliente responde.  Isto é bom para fechar conexões que não respondem mas
# também podem fechar conexões caso não existam rotas para o cliente
# naquele momento (é um problema temporário).  Colocando esta opção como
# "no" por outro lado pode deixar usuários que não tiveram a oportunidade
# de efetuar o logout do servidor dados como "permanentemente conectados"
# no sistema.  Esta opção deve ser ativada/desativada aqui e no programa
# cliente para funcionar.
KeepAlive yes

# Facilidade e nível das mensagens do sshd que aparecerão no syslogd
SyslogFacility AUTH
LogLevel INFO

# Especifica se somente a autenticação via arquivos ~/.rhosts e /etc/hosts.equiv
é
# suficiente para entrar no sistema.  Não é muito bom usar "yes" aqui.
RhostsAuthentication no

# Mesmo que o acima com o acréscimo que o arquivo /etc/ssh/ssh_known_hosts também
# é verificado.  Também evite usar "yes" aqui.
RhostsRSAAuthentication no
```

```
# Especifica se a autenticação via RSA é permitida (só usado na versão 1 do
# protocolo ssh).  Por padrão "yes".
RSAAuthentication yes

# Permite autenticação usando senhas (serve para ambas as versões 1 e 2 do ssh).
# O padrão é "yes".
PasswordAuthentication yes

# Se a PasswordAuthentication for usada, permite (yes) ou não (no) login
# sem senha.  O padrão é "no".
PermitEmptyPasswords no

# Ativa senhas s/key ou autenticação PAM NB interativa.  Nenhum destes é
# compilado por padrão junto com o sshd.  Leia a página de manual do
# sshd antes de ativar esta opção em um sistema que usa PAM.
ChallengeResponseAuthentication no

# Verifica se o usuário possui emails ao entrar no sistema.  O padrão é "no".
# Este módulo também pode estar sendo habilitado usando PAM (neste caso
# cheque a configuração em /etc/pam.d/ssh).
CheckMail no

# Especifica se o programa login é usado para controlar a seções de shell
# interativo.  O padrão é "no".
UseLogin no

# Especifica o número máximo de conexões de autenticação simultâneas feitas
# pelo daemon sshd.  O valor padrão é 10.  Valores aleatórios podem ser
# especificados usando os campos "inicio:taxa:máximo".  Por exemplo,
# 5:40:15 rejeita até 40% das tentativas de autenticação que excedam o
# limite de 5 até atingir o limite máximo de 15  conexõe s, quando
# nenhuma nova autenticação é permitida.
MaxStartups 10
#MaxStartups 10:30:60

# Mostra uma mensagem antes do nome de usuário/senha
Banner /etc/issue.net

# Especifica se o servidor sshd fará um DNS reverso para verificar se o
# endereço confere com a origem (isto é útil para bloquear conexões
# falsificadas - spoofing).  O padrão é "no".
ReverseMappingCheck yes

# Ativa o subsistema de ftp seguro.  Para desabilitar comente a linha
# abaixo
Subsystem     sftp   /usr/lib/sftp-server
```

7 SERVIÇOS DE REDE

Adaptado do Guia Foca GNU/Linux Avançado – Capítulo 4.

Serviço de rede é o que está disponível para ser acessado pelo usuário. No TCP/IP, cada serviço é associado a um número chamado *porta* que é onde o servidor espera pelas conexões dos computadores clientes. Uma porta de rede pode se referenciada tanto pelo número como pelo nome do serviço.

Abaixo, alguns exemplos de portas padrões usadas em serviços TCP/IP:

- 21 - FTP (transferência de arquivos)
- 23 - telnet (terminal virtual remoto)
- 25 - SMTP (envio de e-mails)
- 53 - DNS (resolvedor de nomes)
- 79 - finger (detalhes sobre usuários do sistema)
- 80 - HTTP (protocolo www - transferência de páginas Internet)
- 110 - pop-3 (recebimento de mensagens)
- 119 – nntp (usado por programas de noticias)

O arquivo padrão responsável pelo mapeamento do nome dos serviços e das portas mais utilizadas é o */etc/services* (para detalhes sobre o seu formato, veja a seção 3.6.1).

7.1 Serviços iniciados como *daemons* de rede

Serviços de rede iniciados como *daemons* ficam residentes o tempo todo na memória, esperando que alguém se conecte (também chamado de *modo standalone*). Um exemplo de *daemon* é o servidor proxy `squid` e o servidor web `apache` operando no modo *daemon*.

Alguns programas servidores oferecem a opção de serem executados como *daemons* ou através do `inetd`. É recomendável escolher *daemon* se o serviço for solicitado freqüentemente (como é o caso dos servidores web ou proxy).

Para verificar se um programa está rodando como *daemon*, basta digitar ps ax e procurar o nome do programa, em caso positivo ele é um *daemon*.

Normalmente os programas que são iniciados como *daemons* têm seus próprios recursos de segurança/autenticação para decidir quem tem ou não permissão de se conectar.

7.2 Serviços iniciados através do `inetd`

Serviços iniciados pelo `inetd` são carregados para a memória somente quando são solicitados. O controle de quais serviços podem ser carregados e seus parâmetros, é feitos através do arquivo `/etc/inetd.conf`.

Um *daemon* chamado `inetd` lê as configurações deste arquivo e permanece residente na memória, esperando pela conexão dos clientes.

Quando uma conexão é solicitada, o *daemon* `inetd` verifica as permissões de acesso nos arquivos `/etc/hosts.allow` e `/etc/hosts.deny` e carrega o programa servidor correspondente no arquivo /etc/inetd.conf. Um arquivo também importante neste processo é o /etc/services que faz o mapeamento das portas e nomes dos serviços.

Alguns programas servidores oferecem a opção de serem executados como *daemons* ou através do `inetd`. É recomendável escolher `inetd` se o serviço não for solicitado freqüentemente (como é o caso de servidores `ftp`, `telnet`, `talk`, etc).

7.2.1 /etc/inetd.conf

O arquivo /etc/inetd.conf é um arquivo de configuração para o daemon servidor *inetd*. Sua função é dizer ao inetd o que fazer quando receber uma requisição de conexão para um serviço em particular. Para cada serviço que deseja aceitar conexões, você precisa dizer ao *inetd* qual daemon servidor executar e como executa-lo.

Seu formato é também muito simples. É um arquivo texto com cada linha descrevendo um serviço que deseja oferecer. Qualquer texto em uma linha seguindo uma "#" é ignorada e considerada um comentário. Cada linha contém sete campos separados por qualquer número de espaços em branco (tab ou espaços). O formato geral é o seguinte:

`serviço tipo_soquete proto opções usuário caminho_serv. opções_serv.`

serviço

É o serviço relevante a este arquivo de configuração pego do arquivo /etc/services.

tipo_soquete

Este campo descreve o tipo do soquete que este item utilizará, valores permitidos são: stream, dgram, raw, rdm, ou seqpacket. Isto é um pouco técnico de natureza, mas como uma regra geral, todos os serviços baseados em *tcp* usam stream e todos os protocolos baseados em *udp* usam dgram. Somente alguns tipos de daemons especiais de servidores usam os outros valores.

`protocolo`

O protocolo é considerado válido para esta item. Isto deve bater com um item apropriado no arquivo /etc/services e tipicamente será tcp ou udp. Servidores baseados no Sun RPC (*Remote Procedure Call*), utilizam rpc/tcp ou rpc/udp.

`opções`

Existem somente duas configurações para este campo. A configuração deste campo diz ao *inetd* se o programa servidor de rede libera o soquete após ele ser iniciado e então se inetd pode iniciar outra cópia na próxima requisição de conexão, ou se o inetd deve aguardar e assumir que qualquer servidor já em execução pegará a nova requisição de conexão.

Este é um pequeno truque de trabalho, mas como uma regra, todos os servidores tcp devem ter este parâmetro ajustado para *nowait* e a maior parte dos servidores udp deve tê-lo ajustado para *wait*. Foi alertado que existem algumas excessões a isto, assim deixo isto como exemplo se não estiver seguro.

usuário

Este campo descreve que conta de usuário usuário no arquivo /etc/passwd será escolhida como *dono* do daemon de rede quando este for iniciado. Isto é muito útil se você deseja diminuir os riscos de segurança. Você pode ajustar o usuário de qualquer item para o usuário *nobody*, assim se a segurança do servidor de redes é quebrada, a possibilidade de problemas é minimizada. Normalmente este campo é ajustado para *root*, porque muitos servidores requerem privilégios de usuário root para funcionarem corretamente.

caminho_servidor

Este campo é o caminho para o programa servidor atual que será executado.

argumentos_servidor

Este campo inclui o resto da linha e é opcional. Você pode colocar neste campo qualquer argumento da linha de comando que deseje passar para o daemon servidor quando for iniciado.

Uma dica que pode aumentar significativamente a segurança de seu sistema é comentar (colocar uma #no inicio da linha) os serviços que não serão utilizados.

Abaixo um modelo de arquivo /etc/inetd.conf usado em sistemas Debian:

```
# /etc/inetd.conf:  veja inetd(8) para mais detalhes.
#
# Banco de Dados de configurações do servi dor Internet
#
#
# Linhas iniciando com "#:LABEL:" ou "#<off>#" não devem
# ser alteradas a não ser que saiba o que está fazendo!
#
#
# Os pacotes devem modificar este arquivo usando update -inetd(8)
#
# <nome_serviço> <tipo_soquete> <proto> <opções> <usuá rio> <caminho_servidor>
<args>
#
#:INTERNO: Serviços internos
#echo           stream  tcp nowait  root    internal
#echo           dgram   udp wait    root    internal
#chargen        stream  tcp nowait  root    internal
#chargen        dgram   udp wait     root    internal
#discard        stream  tcp nowait  root    internal
#discard        dgram   udp wait    root    internal
#daytime        stream  tcp nowait  root    internal
#daytime        dgram   udp wait    root    internal
time            stream  tc p nowait  root    internal
#time           dgram   udp wait    root    internal

#:PADRÕES: Estes são serviços padrões.

#:BSD: Shell, login, exec e talk são protocolos BSD.
```

```
#shell          stream  tcp nowait  root     /usr/sbin/tcpd  /usr/sbin/in.rshd
#login          stream  tcp nowait  root     /usr/sbin/tcpd  /usr/sbin/in.rlogind
#exec           stream  tcp nowait  root     /usr/sbin/tcpd  /usr/sbin/in.rexecd
talk            dgram   udp wait    nobody.tty  /usr/sbin/tcpd
/usr/sbin/in.talkd
ntalk           dgram   udp wait    nobody.tty  /usr/sbin/tcpd
/usr/sbin/in.ntalkd

#:MAIL: Mail, news e serviços uucp.
Smtp            stream  tcp nowait  mail     /usr/sbin/exim exim  -bs

#:INFO: Serviços informativos

#:BOOT: O serviço Tftp é oferecido primariamente para a inicialização.  Alguns
sites
# o executam somente em máquinas atuando como "servidores de inicialização".

#:RPC: Serviços baseados em RPC

#:HAM-RADIO: serviços de rádio amador

#:OTHER: Outros serviços
```

8 DNS

Adaptado do "DNS HowTo", por Nicolai Langfeldt
Tradução Original da Conectiva Informática - Março de 1999

8.1 Introdução ao DNS: O que é e o que não é.

Para iniciantes, DNS é o Servidor de Nomes do Domínio. O DNS converte os nomes das máquinas para números IP, que são os endereços das máquinas, mapeando de nome para endereço e de endereço para nome. Este COMO FAZER documenta como definir tais mapeamentos usando o sistema Linux. Um mapeamento é simplesmente uma associação entre duas informações, neste caso um nome de máquina, como ftp.linux.org, e o número IP da máquina, como por exemplo 199.249.150.4.

DNS é, para os não iniciados (você), uma das áreas mais opacas da administração de rede. Este COMO FAZER tentará clarificar alguns conceitos e aspectos sobre este tema. Ele descreve como configurar um nome do servidor DNS *simples*. Começando com um único servidor de cache e seguindo até a configuração de um servidor primário DNS para um domínio. Para configurações mais complexas pode-se checar a seção de Perguntas e Respostas deste documento. Caso não esteja lá descrito, pode ser necessário *ler* a documentação que acompanha os fontes. Esclareceremos em que consiste esta documentação no último capítulo.

Antes de começar, há que se configurar uma máquina para que ela possa se conectar interna e externamente e assim permitir as conexões à rede. Deve ser possível executar o comando telnet 127.0.0.1 e ter acesso à máquina local (teste agora!). É necessário ainda ter-se arquivos de exemplo /etc/nsswitch.conf (ou /etc/host.conf), /etc/resolv.conf e /etc/hosts como ponto de partida, uma vez que não explicaremos aqui a sua função. Caso ainda não se tenha tudo isso configurado e operando, o documento NET-3 ou o COMO FAZER PPP explicam como configurá-los.

Ao nos referirmos a 'máquina local', estamos referenciando à máquina na qual se está tentando configurar o DNS e não a qualquer outra máquina que se possa ter à disposição e que esteja conectada à rede.

Presumimos que esta máquina não está atrás de algum firewall que bloqueie as pesquisas de nomes. Caso seja necessária alguma configuração especial, por favor veja a seção de Perguntas e Respostas.

O serviço de nomes no Unix é feito por um programa servidor denominado named. Ele é integrante do pacote de bind que é coordenado por Paul Vixie para o Consórcio de Programas Para a Internet. O Servidor de nomes está incluído na maioria das distribuições GNU/Linux e é usualmente instalado como /usr/sbin/named. Caso se tenha um named à disposição pode-se usá-lo; caso contrário é possível obter-se um binário a partir de um site ftp Linux, ou conseguir os fontes mais recentes em ftp.isc.org:/isc/bind/src/cur/bind-8/. Este COMO FAZER trata sobre o bind em sua versão 8. A versão antiga do COMO FAZER, que tratava sobre o bind 4,

ainda está ainda disponível em http://www.math.uio.no/~janl/DNS/ no caso de se necessitar utilizar o bind 4. Caso a página do manual sobre o servidor de nomes fale sobre named.conf então tem-se disponível o bind 8, caso mencione o named.boot então trata-se do bind 4. Caso se tenha o 4 e se esteja com problemas de segurança, deve-se atualizar para a versão 8 mais recente.

O DNS é um banco de dados distribuído por toda a rede. É necessário ter-se extremo cuidado com tudo o que for colocado nele. Ao se colocar dados sem significado, outros utilizarão estes dados e certamente tudo ficará um pouco "estranho". O DNS deve estar sempre atualizado e arrumado, evitando-se assim problemas desagradáveis. Deve-se aprender a usá-lo, administrá-lo, depurá-lo para tornar-se um bom administrador da rede, evitando sobrecargas geradas por problemas de administração.

Neste documento são afirmadas categoricamente algumas coisas que não são completamente verdadeiras (sendo então pelo menos meias verdades). Tudo em nome da simplificação. As coisas (provavelmente!) funcionarão quando o leitor acreditar no que está dito!

Dica: Devem ser feitas cópias de segurança de todos os arquivos. É aconselhável, ainda, que elas sejam alteradas de tempos em tempos. Assim se depois de todas as tentativas, nada funcionar, pode-se retornar à situação anterior.

8.2 Um Servidor de Nomes Somente Para Cache.

Um servidor de nomes somente para cache deve ser capaz de encontrar as respostas às pesquisas de nomes e endereços e deve ainda guardar as respostas, para a próxima em que sejam necessárias. Isto diminuirá o tempo de espera significativamente, especialmente quando se tem uma conexão lenta.

Inicialmente é necessário ter-se um arquivo /etc/named.conf, o qual será lido quando o servidor de nomes for inicializado. Por enquanto ele pode conter simplesmente:

```
// Configuração  do arquivo para um servidor de nomes somente para cache

options{
        directory "/var/named";

        // Não comentar isto pode ajudar caso se tenha um firewall presente
        // e as coisas não estejam funcionando:

        // endereço de pesquisa: porta 53;
};

zone "." {
        type hint;
        file "rott.hints ";
};

zone "0.0.127.in-addr.arpa" {
        type master;
```

```
        file "pz/127.0.0";
};
```

A linha `directory' indica onde os arquivos devem estar localizados. Todos os arquivos subseqüentes serão relativos a este. Assim pz é um diretório sob /var/named, ou seja estará localizado em /var/named/pz. /var/named é o diretório definido pelo *Padrão de Sistemas de Arquivos Linux*.

O arquivo denominado /var/named/root.hints deve conter:

```
.                       6D IN NS        G.ROOT -SERVERS.NET.
.                       6D IN NS        J.ROOT -SERVERS.NET.
.                       6D IN NS        K.ROOT -SERVERS.NET.
.                       6D IN NS        L.ROOT -SERVERS.NET.
.                       6D IN NS        M.ROOT -SERVERS.NET.
.                       6D IN NS        A.ROOT -SERVERS.NET.
.                       6D IN NS        H.ROOT -SERVERS.NET.
.                       6D IN NS        B.ROOT -SERVERS.NET.
.                       6D IN NS        C.ROOT -SERVERS.NET.
.                       6D IN NS        D.ROOT -SERVERS.NET.
.                       6D IN NS        E.ROOT -SERVERS.NET.
.                       6D IN NS        I.ROOT -SERVERS.NET.
.                       6D IN NS        F.ROOT -SERVERS.NET.

G.ROOT-SERVERS.NET.     5w6d16h IN A    192.112.36.4
J.ROOT-SERVERS.NET.     5w6d16h IN A    198.41.0.10
K.ROOT-SERVERS.NET.     5w6d16h IN A    193.0.14.129
L.ROOT-SERVERS.NET.     5w6d16h IN A    198.32.64.12
M.ROOT-SERVERS.NET.     5w6d16h IN A    202.12.27.33
A.ROOT-SERVERS.NET.     5w6d16h IN A    198.41.0.4
H.ROOT-SERVERS.NET.     5w6d16h IN A    128.63.2.53
B.ROOT-SERVERS.NET.     5w6d16h IN A    128.9.0.107
C.ROOT-SERVERS.NET.     5w6d16h IN A    192.33.4.12
D.ROOT-SERVERS.NET.     5w6d16h IN A    128.8.10.90
E.ROOT-SERVERS.NET.     5w6d16h IN A    192.203.230.10
I.ROOT-SERVERS.NET.     5w6d16h IN A    192.36.148.17
F.ROOT-SERVERS.NET.     5w6d16h IN A    192.5.5.241
```

Este arquivo descreve o nome dos servidores raiz no mundo. Este conteúdo pode mudar com o passar do tempo e *tem que* ser atualizado permanentemente. Veja a seção de manutenção para saber como mantê-lo atualizado.

A próxima seção em named.conf é zone. Explicaremos o seu uso num capítulo adiante. Por hora somente façamos deste um arquivo chamado 127.0.0 no subdiretório pz:

```
@               IN      SOA     ns.linux.bogus.hostmaster.linux.bogus.   (
                                1     ; Serial
                                8H      ; Atualização
                                2H    ; Tentativas
                                 1W      ; Expiração
                                1D)     ; TTL mínimo
                NS      ns.linux.bogus.
```

```
1               PTR     localhost
```

Em seguida, será necessário um arquivo /etc/resolv.conf com o seguinte conteúdo:

```
search subdomínio.seu\_domínio.edu.br seu\_domínio.edu.br
nome\_do\_servidor 127.0.0.1
```

A linha `search' especifica que o domínio deve ser pesquisado para qualquer nome de máquina com a qual se queira conectar. A linha `nameserver' especifica o endereço do servidor de nomes. Neste caso, a própria máquina, uma vez que é nela que o programa named é executado (já que 127.0.0.1 foi informado, não importando se a máquina tem também outro endereço). Caso se queira indicar vários servidores de nomes, deve-se criar uma linha `nameserver' para cada um deles. (Nota: O programa named nunca lê este arquivo, e sim o resolvedor que utilizar o named).

Vamos ilustrar um pouco mais a função deste arquivo: caso um cliente tente procurar por itamaraca, então itamaraca.subdomínio.seu_domínio.edu.br será a primeira tentativa, então será tentado itamaraca.seu_domínio.edu.br e finalmente somente itamaraca. Se um cliente tentar procurar metalab.unc.edu, metalab.unc.edu.subdomínio.seu_domínio.edu.br será tentado inicialmente (sim, não faz muito sentido, mas é o jeito que ele funciona), então metalab.unc.edu.seu_domínio.edu.br, e finalmente metalab.unc.edu. Caso se queira colocar muitos domínios na linha search, isso pode provocar uma sobrecarga nos tempos de pesquisa.

O exemplo presume que a máquina pertence ao domínio subdomínio.seu_domínio.edu.br, sendo provavelmente o servidor de nomes nome_da_máquina.subdomínio.seu_domínio.edu.br. A linha de busca não deve conter o TLD (Domínio Raiz `edu.br' neste caso). Caso seja necessário conectar-se com freqüência a máquinas de outros domínios, deve-se acrescentar aqueles domínios à linha de busca, como por exemplo:

```
search subdomínio.seu_domínio.edu.br seu_domínio.edu.br outro_domínio.com.br
```

e assim por diante. Obviamente deve-se utilizar nomes reais de domínios. Os aqui colocados servem somente como exemplos. Por favor atente para a falta de pontos no final dos nomes dos domínios.

A seguir, dependendo da versão da biblioteca libc, tanto pode ser necessário atualizar o /etc/nsswitch.conf ou o /etc/host.conf. Caso se tenha o nsswitch.conf este será utilizado, caso contrário, atualizaremos o host.conf.

/etc/nsswitch.conf

Este é um arquivo longo que especifica onde podem ser obtidos diferentes tipos de dados, de que arquivos e de qual base de dados. Usualmente contém comentários úteis no topo, que podem ser lidos agora. Depois disso, deve ser encontrada uma linha que comece com `hosts:', onde se pode ler:

```
hosts:      files dns
```

Caso não haja nenhuma linha iniciada com `hosts:' então deve ser incluída a linha acima. Ela indica que os programas devem primeiramente pesquisar o arquivo /etc/hosts, e após então verificar o DNS de acordo com o configurado no arquivo resolv.conf.

/etc/host.conf

Provavelmente contém várias linhas, uma delas deve começar com order e deve ter o seguinte:

```
order hosts, bind
```

Se não houver nenhuma linha `order', então uma deve ser criada. Esta linha indica que a resolução de nomes de máquinas deve pesquisar inicialmente no arquivo /etc/hosts, e depois pesquisar junto ao servidor de nomes (definido em resolv.conf como 127.0.0.1). Estes dois últimos arquivos estão documentados na página de manual do utilitário resolver(8) (para acessá-la execute man 8 resolv) na maioria das distribuições Linux. Aquela página do manual é clara e em nossa opinião, todos devem lê-la (especialmente os administradores de DNS). Faça-o agora caso você seja um daqueles que diz para si mesmo: "Eu vou ler mais tarde", mas nunca o faz.

8.2.1 Iniciando o named

Após tudo isto é hora de iniciar o servidor de nomes. Caso se esteja usando uma conexão discada, primeiro deve-se estabelecer a conexão. Deve-se digitar então `ndc start', sem opções. Caso isto não funcione, pode-se tentar `/usr/sbin/ndc start'. Caso isto não funcione, deve-se verificar a seção Perguntas e Respostas. Agora é possível testar a configuração. Ao se visualizar o arquivo de mensagens syslog (usualmente chamado /var/adm/messages; podem ser examinados também o diretório /var/log e o arquivo syslog) ao se iniciar o servidor de nomes (executando-se tail -f/var/log/messages) deve-se obter algo como:

(linhas terminadas em \ continuam na linha seguinte)

```
Feb 15 01:26:17 roke named[6091]: starting.  named 8.1.1 Sat Feb 14  \
  00:18:20 MET 1998 ^Ijanl@roke.uio.no:/var/tmp/bind -8.1.1/src/bin/named
Feb 15 01:26:17 roke named[6091]: cache zone "" (IN) loaded (serial 0)
Feb 15 01:26:17 roke named[6091]: master zone "0.0.127.in -addr.arpa" \
  (IN) loaded (serial 1)
Feb 15 01:26:17 roke named[6091]: listening [127.0.0.1].53 (lo)
Feb 15 01:26:17 roke named[6091]: listening [129.240.230.92].53 (ippp0)
Feb 15 01:26:17 roke named[6091]: Forwarding source address is [0.0.0.0].1040
Feb 15 01:26:17 roke named[6092]: Ready to answer queries.
```

Se houver alguma mensagem de erro, ela deve ser examinada. O named indicará o arquivo onde o problema se encontra (ou named.conf. ou root.hints, esperamos). O servidor de nomes deve ser finalizado e os arquivos devem ser corrigidos.

Agora é hora de iniciar o nslookup para examinar o trabalho realizado até aqui.

```
$ nslookup
default Server:  localhost
Address:  127.0.0.1

>
```

Caso este seja o resultado obtido, parabéns, está funcionando. Esperamos que sim. Caso se obtenha um resultado diferente, deve-se retornar e verificar todos os passos. Cada vez que se altere o arquivo named.conf será necessário reiniciar o servidor de nomes usando o comando ndc restart.

Agora podemos fazer pesquisas no sistema. Podemos procurar por alguma máquina próxima; A pat.uio.no está próxima a mim na Universidade de Oslo:

```
> pat.uio.no
Server:  localhost
Address:  127.0.0.1

Name:    pat.uio.no
Address:  129.240.130.16
```

O nslookup agora perguntou ao seu servidor de nomes para procurar a máquina pat.uio.no. Este contactou uma dos servidores de nomes listados no arquivo root.hints, e perguntou a um deles qual o caminho para a máquina desejada. Pode levar bem pouco tempo antes de se obter o resultado, enquanto named procura todos os domínios definidos em /etc/resolv.conf.

Ao se pesquisar novamente, tem-se:

```
> pat.uio.no
Server:  localhost
Address:  127.0.0.1

Non-authoritative answer:
Name:    pat.uio.no
Address:  129.240.2.50
```

Note a linha `Non-authoritative answer:' que obtivemos desta vez. Isto indica que o servidor de nomes não saiu pela rede para perguntar sobre a máquina desejada. Ao invés disto procurou em seu cache e encontrou-o lá. Mas a informação do cache *pode* estar desatualizada (antiga). Então se está informado deste perigo (muito pequeno) quando o sistema informa `resposta Não autorizada:'. Quando o nslookup disser isto pela segunda vez para a mesma máquina, pode-se estar certo de que o cache está funcionando e fornecendo a informação certa. Pode sair-se do comando nslookup digitando-se `exit'.

Agora que sabemos como configurar um servidor de nomes de cache, aproveite para tomar uma cerveja, leite, ou qualquer coisa que se queira para comemorar este fato memorável.

8.3 Um Domínio *Simples*: como configurar um domínio próprio

8.3.1 Um pouco de teoria

Antes de *realmente* começarmos esta seção, forneceremos alguns ensinamentos sobre o funcionamento do DNS; é preciso lê-los porque é fundamental para um administrador de rede. Caso não se queira, deve-se pelo menos pesquisá-los rapidamente, até chegar aonde se quer ir no arquivo named.conf.

DNS é uma sistema hierárquico. O mais alto nível é representado por `.' e denominado "raiz". Sob "." há diversos Domínios de Alto Nível (TLDs), sendo os mais conhecidos ORG, COM, EDU e NET, mas existem muitos mais.

Ao se procurar uma máquina, a pesquisa ocorre recursivamente dentro da hierarquia, começando no topo. Caso se queira descobrir o endereço de

prep.ai.mit.edu, o servidor de nomes local tem que encontrar um nome de servidor que responda pelo domínio edu. Ele pergunta a um servidor . (ele já conhece os servidores ., a partir do arquivo root.hints), e o servidor . fornecerá uma lista dos servidores do domínio edu:

```
$ nslookup
Default Server: localhost
Address:  127.0.0.1
```

Começaremos perguntando por um servidor raiz:

```
> server c.root -servers.net.
Default Server:  c.root  -servers.net
Address:  192.33.4.12
```

A seguir definiremos o tipo de pesquisa que desejamos fazer. Neste caso NS (registros de servidores de nomes):

```
> set q=ns
```

A seguir perguntaremos pelos servidores que respondem pelo domínio edu:

```
> edu.
```

O ponto após edu é significativo. Ele indica ao servidor que estamos pesquisando os servidores sob os quais o domínio edu está configurado (isto de alguma maneira simplifica a busca):

```
edu      nome do servidor = A.ROOT-SERVERS.NET
edu      nome do servidor = H.ROOT-SERVERS.NET
edu      nome do servidor = B.ROOT-SERVERS.NET
edu      nome do servidor  = C.ROOT-SERVERS.NET
edu      nome do servidor = D.ROOT-SERVERS.NET
edu      nome do servidor = E.ROOT-SERVERS.NET
edu      nome do servidor =  I.ROOT-SERVERS.NET
edu      nome do servidor = F.ROOT-SERVERS.NET
edu      nome do servidor = G.ROOT-SERVERS.NET
A.ROOT-SERVERS.NET      endereço na internet = 198.41.0.4
H.ROOT-SERVERS.NET      endereço na internet = 128.63.2.53
B.ROOT-SERVERS.NET      endereço na internet = 128.9.0.107
C.ROOT-SERVERS.NET      endereço na internet = 192.33.4.12
D.ROOT-SERVERS.NET      endereço na internet = 128.8.10.90
E.ROOT-SERVERS.NET      endereço na internet = 192.203.230.10
I.ROOT-SERVERS.NET      endereço na internet  = 192.36.148.17
F.ROOT-SERVERS.NET      endereço na internet = 192.5.5.241
G.ROOT-SERVERS.NET      endereço na internet = 192.112.36.4
```

A resposta nos indica que *.root-servers.net serve edu., podemos então continuar perguntando, por exemplo ao servidor C.ROOT-SERVERS.NET. Agora queremos saber quem serve o próximo nível do nome da máquina: mit.edu.:

```
> mit.edu.
Server:  c.root-servers.net
Address:  192.33.4.12

Non-authoritative answer:
mit.edu nameserver = W20NS.mit.edu
```

```
mit.edu nameserver = BITSY.mit.edu
mit.edu nameserver = STRAWB.mit.edu

Authoritative answers can be found from:
W20NS.mit.edu   internet address = 18.70.0.160
BITSY.mit.edu   internet address = 18.72.0.3
STRAWB.mit.edu  internet address = 18.71.0.151
```

A resposta indica que strawb, w20ns e bitsy servem o domínio mit. Vamos selecionar um deles e perguntar-lhe sobre ai.mit.edu:

```
> servidor W20NS.mit.edu.
```

Os nomes das máquinas não são sensíveis a maiúsculas e minúsculas, mas sugerimos o uso do mouse para cortar e colar como estão na tela.

```
Servidor:  W20NS.mit.edu
Endereço:  18.70.0.160
> ai.mit.edu.
Server:  W20NS.mit.edu
Address:  18.70.0.160

Non-authoritative answer:
ai.mit.edu      nameserver = ALPHA-BITS.AI.MIT.EDU
ai.mit.edu      nameserver = GRAPE-NUTS.AI.MIT.EDU
ai.mit.edu      nameserver = TRIX.AI.MIT.EDU
ai.mit.edu      nameserver = MUESLI.AI.MIT.EDU
ai.mit.edu      nameserver = LIFE.AI.MIT.EDU
ai.mit.edu      nameserver = BEET-CHEX.AI.MIT.EDU
ai.mit.edu      nameserver = MINI-WHEATS.AI.MIT.EDU
ai.mit.edu      nameserver = COUNT-CHOCULA.AI.MIT.EDU
ai.mit.edu      nameserver = MINTAKA.LCS.MIT.EDU

Authoritative answers can be found from:
AI.MIT.EDU      nameserver = ALPHA-BITS.AI.MIT.EDU
AI.MIT.EDU      nameserver = GRAPE-NUTS.AI.MIT.EDU
AI.MIT.EDU      nameserver = TRIX.AI.MIT.EDU
AI.MIT.EDU      nameserver = MUESLI.AI.MIT.EDU
AI.MIT.EDU      nameserver = LIFE.AI.MIT.EDU
AI.MIT.EDU      nameserver = BEET-CHEX.AI.MIT.EDU
AI.MIT.EDU      nameserver = MINI-WHEATS.AI.MIT.EDU
AI.MIT.EDU      nameserver = COUNT-CHOCULA.AI.MIT.EDU
AI.MIT.EDU      nameserver = MINTAKA.LCS.MIT.EDU
ALPHA-BITS.AI.MIT.EDU     internet address = 128.52.32.5
GRAPE-NUTS.AI.MIT.EDU     internet address = 128.52.36.4
TRIX.AI.MIT.EDU                   internet address = 128.52.37.6
MUESLI.AI.MIT.EDU         internet address = 128.52.39.7
LIFE.AI.MIT.EDU                   internet address = 128.52.32.80
BEET-CHEX.AI.MIT.EDU      internet address = 128.52.32.22
MINI-WHEATS.AI.MIT.EDU    internet address = 128.52.54.11
COUNT-CHOCULA.AI.MIT.EDU  internet address = 128.52.38.22
MINTAKA.LCS.MIT.EDU       internet address = 18.26.0.36
```

Desta forma, obtemos que museli.ai.mit.edu é um dos servidores de nomes de ai.mit.edu:

```
> server MUESLI.AI.MIT.EDU
Default Server:  MUESLI.AI.MIT.EDU
Address:  128.52.39.7
```

Agora mudaremos o tipo de pergunta. Já que encontramos o servidor de nomes, agora podemos perguntar tudo o que quisermos sobre prep.ai.mit.edu.

```
> set q=any
> prep.ai.mit.edu.
Server:  MUESLI.AI.MIT.EDU
Address:  128.52.39.7

prep.ai.mit.edu CPU = dec/decstation-5000.25    OS = unix
prep.ai.mit.edu
        inet address = 18.159.0.42, protocol = tcp
          ftp  telnet  smtp  finger
prep.ai.mit.edu preference = 1, mail exchanger = gnu-life.ai.mit.edu
prep.ai.mit.edu internet address = 18.159.0.42
ai.mit.edu      nameserver = beet-chex.ai.mit.edu
ai.mit.edu      nameserver = alpha-bits.ai.mit.edu
ai.mit.edu      nameserver = mini-wheats.ai.mit.edu
ai.mit.edu      nameserver = trix.ai.mit.edu
ai.mit.edu      nameserver = muesli.ai.mit.edu
ai.mit.edu      nameserver = count-chocula.ai.mit.edu
ai.mit.edu      nameserver = mintaka.lcs.mit.edu
ai.mit.edu      nameserver = life.ai.mit.edu
gnu-life.ai.mit.edu             internet address  = 128.52.32.60
beet-chex.ai.mit.edu            internet address = 128.52.32.22
alpha-bits.ai.mit.edu           internet address = 128.52.32.5
mini-wheats.ai.mit.edu          internet address = 128.52.54.11
trix.ai.mit.edu                         internet  address = 128.52.37.6
muesli.ai.mit.edu               internet address = 128.52.39.7
count-chocula.ai.mit.edu    internet address = 128.52.38.22
mintaka.lcs.mit.edu             internet address = 18.26.0.36
life.ai.mit.edu                         internet address = 128.52.32.80
```

Assim começando por . fomos capazes de descobrir os nomes dos servidores do próximo nível de domínio. Caso se esteja usando um servidor DNS próprio ao invés de usar todos aqueles outros servidores, o named certamente guardaria no cache todas as informações que tenha encontrado, não sendo necessária toda esta pesquisa na próxima vez que fosse realizada uma nova pesquisa de localização desta máquina.

Um tema muito menos comentado, mas também muito importante é in-addr.arpa. Ele também está aninhado como um domínio 'normal'. in-addr.arpa permite-nos conseguir os nomes das máquinas através de seus endereços. Uma coisa importante aqui, é notar que ip#s são escritos ao contrário no campo in-addr.arpa. Caso se tenha o endereço da máquina: 192.128.52.43, named procederá exatamente como no exemplo prep.ai.mit.edu: encontrar

servidores arpa., in-addr.arpa., 192.in-addr.arpa., 128.192.in-addr.arpa., 52.128.192.in-addr.arpa.. Encontrar então os registros necessários para 43.52.128.192.in-addr.arpa. Engenhoso não? (Diga `Sim', por favor!.) Porém não se preocupe endereços reversos somente são confusos nos dois primeiros anos.

Acabamos de contar uma mentira. O DNS não funciona exatamente da maneira aqui descrita. Mas não tenha dúvida que é muito próximo disso.

8.3.2 Nosso Próprio Domínio

Agora vamos definir nosso próprio campo. Vamos criar o domínio *linux.bogus* e definir suas máquinas. Usaremos o nome de domínio bogus para estarmos certos de não estarmos perturbando ninguém.

Mais uma coisa antes de começarmos: nem todos os caracteres são permitidos nos nomes das máquinas. Estamos limitados ao caracteres do alfabeto: a-z e ao números: 0-9, além do caracter '-' (hífen). Devemos nos restringir àqueles caracteres. Os caracteres maiúsculos e minúsculos são idênticos para o DNS, assim pat.uio.no é igual a Pat.UiO.No.

Começaremos esta parte com uma linha em named.conf:

```
zone "0.0.127.in-addr.arpa" {
        type master;
        file "pz/127.0.0";
};
```

Por favor note a falta de `.' no final dos nomes dos campos neste arquivo. Isto nos diz que podemos definir uma zona 0.0.127.in-addr.arpa, na qual somos os servidores principais e que as informações estão guardadas em um arquivo chamado pz/127.0.0. Nós já configuramos este arquivo anteriormente com o seguinte conteúdo:

```
@               IN      SOA     ns.linux.bogus.hostmaster.linux.bogus. (
                                1     ; Serial
                                8H        ; Atualização
                                2H    ; Tentativas
                                1W        ; Expiração
                                1D)       ; TTL mínimo
                NS      ns.linux.bogus.
1               PTR     localhost
```

Por favor note o `.' no final de todos os nomes completos de campo neste arquivo, em contraste ao arquivo acima named.conf. Algumas pessoas gostam de começar cada arquivo de zona com uma diretiva $ORIGIN, mas isto é supérfluo. A origem (onde pertence o DNS na hierarquia) de um arquivo de zona é especificado na seção de zona do arquivo named.conf, a qual neste caso é 0.0.127.in-addr.arpa.

Este `arquivo de zona ' contém 3 `registros de recursos' (RRs): SOA, NS e um PTR. SOA é a contração para Início de Autoridade. O `@' é uma

observação especial que significa origem e desde que a coluna do campo para este arquivo diz 0.0.127.in-addr.arpa, a primeira linha realmente quer dizer

```
0.0.127.in-addr.arpa.   IN      SOA ...
```

NS é o nome do servidor RR. Não há '@' no início desta linha, está *implícito* desde que a última linha começou com o caracter '@'. Economiza-se assim alguma digitação e a possibilidade de cometer algum erro. Assim na linha NS se lê

```
0.0.127.in-addr.arpa.   IN      NS      ns.linux.bogus
```

Indicando ao DNS que a máquina é o servidor de nomes do domínio 0.0.127.in-addr.arpa é chamada ns.linux.bogus. 'ns' é um nome comum para servidor de nomes, mas como em servidores web são costumeiramente chamados www.*domínio*, este nome pode ser qualquer coisa.

E finalmente o registro PTR diz que a máquina no endereço 1 na subrede 0.0.127.in-addr.arpa, ou seja, 127.0.0,1 é denominado localhost.

O registro SOA é o preâmbulo para *todos* os arquivos de zona e deve haver exatamente um em cada arquivo de zona, devendo necessariamente ser o primeiro registro. Ele descreve a zona, sua origem (uma máquina servidor de nomes ns.linux.bogus), quem é a responsável por seu conteúdo (hostmaster@linux.bogus), qual a versão do arquivo de zona (serial: 1) e outras coisas que têm a ver com guarda de dados em cache e servidores secundários de DNS. Para os demais campos, Atualização, Tentativas, Expiração e TTL, pode-se usar os valores aqui indicados e se estará seguro.

Agora reinicializaremos o servidor de nomes (através do comando ndc restart), e usaremos nslookup para examinar o que foi feito:

```
$ nslookup

Servidor Padrão: localhost
Endereço:  127.0.0.1

> 127.0.0.1
Servidor:  localhost
Endereço:  127.0.0.1

Nome:    localhost
Endereço:  127.0.0.1
```

observamos então que é possível chegar a localhost a partir do endereço 127.0.0.1. Agora a nossa tarefa principal, no campo linux.bogus, vamos inserir uma nova seção chamada 'zone' no named.conf:

```
zone "linux.bogus" {
        notify no;
        type master;
        file "pz/linux.bogus";
};
```

Note-se a ausência de `.' no nome do domínio no arquivo named.conf.

No arquivo de zona do domínio linux.bogus colocaremos alguns dados totalmente inventados:

```
;
; Arquivo zona para linux.bogus
;
; O arquivo completo de zone
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151       ; serial, data de hoje +  serial de hoje
#
                        8H              ; Atualização, segundos
                        2H              ; T entativa, segundos
                        1W              ; Expiração, segundos
                        1D )            ; TTL, segundos
;
                NS      ns              ; Endereço Internet do nome do servidor
                MX      10 mail.linux .bogus     ; Servidor de Correio Primário
MX      20 mail.friend.bogus.   ; Servidor de Correio Secundário
;
localhost       A       127.0.0.1
ns              A       192.168.196.2
mail            A       192.168.196.4
```

Dois aspectos devem ser observados sobre o registro SOA. ns.linux.bogus *deve* ser uma máquina real com um registro A. Não é permitido ter um registro CNAME para a máquina mencionada no registro SOA. O nome não precisa ser 'ns', pode ser qualquer nome de máquina válido. Em seguida, a hostmaster.linux.bogus deve ser lido como hostmaster@linux.bogus, o qual deve ser um nome alternativo de correio, ou caixa postal, acessado pela(s) pessoa(s) que mantém o DNS e leiam a correspondência freqüentemente. Qualquer correspondência relativa ao domínio será enviada para o endereço relacionado aqui. O nome não precisa ser 'hostmaster', pode ser qualquer endereço e-mail válido, mas espera-se que o endereço e-mail 'hostmaster' *funcione* bem.

Há um novo tipo RR neste arquivo, o MX, ou registro de recurso de servidor de correio. Este arquivo diz aos sistemas de correspondência para onde enviar a correspondência endereçada para alguém@linux.bogus, ou seja no nosso caso mail.linux.bogus ou mail.friend.bogus. O número antes de cada nome de máquina define a prioridade. O RR com o número mais baixo tem prioridade. Caso ele não esteja ativo ou apresentar algum erro, a mensagem pode ser enviada a um outro servidor de mensagens com um número mais alto, um operador de correspondência secundário, ou seja, no nosso caso, mail.friend.bogus que tem prioridade 20.

Ao se reiniciar o servidor de nomes executando-se ndc restart obteremos os seguintes resultados com o nslookup:

```
$ nslookup
> set q=any
> linux.bogus
```

```
Server:  localhost
Address:  127.0.0.1

linux.bogus
        origin = ns.linux.bogus
        mail addr = hostmaster.linux.bogus
        serial = 199802151
        refresh = 28800 (8 horas)
        retry   = 7200 (2 horas)
        expire  = 604800 (7 dias)
        minimum ttl = 86400 (1 dia)
linux.bogus     nameserver = ns.linux.bogus
linux.bogus     preference = 10, mail exchanger = mail.linux.bogus.linux.bogus
linux.bogus     preference = 20, mail exchanger = mail.friend.bogus
linux.bogus     nameserver =  ns.linux.bogus
ns.linux.bogus  internet address = 192.168.196.2
mail.linux.bogus        internet address = 192.168.196.4
```

Com um exame mais apurado pode-se descobrir um pequeno problema. A linha

```
linux.bogus     preference = 10, mail exchanger = mail.linux. bogus.linux.bogus
```

deveria ser

```
linux.bogus     preference = 10, mail exchanger = mail.linux.bogus
```

Deliberadamente cometemos o erro para que o leitor aprenda com ele:-) Examinando o arquivo de zona, percebemos que na linha

```
                MX      10 mail.linux.bogus     ; Servidor primário de correio
```

está faltando um ponto. Ou seja há 'linux.bogus' demais. Caso um nome de máquina não seja seguido por um ponto num arquivo de zona, a origem será acrescentada ao final causando o duplo linux.bogus.linux.bogus. Portanto

```
                MX      10 mail.linux.bogus.    ; Servidor primário de correio
```

ou

```
                MX      10 mail                 ; Servidor primário de correio
```

estão corretos. Particularmente, sugerimos a última forma, por ser mais econômica e menos sujeita a erros. Existem alguns bem conhecidos usuários de bind que discordam e outros que concordam com isto. Num arquivo de zona, o domínio pode tanto ser totalmente identificado e terminado com um `.' ou não deve ser incluído de forma alguma, utilizando então o padrão da origem.

Devemos salientar que em um arquivo named.conf *não* deve haver `.' depois dos nomes dos domínios. Você não tem idéia de quantas vezes um `.' gerou uma enormidade de problemas e confundiu um punhado de administradores.

Agora que já expressamos nosso ponto de vista. Estamos com o novo arquivo de zona e com informações extras:

```
;
; Arquivo de zona para linux.bogus
;
; O arquivo de zona completo
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151       ; serial, data de hoje + serial de hoje #
                        8H              ; Atualizar, segundos
                        2H              ; Tentativas, segundos
                        1W              ; Expiração, segundos
                        1D )            ; TTL, segundos
;
                TXT     "Linux.Bogus, os especialistas DNS "
                NS      ns              ; Endereço Internet do servidor de nomes
                NS      ns.friend.bogus.
                MX      10 mail         ; Servidor de correio primário
                MX      20 mail.friend.bogus. ; Servidor de correio secundário

localhost       A       127.0.0.1

gw              A       192.168.196.1
                HINFO   "Cisco" "IOS"
                TXT     "O roteador"

ns              A       192.168.196.2
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "Pentium" "Linux 2.0"
www             CNAME    ns

donald          A       192.168.196.3
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "i486"  "Linux 2.0"
                TXT     "DEK"

correio         A       192.168.196.4
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "386sx" "Linux 2.2"

ftp             A       192.168.196.5
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "P6" "Linux 2.0.36"
```

Há diversos RRs novos: HINFO (INFOrmação da Máquina) tem duas partes, sendo aconselhável indicar os dois. A primeira parte é o hardware ou

CPU da máquina, e a segunda parte é o software ou OS da máquina. O servidor de nomes 'ns' tem uma CPU Pentium e executa o Linux 2.0. CNAME (NOME Canônico) é uma maneira de dar a uma mesma máquina vários nomes. Assim www é um nome alternativo para o ns.

O uso do registro CNAME é um pouco controvertido. Mas é seguro seguir a regra onde um registro MX, CNAME ou SOA *nunca* deve referir-se a um registro CNAME , e devem referir-se somente a um registro A, sendo portanto incorreto ter-se:

```
itamaracabar            CNAME   www                     ; NÃO!
```

o correto seria:

```
itamaracabar            CNAME   ns                      ; SIM!
```

É também seguro supor que um CNAME não é um nome de máquina válido para um endereço e-mail, por exemplo webmaster@www.linux.bogus é um endereço ilegal, conforme a configuração acima. Não se deve esperar que muito administradores de servidores de mensagens usem esta configuração, mesmo se ela funcionar localmente. A maneira para evitar isto é usar registros de tipo A (e talvez alguns outros também, como um registro MX):

```
www             A       192.168.196.2
```

Um grande número de magos do DNS, sugerem que o CNAME *não* seja utilizado. Por isso, devemos considerar esta sugestão *muito* seriamente.

Mas como se pode perceber, este COMO FAZER e muitos sites não seguem esta regra.

É possível carregar o novo banco de dados executando-se ndc reload, o que fará com que o named leia seus arquivos novamente.

```
$ nslookup
Default Server:  localhost
Address:  127.0.0.1

> ls -d linux.bogus
```

Isto significa que todos os registros devem ser apresentados. O resultado será:

```
<tscreen><verb>
[localhost]
$ORIGIN linux.bogus.
@                       1D IN SOA       ns hostmaster (
                                        199802151       ; nro. serial
                                        8H              ; atualizar
                                        2H              ; tentativas
                                        1W              ; expiração
                                        1D )            ; mínimo

                        1D IN NS        ns
```

```
                        1D IN NS        ns.friend.bogus.
                        1D IN TXT       "Linux.Bogus, os consultores DNS"
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
gw                      1D IN A         192.168.196.1
                        1D IN HINFO     "Cisco" "IOS"
                        1D IN TXT       "O roteador"
mail                    1D IN A         192.168.196.4
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "386sx" "Linux 1.0.9"
localhost               1D IN A         127.0.0.1
www                     1D IN CNAME     ns
donald                  1D IN A         192.168.196.3
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "i486" "Linux 1.2"
                        1D IN TXT       "DEK"
ftp                     1D IN A         192.168.196.5
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "P6" "Linux 1.3.59"
ns                      1D IN A         192.168.196.2
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "Pentium" "Linux 1.2"
@                       1D IN SOA       ns hostmaster (
                                        199802151       ; nro. serial
                                        8H              ; atualizar
                                        2H              ; tentativas
                                        1W              ; expiração
                                        1D )            ; mínimo
```

Parece ótimo. Como se pode ver parece muito com o arquivo de zona. Vamos verificar o que ele diz para www:

```
> set q=any
> www.linux.bogus.
Server:  localhost
Address:  127.0.0.1

www.linux.bogus canonical name = ns.linux.bogus
linux.bogus     nameserver = ns.linux.bogus
linux.bogus     nameserver = ns.friend.bogus
ns.linux.bogus  internet address = 192.168.196.2
```

Em outras palavras, o nome real de www.linux.bogus é ns.linux.bogus, e ele fornece algumas informações adicionais que ele possua sobre ns, o suficiente para um programa conectar-se a ele.

Agora estamos no meio do caminho.

8.3.3 A zona reversa

Agora os programas podem converter os nomes em linux.bogus para endereços com os quais eles podem se conectar. Porém é pedida também uma zona reversa, que torne o DNS capaz de converter um endereço em um nome. Este nome é usado por muitos servidores de espécies diferentes (FTP, IRC, WWW e outros) para decidir se eles querem conversar com a máquina local ou não, e em caso positivo, também qual a prioridade que deve ser dada a esta máquina. Para o acesso completo a todos os serviços da Internet, uma zona reversa é necessária.

Deve-se colocar o seguinte em named.conf:

```
zone "196.168.192.in-addr.arpa" {
        notify no;
        type master;
        file "pz/192.168.196";
};
```

Estes parâmetros são exatamente iguais para 0.0.127.in-addr.arpa e os conteúdos são semelhantes:

```
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151 ; Nro.Serial, data + nro. série
                        8H      ; Atualizar
                        2H  ; Tentativas
                        1W      ; Expiração
                        1D)     ; TTL mínimo

                NS    ns.linux.bogus.

1               PTR     gw.linux.bogus.
2               PTR     ns.linux.bogus.
3               PTR     donald.linux.bogus.
4               PTR     mail.linux.bogus.
5               PTR     ftp.linux.bogus.
```

Agora ao reinicializar o servidor de nomes (ndc restart) e examinar o trabalho realizado, utilizando-se o nslookup, teremos:

```
> 192.168.196.4
Server:  localhost
Address:  127.0.0.1

Name:    mail.linux.bogus
Address:  192.168.196.4
```

Então caso tudo pareça correto, vamos examinar todas as demais informações:

```
> ls -d 196.168.192.in-addr.arpa
[localhost]
$ORIGIN 196.168.192.in-addr.arpa.
@                       1D IN SOA       ns.linux.bogus. hostmaster.linux.bogus. (
                                        199802151       ; nro. serial
                                        8H              ; atualizar
                                        2H              ; tentativas
                                        1W              ; expiração
```

```
                                        1D )            ;  ttl mínimo

                        1D IN NS        ns.linux.bogus.
1                       1D IN PTR       gw.linux.bogus.
2                       1D IN PTR       ns.linux.bogus.
3                       1D IN PTR       donald.linux.bogus.
4                       1D IN PTR       mail.linux.bogus.
5                       1D IN PTR       ftp.linux.bogus.
@                       1D IN SOA       ns.linux.bogus. hostmaster.linux.bogus. (
199802151       ; nro. serial
                                        8H              ; atualizar
                                        2H              ; tentativas
                                        1W              ; expiração
                                        1D )            ; ttl mínimo
```

Parece bom!

Há algumas coisas que devemos acrescentar. Os números IP usados nos exemplos acima foram tirados dos blocos de 'redes privadas', ou seja, eles não podem ser usados publicamente na Internet. Por isso eles são seguros para serem usados em um exemplo de um COMO FAZER. A segunda coisa é a linha notify no;, a qual indica que o servidor de nomes não notificará o servidor secundário (escravo), quando houver uma atualização para um dos arquivos de zona. No bind-8 o servidor de nomes pode notificar os outros servidores relacionados nos registros NS no arquivo de zona, toda vez que ela for atualizada. Isto é conveniente para o uso diário e usual, mas em nossas experiências particulares com zonas, esta característica deve ser desativada, afinal não queremos que a experiência polua toda a Internet, queremos?

E claro, este domínio é totalmente inventado, assim como todos os endereços que estão nele. Para um exemplo real de um domínio real, veja a próxima seção.

8.4 Converter da versão 4 para versão 8

Esta foi originalmente uma seção sobre o uso da bind 8 escrita por David E. Smith (dave@bureau42.ml.org). Ela foi editada para conter o novo nome da seção.

Não há muito a acrescentar. Exceto pelo uso do servidor named.conf ao invés de servidor named.boot, tudo mais é idêntico; bind8 vem com um programa perl que converte arquivos de estilo velho para o novo formato. Exemplo de um named.boot (velho estilo) para um servidor de nomes somente para cache:

```
directory /var/named
cache   .                               root.hints
primary 0.0.127.IN-ADDR.ARPA            127.0.0.zone
primary localhost                       localhost.zone
```

Na linha de comando, no diretório bind8/src/bin/named (*presume-se aqui que se tenham os fontes da distribuição. Caso se localize somente o pacote binário, o programa estará por perto. -ed.*), digite:

```
./named-bootconf.pl < named.boot > named.conf
```

8.5 Windows 2000 Active Directory e BIND

http://www.revistadolinux.com.br/artigos/005,030,3,160,979.html

Guilherme de Assis Brasil
gabrasil@aspdatasul.com.br

Como todos sabemos, o Windows 2000 trouxe consideráveis mudanças em relação ao seu antecessor, o Windows NT 4.0. Uma destas mudanças foi a implementação do novo serviço de diretórios, o "Active Directory", que guarda informações sobre os recursos da rede e os disponibiliza para usuários e aplicações. Este serviço utiliza-se de vários recursos de integração com padrões da Internet como o DNS (Domain Name Service - Serviço de Nome de Domínio), por exemplo.

Neste aspecto, a migração para a plataforma Windows 2000 pode se tornar um problema se você não usa o servidor de DNS da Microsoft, o que é até uma situação bastante comum. Normalmente, os administradores de DNS mais avançados gostam de manter os arquivo de zona de seus domínios bem documentados e organizados, e quando a função de "Atualização Dinâmica" (RFC 2136) é habilitada, isso se torna uma tarefa impossível. Para desgosto deste seleto time de administradores o Active Directory do Windows 2000 usa extensamente o suporte as "Atualizações Dinâmicas" para funcionar corretamente e caso este não esteja habilitado, algumas partes do serviço de diretórios pode não funcionar, como por exemplo a replicação de dados entre controladores de domínio. Porém existe uma forma simples de deixar o Active Directory funcionando sem a desorganização habitual das atualizações dinâmicas.

Estaremos então mostrando como configurar o BIND (testamos no BIND 8.2.3 instalado em um Linux Slackware 7.1) para que o Active Directory funcione perfeitamente. É importante lembrar que a técnica mostrada aqui vale para qualquer outro servidor de DNS que suporte "Atualizações Dinâmicas" e registros SRV (RFC 2782), com as devidas diferenças de sintaxe de configuração.

Para tal feito será necessária a criação de uma zona primária sem suporte a atualização dinâmica e mais quatro subzonas com este suporte.

O procedimento para configurar corretamente o BIND e deixá-lo pronto para que o Active Directory possa utilizá-lo é bem simples e os passos estão abaixo:

8.5.1 Criar a zona primária.

A zona primária deve ser criada normalmente, usarei o exemplo de um domínio chamado "intranet" para ilustrar os exemplos. Esta zona não terá suporte à atualização dinâmica e todas as inclusões de novas máquinas ocorrerão aqui.

Inclua a seguinte entrada no arquivo de configuração do BIND (geralmente "`named.conf`")

```
zone "intranet" in {
     type master;
     file "intranet.zone";

};
```

Esta é uma entrada padrão para a inclusão de uma zona (domínio) no BIND, o arquivo "`intranet.zone`" deve ser criado normalmente, como qualquer outro arquivo de zona de domínio, mas é importante que os servidores que utilizarão os recursos do Active Directory esteja cadastrados corretamente tanto neste arquivo, como seus devidos reversos.

8.5.2 Criar as subzonas.

Quatro subzonas devem ser criadas, são elas "`_msdcs`", "`_udp`", "`_tcp`", "`_sites`". Estas subzonas terão suporte a atualizações dinâmicas e o acesso para escrever nestas zonas deve ser liberado para as máquinas com o active directory.

As entradas no arquivo "`named.conf`" são:

```
zone "_msdcs.intranet" in {
     type master;
     file "msdcs.zone";
     check-names ignore;
     allow-update { x.x.x.x/xx };
};

zone "_udp.intranet" in {
     type master;
     file "udp.zone";
     check-names ignore;
     allow-update { x.x.x.x/xx };
};

zone "_tcp.intranet" in {
     type master;
     file "tcp.zone";
     check-names ignore;
     allow-update { x.x.x.x/xx };
};

zone "_sites.intranet" in {
     type master;
     file "sites.zone";
     check-names ignore;
     allow-update { x.x.x.x/xx };
};
```

O "x.x.x.x/xx" representa o número IP da rede onde estão as máquinas que usarão a atualização dinâmica. Por exemplo, para permitir atualizações partindo da rede 192.168.0.0 com máscara 255.255.255.0 (classe C) deve-se colocar a entrada: "192.168.0.0/24".

9 NFS

Adaptado do "NFS HowTo", por Nicolai Langfeldt
Tradução Original da Conectiva Informática - Janeiro de 1999

9.1 Introdução ao NFS

NFS, o Sistema de Arquivos em Rede tem três importantes características:

- Possibilita o compartilhamento de arquivos sobre uma rede local.
- Funciona bastante bem.
- Possibilita diversos problemas de segurança que são bem conhecidos por intrusos, e podem ser explorados na obtenção de acesso (leitura, gravação e remoção) de todos os arquivos de um sistema.

Abordaremos todos estes assuntos neste Como Fazer. Por favor, não deixe de ler os itens sobre segurança neste Como Fazer, o que tornará a rede menos vulnerável a riscos tolos de segurança. As passagens sobre segurança serão bastante técnicas e exigirão conhecimento sobre redes IP e sobre os termos usados. Caso não se reconheça algum dos termos aqui usados, verifique o Como Fazer - Redes ou obtenha um livro sobre administração de redes TCP/IP. Esta é uma boa idéia de qualquer forma, caso se esteja administrando máquinas Unix/Linux. Um livro muito bom é *TCP/IP Network Administration* de Craig Hunt, publicado pela O'Reilly & Associates, Inc. E após toda esta leitura, certamente você será mais valorizado no mercado de trabalho, e isso não se pode perder ;-)

Há duas seções de ajuda com problemas no NFS, chamadas *Lista de Verificação* e *FAQs*. Por favor, leia estas seções com atenção caso algo não funcione da maneira esperada.

9.2 Configuração do Servidor NFS

9.2.1 Pré-Requisitos

Antes de continuar a ler este capítulo será necessário poder executar o programa telnet de e para as máquinas que serão usadas como servidor e cliente.

9.2.2 Primeiros Passos

Antes que se possa fazer qualquer coisa, será necessário ter um servidor NFS configurado. Caso se faça parte de alguma rede de um departamento ou rede universitária provavelmente já existirão diversos servidores NFS sendo executados. Casos eles permitam o acesso ou ao invés disso, se esteja lendo este Como Fazer para se obter acesso a um servidor NFS, não é necessário ler esta seção, podendo passar diretamente à seção 9.3 Configuração do Cliente NFS.

Caso se necessite configurar um sistema diferente do GNU/Linux para atuar como servidor, será necessário ler o manual do sistema para descobrir como habilitar o NFS e a exportação de sistemas de arquivos. Há uma seção neste documento explicando como fazer isto em muitos sistemas diferentes. Após se descobrir isso tudo, pode-se continuar na leitura desta seção.

Aqueles que continuaram a sua leitura estão avisados: vamos ter que configurar uma série de programas.

9.2.3 O `portmapper`

O portmapper no GNU/Linux é chamado também de portmap ou rpc.portmap. A página de manual online diz que se trata de "mapeador de portas DARPA para números de programas RPC". Este é o primeiro problema de segurança com o qual nos deparamos neste Como Fazer. A descrição de como evitar estes problemas podem ser encontrada na seção 9.4 Segurança e NFS, a qual eu repito que deve ser lida.

Inicializando o portmapper! Ele é chamado de portmap ou rpc.portmap e deve estar localizado no diretório /usr/sbin (em algumas máquinas ele é chamado de rpcbind). Pode-se inicializá-lo manualmente por hora, mas ele deverá ser reinicializado toda vez que o sistema operacional for ativado, sendo então necessário editar os programas rc. Estes programas são explicados mais detalhadamente na página de manual do processo init, e usualmente estão localizados nos diretórios `/etc/rc.d`, `/etc/init.d` ou `/etc/rc.d/init.d`. Caso haja um programa chamado `inet` ou algo similar, este provavelmente será aquele que deve ser editado. Porém, como fazê-lo está além do escopo deste documento. Deve-se iniciar o programa portmap e verificar se ele está ativo através do comando `ps -aux`. Encontrou-o? Ótimo.

9.2.4 O `mountd` e o `nfsd`

Os próximos programas que necessitam ser executados são chamados mountd e nfsd. Porém, antes, é necessário editar outro arquivo. Desta vez o `/etc/exports`. O formato geral das entradas neste arquivo é o seguinte:

```
diretório computador(opção)
```

Onde:

- `diretório`: caminho completo do diretório a ser exportado. Se o diretório não for seguido por um `computador` ou `opção`, qualquer computador terá acesso apenas de leitura (`ro`) ao diretório;
- `computador`: máquinas que terão acesso ao diretório exportado. Se nenhum computador for fornecido, o diretório será exportado a qualquer computador. Os valores válidos são:
  - Nome de máquina, como `gauss.alfamidia.com.br`;
  - Grupos de computadores, como `*.alfamidia.com.br`;
  - Endereço IP/máscara, como `172.20.0.0/255.255.0.0`;
- `opção`: define o tipo de acesso. Se não for informado, é assumido apenas leitura (ro). Os valores mais comuns são:
  - `ro`: apenas leitura;
  - `rw`: leitura e escrita.

Digamos que se deseje que o sistema de arquivo `/mn/parolin/local`, o qual está localizado na máquina parolin se torne disponível para a máquina chamada batel. Deve-se então utilizar a seguinte configuração no arquivo `/etc/exports` na parolin:

```
/mn/parolin/local       batel(rw)
```

As linhas acima fornecem a batel acesso de leitura e gravação (`rw`) para `/mn/parolin/local`. Ao invés de `rw` poderíamos informar `ro`, o que fornece acesso somente para leitura e que é o padrão quando este parâmetro não é informado. Há diversas opções que podem ser utilizadas e que serão discutidas juntamente com aspectos de segurança mais adiante. Elas estão descritas nas páginas de manual online do comando exports, a qual deve ser lida ao menos uma vez na vida. Há ainda formas melhores de incluir diversas máquinas no arquivo exports. Pode-se por exemplo, usar grupos de rede caso se esteja utilizando NIS (ou NYS) (NIS foi conhecido como YP) e especificar sempre um domínio com caracteres de generalização, ou subredes IP como máquinas que têm permissão para montar algo. Porém é necessário considerar que poderá ser possível obter acesso ao servidor de forma não autorizada caso se utilize autorizações tão genéricas.

Nota: o arquivo exports não tem a mesma sintaxe que em outros Unixes.

Após a definição dos diretórios a serem exportados no arquivo `/etc/exports`, será necessário executar o comando `exportfs` para processar este arquivo e atualizar o arquivo `/var/lib/nfs/xtab`. O arquivo `xtab` contém informações sobre os diretórios exportados e é o arquivo que o `mountd` lê quando está processando a solicitação de montagem do cliente. O formato do comando é o seguinte:

```
[root@gauss /root] # exportfs -r
```

Agora que configuramos o mountd (ou talvez ele seja chamado rpc.mountd) e o nfsd (o qual pode ser chamado rpc.nfsd), ambos irão ler o arquivo exports.

Caso se edite o /etc/exports deve-se estar seguro de que os programas nfsd e mountd fiquem cientes destas alterações. A forma tradicional é através da execução do comando exportfs. Muitas distribuições GNU/Linux não possuem o programa exportfs. Caso este seja o seu caso, pode-se instalar o seguinte programa na máquina local:

```
#!/bin/sh
killall -HUP /usr/sbin/rpc.mountd
killall -HUP /usr/sbin/rpc.nfsd
echo re-exportando sistemas de arquivos
```

O programa acima deve ser salvo, por exemplo como /usr/sbin/exportfs, e deve ser executado o comando chmod a+rx exportfs. Agora, toda vez que uma alteração for efetuada, deve-se executar o comando exportfs a seguir, com privilégios de superusuário.

Agora deve-se checar se mountd e nfsd estão sendo adequadamente executados. Inicialmente deve-se executar o comando rpcinfo -p. Ele deverá apresentar uma saída similar a:

```
   program vers proto   port
    100000    2   tcp    111  portmapper
    100000    2   udp    111  portmapper
```

```
100005    1   udp    745  mountd
100005    1   tcp    747  mountd
100003    2   udp   2049  nfs
100003    2   tcp   2049  nfs
```

Como se pode perceber, o portmapper anunciou os seus serviços, assim como mountd e nfsd.

Caso se obtenha uma mensagem similar a rpcinfo: não foi possível contactar o portmapper: RPC: Erro no sistema remoto - Conexão recusada ou algo similar, possivelmente o portmapper não esteja sendo executado. Caso se obtenha uma mensagem similar a Nenhum programa remoto registrado. Então, ou o portmapper não deseja falar com a máquina local ou existe algum erro. Pode-se finalizar o nfsd, o mountd e o portmapper e tentar reiniciá-los nesta ordem novamente.

Após verificar os serviços disponíveis segundo o portmapper, pode-se fazer uma checagem através do comando ps. O portmapper continuará a reportar um serviço, mesmo após o programa responsável ter sido finalizado com erro, por exemplo. Então um comando ps poderá ser a maneira mais simples de descobrir que programas estão efetivamente sendo executados.

Evidentemente, será necessário modificar os arquivos rc do sistema para inicializar o mountd e o nfsd, assim como o portmapper, quando o sistema operacional for carregado. É muito provável que estes programas já existam na máquina local e que se deva somente descomentar as seções adequadas ou ativá-los nos níveis de execução corretos.

Páginas de manual online que já devem ter sido visitadas até agora: portmap, mountd, nfsd, e exports.

Bem, caso tudo tenha sido feito exatamente como foi descrito aqui, já temos à disposição todo o conjunto de ferramentas necessárias para iniciar um cliente NFS.

9.3 Configuração do Cliente NFS

Inicialmente é necessário ter um kernel com o suporte a sistemas de arquivo NFS compilado ou como um módulo. Isso deve ser configurado antes da compilação do kernel. Caso se esteja utilizando uma distribuição muito boa e nunca se tenha lidado com o kernel ou módulos, nfs está magicamente à sua disposição.

O comando showmount lista os diretórios que o servidor exporta e os clientes que têm permissão para montar esses diretórios:

```
[root@gauss /root] # showmount –-exports parolin
/mn/parolin/local       batel(rw)
```

Pode-se agora, na linha de comandos do superusuário, informar o comando de montagem apropriado e o sistema de arquivos estará disponível. Continuando com nosso exemplo anterior, desejamos montar `/mn/parolin/local` a partir de parolin. Isso deve ser feito através do seguinte comando:

```
mount –t nfs parolin:/mn/parolin/local /mnt/nfs
```

ou

```
mount -o rsize=1024,wsize=1024 parolin:/mn/parolin/local /mnt/nfs
```

(Retornaremos posteriormente às opções rsize e wsize). O sistema de arquivos está agora disponível sob /mnt e pode-se acessá-lo através do comando cd, assim como verificar o seu conteúdo através do comando ls, e observar os arquivo individualmente. Pode-se perceber que ele não é tão rápido quando um sistema local, mas muito mais amigável que o uso do ftp. Se, ao invés de montar um sistema de arquivos, o comando mount apresente uma mensagem de erro como mount:parolin:/mn/parolin/local falhou, razão fornecida pelo servidor: Permissão negada, então o arquivo exports contém algum erro. Caso ele informe mount clntudp_create: RPC: Programa não registrado isso significa que os programas nfsd ou mountd não estão sendo executados no servidor.

Para desmontar o sistema de arquivos basta comandar:

```
umount /mnt
```

Para que um sistema de arquivos nfs seja montado na inicialização do sistema operacional, deve-se editar o arquivo /etc/fstab da forma usual. No caso de nosso exemplo, deve-se adicioar a seguinte linha:

```
# dispositivo  pto.montagem tipo_sist_arqs  opções          dump ordem verif.

parolin:/mn/parolin/local  /mnt    nfs  rsize=1024,wsize=1024 0    0
```

Bem, parece tudo. Quase. Continue a leitura por favor.

9.3.1 Opções de Montagem

Há algumas opções que devem ser consideradas. Eles definem a forma como o cliente NFS lida com uma queda do servidor ou da rede. Um dos aspectos mais interessantes sobre NFS é que ele lida com estas situações com elegância, desde que o cliente esteja corretamente configurado. Há dois tipos distintos de parâmetros de tratamento de falhas:

- **soft** O cliente NFS reporta um erro ao processar o acesso a um arquivo localizado em um sistema de arquivos montado via NFS. Alguns programas podem lidar com isto com compostura, outros não. Esta opção não é recomendada.
- **hard** O programa que acessa um arquivo em um sistema de arquivos montado via NFS irá travar sempre que o servidor não responder. O processo não pode ser interrompido ou finalizado a menos que se tenha especificado intr. Quando o servidor NFS esteja novamente ativo, o programa irá continuar a partir do ponto onde tenha parado. Isso é provavelmente o que se deseje. Recomendamos o uso do parâmetro hard,intr em todos os sistemas de arquivos montados via NFS.

A partir do exemplo anterior, esta seria a entrada no arquivo fstab:

```
# dispositivo  pto.montagem tipo_sist_arqs  opç ões          dump ordem verif.
parolin:/mn/parolin/local  /mnt    nfs  rsize=1024,wsize=1024,hard,intr 0 0
```

9.3.2 Otimizando o NFS

Normalmente, caso as opções rsize e wsize seja especificados, o NFS irá ler e gravar blocos de 4096 e 8172 bytes. Algumas combinações de kernel do GNU/Linux e placas de rede não podem lidar com blocos grandes e não

podem ser otimizados. Então vamos tentar descobrir como encontrar os parâmetros rsize e wsize que funcionem da maneira mais otimizada possível. É possível testar a velocidade das opções com um simples comando. Dado o comando mount conforme descrito acima, logo temos acesso de gravação ao disco, podendo executar um teste de performance de gravação seqüencial:

```
time dd if=/dev/zero of=/mnt/testfile bs=16k count=4096
```

Este comando criará um arquivo de 64 Mb de bytes zerados (que deve ser grande o suficiente para que o cache não altere significativamente a performance. Pode ser usado um arquivo maior caso o sistema local tenha muita memória). Isso pode ser feito algumas vezes (5-10?), para que se possa ter uma média bem fundamentada. Neste casos, o importante é medir o tempo de "relógio" e o tempo efetivamente gasto na conexão. Após, pode-se testar a performance da leitura ao se ler o arquivo de volta:

```
time dd if=/mnt/testfile of=/dev/null bs=16k
```

Isso pode ser feito algumas vezes. Após deve-se executar o comando mount e umount novamente com tamanhos maiores em rsize e wsize. Eles devem ser provavelmente múltiplos de 1024, e não maior que 16384 visto que este é o tamanho máximo do NFS versão 2. Exatamente após a montagem de um tamanho maior, acesse o sistema de arquivos montado através do comando cd e explore-o através do comando ls, para estar seguro que ele está funcionando perfeitamente. Caso os parâmetros rsize/wsize sejam muito grandes, os sintomas não são *muito* óbvios. Um típico sintoma é uma lista incompleta dos arquivos produzida pelo comando ls e nenhuma mensagem de erro. Ou ao se ler um arquivo ele falha misteriosamente, sem mensagens de erro. Após definir que os parâmetros rsize/wsize funcionam perfeitamente deve-se executar os testes de performance. SunOS e Solaris tem a reputação de funcionar muito melhor com blocos de 4096 bytes.

kernels mais recentes do GNU/Linux (desde o 1.3) executam a leitura antecipada para rsizes maiores ou iguais ao tamanho de página da máquina. Em máquinas Intel o tamanho de página é de 4.096 bytes. A leitura adiantada aumenta *significativamente* a performance de leitura do NFS. Ou seja, sempre que possível deve-se usar o rsize de 4.096 bytes em máquinas Intel.

Lembre-se de editar o arquivo /etc/fstab com os valores de rsize/wsize encontrados.

Uma sugestão para incrementar a performance de gravação do NFS é desabilitar o sincronismo de gravação do servidor. A especificação NFS indica que a gravação NFS solicitada não pode ser considerada finalizadas antes dos dados serem gravados em um meio não volátil (normalmente o disco rígido). Isso restringe a performance de gravação de alguma forma, enquanto que gravações assíncronas irão aumentar a velocidade do NFS. O servidor GNU/Linux nfsd nunca faz gravações síncronas uma que a própria implementação do sistema de arquivos não o faz, mas em servidores em sistemas operacionais diferentes isso pode aumentar a performance através do seguinte parâmetro no arquivo exports:

```
/dir    -async,access=linuxbox
```

ou algo similar. Por favor verifique a página de manual online da máquina em questão. Cabe salientar que esta opção aumenta o risco de perda de dados no caso de algum problema ocorrer antes da efetiva gravação dos dados.

9.4 Segurança e NFS

Não me considero um expert em segurança de computadores. Porém existem algumas *sugestões* importantes. É importante ressaltar que esta não é uma lista completa de todos os aspectos relacionados com segurança e caso se imagine que implementando somente estes não se poderá ter qualquer problema relacionado com o tema segurança, por favor me envie seu email que eu tenho uma ponte e desejo vendê-la.

Esta seção é provavelmente fora de questão caso se esteja em uma rede *fechada* onde todos os usuários são conhecidos e ninguém que não seja confiável pode acessar a rede, ou seja não há forma de discar para a rede e não há forma de conectar-se a outras redes onde existam usuários não confiáveis. Isso soa como paranóia? Não sou paranóico. Isso é somente um aviso *básico* de segurança. E lembre-se, o que aqui for dito é somente uma base para o tema. Um site *seguro* necessita de um administrador diligente e com conhecimento que consiga encontrar informações sobre problemas de segurança correntes e potenciais.

NFS é um problema básico, caso o cliente não seja informado do contrário irá confiar no servidor NFS e vice e versa. Isso pode ser ruim, pois se a senha do superusuário no servidor NFS for quebrada, a senha dos superusuários dos clientes também o será com relativa facilidade. E vice e versa. Há algumas estratégias para se evitar isso, as quais mencionaremos adiante.

Um leitura obrigatória são os avisos do CERT sobre NFS, onde muitos dos textos lidam com conselhos sobre segurança. Veja em ftp.cert.org/01-README por uma lista atualizada dos avisos CERT. Aqui estão alguns dos relacionados com NFS:

`CA-91:21.SunOS.NFS.Jumbo.and.fsirand` `12/06/91`

Vulnerabilidade preocupa Sun Microsystems, Inc. (Sun) Sistema de Arquivos em Rede (NFS) e o programa fsirand. Estas vulnerabilidades afetam o SunOS versões 4.1.1, 4.1 e 4.0.3 em todas as arquiteturas. Atualizações estão disponíveis para SunOS 4.1.1. Uma atualização inicial para o NFS SunOS 4.1 está também disponível. Sun irá disponibilizar atualizações completas para as versões SunOS 4.1 e SunOS 4.0.3 em uma versão posterior.

`CA-94:15.NFS.Vulnerabilidades` `12/19/94`

Este aviso descreve as medidas de segurança a serem tomadas para evitar diversas vulnerabilidades do Sistema de Arquivos em Rede (NFS). Os avisos foram gerados devido ao incremento do comprometimento de superusuários através de invasores usando ferramentas que exploram estas falhas.

`CA-96.08.pcnfsd` `04/18/96`

Este aviso descreve a vulnerabilidade do programa pcnfsd (também conhecido como rpc.pcnfsd). Uma atualização está incluída.

9.4.1 Segurança no Cliente

No cliente, podemos decidir se desejamos ou não confiar no servidor, através de algumas opções na montagem. Por exemplo, é possível proibir programas suid a funcionarem em sistemas de arquivos NFS através da opção nosuid. Esta pode ser uma boa idéia que deve ser considerada no uso de todos os discos montados via NFS. Esta opção indica que o superusuário do servidor não pode fazer um programa com características de suid no sistema de arquivos, o que possibilitaria que ele acessasse o cliente como um usuário normal e usasse o programa suid-superusuário para tornar-se root na máquina cliente. Deve-se proibir também a execução de arquivo em sistemas de arquivos montados, através da opção noexec. Porém isso pode ser impraticável por vezes, assim como o nosuid uma vez que um sistema de arquivos normalmente contém *alguns* programas que necessitam ser executados. Estes parâmetros podem ser informados na coluna opções, juntamente com os parâmetros rsize e wsize, separados por vírgulas.

9.4.2 Segurança no Servidor: nfsd

No servidor pode-se decidir sobre a possibilidade de confiar na conta do superusuário do cliente. Isso é definido através do uso da opção root_squash no arquivo exports:

```
/mn/parolin/local batel(rw,root_squash)
```

Agora caso um usuário com número de identificação igual a 0 (UID) tentar acessar (ler, gravar, remover) o sistema de arquivos, o servidor substituirá o UID pela identificação da conta "nobody" (ninguém). Isso faz com que o superusuário da máquina cliente não possa acessar arquivos ou executar mudanças autorizadas somente para o superusuário do servidor. Isso é aconselhável e provavelmente deva-se usar root_squash em todos os sistemas exportados. "Porém o superusuário cliente pode ainda usar o comando 'su' para tornar-se qualquer outro usuário e acessar e alterar quaisquer arquivos", é o que se pode pensar à primeira vista. A resposta é: sim, é desta forma que as coisas funcionam com Unix e NFS. Isso traz uma implicação importante: todos os binários e arquivos importantes devem pertencer ao superusuário root, e não a bin ou outra conta diferente, uma vez que somente a conta do superusuário da máquina cliente pode acessar a conta do superusuário no servidor. Na página de manual online do nfsd há diversas outras opções squash que podem ser usadas, então o administrador deve decidir quem não pode ter acesso à conta do superusuário. Existem opções de se evitar o uso de faixas ou de qualquer UID ou GID que se deseje. Isso está descrito na mesma página de manual.

root_squash é na verdade o padrão do nfsd do Linux. Para permitir acesso a um sistema de arquivos como superusuário deve-se usar a opção no_root_squash.

Outro aspecto importante é garantir que o nfsd verifique que todas as requisições são provenientes de uma porta autorizada. Caso se aceite requisições de qualquer porta antiga de um usuário sem privilégios especiais, torna-se simples acessar o sistema de arquivos através da Internet, por exemplo. Basta usar o protocolo nfs e identificar-se como qualquer usuário que se deseje. Ooopss. O nfsd do GNU/Linux realiza esta verificação por padrão,

em outros sistemas operacionais deve-se habilitar esta opção. Isso deverá estar descrito na página de manual do servidor nfs do sistema.

Um dado adicional. Nenhum sistema de arquivo deve ser exportado para o 'localhost' ou 127.0.0.1. Acredite em mim.

9.4.3 Segurança no Servidor: o portmapper

O portmapper básico em combinação com nfsd tem um problema de desenho que torna possível obter arquivos em servidores NFS sem a necessidade de quaisquer privilégios. Felizmente o portmapper do GNU/Linux é relativamente seguro contra este tipo de ataque, o que pode ser evitado através da configuração de uma lista de acessos em dois arquivos,

Inicialmente editaremos o /etc/hosts.deny. Ele deverá conter a seguinte linha:

```
portmap: ALL
```

através da qual o acesso será bloqueado para *todos* os cliente. Isto talvez seja um pouco drástico, então podemos tornar as definições um pouco mais maleáveis através da edição do arquivo /etc/hosts.allow. Inicialmente é necessário definir que será colocado nele. Ele contém basicamente uma lista de todas as máquinas que podem acessar o portmapper local. Em um sistema GNU/Linux há normalmente poucas máquinas que necessitem este tipo de acesso, qualquer que seja a razão. O portmapper administra os programas nfsd, mountd, ypbind/ypserv, pcnfsd, e serviços 'r' como ruptime e rusers. Todas as máquinas que necessitam acessar os serviços da máquina local deve ter permissão para tanto. Digamos que o endereço da máquina local seja 129.240.223.254 e que ela está conectada à subrede 129.240.223.0, a qual deve ter acesso à máquina local (em caso de dúvida verifique o Como Fazer - Redes para refrescar a memória sobre estes conceitos). Para tanto basta executar:

```
portmap: 129.240.223.0/255.255.255.0
```

no arquivo hosts.allow. Este é o mesmo endereço de rede fornecido para o comando route e a máscara de subrede informada no ifcongif. No dispositivo eth0 desta máquina ifconfig mostraria:

```
...
eth0      Link encap:10Mbps Ethernet  HWaddr 00:60:8C:96:D5:56
          inet addr:129.240.223.254  Bcast:129.240.223.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:360315 errors:0 dropped:0 overr uns:0
          TX packets:179274 errors:0 dropped:0 overruns:0
          Interrupt:10 Base address:0x320
...
```

e netstat -rn apresentaria

```
Tabela de Roteamento do Kernel
Destinação     Cam.Padrão         Máscara    Indics Métrica Ref  Uso   Iface
...
129.240.223.0   0.0.0.0         255.255.255.0   U     0      0   174412 eth0
...
```

o endereço de rede na primeira coluna.

Os arquivos hosts.deny e hosts.allow são descritos nas página de manual de mesmo nome.

IMPORTANTE: *não* coloque *nada* exceto *ENDEREÇOS IP* nas linhas do portmap nestes arquivos. Pesquisas por nomes podem indiretamente causar atividade do portmap o qual acionará a pesquisa de nomes de máquinas a qual indiretamente irá causa atividade no portmap.,

As sugestões acima certamente deixarão o servidor mais seguro. As questões restantes residem em alguém que tenha descoberto a senha do superusuário (ou inicializando um MS-DOS) em uma máquina confiável e usando este privilégio para enviar requisições a partir de uma porta segura como qualquer outro usuário real.

9.4.4 NFS e Firewalls

É uma boa idéia proteger o servidor nfs e as portas portmap no roteador ou no firewall. O nfsd opera normalmente na porta 2049, nos protocolos udp e tcp. O portmapper na porta 111, tcp e udp e o mountd na porta 745 e 747, tcp e udp. Estas informações devem ser checadas através do comando rpcinfo -p.

Por outro lado, caso se deseje permitir o acesso ao NFS através de um firewall, há opções em programas mountd e nfsd mais recentes que permitem o uso específico e não padronizado de portas que podem ser abertas através de um firewall.

9.4.5 Resumo

Caso se utilize hosts.allow/deny, root_squash, nosuid e funcionalidades de portas privilegiadas para os softwares portmapper e nfs pode-se evitar muitos dos problemas atualmente conhecidos sobre segurança e pode sentir-se quase seguro sobre *estes problemas* no mínimo. Porém há mais ainda: quando um intruso tem acesso à rede, ele pode incluir comandos estranhos nos arquivos .forward ou nos arquivos de mensagens, quando /home ou /var/spool/mail são montados via NFS. Pela mesma razão, nunca se deve dar acesso às chaves privadas PGPP sobre nfs. Ou no mínimo, deve-se saber dos riscos envolvidos. Pelo menos isso você já sabe.

NFS e o portmapper criam um subsistema complexo e adicionalmente há problemas que são descobertos e que devem ser solucionados, além da necessidade de se ter em mente o desenho básico de implementação a ser usado. Para estar ciente do que está ocorrendo pode acessar o grupo de notícias comp.os.linux.announce e comp.security.announce eventualmente.

9.5 Pontos de Verificação de Montagem

Esta seção é baseada na lista de verificação de problemas da IBM Corp. Meus agradecimentos a eles por tornarem ela disponível para este Como Fazer. Caso o leitor esteja com algum problema em montar sistemas de arquivos NFS, por favor consulte esta lista. Cada item descreve um problema específico e a sua solução.

1. O sistema de arquivos não foi exportado, ao menos para a máquina cliente em questão.

   Solução: Inclui-lo no arquivo exports.

2. A resolução de nomes não confere com a lista de exports.

   por exemplo: a lista em export indica uma exportação para johnmad mas o nome johnmad é resolvido como johnmad.austin.ibm.com, fazendo com que a permissão de montagem seja negada.

   Solução: Exportar em ambos os formatos de nomes.

   Isso pode ocorrer ainda quando o cliente tem 2 interfaces com nomes diferentes para cada um dos dois dispositivos e o comando export especifica somente um deles.

   Solução: Exportas para ambas as interfaces.

   Isso pode ocorrer também quando o servidor não consegue executar um chamada lookuphostbyname ou lookuphostbyaddr (são funções da biblioteca) no cliente. Estejá seguro de que o cliente pode executar máquina <nome>; máquina <endereço_ip>; e que ambos mostram a mesma máquina.

   Solução: ajustar a resolução de nomes no cliente.

3. O sistema de arquivos foi montado após a inicialização do NFS (no servidor). Neste caso o sistema de arquivos está exportado sob um ponto de montagem.

   Solução: Desativar nfsd e reinicializá-lo.

   Nota: os clientes que tenham pontos de montagem sob sistemas de arquivos terão problemas no acesso após a reinicialização.

   N.T.: nestes casos é recomendada a execução do comando mount -a, como superusuário, na máquina cliente.

4. As datas estão estranhamente diferentes em ambas as máquinas (o que pode gerar inconsistências com os arquivos).

   Solução: Ajustar as datas.

   O autor do Como Fazer sugere o uso do NTP para sincronismo de relógios. Uma vez que existem restrições de exportação do NTP para fora dos EUA, pode-se obter uma cópia em uma distribuição GNU/Linux ou em ftp://ftp.hacktic.nl/pub/replay/pub/linux ou em um site espelho.

5. O servidor não aceita uma montagem de um usuário presente em mais de 8 grupos.

   Solução: diminuir o número de grupos aos quais o usuário pertença ou alterar o usuário na montagem.

9.6 FAQ

Esta é uma seção de perguntas e respostas. Muito do que está contido aqui foi escrito por Alan Cox.

1. Obtive uma série de erros de manipulação de arquivos nfs ao usar o GNU/Linux como servidor.

   Isso é causado por uma antiga versão do nfsd. Está corrigida a partir da versão nfs-server2.2beta16.

2. Ao tentar montar um sistema de arquivos, surge a mensagem:
   3. não foi possível registrar-se no portmap: erro do sistema no envio
   4.

   Provavelmente se está utilizando o sistema da Caldera. Há um problemas com os programas rc. Por favor entre em contato com eles para correção do problema.

5. Por que não é possível executar um arquivo após copiá-lo para o servidor NFS?

   A questão reside no fato do nfsd criar caches de manipulação de arquivos por questões de performance (lembre-se que ele é executado em um espaço de usuário). Enquanto nfsd tem um arquivo aberto (como no caso em que ele esteja sendo gravado), o kernel não permite a sua execução. Os programas NFSd a partir de 95 liberam os arquivos após alguns segundos, já versões mais antigas podem levar dias.

6. Os arquivos NFS estão todos com permissões somente de leitura.

   O padrão do servidor NFS GNU/Linux é somente fornecer permissões de leitura para arquivos montados. O arquivo /etc/exports deve ser alterado caso se deseje algo diferente.

7. Existe um sistema de arquivos montado a partir de um servidor nfs GNU/Linux e enquanto o comando ls trabalha, a leitura e gravação de arquivos não funcionam.

   Em versões mais antigas do Linux, deve-se montar um servidor NFS com os parâmetros rsize=1024,wsize=1024.

8. Ao montar a partir de um servidor NFS GNU/Linux com um bloco de tamanho entre 3500-4000 ele trava regularmente.

   Bem...não faça mais isso!

9. O GNU/Linux pode executar NFS sobre TCP?

   Não no momento.

10. Ao se montar a partir de uma máquina Linux, obtém-se inúmeros erros.

Esteja certo de que os usuários utilizados estão presentes em no máximo 8 grupos. Servidores mais antigos requerem isso.

11. Ao reinicializar a máquina, ela algumas vezes trava ao tentar desmontar um servidor NFS.

    Não desmonte servidores NFS ao reinicializar ou desligar. Simplesmente ignore-os. Isso não irá machucar nimguém. O comando é umount -avt nonfs.

12. Clientes GNU/Linux NFS são muito lentos ao tentar gravar em sistemas Sun e BSD.

    NFS executa gravações síncronas (que podem ser desabilitadas caso não haja nenhum grande problema em se perder algum dado). Kernels derivados do BSD tendem a trabalhar mal com pequenos blocos. Porém ao se gravar blocos de 4 Kb de dados a partir de uma máquina Linux, usando pacotes de 1 Kb, faz com que o GNU/Linux use a rotina BSD na seguinte forma:

    ler página de 4K
    alterar para 1K
    gravar 4K no disco rígido
    ler página de 4K
    alterar para 1K
    gravar 4K no disco rígido
    etc.

10 NIS

10.1 Introdução

O serviço NIS (*Network Information Service*, ou Serviço de Informações de Rede) fornece para os usuários da rede local um ambiente de rede transparente. Os dados de contas do usuário e nomes das máquinas são mantidos no servidor, permitindo que os usuários usem qualquer máquina da rede sem se preocupar com senhas diferentes e cópia de arquivos.

O NIS é baseado em RPC (*Remote Procedure Call*, ou Chamada de Procedimento Remoto), e é composto basicamente do servidor, que armazena as informações, do cliente, que acessa o servidor, e das informações administrativas. Originalmente, o NIS era chamado de YP (*Yellow Pages*, ou Páginas Amarelas). Infelizmente esse nome é uma marca registrada da *British Telecom*, e o nome do serviço teve que ser mudado. Mas os nomes dos comandos permaneceram, por isso temos o `ypserv` (servidor) e `ypbind` (cliente).

10.1.1 Mapas

Mapas são estruturas de dados geradas a partir dos arquivos mestre, tais como /etc/passwd e /etc/hosts. Para alguns arquivos são criados vários tipos de mapas.

10.1.2 Tipos de Servidores

Existem dois tipos de servidores NIS: mestre e escravo. Um servidor mestre é uma máquina exclusiva, com um domínio particular, que mantém os mapas autenticados. O mestre executa o daemon `ypupdated`, que alerta os servidores escravos da necessidade da atualização de seus mapas. Todas as mudanças de mapas são feitas no mestre, e depois propagadas para os escravos.

Os servidores escravos agem como intermediários entre os clientes e o mestre, mantendo réplicas exatas dos mapas contidos no servidor principal.

As máquinas cliente, por sua vez, executam o servidor ypbind, que as torna capazes de executar processos para obter informações de um servidor.

10.2 Configurando o Servidor **Mestre**

A primeira coisa a fazer é verificar se os pacotes ypserv, ypbind e yp-tools estão instalados:

```
[root@gauss:~] # rpm -qa | grep yp-tools
[root@gauss:~] # rpm -qa | grep ypbind
[root@gauss:~] # rpm -qa | grep ypserv
```

Caso não, instale os pacotes seguindo os seguintes comandos:

```
[root@gauss:~] # rpm -Uvh yp-tools*
[root@gauss:~] # rpm -Uvh ypbind*
[root@gauss:~] # rpm -Uvh ypserv*
```

Verifique se os arquivos /etc/gshadow e /etc/netgroup estão presentes, caso não crie os arquivos com o comando abaixo:

```
[root@gauss:~] # touch /etc/gshadow
[root@gauss:~] # touch /etc/netgroup
```

Depois disso, deve-se verificar se o nome do domínio NIS está correto. Para isso, basta executar o comando `nisdomainname` e verificar o resultado. É importante ressaltar que o domínio NIS não tem nenhuma relação com o domínio DNS, embora na maioria das vezes o nome seja o mesmo.

```
[root@gauss:~] # nisdomainname
alfamidia.
```

É importante a presença do ponto no final do nome. Se o nome estiver incorreto, basta executar:

```
[root@gauss:~] # nisdomainname alfamidia.
```

Em seguida, deve-se executar os serviços `portmap` e `ypserv`:

```
[root@gauss:~] # touch /etc/init.d/portmap start
Iniciando o portmapper...OK
[root@gauss:~] # touch /etc/init.d/ypserv start
Iniciando o YP server...OK
```

O próximo passo é incluir o próprio servidor na lista de servidores:

```
[root@gauss:~] # /usr/lib/yp/ypinit –m
At this point, we have to construct a list of the hosts which
will run NIS servers. gauss is in the list of server hosts.
Please continue to add the names for the other hosts, one per
Line.  When you are done with the list, type a <Ctrl -D>.
       Next host to add: gauss
       Next host to add:
Is this correct? [y/n: y] y
We need some minutes to build the databases...
(…)
```

Este comando irá gerar a base de dados do NIS e atualizar os mapas. Depois disso, é preciso inicializar o servidor de senhas:

```
[root@gauss:~] # touch /etc/init.d/yppasswd start
Iniciando o YP passwd...OK
```

Este serviço permite que o usuário possa alterar sua senha em qualquer máquina da rede, através do comando `yppasswd`.

É importante ressaltar que se qualquer arquivo relacionado ao servidor for atualizado, os mapas devem ser gerados novamente, e para isso deve-se executar o comando `make` no diretório `/var/yp`. Por exemplo, se for acrescentado um usuário ao sistema:

```
[root@gauss:~] # cd /var/yp
[root@gauss yp]# make
```

Pode-se alterar a configuração padrão do servidor no arquivo `/etc/ypserv.conf`. Veja a documentação do NIS para mais detalhes de como fazê-lo.

10.3 Configurando o cliente

A primeira coisa a fazer é verificar se os pacotes ypbind e yp-tools estão instalados:

```
[root@tales ~]# rpm -qa | grep yp-tools
[root@tales ~]# rpm -qa | grep ypbind
```

Caso não, instale os pacotes seguindo os seguintes comandos:

```
[root@tales ~]# rpm -Uvh yp-tools*
[root@tales ~]# rpm -Uvh ypbind*
```

As máquinas cliente são configuradas pelo arquivo /etc/yp.conf. Ele é utilizado pelo ypbind para fazer a ligação com o servidor. O arquivo deve possuir uma das seguintes entradas válidas:

```
domain NISDOMAIN server HOSTNAME
```

Esse formato usa o nome do servidor (HOSTNAME) para o domínio NIS (NISDOMAIN).

```
domain NISDOMAIN broadcast
```

Esse formato faz um broadcast na rede para descobrir qual é o servidor do domínio NISDOMAIN.

```
ypserver HOSTNAME
```

Esse formato usa o servidor HOSTNAME, qualquer que seja o domínio.

Um exemplo de configuração:

```
domain alfamidia. broadcast
```

Depois disso, deve-se configurar o nome do domínio a que o cliente pertence. Isso é feito no arquivo `/etc/sysconfig/network`:

```
NISDOMAIN=domínio.
```

No nosso exemplo:

```
NISDOMAIN=alfamidia.
```

10.3.1 Executando os serviços no cliente

Por último, temos que inicializar os serviços do NIS no cliente:

```
[root@tales ~]# touch /etc/init.d/portmap start
Iniciando o portmapper...OK
[root@tales ~]# touch /etc/init.d/ypbind start
Ligando ao domínio NIS...OK
Aguardando por um servidor de domínio NIS: ... gauss.alfamidia.
```

10.3.2 Configurando o sistema de busca

Para que o NIS seja usado para busca de nomes de máquinas da rede, deve ser acrescentada a opção “`nis`” na llinha `order` do arquivo `/etc/host.conf`:

```
Order hosts,bind,nis
```

Para maior detalhes, consulte a documentação do `resolv`.

Opcionalmente, pode ser usado também o arquivo /etc/nsswitch.conf para configurar o sistema de buscas do NIS. Veja mais detalhes sobre o arquivo na sua documentação.

11 FTP

11.1 Introdução ao FTP

O FTP (*File Transfer Protocol*, ou Protocolo de Transferência de Arquivos) permite que o usuário realize transferências de arquivos entre a sua máquina e um ponto remoto da rede. A porta padrão utilizada pelo FTP é a 21.

A autenticação dos usuários é feita usando as próprias senhas do sistema. Existe também uma forma sem autenticação, chamada FTP Anônimo (*Anonymous* FTP), onde o usuário tem acesso apenas ao diretório home do usuário `ftp`.

O servidor ftp mais utilizado com o Linux é o wu-ftp.

11.2 Configuração do servidor FTP

Primeiro devemos verificar se o servidor FTP está ativo no arquivo `/etc/inetd.conf`:

```
ftp  stream  tcp  nowait  root  /usr/sbin/tcpd  in.ftpd  –l –a
```

Geralmente, basta descomentar a linha, removendo o carctere "#" do início.

O parâmetro "-l" ativa o log, o "-a" habilita o arquivo de configuração `/etc/ftpaccess`.

Pronto, já é possível acessar o sevidor ftp.

11.2.1 Arquivos de configuração adicionais

O principal arquivo de configuração do `ftpd` é o `/etc/ftpaccess`:

```
# Define uma classe de usuários: reais, convidados e anônimos
class   all         real,guest,anonymous    *
class   rmtusers    guest,anonymous         *
class   local       real                    *
# Para restringir o acesso de usuários reais, basta retirar o real da linha
anterior
# Para restringir o acesso de usuários convidados, basta retirar o guest da linha
anterior
# Para restringir o acesso de usuários anônimos, basta retirar o anonymous da
linha anterior
# É possível restringir o acesso de tipos de usuários de um certo domínio :
class   renegados     real    !*.domínio.com.br
# O "!" funciona como negação e renegados = o nome da classe

# Proíbe qualquer máquina do domínio conectiva.com.br de efetuar login e
# exibe o arquivo /etc/wu-ftpd/access_deny_msg
deny *.conectiva.com.br /etc/wu-ftpd/access_deny_msg

# Proíbe que as máquinas/domínios listados no arquivo
# /etc/wu-ftpd/deny_hosts efetuem login e exibe a mensagem contida no
# arquivo /etc/wu-ftpd/access_deny_msg
```

```
deny /etc/wu-ftpd/deny_hosts /etc/wu-ftpd/access_deny_msg

# Definição de grupos de convidados, mesmo sendo um usuário real, caso o
# grupo do mesmo esteja na lista de guestgroup a sessão de ftp dele será
# automaticamente passada para uma sessão anônima
guestgroup users

# Define um usuário convidado. No /etc/passwd deve estar definido o seu
# diretório root e também o $HOME deste usuário ao utilizar o<htmlurl url=" ftp."
name=" ftp."> Esses
# diretórios devem ser separados por "/./", por exemplo:
# convidado:senha:id:gid:Conta de FTP  - convidado:/ftp/./incoming:/etc/ftponly
guestuser convidado

# Definição de grupos/usuários reais, por exemplo, caso se tenha utiizado a
# opção guestgroup * mas ainda deseja-se que os usuários do grupo suporte
# queiram acessar como reais:
realgroup suporte
realuser  usuario

# Toda vez que um usuário da classe abaixo acessar o ftp, o processo do
# servidor ftp irá ter prioridade  = valor nice
nice <valor nice> [<classe que se logar>]

# Define o valor de umask para cada classe ou geral
defumask 011 rmtusers
defumask 022 local

# Segundos que o serviço espera pela vinda de uma conexão PASV
timeout accept 120

# Segundos que o serviço espera para realizar a conexão de dados com a porta
timeout connect 120

# Segundos que o serviço espera por atividades na conexão de dados
timeout data 1200

# Segundos que o serviço espera pelo próximo comando
timeout idle 900

# Segundos que o serviço espera pelo próximo comando (valor máximo, não pode
# ultrapassar). Caso o timeout idle ultrapasse o valor do maxidle, é sempre
# válido o maxidle

timeout maxidle 1200

# Segundos que o serviço disponibiliza para toda a conversação RFC931:
# <htmlurl url="http://andrew2.andrew.cmu.edu/rfc/rfc931.html"
name="http://andrew2.andrew.cmu.edu/rfc/rfc931.html">
```

```
timeout RFC931 10

# Quantidade de arquivos máxima que um usuário da classe pode transferir
# raw total: total de arquivos: entrada + saída
# raw out:   saída
# raw in:    entrada
file-limit raw total 50 rmtusers

# Quantidade de bytes máxima que um usuário da classe pode transferir
# raw total: total de arquivos: entrada + saída
# raw out:   saída
# raw in:    entrada
byte-limit

# Tempo máximo (em minutos) da conexão para usuários convidados, anônimos e ou os
dois:
# limit-time guest     60
# limit-time anonymous 60
# limit-time *         60
# Por esta opção, usuários reais não possuem tempo máximo de conexão
limit-time guest 10

# Não permite a transferência de arquivos.
# Sintaxe: noretrieve [absolute|relative] [class=<classname>] ... [ -] <filename>
<filename>
noretrieve class=rmtusers /home/*/.bashrc

# Após 3 tentativas de acesso sem sucesso, é guardado no log informações de
# que após 3 tentativas não foi possível o acesso e é terminada a conexão
loginfails 3

# Define o modo da mensagem de "boas-vindas" ao se conectar:
# full:  exibe o nome do servidor e qual serviço utiliza
#        (proftpd/wu-ftpd/etc)
# brief: exibe apenas o nome da máquina
# terse: exibe apenas a mensagem: "FTP server ready"
greeting full

# Define uma mensagem padrão de conexão  - NOTE: caso seja definido o tipo de
# mensagem (acima), o texto abaixo não irá aparecer
greeting text Servidor de FTP

# Logo após entrar com usuário e senha (com sucesso) será mostrado o
# conteúdo do arquivo.
banner /etc/ftp/banner

# Define o nome padrão do servidor ftp  - pode ser acessado através de %L,
# caso seja um servidor virtual, este valor não é utilizado
hostname lilica
```

```
# Define o email do administrador do servidor  - pode ser acessado por %E
email administrador@servidorftp.com.br

# Exibe o conteúdo do arquivo <path> quando o usu ário acessar o servidor
# <LOGIN> ou quando entrar em algum diretório <CWD>
# message <path> {<when> {<class> ...}}
message /etc/ftp/message CWD=/home/ftp/pub all

# No arquivo de mensagens, podem haver os seguintes "atalhos":
# %T      tempo (formato Thu  Nov 15 17:12:42 1990)
# %F      espaço livre na partição (kbytes) [não suportado em todos]
# %C      diretório local (equivale ao pwd)
# %E      email do administrador
# %R      nome da máquina que está acessando o servidor
# %L      nome do servidor
# %u      nome do usuário determinado via autenticação RFC931
# %U      nome do usuário que está logado (que foi digitado)
# %M      número máximo de usuários permitidos nesta classe
# %N      número de usuários correntes desta classe

...... Daqui prá baixo complicou a tradução ........
# %B      absolute limit on disk blocks allocated  #
# %b      preferred limit on disk blocks           #
# %Q      current block count                      #
# %I      maximum number of allocated inodes (+1)  #
# %i      preferred inode limit                    #
# %q      current number of allocated inodes       #
# %H      time limit for excessive disk use        #
# %h      time limit for excessive files           #
.................................................

# readme <path> {<when> {<class>}}
# Exibe uma indicação de que o usuário deve verificar um certo arquivo
# <path>
readme /home/ftp/README LOGIN all

# Guarda o log dos comandos dos usuários, dividido por tipos, real, guest,
# anonymous e separados por vírgulas
log commands guest,anonymous

# Guarda o log das transferências de arquivos de usuários anônimos,
# convidados e/ou reais.
# inbound = arquivos enviados ao servidor
# oubound = arquivos baixados do servidor
log transfers real,anonymous,guest inbound

# Guarda o log das violações das regras de segurança para usuários reais,
# anônimos ou convidados.
```

```
log security anonymous,guest

# Redireciona os avisos de transferências de arquivos para o syslog, sem isso,
# irá para xferlog
log syslog

# Redireciona os avisos de transferências de arquivos para o syslog e
# xferlog
log syslog+xferlog

# Permite utilizar-se de apelidos para diretórios
alias atualizacoes /pub/conectiva/atualizacoes

# Isto define um diretório de busca que seja usado ao mudar diretórios.
cdpath /pub/conectiva/atualizacoes
cdpath /pub/conectiva/beta

# O sistema irá verificar todo o tempo se o servidor será desligado, caso
# sim, o usuário que estiver acessando o servidor será avisado, novos
# acessos não serão permitidos antes de certo tempo ( definido no arquivo de
# configuração) de o servidor ser desligado.
shutdown /etc/ftp_shutdown
# Parâmetros para o arquivo:
# <ano> <mês> <dia> <hora> <minuto> <hora1> <hora2>
# <texto>
# onde
# <ano> qualquer ano > 1970
# <mês> 0-11
# <hora> 0-23
# <minuto> 0-59
# <hora1> e <hora2> no formato HHMM, definem a hora em que os usuários serão
# avisados do desligamento do servidor e quando será realizado o
# desligamento automático de usuários.
# <texto> Segue o mesmo padrão da tag "message" com as opções :
# %s hora em que o servidor será desligado
# %r hora em que conexões não serão mais permitidas
# %d hora em que as conexões serão desligadas

# Para a criação de domínios virtuais (Virtual hosts) é necessário que o seu
# servidor deve estar com seu nome cadastrado como válido no  servidor de
# nomes :
# virtual <address> <root|banner|logfile> <path>
virtual<htmlurl url=" ftp.suporte" name=" ftp.suporte"> root /home/ftp/suporte
banner /etc/ftp/sup_banner logfile xferlog_suporte

# Define os usuários que podem ou não ter acesso ao servidor virtual de<htmlurl
url=" ftp." name=" ftp.">
# virtual <address> allow <username> [<username> ...]
# virtual <address> deny <username> [<username> ...]
```

```
virtual<htmlurl url=" ftp.suporte" name=" ftp.suporte"> allow *
virtual<htmlurl url=" ftp.suporte" name=" ftp.suporte"> deny usuario

# Geralmente, é permitido o acesso de usuários anônimos em domínios
# virtuais, com a opção a seguir, é retirada esta permissão.
virtual<htmlurl url=" ftp.suporte" name=" ftp.suporte"> private

# Define os usuários que podem ou não ter acesso ao servidor de<htmlurl url="
ftp." name=" ftp.">
defaultserver allow usuario
defaultserver deny *

# Geralmente, é permitido o acesso de usuários anônimos em servidores ftp,
# com a opção a seguir, é retirada e sta permissão.
defaultserver private

# passive address <externalip> <cidr>
# Permite que certos intervalos de ip's acessem um certo servidor ao invés
# do padrão.
passive address 10.0.1.15   10.0.0.0/8
passive address 192.168.1.5 0.0.0.0/0

# Especifica o nome de um servidor de email, para onde será enviada as
# mensagens de notificação de "uploads". São aceitos vários servidores, onde
# o serviço de ftp irá mandar mensagens até que um dos servidores definidos
# aceite a mensagem.
mailserver mail.conectiva.com.br
mailserver mail.domínio.com.br
mailserver<htmlurl url=" www.emails.com" name=" www.emails.com">

# Define o email de quem irá receber informações sobre os "uploads"
# anônimos. Pode-se definir mais de um email, todos serão notificados. Caso
# não seja definido nenhum email, não haverá avisos.
incmail usuario@conectiva.com.br
incmail usuario@hotmail.com

# Define o email de quem irá receber informações sobre os "uploads" anônimos
# do domínio virtual. Caso não seja definido, serão enviadas notificaçõ es
# para o email geral.
virtual<htmlurl url=" ftp.suporte" name=" ftp.suporte"> incmail
usuario@ftp.suporte

# Define o email de quem irá receber informações sobre os "uploads"
# anônimos no servidor padrão (excluindo os virtuais).
defaultserver incmail usuario@conectiva.com.br

# Define o email do usuário que enviará a mensagem
mailfrom anonftp@conectiva.com.br
```

```
# Define o email do usuário que enviará a mensagem para o servidor virtual
virtual<htmlurl url=" ftp.suporte" name=" ftp.suporte"> mailfrom
anonftp@ftp.suporte

# Define o email geral do usuário que enviará a mensagem (apenas do servidor
# padrão)
defaultserver mailfrom anonftp@conectiva.com.br

# Permite ou não executar a especificada função. Por padrão, todas são
# permitidas. Pode-se utilizar classes de usuários, bastando seguir a
# sintaxe class=nome_da_classe
# chmod <yes|no> <typelist>
chmod no anonymous
# delete <yes|no> <typelist>
delete yes real,guest,anonymous
# overwrite <yes|no> <typelist>
overwrite yes real,guest,anonymous
# rename <yes|no> <typelist>
rename no guest
# umask <yes|no> <typelist>
umask no real

# Define o nível de checagem das senhas do usuário anônimo.
# passwd-check <none|trivial|rfc822> (<enforce|warn>)
# none, não realiza checagem alguma
# trivial, deve haver "@"
# rfc822, a senha deve ser compatível com a regra rfc822:
# <htmlurl url="http://www.faqs.org/rfcs/rfc822.html"
name="http://www.faqs.org/rfcs/rfc822.html">
# enforce, caso a senha não seja aceita, não realiza o acesso
# warn, caso a senha não seja aceita, apenas avisa o usuário e então acessa
passwd-check rfc822 enforce

# Define as senhas que não serão aceitas (dependerá da configuração acima se
# o usuário poderá ou não acessar o servidor)
deny-email usuario@conectiva.com.br
deny-email info@conectiva.com.br
deny-email suporte@conectiva.com.br

# Define se um diretório tem ou não permissão para realizar "uploads"
# upload [absolute|relative] [class=<classname>]... [ -] <root-dir> <dirglob>
# <yes|no> <owner> <group> <mode> ["dirs"|"nodirs"] [<d_mode>]
# Não permitir upload
upload /var/ftp *              no
# Permitir upload no diretório incoming
upload /var/ftp /incoming      yes ftp daemon 0666
# Permitir upload e criação de subdiretórios no diretório incoming/gifs
upload /var/ftp /incoming/gifs yes jlc guest  0 600 nodirs
```

```
# Define a taxa máxima de transferência para determinados arquivos, caso
# utilize-se "*" é válido como todos os arquivos do diretório definido
throughput /e/ftp *    *      oo    -    *
# Taxa de transferência máxima para todos os arquivos local izados no
# diretório /e/ftp
throughput /e/ftp /sw* *      1024 0.5 *
# Taxa de transferência máxima = 1024 bps para todos os arquivos localizados
# em /e/ftp/sw
throughput /e/ftp /sw* README oo    -    *
# Taxa de transferência máxima para arquivos chamados  README
throughput /e/ftp /sw* *      oo    -    *.foo.com
# Taxa de transferência máxima para usuários do domínio foo.com
# Os caraceteres "oo" significam que não há restrição e " -","0.5" são os
# fatores por qual será multiplicado o valor de bps definido p ara cada nova
# transferência realizada no diretório indicado

# Define o diretório raiz (/) de usuários anônimos, onde real e rmtusers são
# classes de usuários
anonymous-root /home/ftp/uploads real
anonymous-root /home/ftp/pub rmtusers

# Define o diretório raiz de usuários convidados, pode ser passado o
# parâmetro de um intervalo de id's dos usuários
guest-root /home/users
# Define para todos os usuários o diretório raiz = /home/users
guest-root /home/staff %100-999 usuario
# Define para os usuários com  id >= 100 e <= 900 e o usuário usuario a raiz
# = /home/staff
guest-root /home/users/barea/ftp barea
# Define para o usuário barea o diretório raiz = /home/users/barea

# Permite ou não o acesso de usuários através de seus id's, gid's ou nome
# (usuário e grupo)
deny-gid %-99 %65535
deny-uid %-99 %65535
allow-gid ftp
allow-uid ftp

# Permite ou não o acesso de usuários além de seu home
restricted-uid usuario
unrestricted-gid root
```

Outro arquivo importante é o `/etc/ftpconversions`, que define as conversões automáticas de arquivos:

```
:.Z:  :  :/bin/compress -d -c %s:T_REG|T_ASCII:O_UNCOMPRESS:UNCOMPRESS
:   : :.Z:/bin/compress -c %s:T_REG:O_COMPRESS:COMPRESS
:.gz: :  :/bin/gzip -cd %s:T_REG|T_ASCII:O_UNCOMPRESS:GUNZIP
:   : :.gz:/bin/gzip -9 -c %s:T_REG:O_COMPRESS:GZIP
:   : :.tar:/bin/tar -c -f - %s:T_REG|T_DIR:O_TAR:TAR
```

```
:   : :.tar.Z:/bin/tar -c -Z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+COMPRESS
:   : :.tar.gz:/bin/tar -c -z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+GZIP
```

Sintaxe do arquivo:

Campo	Descrição
1	strip prefix
2	strip postfix
3	addon prefix
4	addon postfix
5	external command
6	types
7	options
8	description

Por último veremos o arquivo /etc/ftpusers, que é uma lista dos nomes dos usuários que não podem realizar ftp

```
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

11.2.2 FTP Anônimo

Como já foi mencionado, quando um usuário conecta-se como anonymous, ele só tem acesso ao diretório `/home/ftp`. Isso faz com que ele não consiga acessar alguns arquivos binários e de configuração do sistema, que são essenciais para a utilização do ftp. Por isso, esses arquivos devem ser copiados para dentro do diretório `/home/ftp`.

Em primeiro lugar, é necessário criar alguns diretórios:

- `/home/ftp`: esta é a raiz do usuário anônimo. O dono do diretório deve ser `root:root`, e nenhum outro usuário deve ter permissão de escrita.
- `/home/ftp/pub`: aqui serão armazenados os arquivos para *download* e *upload*. O dono do diretório deve ser `root:root`. Este

diretório pode ter permissão 777 ou 733, dependendo de o usuário desejar ou não o upload de arquivos no servidor.

- `/home/ftp/bin`: este diretório vai abrigar os arquivos executáveis do servidor. O dono do diretório deve ser `root:root`, e nenhum outro usuário deve ter permissão de escrita. Este diretório vai conter uma cópia do binário do comando `ls`, que deve ter as permissões setadas para `111`.
- `/home/ftp/lib`: este diretório vai abrigar os arquivos de bibliotecas utilizadas pelos programas que estão em `/home/ftp/bin`. O dono do diretório deve ser `root:root`, e nenhum outro usuário deve ter permissão de escrita. Para verificar a dependência de bibliotecas de um determinado programa, pode-se utilizar o `ldd`. No caso do `ls`, deve-se copiar as bibliotecas `libc.so.6` e `ld-linux.so.2`, ambas localizadas em `/lib`.
- `/home/ftp/etc`: este diretório vai abrigar os arquivos executáveis do servidor. O dono do diretório deve ser `root:root`, e nenhum outro usuário deve ter permissão de escrita. Esse diretório deve conter os arquivos `passwd` e `group`, para que o comando `ls` possa exibir os donos dos arquivos. Entretanto, não se deve usar os arquivos do diretório /etc, para evitar exibir informações sobre os usuários do sistema.

  Exemplo de arquivo `passwd` para colocar em `/home/ftp/etc`:

```
root:*:0:0:::
bin:*:1:1:::
operator:*:11:0:::
ftp:*:14:50:::
nobody:*:99:99:::
```

  Exemplo de arquivo `group` para colocar em `/home/ftp/etc`:

```
root::0:
bin::1:
daemon::2:
sys::3:
adm::4:
ftp::50:
nowhere::99:
```

Exercícios

1. Implemente um servidor de ftp anônimo na sua máquina.

12 SEGURANÇA DA REDE E CONTROLE DE ACESSO

Adaptado do Guia Foca GNU/Linux Avançado – Capítulo 4.

Deixe-me iniciar esta seção lhe alertando que a segurança da rede em sua máquina e ataques maliciosos são uma arte complexa. Uma regra importante é: "Não ofereça serviços de rede que não deseja utilizar".

Muitas distribuições vem configuradas com vários tipos de serviços que são iniciados automaticamente. Para melhorar, mesmo que insignificantemente, o nível de segurança em seu sistema você deve editar se arquivo /etc/inetd.conf e comentar (colocar uma "#") as linhas que contém serviços que não utiliza.

Bons candidatos são serviços tais como: shell, login, exec, uucp, ftp e serviços de informação tais como finger, netstat e sysstat.

Existem todos os tipos de mecanismos de segurança e controle de acesso, eu descreverei os mais importantes deles.

12.1 /etc/ftpusers

O arquivo /etc/ftpusers é um mecanismo simples que lhe permite bloquear a conexão de certos usuários via *ftp*. O arquivo /etc/ftpusers é lido pelo programa daemon ftp (*ftpd*) quando um pedido de conexão é recebido. O arquivo é uma lista simples de usuários que não tem permissão de se conectar. Ele se parece com:

```
# /etc/ftpusers - login de usuários bloqueados via ftp
root
uucp
bin
mail
```

12.2 /etc/securetty

O arquivo /etc/securetty lhe permite especificar que dispositivos tty que o usuário *root* pode se conectar. O arquivo /etc/securetty é lido pelo programa login (normalmente /bin/login). Seu formato é uma lista de dispositivos tty onde a conexão é permitida, em todos os outros, a entrada do usuário *root* é bloqueada.

```
# /etc/securetty - terminais que o usuário root pode se conectar
tty1
tty2
tty3
tty4
```

12.3 O mecanismo de controle de acess os tcpd

O programa tcpd que você deve ter visto listado no mesmo arquivo /etc/inetd.conf, oferece mecanismos de registro e controle de acesso para os serviços que esta configurado para proteger. Ele é um tipo de firewall simples

e fácil de configurar que pode evitar tipos indesejados de ataques e registrar possíveis tentativas de invasão.

Quando é executado pelo programa inetd, ele lê dos arquivos contendo regras de acesso e permite ou bloqueia o acesso ao servidor protegendo adequadamente.

Ele procura nos arquivos de regras até que uma regra confira. Se nenhuma regra conferir, então ele assume que o acesso deve ser permitido a qualquer um. Os arquivos que ele procura em seqüência são: /etc/hosts.allow e /etc/hosts.deny. Eu descreverei cada um destes arquivos separadamente.

Para uma descrição completa desta facilidade, você deve verificar a página de manual apropriada (hosts_access (5) é um bom ponto de partida).

12.3.1 /etc/hosts.allow

O arquivo /etc/hosts.allow é um arquivo de configuração do programa /usr/sbin/tcpd. O arquivo hosts.allow contém regras descrevendo que hosts tem permissão de acessar um serviço em sua máquina.

O formato do arquivo é muito simples:

```
# /etc/hosts.allow
#
# lista de serviços: lista de hosts : comando
```

lista de serviços

É uma lista de nomes de serviços separados por vírgula que esta regra se aplica. Exemplos de nomes de serviços são: ftpd, telnetd e fingerd.

lista de hosts

É uma lista de nomes de hosts separada por vírgula. Você também pode usar endereços IP's aqui. Adicionalmente, você pode especificar nomes de computadores ou endereço IP usando caracteres coringas para atingir grupos de hosts.

Exemplos incluem: gw.vk2ktj.ampr.org para conferir com um endereço de computador específico, .uts.edu.au para atingir qualquer endereço de computador finalizando com aquele string. Use 200.200.200. para conferir com qualquer endereço IP iniciando com estes dígitos. Existem alguns parâmetros especiais para simplificar a configuração, alguns destes são: ALL atinge todos endereços, LOCAL atinge qualquer computador que não contém um "." (ie. está no mesmo domínio de sua máquina) e PARANOID atinge qualquer computador que o nome não confere com seu endereço (falsificação de nome). Existe também um último parâmetro que é também útil: o parâmetro EXCEPT lhe permite fazer uma lista de exceções. Isto será coberto em um exemplo adiante.

comando

É um parâmetro opcional. Este parâmetro é o caminho completo de um comando que deverá ser executado toda a vez que esta regra conferir. Ele pode executar um comando para tentar identificar quem esta conectado pelo host remoto, ou gerar uma mensagem via E-Mail ou algum outro alerta para um administrador de rede que alguém está tentando se conectar.

Existem um número de expansões que podem ser incluídas, alguns exemplos comuns são: %h expande o endereço do computador que está

conectado ou endereço se ele não possuir um nome, %d o nome do daemon sendo chamado.

Se o computador tiver permissão de acessar um serviço através do /etc/hosts.allow, então o /etc/hosts.deny não será consultado e o acesso será permitido.

Como exemplo:

```
# /etc/hosts.allow
#
# Permite que qualquer um envie e-mails
in.smtpd: ALL
# Permitir telnet e ftp somente para hosts locais e myhost.athome.org. au
in.telnetd, in.ftpd: LOCAL, myhost.athome.org.au
# Permitir finger para qualquer um mas manter um registro de quem é
in.fingerd: ALL: (finger @%h | mail -s "finger from %h" root)
```

Qualquer modificação no arquivo /etc/hosts.allow entrará em ação após reiniciar o daemon *inetd*. Isto pode ser feito com o comando kill -HUP [pid do inetd], o pid do *inetd* pode ser obtido com o comando ps ax|grep inetd.

12.3.2 /etc/hosts.deny

O arquivo /etc/hosts.deny é um arquivo de configuração das regras descrevendo quais computadores não tem a permissão de acessar um serviço em sua máquina.

Um modelo simples deste arquivo se parece com isto:

```
# /etc/hosts.deny
#
# Bloqueia o acesso de computadores com endereços suspeitos
ALL: PARANOID
#
# Bloqueia todos os computadores
ALL: ALL
```

A entrada PARANOID é realmente redundante porque a outra entrada nega tudo. Qualquer uma destas linhas pode fazer uma segurança padrão dependendo de seu requerimento em particular.

Tendo um padrão *ALL: ALL* no arquivo */etc/hosts.deny* e então ativando especificamente os serviços e permitindo computadores que você deseja no arquivo /etc/hosts.allow é a configuração mais segura.

Qualquer modificação no arquivo /etc/hosts.deny entrará em ação após reiniciar o daemon *inetd*. Isto pode ser feito com o comando kill -HUP [pid do inetd], o pid do *inetd* pode ser obtido com o comando ps ax|grep inetd.

12.3.3 /etc/hosts.equiv e /etc/shosts.equiv

O arquivo /etc/hosts.equiv é usado para garantir/bloquear certos computadores e usuários o direito de acesso aos serviços "r*" (rsh, rexec, rcp, etc) sem precisar fornecer uma senha. O /etc/shosts.equiv é equivalente mas é lido somente pelo serviço ssh. Esta função é útil em um ambiente seguro onde você controla todas as máquinas, mesmo assim isto é um perigo de segurança (veja nas observações). O formato deste arquivo é o seguinte:

```
#Acesso  Máquina                  Usuário
```

```
-       maquina2.dominio.com.br   usuario2
-       maquina4.dominio.com.br   usuario2
+       maquina1.dominio.com.br   +@usuarios
```

O primeiro campo especifica se o acesso será permitido ou negado caso o segundo e terceiro campo confiram. Por razões de segurança deve ser especificado o FQDN no caso de nomes de máquinas. Grupos de rede podem ser especificados usando a sintaxe "+@grupo".

Para aumentar a segurança, não use este mecanismo e encoraje seus usuários a também não usar o arquivo .rhosts.

ATENÇÃO O uso do sinal "+" sozinho significa permitir acesso livre a qualquer pessoa de qualquer lugar. Se este mecanismo for mesmo necessário, tenha muita atenção na especificação de seus campos.

Evita também A TODO CUSTO uso de nomes de usuários (a não ser para negar o acesso), pois é facil forjar o login, entrar no sistema tomar conta de processos (como por exemplo do servidor Apache rodando sob o usuário www-data ou até mesmo o **root**), causando enormes estragos.

12.3.4 Verificando a segurança do TCPD e a sintaxe dos arquivos

O utilitário tcpdchk é útil para verificar problemas nos arquivos hosts.allow e hosts.deny. Quando é executado ele verifica a sintaxe destes arquivos e relata problemas, caso eles existam.

Outro utilitário útil é o tcpdmatch, o que ele faz é permitir que você simule a tentativa de conexões ao seu sistema e observar ser ela será permitida ou bloqueada pelos arquivos hosts.allow e hosts.deny.

É importante mostrar na prática como o tcpdmatch funciona através de um exemplo simulando um teste simples em um sistema com a configuração padrão de acesso restrito:

- O arquivo hosts.allow contém as seguintes linhas:

```
ALL: 127.0.0.1
in.talkd, in.ntalkd: ALL
in.fingerd: 192.168.1.  EXCEPT 192.168.1.30
```

A primeira linha permite o loopback (127.0.0.1) acessar qualquer serviço TCP/UDP em nosso computador, a segunda linha permite qualquer um acessar os servidor TALK (nós desejamos que o sistema nos avise quando alguém desejar conversar) e a terceira somente permite enviar dados do finger para computadores dentro de nossa rede privada (exceto para 192.168.1.30).

- O arquivo hosts.deny contém a seguinte linha:

```
ALL: ALL
```

Qualquer outra conexão será explicitamente derrubada.

Vamos aos testes, digitando: "tcpdmatch in.fingerd 127.0.0.1" (verificar se o endereço 127.0.0.1 tem acesso ao finger):

```
client:   address  127.0.0.1
server:   process  in.fingerd
matched:  /etc/hosts.allow line 1
access:   granted
```

Ok, temos acesso garantido com especificado pela linha 1 do hosts.allow (a primeira linha que confere é usada). Agora "tcpdmatch in.fingerd 192.168.1.29":

```
client:   address  192.168.1.29
server:   process  in.fingerd
matched:  /etc/hosts.allow line 3
access:   granted
```

O acesso foi permitido através da linha 3 do hosts.allow. Agora "tcpdmatch in.fingerd 192.168.1.29":

```
client:   address  192.168.1.30
server:   process  in.fingerd
matched:  /etc/hosts.deny line 1
access:   denied
```

O que aconteceu? como a linha 2 do hosts.allow permite o acesso a todos os computadores 192.168.1.* exceto 192.168.1.30, ela não bateu, então o processamento partiu para o hosts.deny que nega todos os serviços para qualquer endereço. Agora um último exemplo: "tcpdmatch in.talkd www.debian.org"

```
client:   address  www.debian.org
server:   process  in.talkd
matched:  /etc/hosts.allow line 2
access:   granted
```

Ok, na linha 2 qualquer computador pode te chamar para conversar via talk na rede, mas para o endereço DNS conferir com um IP especificado, o GNU/Linux faz a resolução DNS, convertendo o endereço para IP e verificando se ele possui acesso.

No lugar do endereço também pode ser usado a forma daemon@computador ou cliente@computador para verificar respectivamente o acesso de daemons e cliente de determinados computadores aos serviços da rede.

Como pode ver o TCPD ajuda a aumentar a segurança do seu sistema, mas não confie nele além do uso em um sistema simples, é necessário o uso de um firewall verdadeiro para controlar minuciosamente a segurança do seu sistema e dos pacotes que atravessam os protocolos, roteamento e as interfaces de rede. Se este for o caso aprenda a trabalhar a fundo com firewalls e implemente a segurança da sua rede da forma que melhor planejar.

12.4 Firewall

Dentre todos os métodos de segurança, o *Firewall* é o mais seguro. A função do Firewall é bloquear determinados tipos de tráfego de um endereço ou para uma porta local ou permitir o acesso de determinados usuários mas bloquear outros, bloquear a falsificação de endereços, redirecionar tráfego da rede, ping da morte, etc.

A implementação de um bom firewall dependerá da experiência, conhecimentos de rede (protocolos, roteamento, interfaces, endereçamento, masquerade, etc), da rede local, e sistema em geral do Administrador de redes, a segurança de sua rede e seus dados dependem da escolha do profissional

correto, que entenda a fundo o TCP/IP, roteamento, protocolos, serviços e outros assuntos ligados a rede.

Freqüentemente tem se ouvido falar de empresas que tiveram seus sistemas invadidos, em parte isto é devido a escolha do sistema operacional indevido mas na maioria das vezes o motivo é a falta de investimento da empresa em políticas de segurança, que algumas simplesmente consideram a segurança de seus dados e sigilo interno como uma despesa a mais.

Um bom firewall que recomendo é o ipchains, Sinus e o TIS. Particularmente gosto muito de usar o ipchains e o Sinus e é possível fazer coisas inimagináveis programando scripts para interagirem com estes programas

13 TRABALHANDO COM O X-WINDOW EM REDE

O X-Window permite que o usuário execute a aplicação em uma máquina e os resultados sejam mostrados em outra. Este recurso é útil quando um usuário usando o X deseja executar uma aplicação remotamente, sendo que a interface gráfica seja apresentada no seu vídeo.

13.1 O Display

Todo servidor X possui um nome de *display*, no seguinte formato:

```
host:display.screen
```

O `host` especifica o nome da máquina em que o display está fisicamente conectado. Se for omitido, será assumido o *localhost*.

O `display` indica o número do terminal gráfico, normalmente é 0, mas pode ser maior se houver mais de um terminal gráfico conectado à máquina.

O `screen` indica o número do monitor, e normalmente também é 0, podendo ser maior se houver mais de um monitor conectado ao terminal gráfico.

O gerenciador X-Window geralmente abre as aplicações no display indicado na variável de ambiente `DISPLAY`, que por padrão tem o valor ":0.0". O usuário pode mudar esta variável se quiser que as aplicações sejam mostradas em um outro display:

```
DISPLAY=gauss.alfamidia:0.0
```

13.2 O servidor X-Window

O servidor X-Window normalmente não aceita conexões de outras máquinas. Se o usuário permitir que todas as máquinas da rede executem aplicações no seu display, estará permitindo também que outros usuários possam ler o pressionamento de teclas, ações do mouse, etc. Ou seja, permitir acesso ao display pode acarretar riscos de segurança.

A solução é autenticar as conexões, permitindo que somente determinados usuários tenham acesso ao seu servidor X-Window. O xhost permite acesso baseado no nome da máquina. O servidor mantém uma lista de máquinas que têm permissão para conectar-se ao seu display.

A lista de máquinas pode ser controlada pelo programa `xhost`. Por exemplo, para permitir que o resultado de um programa executado em `tales.alfamidia` seja exibido em `gauss.alfamidia`, ou seja, que o cliente X-Window em `tales` acesse o servidor X-Window em `gauss`, deve-se fazer:

```
[aluno@gauss ~]$ xhost +tales.alfamidia
```

Para revogar a permissão:

```
[aluno@gauss ~]$ xhost -tales.alfamidia
```

Pode-se também ser desabilitar a verificação de máquinas, permitindo que qualquer máquina da rede conecte-se ao seu display:

```
[aluno@gauss ~]$ xhost +
```

Isto nunca deve ser feito em uma rede onde não se confie em todos os usuários. Para habilitar novamente a verificação:

```
[aluno@gauss ~]$ xhost -
```

Para verificar a lista de máquinas, basta usar o comando sem nenhum parâmetro:

```
[aluno@gauss ~]$ xhost
```

O `xhost` não é um mecanismo totalmente seguro, já que não consegue distinguir usuários diferentes em uma mesma máquina remota.

13.3 O cliente X-Window

O programa cliente sabe em que *display* deve se conectar, através da variável de ambiente `DISPLAY`. Esta definição pode ser ignorada com a utilização do parâmentro `–display host:display.screen` na inicialização do programa cliente.

Supondo que o usuário `aluno` já esteja conectado à maquina `tales` através de `telnet` ou `ssh`, e queira executar o `xfontsel`, com saída na máquina local:

```
[aluno@tales ~]$ export DISPLAY="gauss.alfamidia:0.0"
[aluno@tales ~]$ xfontsel
```

Ou então:

```
[aluno@tales ~]$ xfontsel –display gauss.alfamidia:0.0
```

Todo o processamento do xfontsel será feito na máquina remota, e apenas a interface é apresentada na máquina do usuário.

13.4 Identificação e encerramento de aplicativos X-Window

Para uma lista de aplicações que está sendo executada no display local, use o comando `xlsclients`.

As aplicações locais podem ser interrompidas especificando seu PID (identificador do processo) ao comando `kill`.

As aplicações locais ou remotas podem ser interrompidas usando o comando xkill, que é um utilitário interativo que serve para fechar conexões com os clientes:

```
[aluno@gauss ~]$ xkill
Select the window whose client you wish to kill with button 1...
```

Será necessário clicar com o botão esquerdo do mouse dentro da janela que se deseja interromper.

Exercícios

1. Analise a seguinte seqüência de comandos:

```
[aluno@gauss ~]$ xhost +tales.alfamidia
tales being added to access control list
[aluno@gauss ~]$ ssh tales.alfamidia
password:
[aluno@tales ~]$ export DISPLAY="gauss .alfamidia:0.0"
```

Marque Verdadeiro ou Falso nas alternativas a seguir:

a) A máquina `tales` pode receber conexões X de `gauss`.

b) A máquina `gauss` pode receber conexões X de `tales`.
c) Qualquer display pode ser acessado pelo usuário `aluno` na máquina `tales`.
d) O usuário aluno pode executar aplicativos gráficos localmente em `gauss`.
e) O usuário aluno pode executar aplicativos gráficos localmente em `tales`.

2. Execute o aplicativo `xfontsel` remotamente, usando o `ssh` para logar-se em outra máquina da rede.

3. Utilize o `xkill` para matar a conexão com o `xfontsel` do exercício anterior.

APÊNDICE A. LICENÇA DE PUBLICAÇÃO LIVRE

Esta é uma tradução não-oficial da Open Publication License versão 1.0, de 8 de junho de 1999, e não é substituto legal para a Licença original, disponível em http://www.opencontent.org/openpub. Entretanto, esta tradução poderá auxiliar pessoas que falem Português a entender melhor a licença. É permitido a qualquer pessoa copiar e distribuir cópias desse documento de licença, desde que sem a implementação de qualquer mudança.

OPEN PUBLIC LICENSE

Draft v1.0, 8 june 1999

I. Requisitos comuns às versões modificadas e não modificadas

Os trabalhos protegidos pela Licença de Livre Publicação (Open Publication License) podem ser reproduzidos e distribuídos no todo ou em parte, em qualquer meio físico ou eletrônico, desde que os termos desta licença estejam incluídos, e que esta licença ou uma incorporação dela por referência (com quaisquer das opções escolhidas pelo autor ou editor) estejam presentes na reprodução.

A forma apropriada para uma incorporação por referência deste livro é:

> Copyright© 2002 Alfamídia Ltda. Este material somente poderá ser distribuído se sujeito aos termos e condições firmados na Licença de Livre Publicação (Open Publication License), versão 1.0 ou superior (a versão mais atual encontra-se disponível em http://www.opencontent.org/openpub/).

Esta referência, devidamente preenchida com os dados da publicação, deve ser seguida imediatamente com quaisquer opções escolhidas pelos autores ou editor do documento (consultar a seção *Termos opcionais*).

É permitida a redistribuição comercial de material licenciado pela Licença de Livre Publicação (Open Publication License).

Qualquer publicação no formato livro padrão (papel) requer obrigatoriamente a citação dos autores e editor originais. Os nomes dos autores e do editor devem aparecer em todas as superfícies externas do livro. Em todas as faces externas do livro, o nome do editor original deve estar impresso em tamanho tão grande quanto o título do trabalho, e citado como proprietário em relação àquele título.

II. Copyright

O *copyright* de todo trabalho protegido pela Licença de Livre Publicação (Open Publication License) pertence aos autores ou proprietários.

III. Escopo da licença

Os termos de licença a seguir aplicam-se a todos os trabalhos protegidos pela Licença de Livre Publicação (Open Publication License), a não ser que explicitamente indicado no trabalho.

A mera adição de trabalhos protegidos pela Licença de Livre Publicação (Open Publication License) ou partes de trabalhos protegidos pela Licença de Livre Publicação (Open Publication License) em uma mesma mídia que contenha outros trabalhos ou programas não protegidos por essa licença não decorre em aplicação da Licença de Livre Publicação (Open Publication License) para esses outros trabalhos. O trabalho resultante deve explicitamente conter uma nota especificando a inclusão do material protegido pela Licença de Livre Publicação (Open Publication License) e o aviso de copyright apropriado.

APLICABILIDADE. Se alguma parte desta licença não puder ser aplicada em alguma jurisdição, as partes restantes deste documento continuam sendo aplicadas.

AUSÊNCIA DE GARANTIA. Os trabalhos protegidos pela Licença de Livre Publicação (Open Publication License) são fornecidos "como estão", sem garantias de qualquer tipo, explícita ou implícita, incluindo, mas não limitado a, as garantias implícitas de comercialização e conveniência para um propósito particular, ou garantia de não-infração.

IV. Requisitos para trabalhos modificados

Todas as versões modificadas de documentos cobertos por esta licença, incluindo traduções, antologias, compilações e documentação parcial, deve seguir os requisitos abaixo:

A versão modificada deve ser indicada como tal.

As pessoas que fizerem as modificações e as datas de modificação devem ser identificadas.

O reconhecimento dos autores e editor originais (se aplicável) deve ser mantido de acordo com as práticas acadêmicas usuais de citação.

O local da versão não-modificada do documento deve ser indicado.

Os nomes originais dos autores não devem ser utilizados para indicar ou garantir seu endosso ao documento resultante sem a autorização expressa dos autores.

V. Práticas recomendadas

Em adição aos requisitos desta licença, é solicitado e extremamente recomendado aos redistribuidores que:

Se os trabalhos protegidos pela Licença de Livre Publicação (Open Publication License) estiverem sendo distribuídos em impressos ou CD-ROM, os autores sejam informados por email, ao menos trinta dias antes, para que os autores tenham tempo de providenciar documentação atualizada. Esta notificação deve descrever as modificaçoes introduzidas no documento, se existirem.

Todas as modificações substanciais (incluindo exclusões) devem ser marcadas claramente no documento, ou então descritas em um anexo ao documento.

Finalmente, mesmo não sendo obrigatório sob esta licença, é considerado de bom tom oferecer uma cópia sem ônus de todo o material modificado (impresso e CD-ROM) para os autores originais.

VI. Termos opcionais

Os autores e editores de documentos protegidos pela Licença de Livre Publicação (Open Publication License) podem escolher certas opções de licença simplesmente incluindo alguns parágrafos após a cópia da licença ou sua referência. Estas opções são consideradas parte da licença e devem ser incluídas com ela (ou com a referência a ela) nos trabalhos derivados.

As opções que se aplicam a este trabalho são:

A:É vedada a distribuição de versões com modificações substanciais deste documento sem a expressa permissão dos proprietários do direito autoral.

B:É vedada a distribuição deste trabalho ou qualquer derivado seu em qualquer formato de livro padrão (papel) sem a prévia autorização dos proprietários do direito autoral.

Políticas de Publicação Livre

(O texto a seguir não é considerado parte da licença.)

Os trabalhos protegidos pela Licença de Livre Publicação (Open Publication License) estão disponíveis e podem ser acessados na home page da Open Publication http://works.opencontent.org/ .

Os autores de trabalhos protegidos pela Licença de Livre Publicação (Open Publication License) podem incluir suas próprias licenças nesses trabalhos, desde que os termos dessa licença não sejam mais restritrivos que os da Licença de Livre Publicação (Open Publication License).

Em caso de dúvidas sobre a Licença de Livre Publicação (Open Publication License), contactar David Wiley <dw2@opencontent.org> ou a lista de autores de publicações <opal@opencontent.org> via email.

Para se inscrever na lista de autores de publicações livres (Open Publication Author's List), mande um email para <opal-request@opencontent.org> com a palavra **subscribe** no corpo da mensagem.

Para enviar mensagens para a lista de autores de publicações livres (Open Publication Author's List), mande um email para **opal@opencontent.org** ou simplesmente responda a uma mensagem postada.

Para se desinscrever na lista de autores de publicações livres (Open Publication Author's List), mande um email para **opal-request@opencontent.org** com a palavra **unsubscribe** no corpo da mensagem.