Guia Foca GNU/Linux
Capítulo 15 - Rede

Este capítulo descreve o que é uma rede, os principais dispositivos de rede no `GNU/Linux`, a identificação de cada um, como configurar os dispositivos, escolha de endereços IP, roteamento.

Parte deste capítulo, uns 70% pelo menos, é baseado no documento NET3-4-HOWTO. (seria perda de tempo reescrever este assunto pois existe um material desta qualidade já disponível).

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

Com os ítens 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, pontes, etc.

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*.

15.2 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 (como é o caso do NetBios) outros para redes mundiais (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.

15.3 Endereço IP

O *endereço IP* são números que identificam seu computador em uma rede. Inicialmente você pode imaginar o IP como um número de telefone. O IP é compostos 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's. Todos os endereços dentro da rede tem um número de dígitos dentro dos endereços em comum. A porção dos endereços que são comuns entre todos os endereços de uma rede são chamados de *porção da rede*. Os dígitos restantes são chamados de *porção dos hosts*. O número de bits que são compartilhados por todos os endereços dentro da rede são chamados de *netmask*(máscara da rede) e o papel da *netmask* é determinar quais endereços pertencem ou não a rede. Por exemplo, considere o seguinte:

```
---------------- ---------------

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* que pertence. O endereço e 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 recebe-las simultaneamente.
Existe 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 tem 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*.

15.3.1 Classes de Rede IP

Por razões administrativas após algum 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:

```
+-----------------------------------------------------+

| Classe | Máscara de | Endereço da Rede |

| | Rede | |

+-----------------------------------------------------+

| A | 255.0.0.0 | 0.0.0.0 - 127.255.255.255 |

| B | 255.255.0.0 | 128.0.0.0 - 191.255.255.255 |

| C | 255.255.255.0 | 192.0.0.0 - 223.255.255.255 |

|Multicast| 240.0.0.0 | 224.0.0.0 - 239.255.255.255 |

+-----------------------------------------------------+
```

O tipo de endereço que você deve utilizar depende exatamente do que estiver fazendo.

15.3.2 Para instalar uma máquina usando o Linux em uma rede existente

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

- Endereço IP de sua 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 escolhe-los e esperar que sua configuração funcione.

15.3.3 Endereços reservados para uso em uma rede Privada

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:

```
+-------------------------------------------------------+

| ENDEREÇOS RESERVADOS PARA REDES PRIVADAS |

+-------------------------------------------------------+

| Classe | Máscara de | Endereço da Rede |

| de Rede | 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.

15.4 Interface de rede

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 Linux (a ? 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
- `eql` - Balanceador de tráfego para múltiplas linhas
- `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.
- `nr?` - Interface Net Rom
- `rs?` - Interfaces Rose
- `st?` - Interfaces Strip (Starmode Radio IP)
- `tr?` - Token Ring

Para maiores detalhes sobre as interfaces acima, consulte o documento *NET3-4-HOWTO*.

15.4.1 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 padrões de inicialização existentes em sua distribuição).

```
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`.

15.4.2 Atribuindo um endereço de rede a uma interface (ifconfig)

Após 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:

```
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:

```
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` Em sistemas `Debian`, o arquivo correto para especificar os dados das interfaces é o `/etc/network/interfaces` (veja Arquivo `/etc/network/interfaces`, Seção 27.8).
Para mais detalhes, veja a página de manual do `ifconfig` ou o *NET3-4-HOWTO*.

15.5 Roteamento

Roteamento é quando uma máquina com múltiplas conexões de rede decide onde entregar os pacotes IP que recebeu, para que cheguem ao seu destino.

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 tem dois dispositivos de rede, um é a interface *loopback* (explicada acima) o outro é um usado para falar com o resto da rede, talvez uma ethernet, talvez uma interface serial PPP ou SLIP.
OK, viu como o roteamento funciona? 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.

15.5.1 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:

```
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, Seção 15.3?)) 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:

```
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*.

15.6 Resolvedor de nomes (DNS)

DNS significa Domain Name System (sistema de nomes de domínio). O *DNS* converte os nomes de máquinas para endereços IPs que todas as máquinas da Internet possuem. Ele faz o mapeamento do nome para o endereço e do endereço para o nome e algumas outras coisas. Um mapeamento é simplesmente uma associação entre duas coisas, neste caso um nome de computador, como www.cipsga.org.br, e o endereço IP desta máquina (ou endereços) como 200.245.157.9.
O *DNS* foi criado com o objetivo de tornar as coisas mais fáceis para o utilizador, permitindo assim, a identificação de computadores na Internet ou redes locais através de nomes (é como se tivéssemos apenas que decorar o nome da pessoa ao invés de um número de telefone). A parte responsável por traduzir os nomes como `www.nome.com.br` em um endereço IP é chamada de *resolvedor de nomes*.
O *resolvedor de nomes* pode ser um banco de dados local (controlador por um arquivo ou programa) que converte automaticamente os nomes em endereços IP ou através de *servidores DNS* que fazem a busca em um

banco de dados na Internet e retornam o endereço IP do computador desejado. Um servidor DNS mais difundido na Internet é o `bind`.
Através do DNS é necessário apenas decorar o endereço sem precisar se preocupar com o endereço IP (alguns utilizadores simplesmente não sabem que isto existe...). Se desejar mais detalhes sobre *DNS*, veja o documento DNS-HOWTO.

15.6.1 O que é um nome?

Você deve estar acostumado com o uso dos nomes de computadores na Internet, mas pode não entender como eles são organizados. Os nomes de domínio na Internet são uma estrutura hierárquica, ou seja, eles tem uma estrutura semelhante aos diretórios de seu sistema.

Um *domínio* é uma família ou grupo de nomes. Um domínio pode ser colocado em um *sub-domínio*.
Um *domínio principal* é um domínio que não é um sub-domínio. Os domínios principais são especificados na RFC-920. Alguns exemplos de domínios principais comuns são:

- `COM` - Organizações Comerciais
- `EDU` - Organizações Educacionais
- `GOV` - Organizações Governamentais
- `MIL` - Organizações Militares
- `ORG` - Outras Organizações
- `NET` - Organizações relacionadas com a Internet
- `Identificador do País` - São duas letras que representam um país em particular.

Cada um dos domínios principais tem sub-domínios. Os domínios principais baseados no nome do país são freqüentemente divididos em sub-domínios baseado nos domínios `.com`, `.edu`, `.gov`, `.mil` e `.org`. Assim, por exemplo, você pode finaliza-lo com: `com.au` e `gov.au` para organizações comerciais e governamentais na Austrália; note que isto não é uma regra geral, as organizações de domínio atuais dependem da autoridade na escolha de nomes de cada domínio. Quando o endereço não especifica o domínio principal, como o endereço `www.unicamp.br`, isto quer dizer que é uma organização acadêmica.

O próximo nível da divisão representa o nome da organização. Subdomínios futuros variam em natureza, freqüentemente o próximo nível do sub-domínio é baseado na estrutura departamental da organização mas ela pode ser baseada em qualquer critério considerado razoável e significantes pelos administradores de rede para a organização.

A porção mais a esquerda do nome é sempre o nome único da máquina chamado *hostname*, a porção do nome a direita do hostname é chamado *nome de domínio* e o nome completo é chamado *nome do domínio completamente qualificado* (*Fully Qualified Domain Name*).
Usando o computador `www.debian.org.br` como exemplo:

- `br` - País onde o computador se encontra
- `org` - Domínio principal
- `debian` - Nome de Domínio
- `www` - Nome do computador

A localização do computador `www.debian.org.br` através de servidores DNS na Internet obedece exatamente a seqüência de procura acima. Os administradores do domínio `debian.org.br` podem cadastrar quantos sub-domínios e computadores quiserem (como `www.non-us.debian.org.br` ou`cvs.debian.org.br`).

15.6.2 Arquivos de configuração usados na resolução de nomes

Abaixo a descrição dos arquivos usados no processo de resolver um nome no sistema `GNU/Linux`.

15.6.2.1 /etc/resolv.conf

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 o endereço de servidores DNS externos são especificados nele. Existem três palavras chaves normalmente usadas que são:
domain

Especifica o nome do domínio 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 de nomes de domínio para resolução de nomes. Pode ser usado várias vezes.

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 (i.e. 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.

15.6.2.2 /etc/host.conf

O arquivo `/etc/host.conf` é o local onde é possível configurar alguns ítens que gerenciam o código do resolvedor de nomes. 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 e *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 da biblioteca resolv (para checar se o endereço pertence realmente àquele nome) e o *spoofalert on* registra falhas desta operação no `syslog`.

15.6.2.3 /etc/hosts

O arquivo `/etc/hosts` faz o relacionamento entre um nome de computador e endereço IP local. Recomendado para IPs constantemente acessados e para colocação de endereços de virtual hosts (quando deseja referir pelo nome ao invés de IP). A inclusão de um computador neste arquivo dispenda a consulta de um servidor de nomes para obter um endereço IP, sendo muito útil para máquinas que são acessadas frequentemente. 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 maquina.dominio.com.br
```

Você pode especificar mais que um nome de computador por linha como demonstrada pela primeira linha, a que identifica a interface loopback. Certifique-se de que a entrada do nome de domínio neste arquivo aponta para a interface de rede e não para a interface loopback, ou terá problema com o comportamento de alguns serviços.

OBS: Caso encontre problemas de lentidão para resolver nomes e até para executar os aplicativos (como o `mc`, etc), verifique se existem erros neste arquivo de configuração.
Estes sintomas se confundem com erros de memória ou outro erro qualquer de configuração de hardware, e somem quando a interface de rede é desativada (a com o IP não loopback). Isto é causados somente pela má configuração do arquivo `/etc/hosts`. O bom funcionamento do `Unix` depende da boa atenção do administrador de sistemas para configurar os detalhes de seu servidor.

15.6.2.4 /etc/networks

O arquivo `/etc/networks` tem uma função similar ao arquivo `/etc/hosts`. Ele contém um banco de dados simples de nomes de redes contra endereços de redes. Seu formato se difere por dois campos por linha e seus campos são identificados como:

```
Nome_da_Rede Endereço_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.

15.6.3 Executando um servidor de nomes

Se você planeja executar um servidor de nomes, você pode fazer isto facilmente. Por favor veja o documento `DNS-HOWTO` e quaisquer documentos incluídos em sua versão do BIND (Berkeley Internet Name Domain).

15.7 Serviços de Rede

Serviços de rede é o que está disponível para ser acessado pelo utilizador. 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 utilizadores 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 /etc/services, Seção 15.9.1).

15.7.1 Serviços iniciados como Daemons de rede

Serviços de rede iniciados como *daemons* ficam residente 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 possuem seus próprios recursos de segurança/autenticação para decidir quem tem ou não permissão de se conectar.

15.7.2 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, são 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).

15.7.2.1 /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.num utilizador 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.
O *número* especificado após o "." é opcional e define a quantidade máxima de vezes que o serviço poderá ser executado durante 1 minuto. Se o serviço for executado mais vezes do que este valor, ele será automaticamente desativado pelo inetd e uma mensagem será mostrada no log do sistema avisando sobre o fato.
Para reativar o serviço interrompido, reinicie o `inetd` com: `killall -HUP inetd`. O valor padrão é `40`.

utilizador

Este campo descreve que conta de utilizador utilizador 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 utilizador de qualquer item para o utilizador *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 utilizador 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 servidor 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> <utilizador> <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 tcp 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.10 nobody.tty /usr/sbin/tcpd /usr/sbin/in.talkd

ntalk dgram udp wait.10 nobody.tty /usr/sbin/tcpd /usr/sbin/in.ntalkd

#:MAIL: Mail, news e serviços uucp.

smtp stream tcp nowait.60 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
```

15.8 Segurança da Rede e controle de Acesso

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.

15.8.1 /etc/ftpusers

O arquivo `/etc/ftpusers` é um mecanismo simples que lhe permite bloquear a conexão de certos utilizadores 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 utilizadores que não tem permissão de se conectar. Ele se parece com:

```
# /etc/ftpusers - login de utilizadores bloqueados via ftp

root

uucp

bin

mail
```

15.8.2 /etc/securetty

O arquivo `/etc/securetty` lhe permite especificar que dispositivos `tty` que o utilizador *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 utilizador *root* é bloqueada.

```
# /etc/securetty - terminais que o utilizador root pode se conectar

tty1

tty2

tty3

tty4
```

15.8.3 O mecanismo de controle de acessos 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).

15.8.3.1 /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`.

15.8.3.2 /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`.

15.8.3.3 /etc/hosts.equiv e /etc/shosts.equiv

O arquivo `/etc/hosts.equiv` é usado para garantir/bloquear certos computadores e utilizadores 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 Utilizador

- 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 utilizadores 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 utilizadores (a não ser para negar o acesso), pois é fácil forjar o login, entrar no sistema tomar conta de processos (como por exemplo do servidor `Apache` rodando sob o utilizador `www-data` ou até mesmo o **root**), causando enormes estragos.

15.8.3.4 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.

15.8.4 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 utilizadores 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...

15.9 Outros arquivos de configuração relacionados com a rede

15.9.1 /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 apelido # comentário
```

name

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 tem 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 caracter "#" é ignorado e tratado como comentário.

15.9.2 /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
```