



ADMINISTRAÇÃO BÁSICA DE REDES DE COMPUTADORES

Prof. Ricardo Mesquita11

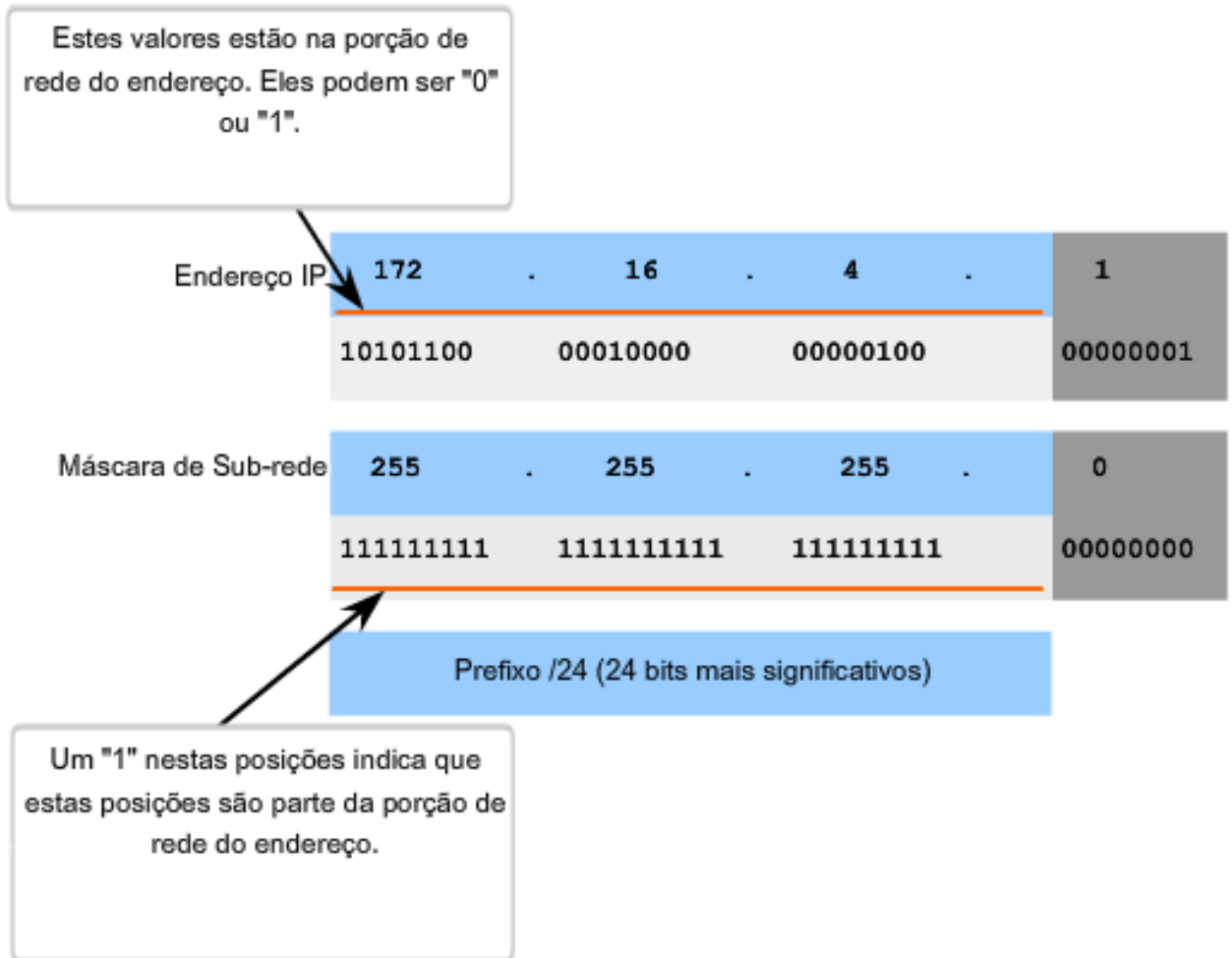
Endereços IP

- Uma máquina pode ter muitos endereços IP, acomodando múltiplas interfaces físicas, redes internas virtuais e muito mais.
- Para ver os endereços que estão ativos em sua máquina Linux, execute:

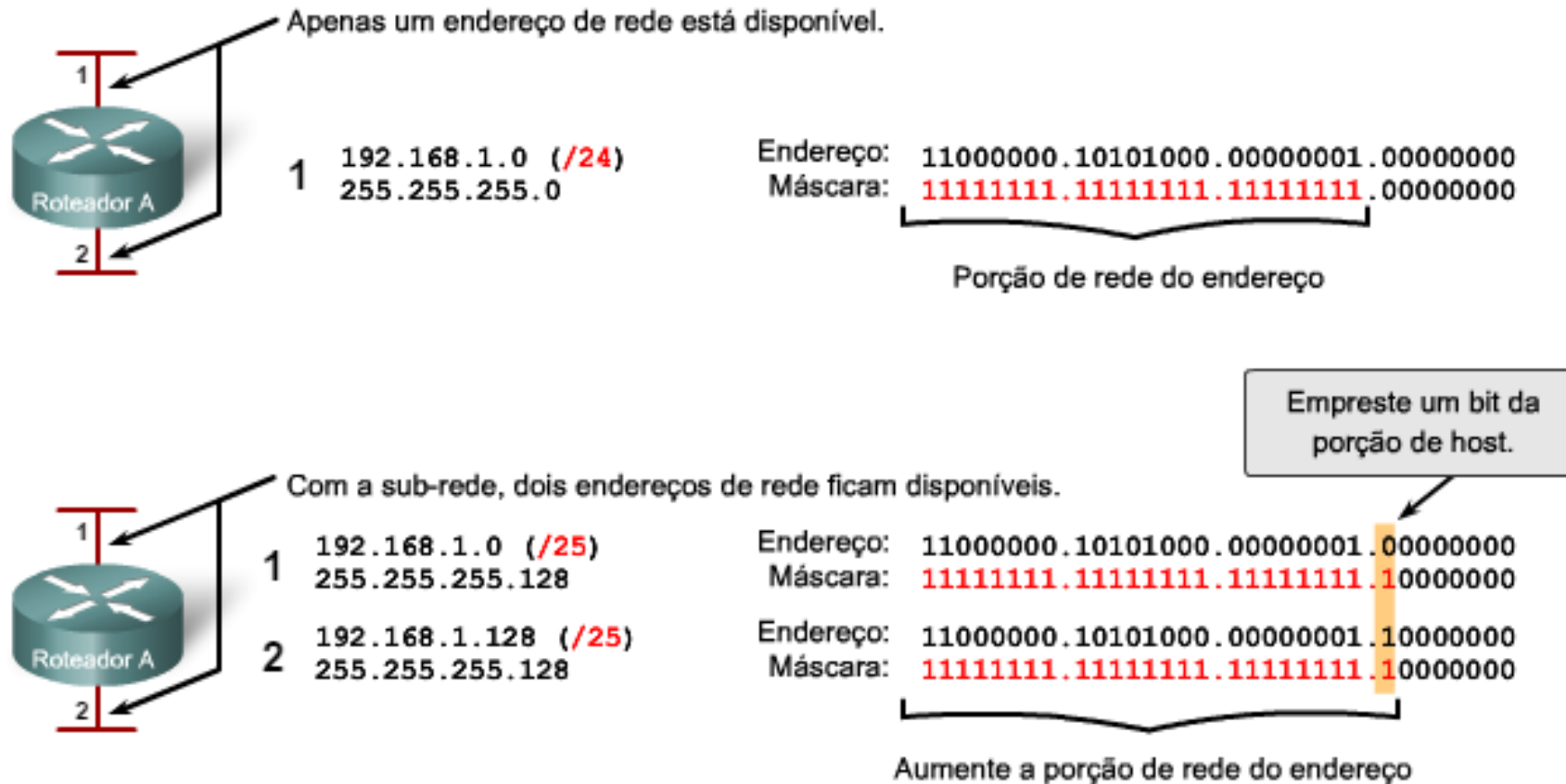
```
$ ip address show
```

- A saída do comando ip inclui muitos detalhes das camadas da Internet e da camada física.
 - Às vezes nem inclui um endereço de Internet! (denotado com ***inet***)

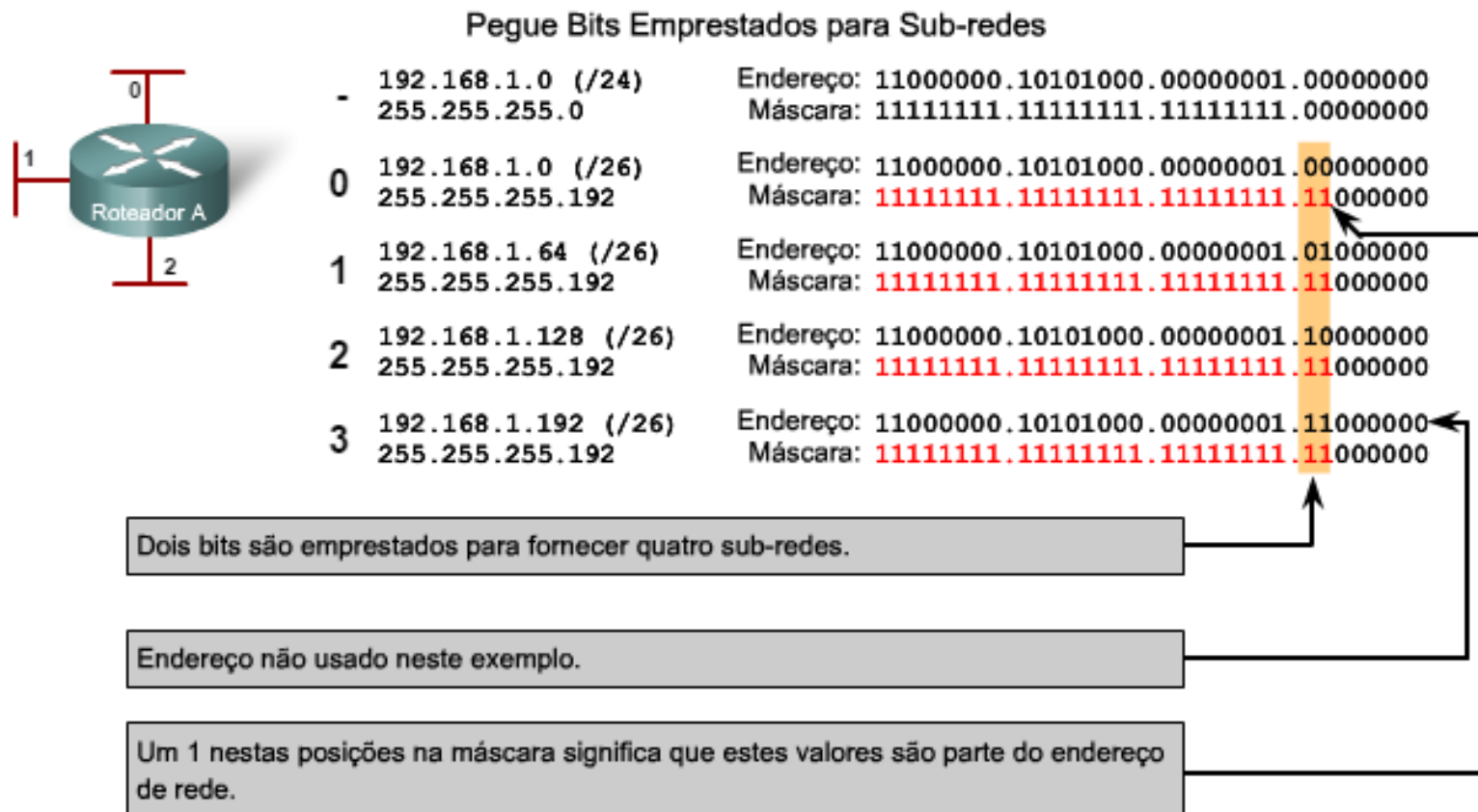
RECORDANDO: SUB-REDES



Recordando: Criação de Sub-redes



Recordando: Criação de Sub-redes



Mais sub-redes estão disponíveis, mas poucos endereços estão disponíveis por sub-rede.

Recordando: Criação de Sub-redes

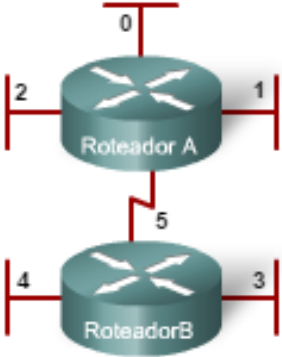
Pegue Bits Emprestados para Sub-redes

Começe com este endereço - 192.168.1.0 (/24)
255.255.255.0

Endereço: 11000000.10101000.00000001.00000000
Máscara: 11111111.11111111.11111111.00000000

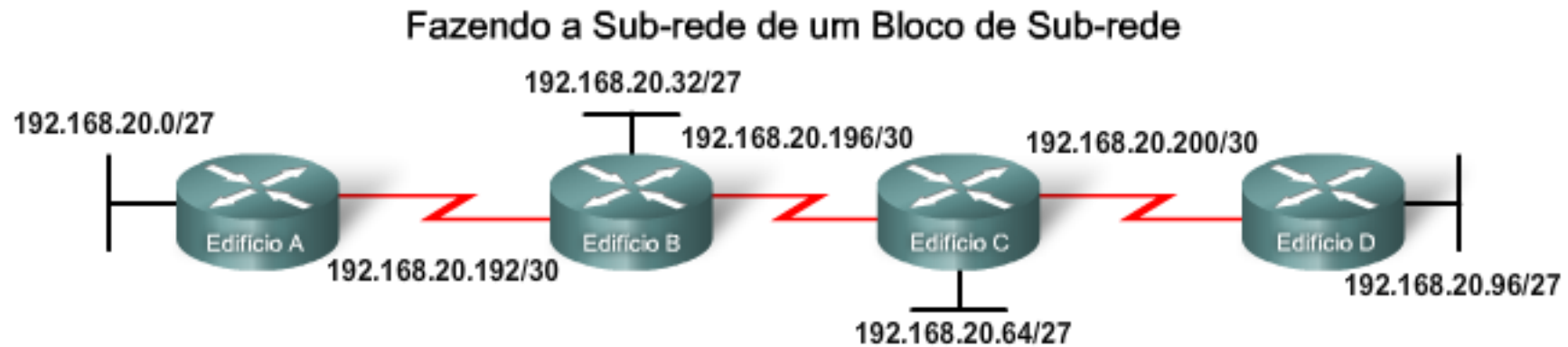
Crie 8 sub-redes

0	192.168.1.0 (/27) 255.255.255.224	Endereço: 11000000.10101000.00000001.00000000 Máscara: 11111111.11111111.11111111.11100000
1	192.168.1.32 (/27) 255.255.255.224	Endereço: 11000000.10101000.00000001.00100000 Máscara: 11111111.11111111.11111111.11100000
2	192.168.1.64 (/27) 255.255.255.224	Endereço: 11000000.10101000.00000001.01000000 Máscara: 11111111.11111111.11111111.11100000
3	192.168.1.96 (/27) 255.255.255.224	Endereço: 11000000.10101000.00000001.01100000 Máscara: 11111111.11111111.11111111.11100000
4	192.168.1.128 (/27) 255.255.255.224	Endereço: 11000000.10101000.00000001.10000000 Máscara: 11111111.11111111.11111111.11100000
5	192.168.1.160 (/27) 255.255.255.224	Endereço: 11000000.10101000.00000001.10100000 Máscara: 11111111.11111111.11111111.11100000
6	192.168.1.192 (/27) 255.255.255.224	Endereço: 11000000.10101000.00000001.11000000 Máscara: 11111111.11111111.11111111.11100000
7	192.168.1.224 (/27) 255.255.255.224	Endereço: 11000000.10101000.00000001.11100000 Máscara: 11111111.11111111.11111111.11100000



Três bits são emprestados para fornecer oito sub-redes.

Recordando: Criação de Sub-redes



Número da sub-rede	Endereço da sub-rede
Sub-rede 0	192.168.20.0/27
Sub-rede 1	192.168.20.32/27
Sub-rede 2	192.168.20.64/27
Sub-rede 3	192.168.20.96/27
Sub-rede 4	192.168.20.128/27
Sub-rede 5	192.168.20.160/27
Sub-rede 6	192.168.20.192/27
Sub-rede 7	192.168.20.224/27

Número da sub-rede	Endereço da sub-rede
Sub-rede 0	192.168.20.192/30
Sub-rede 1	192.168.20.196/30
Sub-rede 2	192.168.20.200/30
Sub-rede 3	192.168.20.204/30
Sub-rede 4	192.168.20.208/30
Sub-rede 5	192.168.20.212/30
Sub-rede 6	192.168.20.216/30
Sub-rede 7	192.168.20.220/30

Rotas e Tabelas de Roteamento

- Conectar sub-redes da Internet é principalmente um processo de envio de dados por meio de hosts conectados a mais de uma sub-rede.
- Para alcançar hosts no restante da Internet, é preciso direcionar a mensagem para um roteador.
- Para mostrar a tabela de roteamento, use o comando **ip route show**.

```
$ ip route show  
default via 10.23.2.1 dev enp0s31f6 proto static metric 100  
10.23.2.0/24 dev enp0s31f6 proto kernel scope link src 10.23.2.4 metri
```


Rotas e Tabelas de Roteamento

- A ferramenta tradicional para visualizar rotas é o comando **route**, (route -n).
 - A opção -n solicita mostrar endereços IP em vez de tentar mostrar hosts e redes por nome.
- Esta é uma opção importante a ser lembrada porque você poderá usá-la em outros comandos relacionados à rede, como **netstat**.

Gateway Padrão

- Na notação CIDR, 0.0.0.0/0 é a rota padrão para IPv4.
- O gateway padrão é para onde você envia mensagens quando não há outra opção.
- Você pode configurar um host sem um gateway padrão, mas ele não conseguirá alcançar hosts fora dos destinos especificados na tabela de roteamento.
- *Normalmente*, o roteador recebe o *primeiro endereço* da sub-rede. (O último é o endereço de broadcast.)

Sobre o IPv6

- No IPv6, os hosts normalmente têm pelo menos dois endereços.
- O primeiro, válido em toda a Internet, é chamado de **endereço unicast global**.
- O segundo, para a rede local, é chamado de **endereço link-local**.
- Os endereços de link local sempre têm um prefixo **fe80::/10**, seguido por um ID de rede de 54 bits totalmente zero e terminam com um ID de interface de 64 bits.
- O resultado é que quando você vê um endereço de link local em seu sistema, ele estará na sub-rede fe80::/64.

Sobre o IPv6

- Endereços unicast globais têm o prefixo 2000::/3.
- Como o primeiro byte começa com 001 com este prefixo, esse byte pode ser completado como 0010 ou 0011.
- Portanto, um endereço unicast global sempre começa com 2 ou 3.

Visualizando a Configuração do IPv6

```
$ ip -6 address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 state UNKNOWN qlen 1000
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s31f6: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 state UP qlen 1000
    inet6 2001:db8:8500:e:52b6:59cc:74e9:8b6e/64 scope global dynamic
        valid_lft 86136sec preferred_lft 86136sec
    inet6 fe80::d05c:97f9:7be8:bca/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

- Além da interface de loopback, você pode ver mais dois endereços:
 - O endereço unicast global é indicado com **scope global**, e o endereço link local.

Visualizando Rotas no IPv6

```
$ ip -6 route show
::1 dev lo proto kernel metric 256 pref medium
2001:db8:8500:e::/64 dev enp0s31f6 proto ra metric 100 pref medium
fe80::/64 dev enp0s31f6 proto kernel metric 100 pref medium
default via fe80::800d:7bff:feb8:14a0 dev enp0s31f6 proto ra metric
```

- A segunda linha é para destinos nas sub-redes de endereços unicast globais conectadas localmente; o host sabe que pode alcançá-los diretamente, e a linha link local abaixo é semelhante.
- Para a rota padrão (também escrita como ::/0 em IPv6; lembre-se de que isso é qualquer coisa que não esteja diretamente conectada), esta configuração organiza o tráfego para passar pelo roteador no endereço de link local fe80::800d:7bff:feb8:14a0 em vez de seu endereço na sub-rede global.

Ferramentas ICMP e DNS

- Vamos examinar alguns utilitários práticos básicos para ajudá-lo a interagir com os hosts de sua rede.
 - Essas ferramentas usam dois protocolos de particular interesse: **Internet Control Message Protocol (ICMP)**, que pode ajudá-lo a eliminar problemas de conectividade e roteamento, e o sistema **Domain Name Service (DNS)**, que mapeia nomes para endereços IP.
- O ICMP é um protocolo da **camada de transporte** usado para configurar e diagnosticar redes de Internet.
 - Ele difere de outros protocolos da camada de transporte porque não carrega nenhum dado real do usuário e, portanto, não há camada de aplicação acima dele.
- O DNS é um protocolo da **camada de aplicação** usado para mapear nomes para endereços da Internet.

Ferramentas ICMP e DNS

- **ping**
 - É uma das ferramentas de depuração de rede mais básicas.
 - Ele envia pacotes de solicitação echo ICMP para um host que solicitam ao host destinatário que devolva o pacote ao remetente.
 - Se o host destinatário receber o pacote e estiver configurado para responder, ele enviará um pacote de resposta de echo ICMP em troca.

```
$ ping 10.23.2.1
PING 10.23.2.1 (10.23.2.1) 56(84) bytes of data.
64 bytes from 10.23.2.1: icmp_req=1 ttl=64 time=1.76 ms
64 bytes from 10.23.2.1: icmp_req=2 ttl=64 time=2.35 ms
64 bytes from 10.23.2.1: icmp_req=4 ttl=64 time=1.69 ms
64 bytes from 10.23.2.1: icmp_req=5 ttl=64 time=1.61 ms
```


Ferramentas ICMP e DNS

- **ping**
- Note:
 - Pacotes de 56 bytes (84 bytes, se incluir os cabeçalhos).
 - Por padrão, um pacote por segundo.
 - O número de bytes retornados é o tamanho do pacote enviado mais 8.
 - Uma lacuna nos números de sequência, geralmente significa que há algum tipo de problema de conectividade.
 - Os pacotes não deveriam chegar fora de ordem, porque o ping envia apenas um pacote por segundo. Se uma resposta demorar mais de um segundo (1.000 ms) para chegar, a conexão está extremamente lenta.
 - Se não houver como chegar ao destino, o retorno será um ICMP “host inacessível”.

Ferramentas ICMP e DNS

- Para encontrar o endereço IP por trás de um nome de domínio, use o comando host:

```
$ host www.example.com
example.com has address 172.17.216.34
example.com has IPv6 address 2001:db8:220:1:248:1893:25c8:1946
```

- Pode-se também usar host ao contrário: insira um endereço IP em vez de um nome de host para tentar descobrir o nome de host por trás do endereço IP.
 - Não espere que isso funcione de maneira confiável, entretanto.
 - Um único endereço IP pode estar associado a mais de um nome de host, e o DNS não sabe como determinar qual desses nomes de host deve corresponder a um endereço IP.
 - Além disso, o administrador desse host precisa configurar manualmente a pesquisa inversa, e os administradores muitas vezes não fazem isso.

Introdução à Configuração de Interface de Rede

Para conectar uma máquina Linux à Internet, você ou um software deve fazer o seguinte:

1. Conecte o hardware de rede e certifique-se de que o kernel possua um driver para ele. Se o driver estiver presente, **ip address show** inclui uma entrada para o dispositivo, mesmo que não tenha sido configurado.
2. Execute qualquer configuração adicional da camada física, como escolher um nome de rede ou senha.
3. Atribua endereços IP e sub-redes à interface de rede do kernel para que os drivers de dispositivo do kernel (camada física) e os subsistemas da Internet (camada da Internet) possam se comunicar entre si.
4. Adicione quaisquer rotas adicionais necessárias, incluindo o gateway padrão.

Configurando interfaces manualmente

- Você pode vincular uma interface à camada da Internet com o comando `ip`. Para adicionar um endereço IP e uma sub-rede para uma interface de rede do kernel, você faria o seguinte:

```
# ip address add address/subnet dev interface
```

- interface é o nome da interface, como `enp0s31f6` ou `eth0`.
- Para ver todas as opções, consulte a página do manual `ip-address(8)`.

Manipulando rotas manualmente

- Com a interface ativada, você pode adicionar rotas, o que normalmente é apenas uma questão de configurar o gateway padrão:

```
# ip route add default via gw-address dev interface
```

- O parâmetro gw-address é o endereço IP do seu gateway padrão; deve ser um endereço em uma sub-rede conectada localmente atribuída a uma de suas interfaces de rede.
- Para remover o gateway padrão, execute:

```
# ip route del default
```

Manipulando rotas manualmente

- Você pode substituir facilmente o gateway padrão por outras rotas.
- Por exemplo, digamos que sua máquina esteja na sub-rede 10.23.2.0/24, você queira acessar uma sub-rede em 192.168.45.0/24 e saiba que o host em 10.23.2.44 pode atuar como um roteador para essa sub-rede.
- Execute este comando para enviar o tráfego vinculado a 192.168.45.0 para esse roteador:

```
# ip route add 192.168.45.0/24 via 10.23.2.44
```

Manipulando rotas manualmente

- Você não precisa especificar o roteador para excluir uma rota:

```
# ip route del 192.168.45.0/24
```

- *Dica:*
 - Você deve manter as coisas o mais simples possível, configurando redes locais para que seus hosts precisem apenas de uma rota padrão.
 - Se você precisar de várias sub-redes e da capacidade de rotear entre elas, geralmente é melhor configurar os roteadores que atuam como gateways padrão para fazer todo o trabalho de roteamento entre diferentes sub-redes locais.

Portas TCP e Conexões

- Você pode identificar uma conexão usando o par de endereços IP e números de porta.
- Para visualizar as conexões atualmente abertas em sua máquina, use netstat.
- Segue um exemplo que mostra conexões TCP; a opção -n desativa a resolução de nome de host (DNS) e -t limita a saída para TCP:

```
$ netstat -nt
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
tcp      0      0 10.23.2.4:47626         10.194.79.125:5222
tcp      0      0 10.23.2.4:41475         172.19.52.144:6667
tcp      0      0 10.23.2.4:57132         192.168.231.135:22
```


Portas TCP e Conexões

- Um host remoto conectado à sua máquina em uma porta conhecida implica que um servidor na sua máquina local está escutando nesta porta.
- Para confirmar isso, liste todas as portas TCP que sua máquina está escutando com `netstat`, desta vez com a opção `-l`, que mostra as portas que os processos estão escutando:

```
$ netstat -ntl
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
❶ tcp        0      0 0.0.0.0:80              0.0.0.0:*
❷ tcp        0      0 0.0.0.0:443             0.0.0.0:*
❸ tcp        0      0 127.0.0.53:53           0.0.0.0:*
```

Note:

- Escuta as portas 80 e 443.
- Na linha 3: algo está escutando conexões apenas na interface localhost

Portas TCP e Conexões

- Para descobrir as portas, você pode procurar em `/etc/services`, que traduz números de portas conhecidos em nomes.
- Este é um arquivo de texto simples.
- Você deverá ver entradas como esta:

```
ssh          22/tcp      # SSH Remote Login Protocol
smtp         25/tcp
domain       53/udp
```

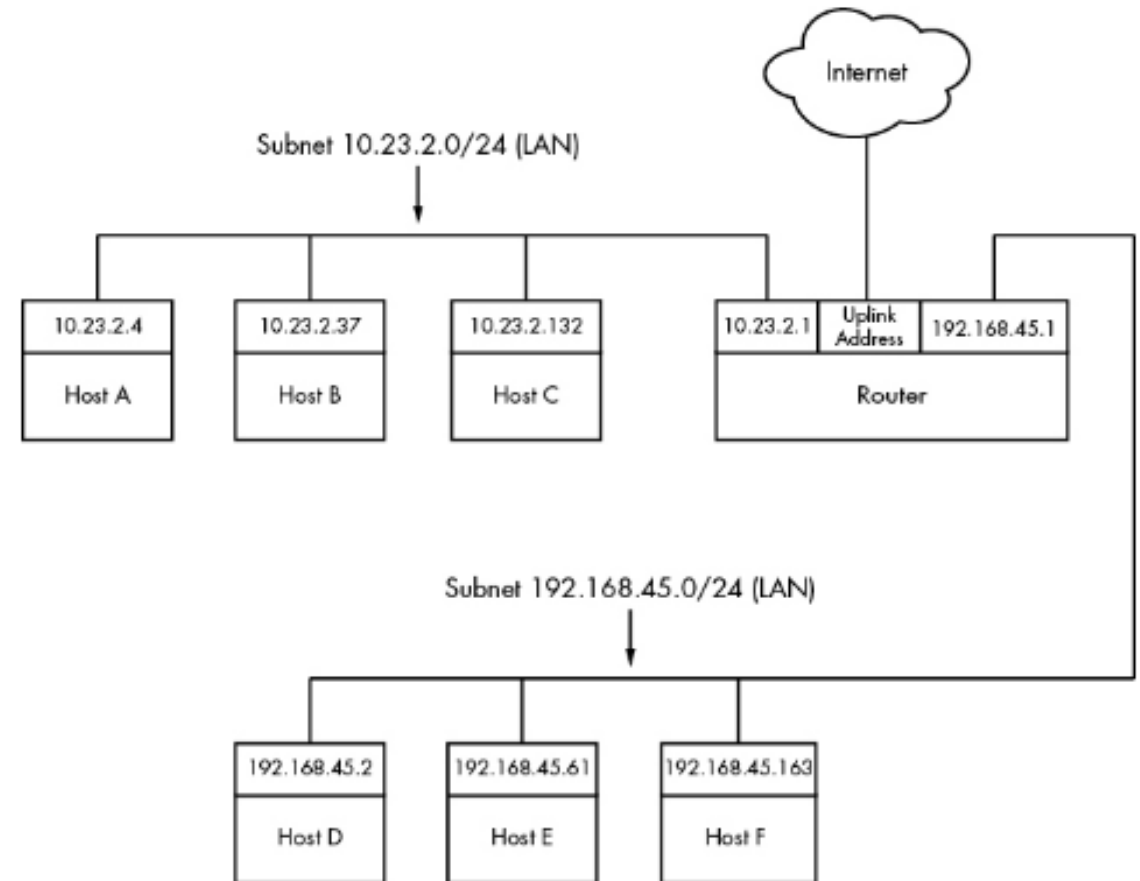
Cliente DHCP Linux

- O cliente padrão tradicional é o programa dhclient do Internet Software Consortium (ISC).
- No entanto, o systemd-networkd também inclui um cliente DHCP integrado.
- Na inicialização, o dhclient armazena seu ID de processo em /var/run/dhclient.pid e suas informações de "arrendamento" em /var/lib/dhcp/dhclient.leases.
- Você pode testar o dhclient manualmente na linha de comando, mas antes de fazer isso você deve remover qualquer rota de gateway padrão.
- Para executar o teste, basta especificar o nome da interface de rede (aqui é enp0s31f6):

```
# dhclient enp0s31f6
```

Configurando o Linux como Roteador

- Digamos que você tenha duas sub-redes LAN, 10.23.2.0/24 e 192.168.45.0/24.
- Para conectá-los, você tem uma máquina roteadora Linux com três interfaces de rede: duas para as sub-redes LAN e uma para uplink de internet, conforme mostrado na Figura.



Configurando o Linux como Roteador

- Agora digamos que os hosts em cada sub-rede tenham o roteador como gateway padrão (10.23.2.1 para 10.23.2.0/24 e 192.168.45.1 para 192.168.45.0/24).
- Se 10.23.2.4 quiser enviar um pacote para qualquer lugar fora de 10.23.2.0/24, ele passa o pacote para 10.23.2.1.
- Por exemplo, para enviar um pacote de 10.23.2.4 (Host A) para 192.168.45.61 (Host E), o pacote vai para 10.23.2.1 (o roteador) por meio de sua interface enp0s31f6 e depois retorna pela interface enp0s1 do roteador.

Configurando o Linux como Roteador

- Entretanto, nas configurações básicas, o kernel do Linux não move pacotes automaticamente de uma sub-rede para outra.
- Para habilitar esta função básica de roteamento, você precisa habilitar o encaminhamento de IP no kernel do roteador com este comando:

```
# sysctl -w net.ipv4.ip_forward=1
```

- Assim que você inserir este comando, a máquina deverá começar a rotear pacotes entre sub-redes, *assumindo que os hosts nessas sub-redes saibam que devem enviar seus pacotes para o roteador que você acabou de criar.*

Network Address Translation

- Para configurar uma máquina Linux para funcionar como um roteador NAT, você deve ativar todos os seguintes itens dentro da configuração do kernel: filtragem de pacotes de rede (“suporte a firewall”), rastreamento de conexão, suporte a iptables, NAT completo e suporte a alvo MASQUERADE.
- A maioria dos kernels de distribuição vem com este suporte.
- Em seguida, você precisa executar alguns comandos iptables de aparência complexa para fazer o roteador executar NAT para sua sub-rede privada.

Network Address Translation

- Aqui está um exemplo que se aplica a uma rede Ethernet interna em enp0s2 compartilhando uma conexão externa em enp0s31f6

```
# sysctl -w net.ipv4.ip_forward=1
```

```
# iptables -P FORWARD DROP
```

```
# iptables -t nat -A POSTROUTING -o enp0s31f6 -j MASQUERADE
```

```
# iptables -A FORWARD -i enp0s31f6 -o enp0s2 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

```
# iptables -A FORWARD -i enp0s2 -o enp0s31f6 -j ACCEPT
```


Firewall

- Visualize a configuração atual:

```
# iptables -L
```

- A saída é geralmente algo assim:

```
Chain INPUT (policy ACCEPT)
target      prot opt source                destination

Chain FORWARD (policy ACCEPT)
target      prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target      prot opt source                destination
```

Firewall

- Cada cadeia de firewall possui uma política padrão que especifica o que fazer com um pacote se nenhuma regra corresponder ao pacote.
- A política para todas as três cadeias neste exemplo é ACCEPT, o que significa que o kernel permite que o pacote passe pelo sistema de filtragem de pacotes.
- A política DROP diz ao kernel para descartar o pacote.
- Para definir a política em uma cadeia, use iptables -P assim:

```
# iptables -P FORWARD DROP
```

Firewall

- Digamos que alguém em 192.168.34.63 esteja incomodando você.
- Para evitar que eles falem com sua máquina, execute este comando:

```
# iptables -A INPUT -s 192.168.34.63 -j DROP
```

- O parâmetro -A INPUT anexa uma regra à cadeia INPUT.
- A parte -s 192.168.34.63 especifica o endereço IP de origem na regra e -j DROP informa ao kernel para descartar qualquer pacote que corresponda à regra.
 - Portanto, sua máquina irá descartar qualquer pacote vindo de 192.168.34.63.

Firewall

- Para ver a regra em vigor, execute iptables -L novamente:

```
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
DROP       all  --  192.168.34.63          anywhere
```

Firewall

- Infelizmente, seu amigo em 192.168.34.63 disse a todos em sua sub-rede para abrirem conexões com sua porta SMTP (porta TCP 25).

- Para se livrar desse tráfego também, execute:

```
# iptables -A INPUT -s 192.168.34.0/24 -p tcp --destination-port 25 -j DROP
```

- Este exemplo adiciona um qualificador de máscara de rede ao endereço de origem, bem como -p tcp para especificar apenas pacotes TCP.
- Uma restrição adicional, --destination-port 25, diz que a regra deve ser aplicada apenas ao tráfego para a porta 25.

Firewall

- A lista da tabela IP para INPUT agora se parece com isto:

```
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
DROP       all  --  192.168.34.63          anywhere
DROP       tcp  --  192.168.34.0/24        anywhere        tcp dpt:sm
```

Firewall

- Tudo está bem até você ouvir de alguém que você conhece em 192.168.34.37 dizendo que ela não pode enviar e-mail para você porque você bloqueou a máquina dela.
- Pensando que isso é uma solução rápida, você executa este comando:

```
# iptables -A INPUT -s 192.168.34.37 -j ACCEPT
```

- No entanto, isso não funciona. Para ver por quê, observe a nova cadeia:

```
Chain INPUT (policy ACCEPT)
target     prot opt source                destination
DROP       all  --  192.168.34.63          anywhere
DROP       tcp  --  192.168.34.0/24        anywhere      tcp dpt:sm
ACCEPT     all  --  192.168.34.37          anywhere
```

Firewall

- O kernel lê a cadeia de cima para baixo, usando a primeira regra que corresponder.
- A primeira regra não corresponde a 192.168.34.37, mas a segunda sim, porque se aplica a todos os hosts de 192.168.34.1 a 192.168.34.254 e esta segunda regra diz para descartar pacotes.
- Quando uma regra corresponde, o kernel executa a ação e não procura mais abaixo na cadeia.
 - Você pode notar que 192.168.34.37 pode enviar pacotes para qualquer porta da sua máquina, exceto a porta 25, porque a segunda regra se aplica apenas à porta 25.

Firewall

- A solução é mover a terceira regra para o topo. Primeiro, exclua a terceira regra com este comando:

```
# iptables -D INPUT 3
```

- Em seguida, insira essa regra no topo da cadeia com iptables -I:

```
# iptables -I INPUT -s 192.168.34.37 -j ACCEPT
```

- Para inserir uma regra em outro lugar de uma cadeia, coloque o número da regra após o nome da cadeia (por exemplo, iptables -I INPUT 4 ...).



Continua...