CAMADA DE REDE

Resumo

Leonardo Brito (LMPB)

SUMÁRIO

OBJETIVOS E FUNCIONALIDADES

A tarefa da camada de rede é *transferir dados (tipicamente segmentos da camada transporte) entre dois hosts.*

Os serviços oferecidos pela camada são:

- **Roteamento:** algoritmos de roteamento decidem qual o caminho geral que o datagrama fará para chegar ao destino
- **Encaminhamento:** roteadores encaminham datagramas para a interface de saída correta
- (em alguns casos**) Estabelecimento de conexão:** alguns protocolos oferecem um serviço orientado a conexão (call setup)

MODELOS DE SERVIÇO

ORIENTADO A CONEXÃO (VCS: VIRTUAL CIRCUITS)

Resumidamente: modelo herdado das redes telefônicas (end-systems "burros" → complexidade empurrada para o núcleo da rede). Pode trazer garantias**. Não escala.**

- Antes de começar a transferir o payload propriamente dito, os nós esperam a montagem de um **circuito virtual**, que é um conjunto de estados em vários roteadores interligando o par de hosts que querem se comunicar.
    - Há três etapas na vida dum VC: *call setup, transfer* e *teardown*, referentes ao estabelecimento da conexão, transferência de dados e encerramento da conexão, respectivamente.
- Os roteadores guardam os estados de todos os VC dos quais participam: por um lado, isso pode oferecer **garantas** (delay máximo, bitrate mínimo etc); por outro, **dificulta a escalabilidade**, pois a medida que cresce o número de nós da rede, torna-se infactível armazenar todas as conexões simultaneamente.

- Exemplos de serviços: ABR (available bitrate: garante uma taxa mínima de bitrate, chamada MCR); CBR (constant bitrate: garante um fluxo constante de bits na velocidade especificada); UBR (undefined bitrate: sem garantias de bitrate).

NÃO-ORIENTADO A CONEXÃO (REDE DE DATAGRAMAS)

Resumidamente: end-systems são computadores → complexidade pôde ser empurrada para a borda da rede. Processamento maior nos end-systems aliado a protocolos simples para o núcleo da rede resultaram num modelo com **grande escalabilidade**.

- Não há estabelecimento de conexão
- Datagramas são estampados com endereço de destino. Cada roteador lê esse endereço e escolhe para qual interface de saída deve manda-lo (**encaminhamento**). O conjunto de encaminhamentos sucessivos vai entregar o datagrama ao seu destino final (**roteamento**).
- Núcleo da rede com baixa complexidade: roteadores "stateless" apenas decidem para qual saída enviar cada datagrama que chega, sem se preocupar com o caminho completo a ser percorrido pelo datagrama
- Altamente escalável: end-systems inteligentes absorvem a complexidade, permitindo protocolos simples no núcleo da rede.

COMO O ROTEAMENTO É FEITO?

O **roteamento** é apenas o nome dado ao **conjunto de encaminhamentos sucessivos** sofridos por um datagrama. O roteamento, portanto, depende do sucesso dos encaminhamentos.

Por sua vez, cada encaminhamento é decidido pela **tabela de encaminhamento**, que responde à pergunta: "qual o próximo roteador que devo ir para chegar no endereço de destino?". A tabela está presente em cada host e roteador da rede.

Quando um host gera um datagrama destinado ao endereço IP a.b.c.d, o adaptador de rede do host busca em sua tabela de encaminhamento qual o próximo roteador no caminho para chegar à subrede a.b.c.d/x.

Exemplo: um host A gera um pacote destinado ao host B no endereço IP 223.200.2.1, pertencente à subrede 223.200.2.0/24. Na tabela de encaminhamento do host A, vê-se que para chegar à subrede 223.200.2.0/24, deve-se ir ao roteador 178.2.3.44; o host envia o datagrama a esse roteador. Chegando lá, o roteador repete o processo: lê sua própria tabela de encaminhamento, procura o próximo roteador e despacha o datagrama, e assim sucessivamente, até que se chegue em um nó onde o campo "próximo roteador" esteja vazio.

Isso significa que o datagrama chegou à subrede de destino; manda-se então o datagrama para a interface de saída adequada; agora o protocolo de enlace se encarregará de entregar o quadro ao host certo.

COMO SE MONTA A TABELA DE ENCAMINHAMENTO?

A tabela de encaminhamento é montada executando-se algoritmos de roteamento.

Há vários algoritmos de roteamento (Dijkstra, Distance Vector, etc). No entanto, os nós da rede precisam concordar sobre qual algoritmo usar. Como cada algoritmo tem suas vantagens e desvantagens, é infactível usar um único algoritmo para toda a Internet e todas suas subredes (e.g. há algoritmos que precisam de informação completa, como Dijkstra, o que obviamente é impossível num ambiente como a Internet).

A solução é a divisão da rede geral em várias redes menores chamadas **Autonomous Systems (AS)**, que, para se comunicarem uns com os outros, usam um único algoritmo de roteamento, mas internamente usam cada qual seu algoritmo de roteamento, possivelmente diferente.

Dessa forma, hierarquizamos a tarefa de roteamento: garantimos a comunicação entre quaisquer dois hosts de quaisquer dois AS distintos, mesmo que rodem internamente algoritmos de roteamento distintos.

Isso só é possível pois em cada AS é eleito um (ou mais) **gateways**, que são hosts "bilíngues", i.e., que rodam tanto o algoritmo de roteamento interno (intra-AS) quanto o algoritmo externo (inter-AS). Assim, quando um host dum AS "A" quer se comunicar com um host do AS "B", os dados são roteados primeiro para o gateway de A, depois para o gateway de B, e aí sim para o host de destino no AS "B".

IP – INTERNET PROTOCOL

É a parte da camada de rede responsável por transferir dados – tipicamente segmentos – entre dois hosts.

IPV4

(RFC 791, 1981: http://www.faqs.org/rfcs/rfc791.html)

Definido ainda na era DARPA. O espaço de endereçamento (2^32) está praticamente esgotado.

Do RFC:

"The internet protocol provides for ***transmitting blocks of data called datagrams from sources to destinations, where sources and destinations are hosts identified by fixed length addresses.*** *The internet protocol also provides for fragmentation and reassembly of long datagrams, if necessary, for transmission through "small packet" networks."*

ENDEREÇAMENTO

O IP tem endereços de 32 bits, geralmente representados na notação dotted-decimal (quatro inteiros separados por pontos).

Esse endereço geralmente é compreendido em duas partes: a parte da **subrede** e a parte do **host.**

SUBREDES

Uma subrede é um conjunto de interfaces com um mesmo prefixo IP.

Graficamente, podemos identificar uma subrede “cortando” as conexões dos enlaces às interfaces. Cada “ilha” de enlaces que se forma é uma subrede distinta.

Como vimos anteriormente, o conceito de subrede é importante para o roteamento, pois é extensivamente usado no encaminhamento de datagramas.

CLASSFUL ADDRESSING

Inicialmente, os endereços eram divididos em **classes (classful network)**:

Class	**Leading bits**	**Size of *network number* bit field**	**Size of *rest* bit field**	**Number of networks**	**Addresses per network**	**Start address**	**End address**
Class A	0	8	24	128 (2^7)	16,777,216 (2^{24})	0.0.0.0	127.255.255.255
Class B	10	16	16	16,384 (2^{14})	65,536 (2^{16})	128.0.0.0	191.255.255.255
Class C	110	24	8	2,097,152 (2^{21})	256 (2^8)	192.0.0.0	223.255.255.255
Class D (multicast)	1110	not defined	not defined	not defined	not defined	224.0.0.0	239.255.255.255
Class E (reserved)	1111	not defined	not defined	not defined	not defined	240.0.0.0	255.255.255.255

No entanto, o uso de classes rígidas impedia o aproveitamento máximo dos endereços, e promovia o desperdício de faixas de endereços.

Isso ocorria por conta das máscaras de subrede de tamanhos fixos e distantes uns dos outros (8, 16 e 24 bits).

Se não pensarmos em classes, uma organização precisando de 400 endereços compraria uma faixa de 2^9 = 512 endereços IP, totalizando um desperdício de 112 endereços ou aproximadamente 22% da faixa de endereços adquirida ((512-400)/512).

Já com máscaras de subrede fixas, essa organização só poderia comprar uma faixa de classe B, com 2^16 = 65 536 endereços; um desperdício de 99% dos endereços adquiridos.

CLASSLESS ADRESSING

Para resolver esse problema, foi proposto o esquema sem classes **CIDR (classless interdomain routing),** em 1993. (RFC 1518: http://tools.ietf.org/html/rfc1518 e RFC1519: http://tools.ietf.org/html/rfc1519).

O CIDR usa a idéia de **variable-length subnet masking (VLSM)**, ou seja, dá suporte a comprimentos variáveis de máscaras de subrede. O uso de máscaras de comprimento variável permite uma alocação mais eficiente e com menor desperdício dos endereços, como já mostramos no exemplo de desperdício na seção anterior.

REPRESENTAÇÃO

Endereços IP na notação CIDR são representados da seguinte forma:

a.b.c.d/n

onde a,b,c,d são cada qual grupos de oito bits representados em forma decimal, e n é o comprimento da máscara de subrede, geralmente representada de forma decimal.

ALOCAÇÃO DE ENDEREÇOS

O **ICANN (internet Corporation for assigned names and numbers)** administra domínios e endereços IP (além de servidores raiz DNS).

Exemplo de alocação:

Um ISP A, de tier alto, de alcance nacional, compra uma grande faixa de endereços do ICANN (ou do RIR, regional internet registry, local), isto é, uma faixa com **máscara de subrede pequena.** Por exemplo, a faixa x.0.0.0/8 terá 2^24 =~ 16,7 de endereços possíveis para hosts.

O ISP A, de tier alto, vende uma faixa menor de seus endereços, digamos x.a.0.0/16, para um ISP B de tier inferior, de alcance regional. O ISP A ainda pode alocar para outros ISPs ou organizações qualquer parte dos 2^24 – 2^16 endereços restantes.

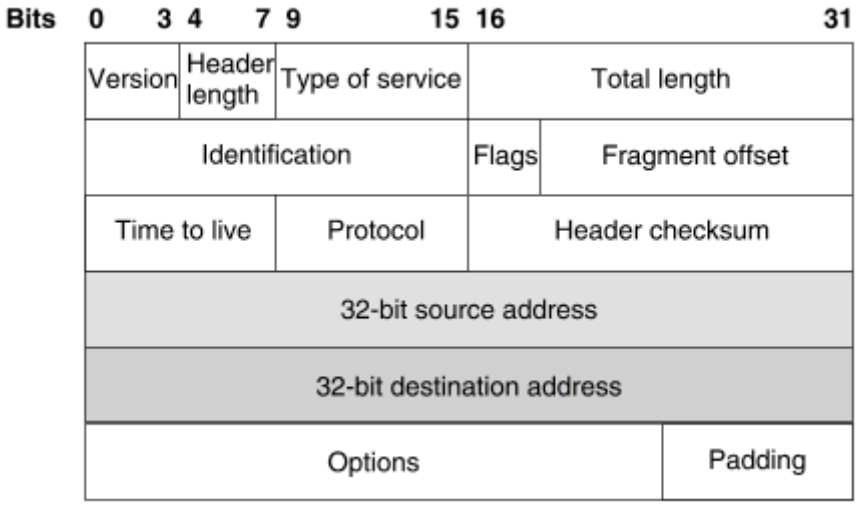
O ISP B por sua vez aloca endereços IP diretamente a clientes residenciais. Assim, o ISP B poderá ter 2^16 =~ 65,5k clientes conectados simultaneamente.

Outra vantagem do CIDR é a "portabilidade de endereço IP": pode-se facilmente mudar de ISP e preservar o endereço (ou faixa de endereços) IP, bastando cadastrar o endereço (ou faixa) no novo ISP. Neste caso, pode parecer que há um problema: o ISP antigo ainda é dono da faixa de endereços IP que contém o endereço do cliente que trocou de provedor, e no entanto o endereço do cliente também está cadastrado no novo ISP.

Quando ambiguidades assim acontecem, o datagrama sempre é roteado à subrede com prefixo em comum mais longo, ou seja, à subrede com o endereço "mais semelhante" ao endereço de destino.

Supondo um host com endereço IP x.a.22.1, cliente do ISP B. O cliente poderá passar a um ISP C: cadastrando-se seu endereço no novo ISP, os datagramas destinados a x.a.22.1 sempre serão enviados a C, pois num datagrama destinado a x.a.22.1, temos 32 bits coincidindo com "x.a.22.1" (que é o endereço cadastrado no ISP C), enquanto temos apenas 16 bits coincidindo com o ISP B ("x.a.0.0/16").

CABEÇALHO



1. **Versão:** versão do protocolo (IPv4, IPv6)
2. **Header length:** tamanho do cabeçalho. Necessário pois pode haver ou não opções
3. **Tipo de serviço:** não é muito utilizado, descrevia alguns serviços requeridos: baixo delay, etc
4. **Datagram length:** tamanho total do datagrama
5. **ID:** identificação para o fragmento
6. **Fragflag:** se é ou não fragmento
7. **Offset:** onde começar a inserir os bits de dados ao recompor fragmentos
8. **TTL:** quantos hops faltam para o datagrama expirar e não ser mais processado
9. **Protocolo:** a qual protocolo de transporte entregar o payload
10. **Checksum:** detecção de erro
11. **Endereço de origem**
12. **Endereço de destino**
13. **Opções:** não é muito utilizado. Incluía opções diversas (segurança, se o datagrama é confidencial ou não, etc)

FRAGMENTAÇÃO DE DATAGRAMAS

Datagramas podem ter tamanho teórico de até ~65KB (campo “Datagram length” tem 2^16 bits); porém, o **MTU (maximum transmission unit)**, que é o tamanho máximo do payload de algum protocolo de enlace, dificilmente chega a tal e variam muito entre si. Por exemplo, há enlaces de MTU = 1500 bytes e enlaces com MTU de 576 bytes.

Para que os datagramas “grandes” consigam passar por enlaces com MTU “pequenos”, o IPv4 implementa a **fragmentação de datagramas**. A fragmentação é feita “just-in-time” pelo **roteador** quando imediatamente necessário, e a remontagem é feita pelo host destinatário final do datagrama.

Apenas o **payload** do datagrama é fragmentado: o header original é desprezado, pois são formados novos headers para cada fragmento.

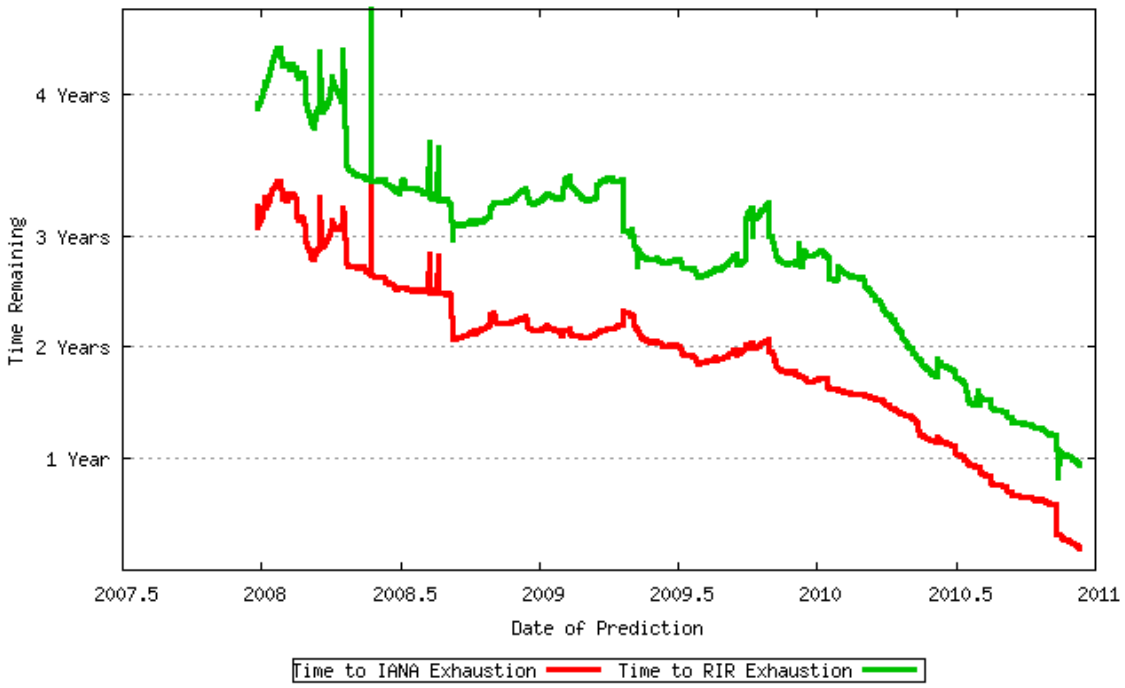
Exemplo: datagrama de 2000 bytes e enlace de MTU = 1500 bytes, supondo cabeçalhos de IP de 20 bytes.

O primeiro datagrama terá 20 bytes de header + 1480 bytes de dados. Terá fragflag=1, offset=0, e o ID estabelecido pelo roteador que o fragmentou.

O segundo datagrama terá 20 bytes de header + 520 bytes de dados (2000-1480). Terá fragflag=0 (pois é o último fragmente) e offset = 1480.

IPV6

O espaço de endereçamento do IPv4 irá acabar em breve (está previsto para 2011). Soluções para melhorar o uso dos endereços de 32 bits do IPv4 como o endereçamento CIDR e o NAT retardaram a exaustão de endereços, mas atualmente mais de 90% dos endereços IPv4 estão ocupados.



Pensando nisso, foi proposto a versão 6 do Internet Protocol. A principal diferença é o espaço de endereçamento imenso (128 bits), o que acabaria de vez com o problema da exaustão de endereços.

Aproveitando a necessidade de uma nova versão do IP, foram também mudadas algumas características do protocolo em relação à versão anterior, visando principalmente **reduzir o processamento nos roteadores**:

- **Fim da fragmentação:** a fragmentação atualmente é feita pelos roteadores, o que pode custar processamento significativo. O IPv6 não permite fragmentação. Se algum datagrama precisar passar por um MTU menor que seu tamanho, ele será descartado e o ICMPv6 enviará uma mensagem de erro ao emissor do datagrama, que poderá então ele próprio fragmentar o datagrama e reenvia-lo.
- **Fim do checksum:** pensa-se que já é suficientemente redundante o uso de checksum e CRC nas demais camadas, podendo dispensar o processamento extra causado pelo cálculo do checksum a cada hop.
- **Header de tamanho fixo:** as opções, antes presentes ou não no corpo do cabeçalho, agora podem ficar apenas no payload do datagrama, sendo apontadas pelo campo do cabeçalho nextHeader.

TRANSIÇÃO IPV4 – IPV6

Não se pode fazer um “flag day” para que todos os aparelhos IPv4 do planeta troquem para IPv6. Por isso, a transição será gradual, e inicialmente os dois protocolos irão conviver juntos.

Para que isso se dê, uma das propostas é a de **tunelamento**. Roteadores “bilíngues” (escrevem e leem datagramas IPv4 e IPv6) seriam postos nas fronteiras entre redes puramente IPv6 e redes puramente IPv4. Ao receber um pacote IPv6 que deve passar por uma rede IPv4, o roteador de fronteira encapsula o datagrama IPv6 inteiro dentro do payload dum datagrama IPv4. Do outro lado do “túnel”, o roteador receptor recebe o datagrama IPv4, extrai do payload o datagrama IPv6 e o processa normalmente.

Roteadores “bilíngues” também são chamados de **dual-stack**.

ICMP (INTERNET CONTROL MESSAGE PROTOCOL)

É utilizado para enviar mensagens de controle e de erro. Faz parte da camada de rede, mas está acima do IP: mensagens ICMP são enviadas como payload do datagrama IP.

É usdado no traceroute e ping.

DHCP (DYNAMIC HOST CONFIGURATION PROTOCOL)

Sua função é fornecer endereços IP a hosts de maneira “plug-and-play”. É um protocolo cliente-servidor: há um servidor DHCP (geralmente um roteador com essa função) que fornece o serviço aos clientes (hosts).

Vamos supor um cliente querendo entrar numa certa rede. No entanto, o cliente não o endereço da rede, nem, claro, o endereço do servidor DHCP. São tomados os seguintes passos:

1. **DHCP DISCOVERY:** Cliente envia ao endereço de broadcast (255.255.255.255) um segmento UDP na porta 67, com IP de origem igual a 0.0.0.0.
2. **DHCP OFFER:** Servidor DHCP recebe o pacote, e gera um novo pacote usando a porta UDP 68. No novo pacote há o endereço IP proposto para o cliente. Servidor envia pacote ao endereço broadcast.
3. **DHCP REQUEST:** Cliente recebe o pacote com o endereço proposto e envia um pacote de resposta informando que aceitou o endereço. O pacote é enviado ao endereço broadcast, porta UDP 67.
4. **DHCP ACK:** Servidor DHCP explicitamente confirma que foi concedido ao cliente o endereço IP informado durante o tempo informado.

O DHCP também ajuda a evitar o desperdício de endereços IP, pois permite que ISPs e organizações aloquem **endereços IP temporários**. Dessa forma, se um ISP tem *x* clientes cadastrados mas nunca mais do que *y* clientes estão online simultaneamente, o ISP poderá ter uma faixa de apenas pouco mais do que *y* endereços IP, em vez de x endereços. Se y for muito menor do que x, o ISP fará uma grande economia e evitará o desperdício de uma boa faixa de endereços.

Endereços IP temporários só são possíveis porque cada endereço IP oferecido por servidores DHCP tem **prazo de validade**. Após o fim do prazo de validade, o cliente perde o endereço (mas poderá recuperá-lo ou pedir um novo endereço).

NAT (NETWORK ADDRESS TRANSLATION)

É a ocultação dos detalhes de uma rede local através do uso de um único endereço IP "externo" para toda a rede.

Supomos uma rede local formada por vários hosts conectados a um roteador. Em vez de designar um endereço IP para cada host, o NAT usa um único endereço IP para a interface "externa" do roteador e compartilha esse endereço entre os nós da rede local. Assim, os hosts da rede local podem ter endereços IP arbitrários sem entrar em conflito com outros hosts da Internet.

O NAT funciona com o uso das portas da camada transporte. No roteador que implementa NAT, há uma tabela relacionando endereços IP internos e portas com endereço IP externo e porta.

A tabela de tradução é construída a medida que os hosts da rede local vão enviando pacotes para fora da rede. Cada vez que um host envia um datagrama para fora da rede, o par (endereço IP origem da rede local, porta) é escrito na tabela e relacionado com um par (endereço IP externo do roteador, porta), e o datagrama original é modificado com os novos dados do novo par.

Vantagens:

- permite diminuir desperdício de endereços IP permitindo que uma rede local inteira tenha apenas um endereço IP externo;
- aumenta a segurança da rede local ao ocultar detalhes internos.

Desvantagens:

- viola a arquitetura em camadas independentes

MATERIAL DE ESTUDO

- http://en.wikipedia.org/wiki/Subnetwork
- http://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing
- http://en.wikipedia.org/wiki/CIDR_notation

- http://www.javvin.com/protocolIP.html

- http://www.potaroo.net/tools/ipv4/index.html

- http://www.ietf.org/rfc/rfc4192.txt