Os Protocolos IP e DHCP

Diego Passos

1 O Protocolo IP

O protocolo IP (do inglês *Internet Protocol*) é o protocolo fundamental da Internet — como o próprio nome indica. Ele é responsável por fornecer uma interface padronizada de comunicação em uma rede que, por projeto, foi criada para rodar sobre tecnologias de enlace bastante diversas.

Sendo assim, ele é também o principal protocolo da camada de rede na Internet, embora não seja o único. Conforme será estudado em aulas posteriores, vários protocolos são executados em conjunto na camada de rede da Internet, colaborando para a implantação das funcionalidades encontradas nesta camada. O protocolo ICMP, por exemplo, é utilizado para reportar condições de erros e sinalizar informações relevantes ao roteamento. Os protocolos RIP, OSPF e BGP são alguns exemplos dos chamados **Protocolos de Roteamento**, responsáveis pela construção e manutenção das tabelas de roteamento dos roteadores.

O protocolo IP, por sua vez, é responsável primariamente pelo estabelecimento de uma série de convenções usadas para gerar e manipular os datagramas na Internet. Ele define, ainda, formatos e convenções de endereçamento dos nós.

Assim como o TCP, o IP evoluiu ao longo da sua história através de versões. Ao contrário do TCP, no entanto, as versões do IP não são retro-compatíveis — embora elas possam, de alguma forma, coexistir, como será visto em aulas posteriores. A versão atualmente dominante do IP é a 4, muitas vezes chamada de IPv4. Há muitos anos, já existe o IPv6 que vem **lentamente** substituindo o IPv4 na Internet pública. Devido à ainda maior popularidade do IPv4 na Internet atual, nesta aula nos dedicaremos exclusivamente a esta versão. De fato, o foco desta disciplina como um todo será sempre no IPv4. No entanto, em aulas posteriores cobriremos brevemente o funcionamento do IPv6.

1.1 Formato do Datagrama IP

O cabeçalho de um datagrama IPv4 tem, no mínimo, 20 bytes. Um campo de *opções* — que, como o próprio nome sugere, são opcionais — pode ser adicionado para certos propósitos, aumentando o tamanho do cabeçalho. Na prática, a enorme maioria dos datagramas IPv4 **não incluem** o campo de opções, tendo, portanto, 20 bytes de cabeçalho.

Os primeiros 4 bits do cabeçalho IP armazenam um número inteiro sem sinal que denota a versão do protocolo pertinente àquele datagrama. Para o IPv4, por exemplo, este campo sempre assume o valor $0100_{(2)} = 4_{(10)}$.

Os 4 bits seguintes são utilizados para informar o tamanho do cabeçalho IP do datagrama. Este campo é fundamental por conta da supracitada existência de um campo de opções: se o tamanho do cabeçalho pode variar, é preciso que o datagrama informe onde começam os dados e onde termina o cabeçalho. Repare que com 4 bits só é possível informar valores entre 0 e $2^4 - 1 = 15$. Como o tamanho mínimo do cabeçalho IP são 20 bytes, se interpretássemos o valor deste campo como o número de bytes do cabeçalho, precisaríamos de mais bits. Para evitar aumentar o número de bits deste campo, o IP adota a convenção de que este tamanho é dado em unidades de *palavras de 32 bits*. Para um cabeçalho mínimo de 20 bytes, portanto, este campo deve ter o valor mínimo de 5 (já que $5 \times 32/8 = 20$).

Os próximos 8 bits originalmente compunham um campo chamado de `ToS` (do inglês *Type-of-Service*. O propósito deste campo era permitir que *hosts* ou roteadores **marcassem** o datagrama com um valor numérico especial que, ao ser lido por outros roteadores, indicaria que aquele datagrama deveria ser tratado de forma distinta. Um uso comum deste campo é para a implementação da diferenciação dos datagramas em classes de tráfego com o objetivo de aplicar políticas de escalonamento distintas — conforme visto na aula anterior. RFCs mais recentes alteraram — mais de uma vez — a semântica destes bits. Atualmente, os 6 bits mais significativos deste conjunto são chamados de `DSCP` (*Differentiated Services Code Point*) — com uma finalidade semelhante à do campo `ToS` original — enquanto os 2 bits restantes formam o campo `ECN` (*Explicit Congestion Notification*) — que permitiria que a rede avisasse os *hosts* sobre a ocorrência de congestionamentos explicitamente. Vale destacar

que a efetiva utilização destes campos na prática — seja o `ToS` original, ou `DSCP` e `ECN` — é muito limitada.

O próximo campo, de 16 bits, informa o comprimento **completo** do datagrama — *i.e.*, contando cabeçalho *e* carga útil. Este campo é contado em bytes, o que faz com que datagramas IPv4 sejam limitados a um tamanho total máximo de $2^{16} - 1 = 65535$ bytes. Na prática, no entanto, datagramas IPv4 na Internet tendem a ser bem menores que este limite. Por outro lado, por conta do tamanho mínimo do cabeçalho IPv4, o menor datagrama possível tem ao menos 20 bytes.

O campo seguinte, de identificação, contém também 16 bits. Ele funciona como uma espécie de número de sequência: a cada novo datagrama gerado por um *host*, ele preenche este campo com um novo valor (*e.g.*, incrementando em uma unidade em relação ao datagrama anterior). Note, no entanto, que, ao contrário do que faz, por exemplo, o TCP, o IP não utiliza este campo para suprimir duplicatas, detectar perdas ou reordenar os pacotes — lembre-se, um receptor IP não realiza nenhum destes controles. A utilidade do campo de identificação surge nas operações de fragmentação e remontagem de datagramas — tópicos discutidos mais adiante. Os dois campos subsequentes — *flags* e *offset* do fragmento — também são utilizados apenas nos casos destas operações e, portanto, deixaremos para discuti-los posteriormente.

Após estes campos ligados à fragmentação, chega-se a um pequeno campo de 8 bits, mas de fundamental importância no IP. Este campo, denominado `TTL` (*Time-To-Live*), **determina o número máximo de saltos que o datagrama ainda pode percorrer**. O nó de origem do datagrama configura este campo para um certo valor inicial (que pode variar dependendo de alguns fatores) e cada roteador intermediário pelo qual o datagrama passa decrementa o valor deste campo em uma unidade. Se, após o decremento, o roteador verificar que o valor do TTL chegou a zero, o datagrama é imediatamente descartado.

O campo TTL é importante porque redes de datagramas, como a Internet, são susceptíveis aos chamados *loops de roteamento*. Em outras palavras, o processo de encaminhamento de um datagrama pode resultar em um ciclo, no qual o datagrama passa indefinidamente por uma mesma sequência de roteadores sem nunca chegar ao destino. *Loops* de roteamento são especialmente prejudiciais porque, não só aquele datagrama particular nunca é entregue ao destinatário, mas ele consome recursos da rede (*e.g.*, espaço em *buffer*, tempo de uso dos enlaces) repetidamente, afetando, assim, outros datagramas de outros fluxos de dados. A existência do campo `TTL`, portanto, ajuda a mitigar este problema, garantindo que, em algum momento, pacotes em *loop* serão descartados.

O campo subsequente é chamado de *protocol*, ou *upper layer*. Este campo informa qual é o protocolo da mensagem encapsulada na carga útil do datagrama (*e.g.*, TCP, UDP, ICMP). Esta informação é necessária quando do recebimento do datagrama pelo destinatário final. Neste momento, após realizar todas as verificações inerentes à camada de rede, o IP realiza o desencapsulamento da carga útil e precisa decidir para qual protocolo enviar este conteúdo para que o pacote continue seu processamento normal. Este campo tem comprimento de 8 bits, e os valores válidos são padronizados pela IANA — por exemplo, o ICMP corresponde ao valor 1, enquanto o TCP corresponde ao valor 6.

Os próximos 16 bits do cabeçalho IP são usados para armazenar um *checksum* calculado **apenas sobre os bits do próprio cabeçalho IP**. Note, portanto, que este *checksum* não é útil para a verificação da integridade da carga útil. Entre outras razões, optou-se por isso porque os protocolos tradicionais de camada de transporte na Internet — UDP e TCP — já contém seus próprios *checksums* que englobam seus cabeçalhos, bem como o restante do pacote. Desta forma, o *checksum* do IP **não é redundante com o *checksum* de TCP ou UDP**. O cálculo deste *checksum* é feito exatamente da mesma forma que o *checksum* usado nos protocolos de camada de transporte estudados anteriormente. Repare que o *checksum* precisa ser alterado a cada salto, já que, minimamente, o roteador realizará o decremento do campo `TTL`.

Os dois campos seguintes têm 32 bits de comprimento, cada. Ambos representam endereços dos dispositivos que participam da comunicação do datagrama em questão. O primeiro endereço identifica o dispositivo que originou o datagrama, enquanto o segundo armazena o endereço do destinatário. **Note que, em ambos os casos, estamos nos referindo às pontas da comunicação, ou seja, ao nó que originou aquele datagrama e ao destinatário final**. Estes campos, portanto, não são — em condições normais — alterados pelos roteadores intermediários, mantendo seus valores do início ao fim da rota percorrida pelo datagrama.

O último campo do cabeçalho IP é o campo de opções. Não entraremos em detalhes de quais são as opções suportadas por este campo, e nem do formato específico deste campo. Nos limitaremos a comentar um exemplo de uso deste campo: a opção `Record Route`. Quando habilitada, ela instrui os roteadores intermediários a incluírem seus endereços (também no campo de opções) à medida que o datagrama percorre seu caminho. Esta opção é usada por certas implementações da ferramenta `ping` para obter a informação da rota utilizada pelos pacotes enviados.

1.2 Fragmentação

Fragmentação é o processo pelo qual um datagrama **grande** é quebrado em uma sequência de datagramas menores. No IPv4 a fragmentação *pode ocorrer* nos roteadores intermediários pelos quais um datagrama passa.

Para entender a razão pela qual a fragmentação existe, é preciso voltarmos uma vez mais às características básicas da Internet. Em particular, devemos lembrar que a Internet foi pensada como uma rede que pudesse ser executada sobre uma grande variedade de tecnologias de enlace diferentes, com o protocolo IP provendo a interface padronizada que garantiria a intercomunicação. Ocorre que tecnologias de enlaces de comunicação baseadas em comutação de pacotes normalmente definem um **tamanho máximo** para os pacotes transmitidos pelos seus enlaces — o Ethernet, por exemplo, coloca este limite em 1514 bytes. Além disso, tecnologias diferentes muitas vezes estabelecem limites distintos — por exemplo, o Wi-Fi permite a transmissão de pacotes de até 2346 bytes.

Esta variabilidade dos limites de tamanho dos pacotes transmitidos por cada enlace cria o seguinte dilema: qual é o tamanho máximo de um datagrama gerado por um determinado *host* de origem? O *host* poderia verificar o limite da sua própria interface, e utilizar este valor para balizar o tamanho máximo dos seus datagramas. Mas isso não garante que, ao longo do caminho até o destinatário final, este datagrama não precisará passar por um enlace mais restritivo. Se isso acontecer, o que o roteador intermediário deverá fazer?

É justamente neste ponto que entra a fragmentação. Se um roteador intermediário precisa encaminhar um datagrama grande demais por um certo enlace de saída restritivo, ele pode quebrar o datagrama original em fragmentos menores — que respeitem o limite de tamanho imposto pelo enlace — e transmiti-los um a um. Cada fragmento é, na verdade, um novo datagrama IP, no sentido de que cada datagrama conterá seu próprio cabeçalho e será encaminhado de maneira independente dos demais fragmentos do datagrama original — *e.g.*, um fragmento pode ser perdido, enquanto os demais são entregues ao destinatário final, ou mesmo encaminhado por um caminho diferente dos outros fragmentos.

O processo inverso à fragmentação é comumente chamado de *remontagem*, no qual os fragmentos de um datagrama são juntados para formar novamente um datagrama idêntico ao original. É importante ressaltar que no IPv4 **a remontagem é realizada apenas pelo destinatário final**. Uma das justificativas para esta escolha tem relação com a complexidade associada ao processo de remontagem, associada à filosofia de manter a inteligência nas bordas na Internet.

No protocolo IP, o tamanho máximo de um datagrama — **incluindo o cabeçalho IP** — que pode ser transmitido por aquele enlace recebe o nome de MTU (*Maximum Transmission Unit*). Ao tentar encaminhar um datagrama por um enlace de saída, um roteador deve comparar seu comprimento total ao MTU do enlace. Se o datagrama é maior que o MTU, efetua-se a fragmentação.

Para fragmentar um datagrama, o IP utiliza três campos específicos de seu cabeçalho: o campo de identificação, o campo de *offset* do fragmento e a *flag mais fragmentos*.

O campo de identificação é usado para informar ao receptor que certo conjunto de fragmentos fazem parte de um mesmo datagrama original. Basicamente, ao fragmentar um datagrama, o roteador deve incluir o mesmo identificador do datagrama original em todos os fragmentos resultantes.

O campo *offset* do fragmento armazena a posição que a carga útil do fragmento ocupa dentro da carga útil do datagrama original. Se pensarmos na carga útil de um datagrama original como um vetor de bytes, o *offset* de um fragmento diz qual índice deste vetor corresponde ao primeiro byte da carga útil do fragmento. Um detalhe sobre este campo é que, para permitir *offsets* grandes com um campo relativamente pequeno — este campo possui 13 bits — o valor do *offset* é contado em *palavras de 8 bytes*. Por exemplo, se o valor do campo *offset* é 10, isso significa que o primeiro byte da carga útil do fragmento corresponde ao byte de índice $10 \times 8 = 80$ na carga útil do datagrama original.

Finalmente, a *flag mais fragmentos* indica se o fragmento é o último do datagrama original. Se o bit desta *flag* tiver valor 0, este fragmento é o último. Caso contrário, ainda há outros fragmentos após este.

O campo *offset* é fundamental para que o receptor possa saber como — em que ordem — remontar os fragmentos no datagrama original, já que a Internet pode entregar datagramas — e, portanto, fragmentos — fora de ordem. Já a *flag mais fragmentos* serve para indicar ao receptor quando parar de esperar novos fragmentos.

Note que estes campos existem no cabeçalho IP, independentemente de se o datagrama é original ou um fragmento. Para datagramas não fragmentados, ambos o campo *offset* e a *flag mais fragmentos* devem ser iguais a zero. Qualquer outra combinação indica que o datagrama é, na verdade, um fragmento de um datagrama maior.

Ao realizar a fragmentação de um datagrama, é comum — embora não mandatório — que o roteador gere os maiores fragmentos possíveis. Assim, os n primeiros fragmentos teriam comprimento

total igual ao MTU do enlace — o que significa que eles receberiam MTU − tamanho do cabeçalho bytes de carga útil cada (na verdade, o maior múltiplo de 8 menor ou igual a este valor). O último fragmento, por sua vez, receberia o restante da carga útil.

Por fim, repare que a **fragmentação pode ocorre múltiplas vezes ao longo do caminho de um datagrama**. Se um datagrama é fragmentado em um roteador e um ou mais fragmentos resultantes são maiores que o MTU de um enlace posterior do caminho, os fragmentos serão novamente fragmentados.

1.3 Endereçamento IP

Muitas vezes diz-se informalmente que o endereço IP é o identificador de um nó na Internet. Na verdade, endereços IP são atribuídos a interfaces de rede, o que significa que se um dispositivo possui múltiplas interfaces — como, por exemplo, um roteador — ele normalmente possuirá múltiplos endereços IP.

No IPv4, cada endereço possui 32 bits. Na prática, é comum a utilização de uma notação em que os 32 bits são quebrados em quatro bytes — ou octetos, em um jargão comum em Redes de Computadores. Neste caso, o IP é representado pelos valores decimais dos seus octetos separados por pontos, *e.g.*, `200.20.15.166`.

Os endereços IPv4, por sua vez, são organizados nas chamadas **sub-redes**. De uma maneira simplificada — embora não totalmente correta — uma sub-rede diz respeito a um conjunto de interfaces de redes que podem se comunicar sem o intermédio de um roteador. Um corolário desta definição é que o papel de um roteador é conectar sub-redes diferentes, permitindo a comunicação entre dispositivos destas várias sub-redes.

Em uma topologia complexa de rede, podemos identificar as sub-redes olhando para os conjuntos de interfaces que se comunicam *diretamente*: cada um destes conjuntos define uma sub-rede. Um erro comum é desconsiderar uma sub-rede que interconecta interfaces apenas entre roteadores: ainda que não haja interfaces de *hosts* conectadas, trata-se de uma sub-rede.

Em termos de endereçamento, sub-redes correspondem a conjuntos de 2^n endereços numericamente sub-sequentes (considerando-se a representação binária dos 32 bits dos endereços). Em particular, os bits de um endereço IP podem ser quebrados em duas partes: **um prefixo de sub-rede**, denotado pelos bits mais significativos do endereço — *i.e.*, mais à esquerda — e a identificação da interface dentro daquela sub-rede, denotada pelos bits restantes.

Uma pergunta imediata é: quantos bits devem ser alocados para o prefixo? Quanto mais bits alocarmos para o prefixo, maior o **número de sub-redes diferentes**. Por outro lado, estas sub-redes serão **menores, *i.e.* terão menos endereços disponíveis**. Outro fator a se considerar é que nem todas as sub-redes precisam ser do mesmo tamanho. Uma sub-rede utilizada para o endereçamento de dispositivos em uma residência pode ser bem menor que a sub-rede utilizada para endereçar todos os computadores de um grande *data-center*. Por esta razão, a Internet permite a definição de sub-redes com diferentes tamanhos — o que resulta em comprimentos distintos para os prefixos utilizados.

Originalmente, esta alocação de sub-redes na Internet seguia um padrão de endereçamento chamado de **endereçamento por classes**. A ideia era definir algumas *classes* de sub-redes padronizadas, que fornecessem alguma flexibilidade de tamanho, com o propósito de servir bem a redes *pequenas*, *médias* ou *grandes*. Foram justamente estas noções abstratas de tamanho que guiaram a criação das classes. Em particular, foram criadas as seguintes cinco classes:

1. **Classe A**: utilizada para redes *grandes*, usava um prefixo de apenas 8 bits, permitindo, portanto, até $2^{24} = 16777216$ endereços. Todos os prefixos que denotavam sub-redes de Classe A eram iniciados por 0.

2. **Classe B**: utilizada para redes *médias*, usava um prefixo de 16 bits, permitindo, portanto, até $2^{16} = 65536$ endereços. Todos os prefixos que denotavam sub-redes de Classe A eram iniciados por 10.

3. **Classe C**: utilizada para redes *pequenas*, usava um prefixo de 24 bits, permitindo, portanto, até $2^8 = 256$ endereços. Todos os prefixos que denotavam sub-redes de Classe A eram iniciados por 110.

4. **Classes D e E**: reservadas para outros usos mais específicos, como comunicação *multicast* — tópico de aulas futuras — no caso da Classe D.

Eventualmente, percebeu-se que o endereçamento baseado em classes não fornecia uma granularidade suficiente para atender eficientemente a demandas típicas na Internet. Para ver isso, considere uma instituição que precisa conectar 1000 equipamentos à sua sub-rede. Uma sub-rede da Classe C é

pequena demais, mas, ao mesmo tempo, uma sub-rede da classe B resulta em um desperdício de mais de 64 mil endereços.

Para permitir uma alocação de sub-redes mais justas — *i.e.*, com "sobras" menores de endereços — o ideal é que pudéssemos definir **arbitrariamente** o comprimento do prefixo da sub-rede. Esta ideia é a base da solução de endereçamento utilizada atualmente na Internet: o endereçamento CIDR (*Classless Inter-Domain Routing*.

No CIDR, pode-se utilizar sub-redes com qualquer comprimento. Para que se possa saber o tamanho do prefixo da sub-rede a qual um endereço pertence, endereços são denotados por `a.b.c.d/x`, onde `x` é o número de bits no prefixo.

Desta forma, podemos definir sub-redes com qualquer tamanho que seja uma potência de 2. Por exemplo, uma sub-rede `/23` tem prefixo de 23 bits, deixando 9 bits para o endereçamento de interfaces, resultando até $2^9 = 512$ endereços diferentes. Voltando ao exemplo de uma instituição com 1000 equipamentos a serem interligados em uma sub-rede, uma sub-rede `/22` passa a ser a opção mais eficiente, por permitir $2^{32-22} = 1024$ endereços — resultando em uma sobra de apenas 24 endereços.

Outros dois conceitos importantes quando se discutem as convenções de endereçamento no IP são os de **endereço de sub-rede** e **endereço de *broadcast***. O endereço de sub-rede é primeiro endereço da faixa definida pela sub-rede. Em outras palavras, trata-se do endereço IP formado pelo prefixo da sub-rede complementado por todos os demais bits iguais a zero. Já o endereço de *broadcast* é o último, formado pelo prefixo seguido de todos os demais bits iguais a um.

O endereço de *broadcast* é utilizado para permitir o envio de um datagrama IP para todas as interfaces conectadas à sub-rede em questão. Embora até aqui nesta disciplina tenhamos focado exclusivamente na comunicação *unicast* — *i.e.*, aquela em que um pacote tem exatamente um destinatário especificado — comunicação em *broadcast* é relativamente comum em redes e tem importantes aplicações em uma série de protocolos que serão estudados mais à frente na disciplina. **Uma interface, portanto, não pode ser configurada para operar com o endereço de *broadcast* da sua sub-rede**.

Já o endereço de sub-rede é, como o nome sugere, um endereço IP usado para identificar a sub-rede em questão. À princípio, uma interface *poderia* ser configurada para utilizar o endereço de sub-rede da sub-rede na qual se encontra — de fato, alguns equipamentos/plataformas aceitam essa configuração. Entretanto, a RFC 1812 especifica que roteadores devem descartar pacotes endereçados ao endereço de sub-rede da sub-rede de sua interface ou tratá-los como pacotes de **broadcast** — em certo momento da história, o que hoje chamamos de endereço de sub-rede já foi considerado o identificador do endereço de *broadcast*. Na prática, portanto, uma interface configurada para operar com o endereço de sub-rede da sua sub-rede provavelmente não será bem sucedida ao tentar se comunicar com outras interfaces.

A atribuição de semânticas especiais a estes dois endereços faz com que o *tamanho útil* de uma sub-rede — *i.e.*, a quantidade de endereços que efetivamente podem ser utilizados para endereçar interfaces — seja dado por $2^{32-x} - 2$, onde x é o comprimento do prefixo da sub-rede. Esse fator deve ser considerado ao escolher-se o tamanho do prefixo de uma sub-rede.

Por fim, é importante citar que há uma notação alternativa ainda bastante popular para denotar o comprimento de um prefixo no endereçamento CIDR: a chamada **máscara de sub-rede**. A máscara de sub-rede de uma determinada sub-rede X é uma sequência de 32 bits tal que, para qualquer endereço pertencente a X, a operação de *and* bit-a-bit da máscara com este endereço resulte no endereço de sub-rede de X. De uma forma um pouco mais simples e prática, se uma sub-rede possui um prefixo de comprimento x, sua máscara de sub-rede será dada por uma sequência de x bits um concatenada com uma sequência de $32 - x$ bits zero. Por exemplo, uma sub-rede `/22` possui uma máscara de sub-rede `255.255.252.0`.

2 DHCP

Até aqui, discutimos uma série de convenções relacionadas aos endereços IP, incluindo o importante conceito de sub-rede. No entanto, estas convenções, por si só, não respondem a uma pergunta importante: como endereços IP são efetivamente atribuídos a uma interface de rede na Internet? Note que, como um certo endereço IP só faz sentido dentro de uma sub-rede que o contém, não podemos atribuir endereços arbitrários às interfaces de nossos equipamentos. Ao conectar um equipamento a uma sub-rede específica, precisamos atribuir à sua interface um endereço válido. Por fim, repare que, a princípio, endereços IP devem ser **únicos** para que não haja ambiguidade na entrega de um datagrama.

Existem várias respostas para esta pergunta. Em particular, boa parte dos equipamentos de rede permitem a **configuração estática** do endereço IP. Neste caso, o administrador de rede determina

um endereço válido e disponível na sub-rede em questão e este é configurado de maneira permanente no equipamento.

Esta solução de atribuição estática de endereços funciona bem para equipamentos estáticos e para redes controladas — no sentido de controle sobre quais equipamentos estarão conectados à rede. Hoje, no entanto, é comum que redes sejam bastante dinâmicas, com entrada e saída frequentes de equipamentos. Nestes casos, o processo manual envolvido na configuração estática do endereço IP não é prático, ou mesmo viável.

Este tipo de situação motiva a existência do protocolo DHCP (do inglês *Dynamic Host Configuration Protocol*). Este protocolo permite que um *host*, ao se conectar a uma rede, receba automaticamente uma série de configurações básicas pertinentes àquele ambiente. Entre estas configurações, está justamente um endereço IP válido e disponível. É importante destacar, no entanto, que **o endereço IP não é a única informação de configuração que pode ser obtida através do DHCP**. De fato, mesmo em configurações básicas deste serviço, o *host* tipicamente recebe algumas outras configurações, além do endereço IP.

O DHCP utiliza uma arquitetura do tipo *cliente-servidor*: o cliente é o *host* que deseja ingressar na rede, enquanto o servidor é um equipamento que se encontra permanentemente conectado à sub-rede em questão.

Uma característica interessante e importante do DHCP é que o cliente não precisa conhecer — e geralmente não conhece — o endereço do servidor ao ingressar na rede. Para lidar com isso, o DHCP prevê uma mensagem chamada de *DHCP Discovery*. Esta mensagem é enviada pelo cliente em *broadcast* — justamente porque o cliente não sabe o endereço IP do servidor DHCP. Por ser enviada em *broadcast*, todos as interfaces conectadas àquela sub-rede receberão e processarão o pacote, mas apenas equipamentos que possuírem um processo servidor DHCP a responderão.

Como resposta a um *DHCP Discovery*, um servidor DHCP gera uma mensagem do tipo *DHCP offer*. Esta mensagem não só informa ao cliente sobre a existência do servidor, mas também já carrega um endereço IP ofertado pelo servidor para uso pelo cliente.

O cliente, por sua vez, envia como resposta uma mensagem do tipo *DHCP Request*, requisitando autorização para efetivamente utilizar o endereço ofertado. Finalmente, o servidor envia uma mensagem *DHCP Ack*, confirmando a atribuição do endereço para o cliente.

Endereços IP atribuídos via DHCP são temporários. Idealmente, após utilizar um endereço obtido via DHCP, o cliente "devolveria" o endereço ao servidor. De fato, o DHCP prevê uma mensagem específica para este fim (a *DHCP Releasing*), mas esta é raramente enviada porque normalmente *hosts* móveis saem de redes de maneira abrupta. Para garantir que os endereços voltem ao servidor após uso pelos clientes, esta atribuição é feita em um esquema de *lease*, ou empréstimo. Ao enviar o *DHCP Ack* confirmando a atribuição do endereço ao cliente, o servidor informa o tempo de validade daquela *lease*. Se a validade de uma *lease* estiver próxima da expiração e o cliente desejar continuar utilizando o endereço, este pode reenviar a mensagem *DHCP Request*, solicitando uma renovação. Caso contrário, o servidor considere que o endereço não está mais em uso e o armazena em um *pool* de endereços disponíveis.

A princípio pode parece estranha a sequência de mensagens trocadas em uma comunicação DHCP. Em particular, por que não basta que o servidor envie um *DHCP Offer*? Por que o cliente precisa realizar um *DHCP Request* logo em seguida? Uma das razões para isso é o fato de que, em certas situações — que só ficarão claras ao se estudar o conceito de Redes Locais, em Redes II — mais de um servidor DHCP pode receber o *DHCP Discovery* e, portanto, um cliente pode receber múltiplas mensagens do tipo *DHCP Offer*. Neste caso, cabe ao cliente escolher um dos endereços ofertados e realizar uma requisição para efetivamente poder utilizá-lo.

Com exceção do caso de renovação de uma *lease* já existente, a comunicação entre cliente e servidor no DHCP se dá sem que o cliente possua um endereço IP definido. Por este motivo, os datagramas IP que transportam as mensagens DHCP utilizam endereços IP de origem e destino pouco usuais — novamente, vale destacar que estamos nos referindo à obtenção inicial de um endereço, e não à renovação de uma *lease*. Nas mensagens originadas pelo servidor, o endereço IP de origem é o endereço IP da interface do servidor, mas o endereço de destino é o `255.255.255.255`. Este endereço IP especial funciona como um endereço de *broadcast* coringa, independente do prefixo da sub-rede utilizada. Desta forma, mesmo sem possuir ainda um endereço IP definido, o cliente aceitará datagramas endereçados a este endereço. Nas mensagens geradas pelo cliente, a situação é ainda mais exótica: os datagramas tem como endereço de destino o `255.255.255.255` e como endereço de origem o `0.0.0.0`.

Um detalhe importante — e, talvez, pouco intuitivo — sobre o DHCP é que ele é, a rigor, um protocolo de camada de aplicação. Cliente e servidor trocam suas mensagens através de *sockets* UDP — o servidor na porta 67, e o cliente na porta 68. Quando o cliente obtém as informações necessárias para utilizar a rede, a aplicação cliente do DHCP as implementa através, por exemplo, de chamadas de sistema que requisitam ao SO alterações nas configurações de rede.

Por fim, voltemos brevemente à questão das demais configurações que podem ser obtidas pelo DHCP. Além do endereço IP, é bastante comum que servidores DHCP enviem o endereço IP do **roteador de primeiro salto** — também chamado de ***gateway* padrão**. O roteador de primeiro salto é aquele que interconecta a sub-rede a qual o cliente está se conectando ao restante da Internet. Portanto, qualquer datagrama endereço a um endereço exterior a esta sub-rede deverá ser encaminhado através deste roteador. Note que o resultado da recepção desta informação via DHCP é a adição de uma entrada na tabela de roteamento do *host* — a chamada **rota padrão**. Outra informação fundamental é o tamanho do prefixo da sub-rede a qual o endereço IP fornecido pertence — ou, de forma equivalente, sua máscara de sub-rede. É esta informação que permite que o *host* determine quais endereços estão na sua sub-rede e quais são externos. Outro exemplo relativamente comum de configuração fornecida por servidores DHCP é o endereço IP do servidor DNS local que atende àquela sub-rede.

3 Endereçamento Hierárquico

Na seção anterior, discutimos como um *host* pode obter um endereço válido ao conectar sua interface a uma sub-rede IP. Este termo *endereço válido* denota justamente um endereço que pertença à faixa de endereços pertencentes àquela sub-rede. Uma pergunta que não respondemos, no entanto, é como uma sub-rede obtém seus endereços?

No caso de uma empresa, por exemplo, que contrata o serviço de um ISP para conectar seus dispositivos à Internet, a resposta pode ser que o ISP aloca um pedaço da sua própria faixa de endereços IP para a empresa cliente. Suponha, por exemplo, que o ISP possua uma sub-rede com prefixo `/20` — totalizando, portanto, $2^{12} = 4096$ endereços. Suponha que a empresa precise de uma sub-rede suficientemente grande para endereçar cerca de 400 interfaces. Logo, a empresa precisa de uma sub-rede de prefixo `/23`, que provê um total de $2^9 = 512$ endereços.

Repare que a sub-rede do ISP é 8 vezes maior que a sub-rede necessária à empresa. Em particular, repare que podemos sub-dividir a sub-rede do ISP — de prefixo `/20` — em 8 sub-redes de prefixo `/23`. Uma forma alternativa de enxergar isso é pensar nos prefixos: como um prefixo `/23` usa três bits a mais que um prefixo `/20`, há $2^3 = 8$ combinações diferentes de prefixos `/23` compreendidos na sub-rede original.

Esta capacidade de quebrarmos uma sub-rede maior em sub-redes menores — e vice-versa — recebe o nome de **endereçamento hierárquico** e é extensivamente utilizada no projeto lógico de redes IP. Um problema típico desta área é justamente o ilustrado no exemplo anterior: dada uma determinada sub-rede IP "grande" para atender a uma certa instituição e demandas de endereços — possivelmente distintas — para departamentos diferentes, como podemos quebrar a sub-rede original em sub-redes menores alocadas a cada departamento com o menor desperdício possível de endereços?

3.1 Tabelas de Roteamento e Endereçamento Hierárquico

A atribuição de endereços através da sub-divisão de sub-redes em partes menores — o chamado endereçamento hierárquico — tem como efeito colateral (positivo) a agregação de rotas nas tabelas de roteamento da Internet. Devido à esta natureza hierárquica do endereçamento IP, endereços IP numericamente próximos tendem a ser atribuídos a dispositivos *topologicamente próximos*, *i.e.*, dispositivos acessíveis através de rotas semelhantes. A consequência disso é que grandes quantidades de endereços IP sequenciais passam a poder ser agregados em uma mesma faixa associada a cada entrada de uma tabela de roteamento na Internet, contribuindo assim para tabelas de roteamento pequenas[1].

Na prática, a faixa de endereços associada a uma entrada de uma tabela de roteamento nada mais é que uma sub-rede e, por conta do endereçamento hierárquico, geralmente os prefixos de sub-rede que constam em tabelas de roteamento na Internet são curtos — *i.e.*, correspondem a *grandes* sub-redes. Por fim, lembre-se que uma das convenções do IP é a utilização do *casamento por prefixo mais longo*. Isso significa que, se por qualquer motivo um roteador tiver interfaces de saída diferentes associadas a uma determinada sub-rede e a todos os outros endereços de uma sub-rede maior que a contenha, basta a inserção de duas entradas: uma para a sub-rede maior — portanto, com prefixo mais curto — e outra para a sub-rede menor, mais específica — portanto, com prefixo mais longo.

[1] Por uma série de fatores que fogem ao escopo desta disciplina, isso não é estritamente verdadeiro na Internet atual. Para efeito desta disciplina, no entanto, assumiremos isso como fato.

3.2 Endereçamento Hierárquico e ICANN

Na seção anterior, discutimos como uma sub-rede pode ser quebrada em sub-redes menores. Exemplificamos isso com a situação comum em que um ISP quebra sua faixa de endereços IP em sub-redes atribuídas a seus clientes. No entanto, ainda não está absolutamente claro como os ISPs, por sua vez, recebem suas faixas de endereço.

Em última instância, quem coordena a atribuição de endereços IP na Internet é uma entidade denominada ICANN (*Internet Corporation for Assigned Names and Numbers*). O ICANN — mais especificamente, um departamento seu chamado de IANA (*Internet Assigned Numbers Authority*) — é responsável pela atribuição e padronização de uma série de "números" usados na Internet[2], incluindo, por exemplo, os números de porta padrão para protocolos de camada de aplicação. Em particular, o ICANN controla o espaço de endereços IPv4 e distribui grandes blocos para as chamadas entidades de *Registro Regional da Internet*. Estas entidades regionais, por sua vez, aplicam políticas próprias para fragmentar estes grandes blocos em faixas menores — porém ainda tipicamente grandes — e atribuí-las a outras entidades mais localizadas (*e.g.*, como órgãos nacionais de cada país ou diretamente a ISPs).

Na América latina, por exemplo, a entidade de registro nacional é a LACNIC (*Latin America and Caribbean Network Information Centre*). O orgão brasileiro responsável por — entre outras coisas — atribuir as faixas de endereços IP é o CGI.br (Comitê Gestor da Internet no Brasil). O CGI.br recebe faixas atribuídas pela LACNIC e as redistribui de acordo com políticas nacionais para órgãos e entidades brasileiras.

[2] O ICANN também é a entidade responsável pelo gerenciamento do serviço de DNS na Internet, embora isso não seja diretamente relevante à discussão desta aula.