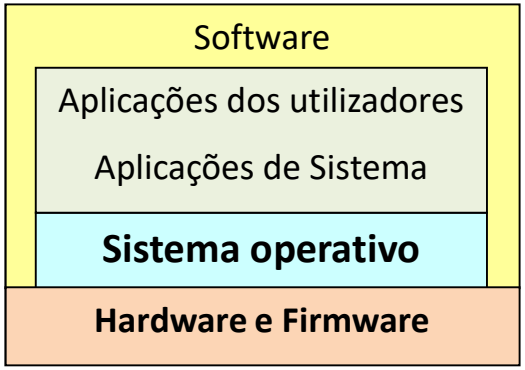
Operating Systems and Computer Networks Basics

SCOMRED, October 2017

Sistemas operativos – abstração do hardware

O sistema operativo é um conjunto de componentes de software (programas) que tem duas missões fundamentais:

Constituir uma **camada de abstração** relativamente ao hardware disponível. Existem muitos fabricantes de hardware, cada componente de hardware tem caraterísticas particulares (CPUs, chips de memória, interfaces de disco, interfaces de vídeo, interfaces de rede, etc.). Não é praticável desenvolver aplicações diferentes para cada componente de hardware específico.

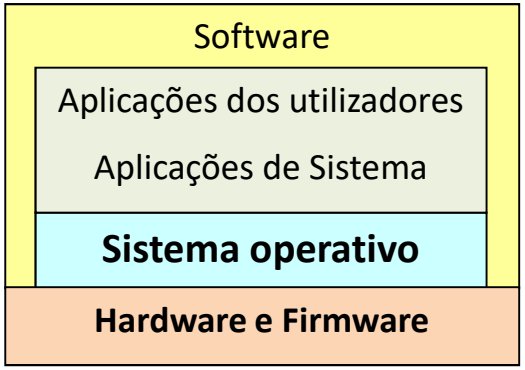


O sistema operativo sim, tem de interagir com o hardware. Para isso utiliza *drivers*, normalmente fornecidos pelo fabricante do hardware. Os drivers são pequenos componentes de software que podem ser integrados no sistema operativo. Permitem a interação com um componente de hardware. O driver terá de ser também o adequado para o sistema operativo em causa.

Assim o sistema operativo gere todo o hardware, as aplicações nunca podem aceder diretamente ao hardware. Isto torna-as independentes do hardware, dependem só do sistema operativo.

Sistemas operativos – plataforma de serviços

Uma vez que as aplicações não podem interagir diretamente com o hardware, terão de o usar indiretamente segundo as regras impostas pelo sistema operativo. O sistema operativo disponibiliza uma API com um conjunto vasto de funções para esse efeito.



Nesta perspetiva, sob o ponto de vista das aplicações, o sistema operativo é uma **plataforma estável de serviços** que podem ser invocados.

O facto de o sistema operativo manter o controlo garante a estabilidade, evitando por exemplo que as aplicações (eventualmente de utilizadores diferentes) interfiram umas com as outras.

O sistema operativo constitui um ponto único de contacto com o hardware, isto permite-lhe gerir todas as questões de concorrência de acesso aos recursos pelas aplicações.

O sistema operativo aplica uma política de direitos e permissões orientada por utilizadores e grupos de utilizadores.

Processos e threads

Um **processo** é uma unidade de execução sequencial, ou seja um programa que se encontra a executar uma sequência de operações.

Uma das funções principais do sistema operativo é gerir os CPUs (*Central Processing Units*) existentes e a memória central (RAM) que lhes está associada.

Para um processo poder executar uma operação tem de utilizar um CPU, mas normalmente temos mais processos do que CPUs. O sistema operativo tem de dividir a utilização dos CPUs pelos vários processos, faz isso atribuindo fatias de tempo a cada processo. Os processos não estão continuamente a realizar operações, mas sim apenas durante a fatia de tempo a que têm direito.

A cada processo o sistema atribui uma determinada quantidade de memória que ele usa para guardar variáveis e objetos, essa memória é privada, outros processos não podem aceder a ela. Isto evita interferências entre processos.

Um ***thread*** é uma unidade de execução sequencial dentro de um processo, um processo pode ter um ou vários *threads*. **A memória do processo é partilhada por todos os seus *threads*.**

Aplicações

Quando uma aplicação é carregada para a memória pelo sistema operativo (executada) é transformada num processo em execução.

O que se passa a partir daí é da responsabilidade da aplicação (dentro do que lhe é permitido pelo sistema operativo), pode criar novos processos e em cada processo pode criar vários *threads*. Quando concluir todos os seu objetivos pode terminar.

Dispositivos externos

Todo o hardware relevante adicional para além dos CPUs e memória central associada são considerados dispositivos externos. O sistema operativo controla totalmente o acesso aos dispositivos externos, o acesso pelas aplicações é conseguido recorrendo aos serviços do sistema operativo.

Um caso notório são os dispositivos de armazenamento externo, discos (HDD, SSD), unidades de DVD, etc. Estes dispositivos permitem o armazenamento persistente de quantidades enormes de bytes, de tal forma que cada um deles pode ser acedido individualmente.

Sistemas de ficheiros

Os dispositivos externos de armazenamento necessitam de ser formatados antes de as aplicações os poderem usar.

A **formatação** consiste em criar no disco um **sistema de ficheiros**, os sistemas de ficheiros permitem a organização da informação em ficheiros e pastas.

O conceito de ficheiro é muito mais adequados às aplicações e utilizadores, além disso facilitam o trabalho do sistema operativo no controlo de concorrência e permissões de acesso.

As aplicações utilizam os sistemas de ficheiros disponibilizados pelo sistema operativo através de funções orientadas por ficheiros como: **open**, **read**, **write** e **close**.

O acesso direto das aplicações aos discos, necessário por exemplo para os formatar, só é permitido para o administrador do sistema operativo.

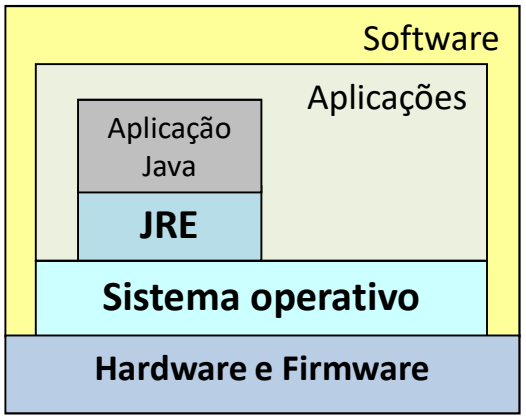
O próprio sistema operativo é colocado em execução através do seu carregamento a partir de um sistema de ficheiros. Igualmente as aplicações são colocadas em execução através do seu carregamento de um ficheiro para a memória.

Java e JRE (*Java Runtime Environment*)

Na maioria das linguagens de programação, quando o código fonte é compilado produz uma aplicação que pode ser executada diretamente pelo sistema operativo.

O inconveniente é que para cada tipo de sistema operativo diferente é necessário compilar novamente. Por outras palavra a aplicação depois de compilada não é portável entre diferentes tipos de sistemas operativo como por exemplo Windows, OS X e Linux.

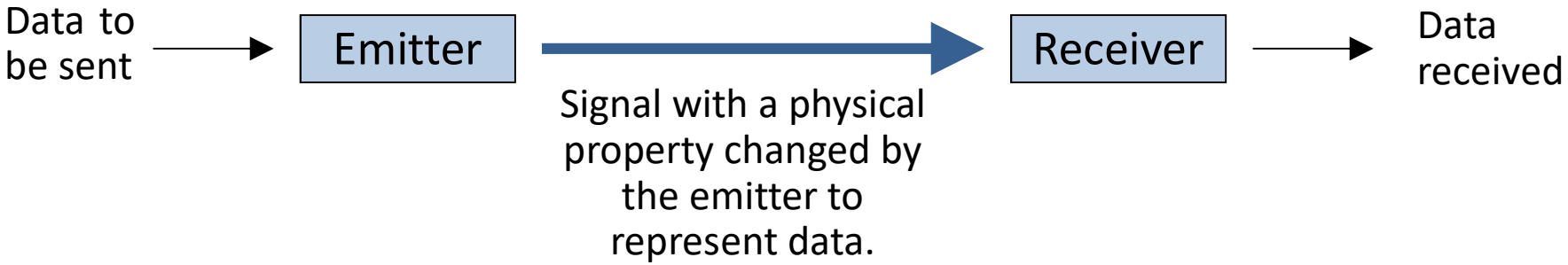
O Java é uma linguagem de programação orientada a objetos em que o resultado da compilação é uma aplicação que só funciona sobre o JRE. O JRE é ele próprio uma aplicação, também conhecido como **máquina virtual Java**.



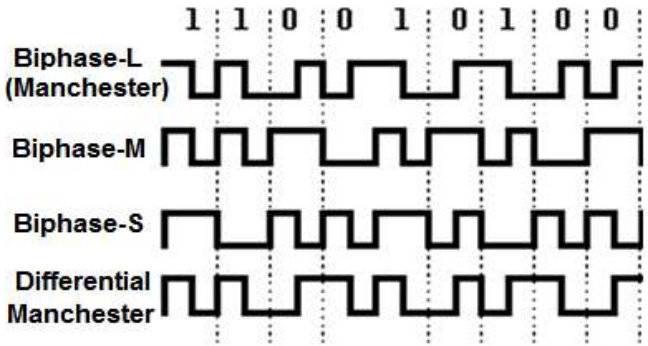
De certa forma o JRE assemelha-se a um pequeno sistema operativo, mas em lugar de interagir com o hardware interage com o sistema operativo. Seja qual for o sistema operativo desde que o JRE esteja instalado, qualquer aplicação Java compilada poderá ser executada.

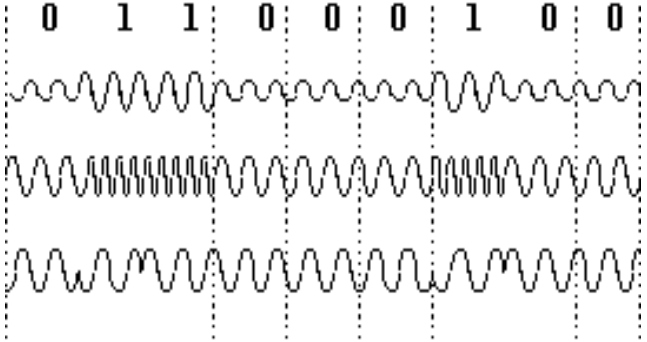
Redes de computadores - Sinais

Na base do funcionamento das redes de computadores estão os sinais. Sinais são fenómenos físicos que se propagam tais como a corrente elétrica, a luz e as ondas eletromagnéticas.



Alterando propriedades do sinal tais como a intensidade, a frequência ou a fase podemos representar informação que dessa forma é transmitida do emissor até ao recetor.





Pacotes

Porque as redes são partilhadas por muitos utilizadores, cada um só pode emitir dados durante um período de tempo máximo, depois tem de parar antes de poder emitir mais dados.

Os dados emitidos continuamente durante estes períodos de tempo constituem um **pacote**.

Os dados são colocados no sinal a uma cadência fixa designada **taxa de transmissão**, expressa em **bits por segundo (bps)**, consequentemente a um período de tempo máximo concedido para emitir cada pacote corresponde também um tamanho máximo que o pacote pode ter.

Exemplo:

- Taxa de transmissão: 1 Mbps (1 000 000 bps)
- Tempo máximo de emissão: 2 ms (0,002 segundos)
- Resulta que o tamanho máximo do pacote é:

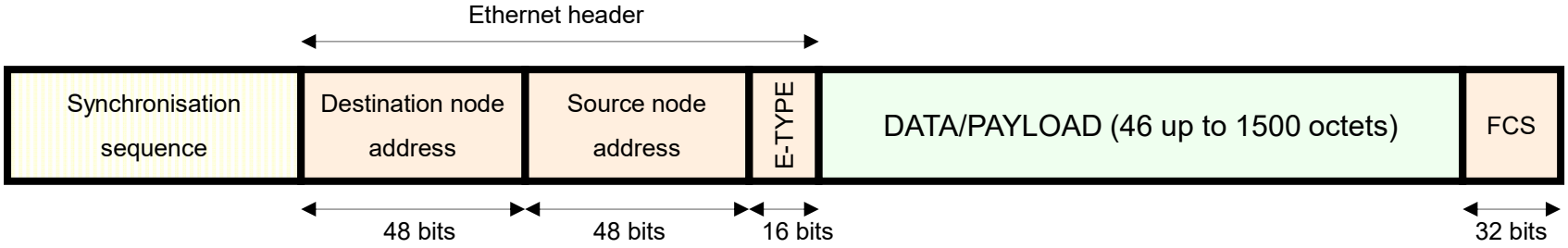
1 000 000 x 0,002 = 2 000 bits, ou seja 250 bytes

Endereços de nó

Nas redes de computadores não temos emissores dum lado e recetores do outro lado, todos os intervenientes emitem e recebem, designam-se **nós de rede.**

Como uma rede não possui apenas dois nós, cada nó tem de possuir um **endereço único**. Tal como um número de telefone ou uma morada, permite identificar um entre muitos.

O **endereço do nó de destino** têm de ser colocados nos pacotes para que a rede saiba onde o deve entregar, tal como colocamos a morada do destinatário numa carta de correio.



A imagem acima representa a estrutura de um **pacote Ethernet.** Tal como numa carta de correio, também colocamos o remetente para que o destinatário nos possa responder, ou a rede possa avisar que não conseguiu fazer a entrega.

Tecnologias de nível dois

As tecnologias que permitem transmitir pacotes sob a forma de sinais designam-se de nível dois (Layer 2). O problema é que existem várias alternativas, cada uma usa endereços de nó e pacotes de diferentes formatos que são incompatíveis entre si.

Normalmente os utilizadores finais contactam diretamente com as tecnologias de nível dois Ethernet, WiFi e GSM. Para isso utilizam uma **interface de rede** da tecnologia pretendida.

Ethernet e WiFi são tecnologias de rede local, respetivamente por cabo e wireless. São compatíveis entre si, utilizam endereços de nó de 48 bits normalmente conhecidos por endereços MAC, por exemplo: **00:60:B0:3C:93:DB**.

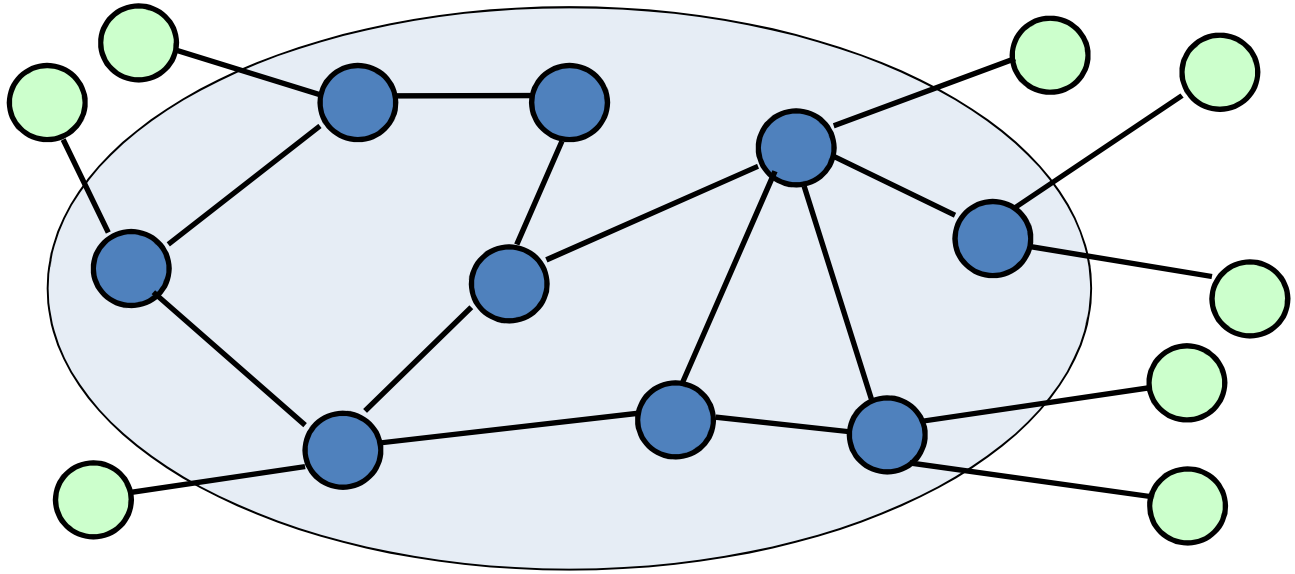
Devemos no entanto ter presente que quando a informação circula entre redes locais diferentes através de estrutura complexa da internet terá de atravessar um vasto conjunto de diferentes tecnologias de nível dois.

Sem a ajuda de algo mais nunca seria possível existir uma infraestrutura de comunicação global como a internet.

Encaminhamento de pacotes

Já percebemos que a rede é uma infraestrutura que recebe pacotes dos nós e tem a missão de os entregar no nó cujo endereço consta como endereço de destino.

Uma rede, além dos **nós finais** que emitem e recebem pacotes é também constituída por **nós intermédios** que retransmitem pacotes.



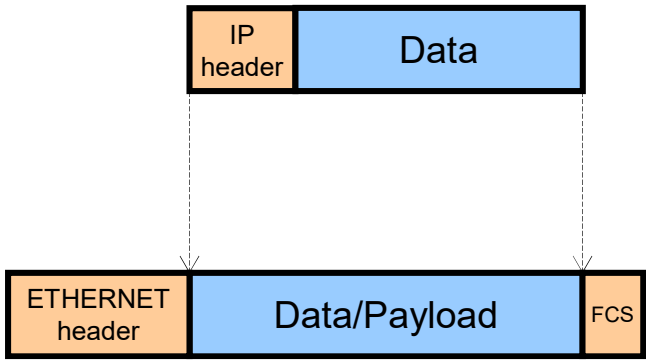
Quando um nó intermédio recebe um pacote tem de analisar o endereço do nó de destino, e em função do que conhece da rede determinar para onde o deve retransmitir. Os nós intermédios que encaminham pacotes de nível dois designam-se ***switches***.

Nível três – a camada de rede

O principal objetivo da camada de rede é homogeneizar, é aqui que se situa o protocolo IP (*Internet Protocol*). Este protocolo estabelece um formato de pacote próprio e usa endereços de nó de 32 bits. Este formato de pacote é usado por todos os nós ligados à internet.

Ao contrário do que acontece com os pacotes de nível dois, os pacotes de rede não são diretamente convertidos em sinais. Em vez disso são transportados por pacotes de nível 2, isto designa-se encapsulamento.

Na figura está representado o encapsulamento de um pacote IP num pacote Ethernet.

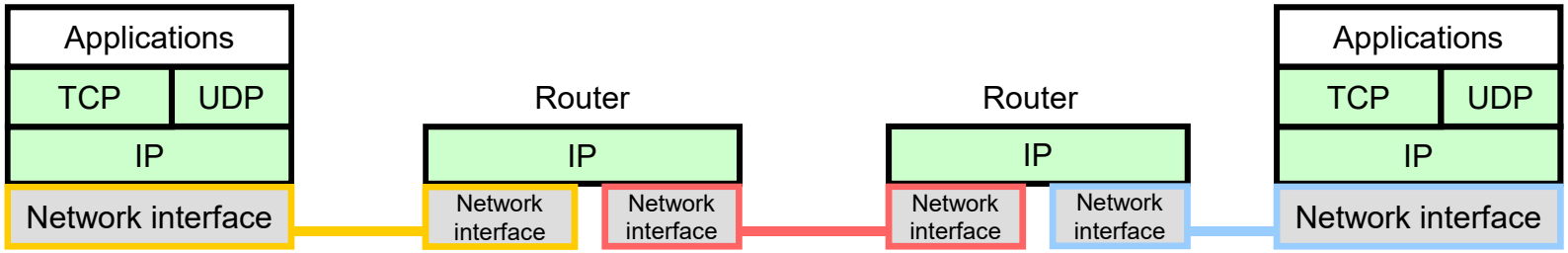


Note-se que em primeira análise o problema persiste porque continua a não ser possível transferir um pacote IP entre tecnologias de nível dois diferentes.

Nós intermédios do nível três – routers

O problema é resolvido pelos nós intermédios que retransmitem pacotes IP em vez de pacotes de nível dois como os *switches*.

Os nós que retransmitem pacotes IP designam-se ***routers***. O problema fica resolvido porque um *router* pode perfeitamente estar ligado a mais do que uma tecnologia de nível dois.



A figura acima ilustra essa situação, as diferentes cores na camada inferior pretendem representar diferentes tecnologias de nível dois.

Operando no nível três, um router nunca retransmite pacotes de nível dois. Quando recebe um pacote de nível dois extrai o pacote IP que está no seu interior e retransmite-o noutra interface de rede encapsulando-o noutro pacote de nível dois. **A tecnologia de nível dois usada para emitir (retransmitir) é totalmente independente da foi usada para receber.**

A camada de rede IP (nível 3) proporciona um serviço de transmissão de pacotes global através da internet baseado em endereços de nó de 32 bits. Para comunicar através da internet, cada nó terá de possuir um endereço único.

No entanto, o IP não apresenta ainda funcionalidades suficientes para ser usado diretamente pelas aplicações de rede normais. Existem dois problemas:

- **Não é fiável.** Quando um nó emite um pacote IP não existe garantia alguma de que ele chegue ao destino, pior ainda, o emissor não é informado se o pacote foi entregue ou não.

- **Não identifica aplicações.** O IP só usa endereços de nó, se tudo correr bem o pacote é entregue no nó de destino. O problema é que num nó não existe apenas uma aplicação de rede, existem muitas. É necessário dispor de um mecanismo que permita identificar aplicações individuais no nó, caso contrário só poderia existir uma aplicação de rede em cada nó.

Nível 4 – Números de porto

Os protocolos de nível quatro TCP e UDP fazem-se transportar através do encapsulamento em pacotes IP, estes protocolos destinam-se a suportar diretamente as aplicações dos utilizadores. Existem por isso aplicações de rede UDP e TCP conforme utilizem os protocolos UDP e TCP

Ambos os protocolos definem o conceito de número de porto (***port number***), trata-se números únicos de **16 bits** que são atribuídos por cada nó às suas aplicações de rede. Os dados transferidos através dos protocolos UDP e TCP transportam dois números de porto:

- **número de porto de origem** - identifica a aplicação que emitiu os dados no nó de origem.
- **número de porto de destino** - identifica a aplicação que deve receber os dados no nó de destino.

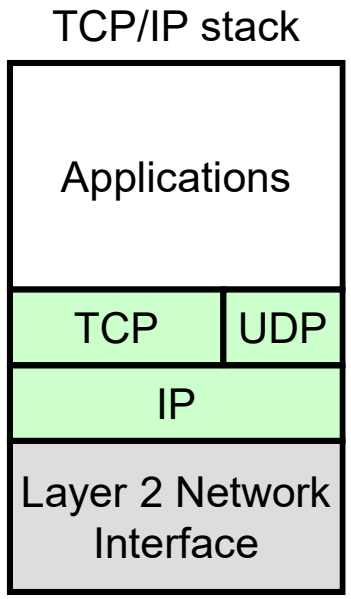
O caráter único dos números de porto só precisa de ser garantido dentro de cada nó. Juntamente com os endereços de nó IP, os números de porto identificam as aplicações a nível global.

Nível 4 – Fiabilidade

O único valor acrescentado pelo UDP (*User Datagram Protocol*) é a implementação dos números de porto, consequentemente mantém a falta de fiabilidade do IP. Quando um pacote UDP é emitido não há garantia de que seja entregue, nem feedback.

O protocolo TCP (*Transmission Control Protocol*) vai bastante mais longe, além dos números de porto garante a entrega dos dados à aplicação no nó de destino.

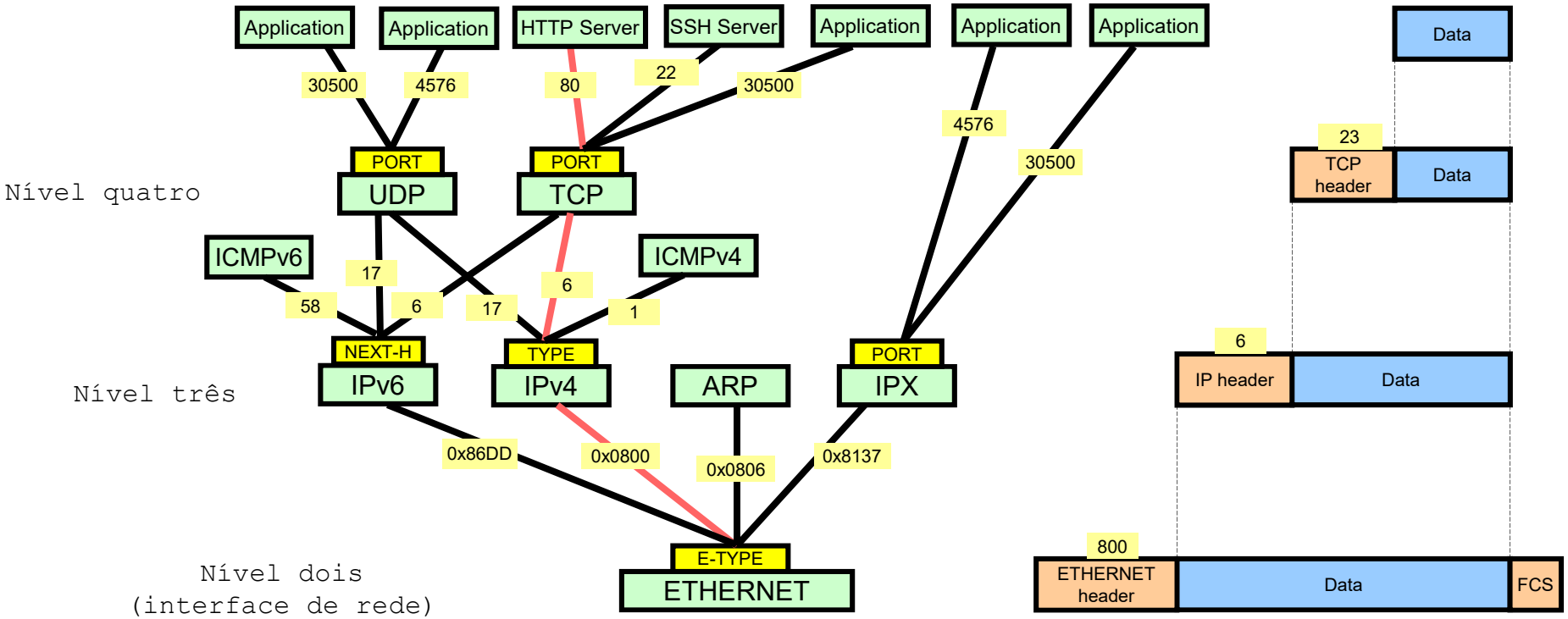
O TCP é também um protocolo *connection-oriented*. Isto significa que antes de se poderem enviar e receber dados é necessário estabelecer uma ligação lógica entre as aplicações, a conexão TCP.



Existem algumas coisas que as aplicações conseguem fazer com o UDP que não são possíveis com o TCP, a mais notória é enviar dados para endereços especiais designados *multicast* e *broadcast*. Enviar pacotes UDP para estes endereços leva a rede a entregar uma cópia em múltiplos nós da rede. Como o TCP é *connection-oriented*, isto impossível em TCP.

A pilha de protocolos

No interior de cada nó, nos vários níveis, são usadas etiquetas numéricas para identificar os dados. São colocadas quando os dados são emitido e são usadas para o encaminhar para a camada correta quando os dados são recebidos.

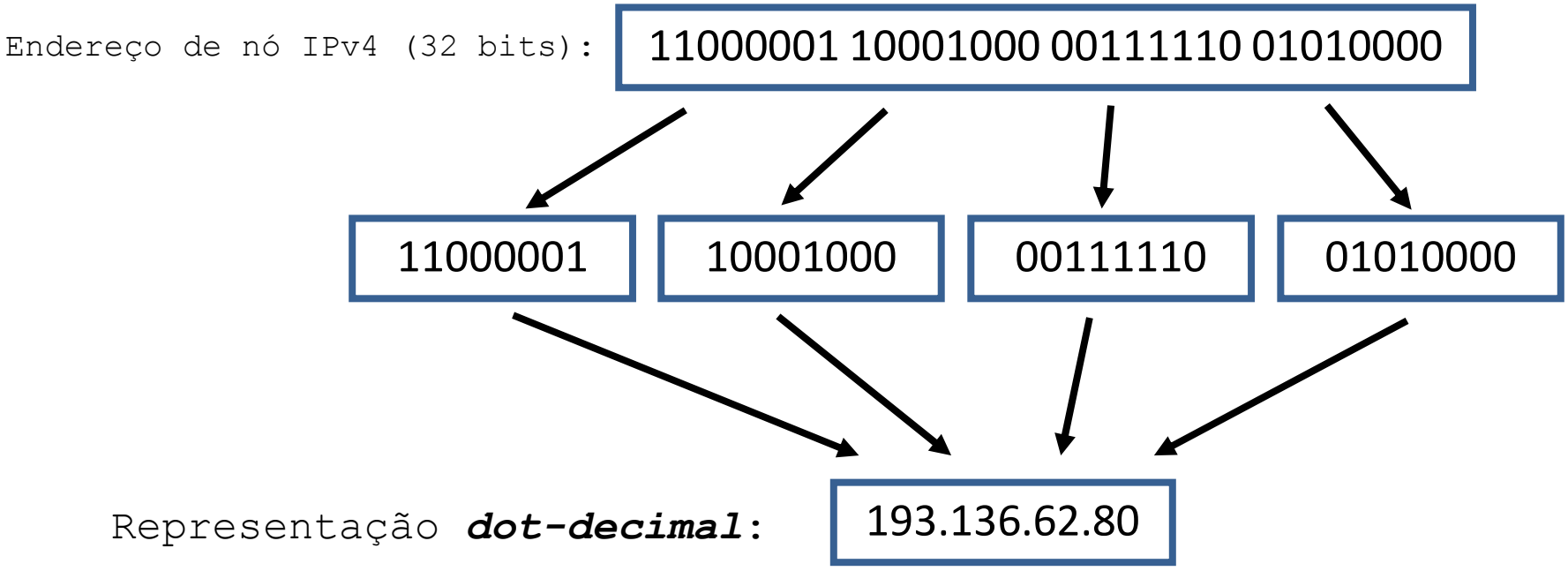


Os números de porto estão no topo, servem para identificar aplicações individuais no nó. Alguns números de porto, são standard, por exemplo o 80 deve ser usado por servidores HTTP.

Endereço de nó IPv4

Cada nó da internet possui um endereço único de 32 bits. Para melhor legibilidade humana, são divididos em 4 conjuntos de 8 bits (octetos). Cada octeto é representado em notação decimal, para separar os quatro octetos usa-se um ponto. Esta é a representação ***dot-decimal***.

Exemplo:



Endereço de rede IPv4

O conjunto de 32 bits do endereço de um nó IPv4 identifica também a rede, um determinado número de bits mais significativos (da esquerda) identificam a rede, os restantes bits identificam o nó na rede.

Os endereços de redes facilitam a vida dos *routers*, ao contrários dos nós intermédios de nível dois (*switches*) não precisam de conhecer todos os nós, basta conhecer todas a redes.

O número de bits que estão a ser usados para identificar a rede designa-se ***prefix length***. O endereço de uma rede IPv4 ou ***network prefix*** é uma representação *dot-decimal* do endereço que se obtém removendo (colocando a zero) todos bits para além do *prefix length*.

Uma rede IP é identificada através do conjunto:

network prefix/prefix length

Exemplos: 193.136.62.0/24

10.30.0.0/16 130.44.5.0/16 (incorreto)

Mascara de rede IPv4

Em alguns sistemas ainda se usam mascaras de rede para definir o *prefix length* no IPv4. A mascara de rede é uma representação *dot-decimal* onde todos os bits dentro do *prefix length* têm o valor 1. Exemplos:

Prefix Length	Network Mask
/8	255.0.0.0
/16	255.255.0.0
/24	255.255.255.0
/26	255.255.255.192

Para poder operar, um nó IP tem de conhecer o seu endereço de nó e o *prefix length*. Com estes dados determina qual é a sua rede, basta eliminar todos os bits do seu endereço que estão para além do *prefix length*. Exemplos:

Node address/prefix length	Network prefix
190.30.57.28/16	190.30.0.0/16
192.168.32.47/24	192.168.32.0/24
172.16.100.130/25	172.16.100.128/25

Número máximo de nós numa rede IP

Dois nós IP pertencem à mesma rede IP se os seus endereços de nó têm prefixos iguais, os restantes bits identificam cada nó no interior da rede e serão necessariamente únicos.

O *prefix length* determina o número máximo de nós que a rede IP pode comportar. São os bits que ficam fora do prefixo de rede permitem identificar diferentes nós na rede (bits de nó). Nenhum nó pode usar um endereço com todos os bits de nó a zero (porque isso é o prefixo da rede), no caso do IPv4 também nenhum nó pode usar um endereço com todos os bits de nó a um (este endereço está reservado para broadcast). Exemplos:

$$Nós = 2^{(32-Prefix\ length)} - 2$$

Prefix length	Número máximo de nós
/16	65534
/24	254
/30	2

O **endereço de broadcast** de uma rede é constituído pelo prefixo da rede seguido de todos os bits de nó com o valor um. Quando um pacote é enviado para este endereço será entregue uma cópia a todos os nós que fazem parte da rede.

Endereços IP especiais

Para garantir que todos os endereços IP são únicos existe uma entidade global que faz a sua gestão, trata-se da Internet Assigned Numbers Authority (IANA). Alguns endereços IPv4 foram reservados para utilizações específicas, exemplos:

Prefixes	Número máximo de nós
10.0.0.0/8 172.16.0.0/12 192.168.0.0/16	Reservados para uso privado local, não são reconhecidos pela internet. Destinam-se a ser usados localmente em atividades de experimentais. Apesar de não serem reconhecidos na internet, usado a técnica NAT (*Network Address Translation*) é possível esconder um grande conjunto de endereços privados atrás de um endereço não privado e desta forma os nós privados conseguem usar a internet.
127.0.0.0/8	Endereços de *loopback*, apenas são válidos no interior de cada nó, normalmente os nós usam o endereço 127.0.0.1/8.
224.0.0.0/4	Multicast, permitem o envio para conjuntos de nós (grupos de multicast), o broadcast é um caso particular de multicast.
255.255.255.255	Endereço de broadcast genérico na rede local. É útil para ser usado pelas aplicações em lugar do endereço de broadcast de uma rede especifica, desse modo a aplicação poderá funcionar em qualquer rede.

Configuração de nós IPv4

Os dois parâmetros base para configurar um nó de rede IP são o **endereço de nó** e o ***prefix length*** ou **mascara de rede.** Com estes dados o nó determina o prefixo da sua rede.

Estes elementos são suficientes para comunicar com outros nós da mesma rede IP, para comunicar com nós de outras redes IP é necessário recorrer a um *router*, o *router* transfere pacotes entre redes IP diferentes.

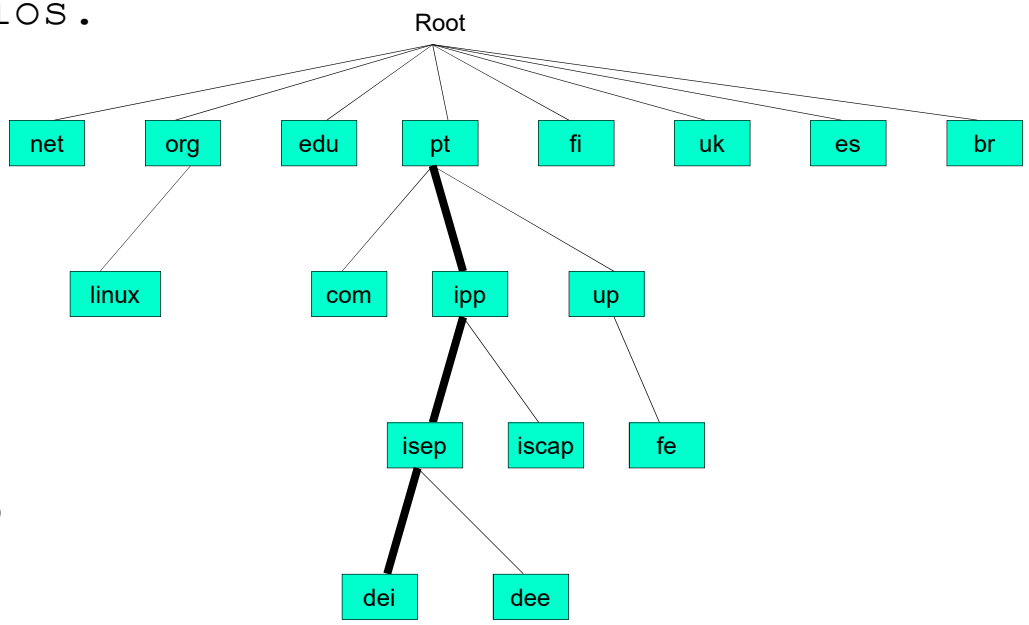
Assim, para que um nó possa comunicar com nós de outras redes IP tem de conhecer o **endereço IP de um router da sua rede**, é o chamado ***default gateway*** que deve ser adicionado à configuração da rede.

Ao nível das aplicações, o manuseamento pelos utilizadores de endereços IP não é cómodo, em seu lugar podem ser usados nomes DNS (*Domain Name System*).

Para o efeito é necessário definir na configuração de rede os **endereços IP dos servidores DNS** locais. Adicionalmente podem também ser indicados nomes de domínio a aplicar quando são fornecidos nomes não qualificados.

Configuração de nós IPv4

O DNS é uma infraestrutura em árvore de resolução de nomes organizada em nomes de domínios.



Os servidores DNS apenas resolvem nomes qualificados, ou seja nomes que identificam em termos absolutos o ramo da árvore, como por exemplo www.dei.isep.ipp.pt.

Para resolver nomes não qualificados, em cada nó podem ser definidos nomes de domínios a usar para qualificar esses nomes.

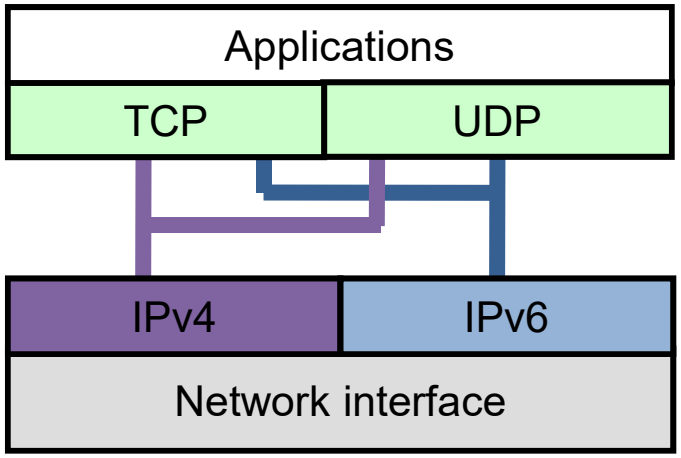
Na maioria das redes locais, todos os parâmetros de configuração de rede são fornecidos automaticamente aos postos de trabalho através do serviço **DHCP** (*Dynamic Host Configuration Protocol*): endereço IP, *prefix length*, *default gateway* e endereços dos servidores DNS.

IPv6

A versão mais recente do IP é o IPv6 que está a ser gradualmente introduzido na internet coexistindo IPv4 e IPv6. As principais diferenças são: endereços de nó de 128 bits, não existe broadcast (apenas multicast) o *default gateway* e o prefixo da rede são definidos automaticamente.

Atualmente a maioria dos nós de rede são ***dual-stack*** IPv4/IPv6, ou seja suportam os dois protocolos, as aplicações usam ambos os protocolos de forma transparente através do UDP e TCP

Apesar de existirem duas implementações do nível três (IPv4 e IPv6), existe uma única camada UDP e uma única camada TCP. Se a aplicação usa um endereço IPv6, as camadas TCP/UDP usam IPv6, se a aplicação usa um endereço IPv4, as camadas TCP/UDP usam IPv4.



Quando se usam nomes DNS, tudo depende da resposta do servidor DNS. Um nome DNS pode ter associado a si um endereço IPv4 e um endereço IPv6, nesse caso o sistema operativo é que decide.

Representação de endereços IPv6

Tudo o que foi dito sob a forma de usar endereços IPv4 aplica-se também ao IPv6, nomeadamente os conceitos de **endereço de nó**, ***prefix length*** e endereço da rede (***network prefix***). Como não existe broadcast, o endereço em que os bits de nó têm todos o valor um passa a ser válido. No IPv6 não se usam mascaras de rede para representar o *prefix length*.

Os endereços de 128 bits do IPv6 são representados sob a forma de uma sequência de 8 conjuntos de 16 bits, em notação hexadecimal separadas pelo símbolo de dois pontos:

xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx

Em cada conjunto, os zeros à esquerda devem ser omitidos. Em cada endereço a maior sequência de conjuntos com o valor zero deve ser substituída por um duplo símbolo de dois pontos.

Por exemplo: **fd1e:3278:0a04:0005:0000:0000:0000:0034/64**

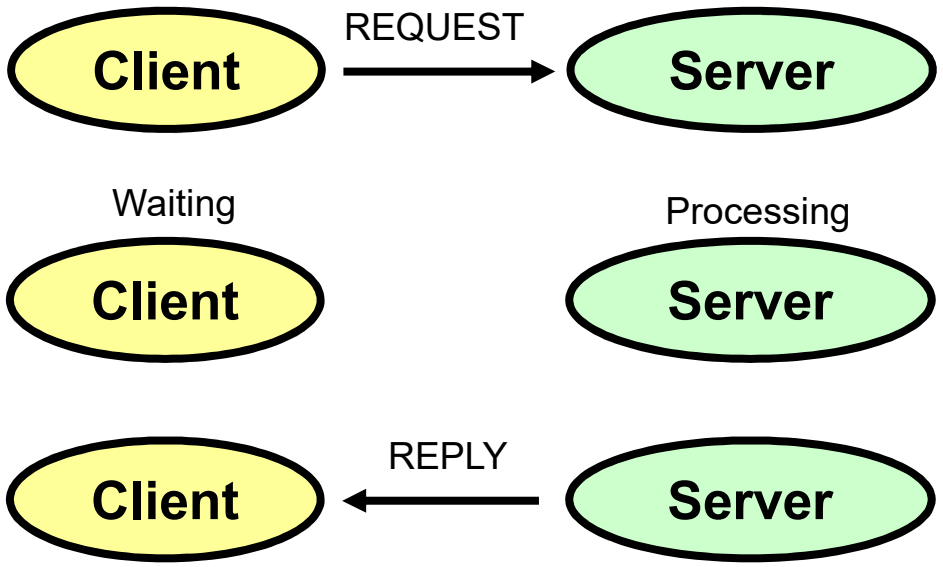
Deve ser representado por: **fd1e:3278:a04:5::34/64**

Modelo de comunicação cliente-servidor

A maioria das aplicações de rede utiliza um modelo de diálogo muito simples designado cliente-servidor.

As duas aplicações assume dois papeis distintos o servidor está permanentemente à espera de pedidos de clientes.

A iniciativa cabe ao cliente que envia um pedido ao servidor, para isso necessita de conhecer o **endereço IP** do servidor e o respetivo **número de porto.**



Quando o servidor recebe o pedido processa-o, entretanto o cliente fica à espera da resposta.

Finalmente o servidor envia a resposta ao cliente, para determinar o destino da resposta copia do pedido o endereço IP de origem e o número de porto de origem.

Atividades práticas com o *Packet Tracer*

https://www.netacad.com/courses/packet-tracer-download/

(Registo gratuito -> Download do *Packet Tracer*)