

Fundamentos de Redes de Comunicação

CTESP – Redes e Sistemas Informáticos 5 – Transporte

António Godinho

Fundamentos de Redes de Comunicação



1

Transporte

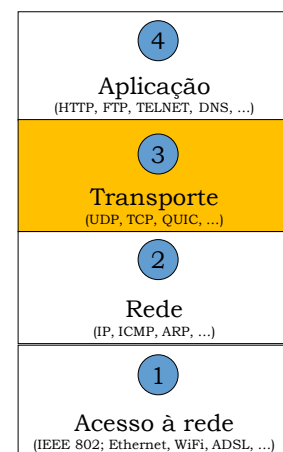
Terminado o estudo do nível Rede do modelo TCP/IP vamos passar à **camada de transporte**.

O protocolo IP (Camada de Rede) tem algumas características que o tornam “incompleto” para algumas aplicações:

- Não garante a entrega dos pacotes
- Não solicita a retransmissão em caso de perdas
- Não implementa controlo de fluxo em caso de congestionamento.

É função da Camada de Transporte implementar estas funções. Os principais protocolos desta camada são o **UDP** e **TCP**.

Fundamentos de Redes de Comunicação - Cap 5 Transporte



2

2

Transporte – Portos e Sockets

Na comunicação entre dois processos há a necessidade de os identificar pois cada máquina pode estar a correr vários processos em simultâneo

Exemplo: se estivermos a receber um e-mail e a ouvir rádio online, a nossa máquina recebe pacotes destinados a cada um destes processos. Se apenas estivessem identificados pelo nosso endereço IP, a distinção entre eles não era possível

Surge então o conceito de **porto** e **socket**: permitem a identificação unívoca do canal lógico de comunicação estabelecido entre dois processos que estejam em comunicação

Fundamentos de Redes de Comunicação - Cap 5 Transporte

3

3

Transporte – Portos e Sockets

Cada processo que pretenda comunicar com outro deve identificar-se na arquitetura TCP/IP por um ou mais portos (ou portas):

- **Porto** (ou porta): número de 16 bits - valores entre 0 e 65535; Todos os pacotes trocados entre dois sistemas têm de identificar o endereço IP de destino e o porto de destino.
- O conjunto formado pelo Endereço IP + porto é o **socket**
- Os portos podem ser Bem-Conhecidos/well-know ou Efêmeros/cliente

Portos Bem-conhecidos ou “well-know ports” [1 a 1023] (geridos e atribuídos pela IANA) - algumas aplicações utilizam sempre o mesmo porto. Alguns exemplos na tabela seguinte:

Fundamentos de Redes de Comunicação - Cap 5 Transporte

4

4

Portos bem conhecidos

Porta	TCP ou UDP	Nome do protocolo ou serviço	Nome do serviço	Usado por/informações adicionais
7	TCP/UDP	echo	echo	ping, traceroute
20	TCP	File Transport Protocol (FTP)	ftp-data	-
21	TCP	Controle de FTP	ftp	-
22	TCP	Secure Shell (SSH)	ssh	Acesso Remoto
23	TCP	Telnet	telnet	-
25	TCP	Simple Mail Transfer Protocol (Protocolo simples de transferência de e-mails - SMTP)	smtp	Mail (para enviar e-mail)
53	TCP/UDP	Sistema de nomes de domínio (DNS)	domínio	
67	TCP	File Transfer Protocol (FTP)	ftp	
68	TCP	File Transfer Protocol (FTP)	ftp	
69	UDP	Trivial File Transfer Protocol (TFTP)	tftp	
80	TCP	Hypertext Transfer Protocol (HTTP)	http	
110	TCP	Post Office Protocol (POP3)	pop3	Mail (para receber e-mail)
143	TCP	Internet Message Access Protocol (IMAP)	imap	Mail (para receber e-mail)
443	TCP	Secure Sockets Layer (SSL ou "HTTPS")	https	
993	TCP	SSL para IMAP no Mail	imaps	
995	TCP/UDP	SSL para POP no Mail	pop3s	-

Fundamentos de Redes de Comunicação - Cap 5 Transporte

5

5

Transporte – Portos Efêmeros

Portos Efêmeros/cliente : [1024, 65535]

Portos não alocados a uma aplicação ou serviço específico e que podem ser utilizados pelos clientes que iniciam a comunicação com o servidor.

Tipicamente são escolhidos pelo sistema operativo no início de uma comunicação.

Por exemplo, se desenvolverem uma nova aplicação poderão escolher um destes portos para a comunicação com o respetivo servidor.

Uma vez que ao nível do pacote IP se faz a distinção entre os protocolos de transporte, pode-se usar o mesmo número de porto para dois processos distintos, um que utilize o UDP e outro que utilize o TCP

Fundamentos de Redes de Comunicação - Cap 5 Transporte

6

6

Protocolos de Transporte

A camada de transporte tem dois principais protocolos: **UDP** e **TCP**

	UDP	TCP
Ligação	Connectionless – Ausência de ligação	Modo de ligação – exige estabelecimento de ligação prévia à troca de dados
Consumo de recursos	Mais leve	Mais complexo (pesado)
Controlo de fluxo	Não	Sim
Controlo de congestionamento	Não	Sim
Garante Fiabilidade?	Não	Sim (os pacotes chegam todos ao destino, por ordem e sem duplicações)
Nome dos pacotes	Datagramas	Segmentos
Broadcast	Sim	Não

Fundamentos de Redes de Comunicação - Cap 5 Transporte

7

7

Protocolo UDP

User Datagram Protocol

Fornecer um serviço não orientado à conexão (Connectionless)

Não fiável (não há garantia de entrega do datagrama)

Simples (não introduz muito overhead na rede): Apenas acrescenta ao pacote IP a identificação do porto do processo de aplicação remetente e do porto do processo de aplicação destinatário (porto origem e porto destino)

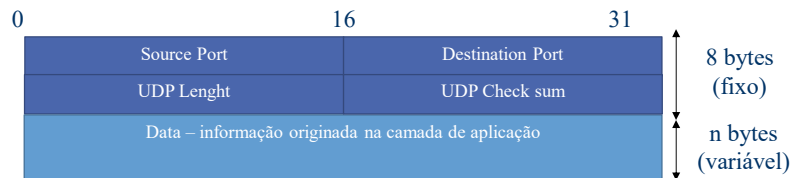
Permite comunicação unicast, broadcast, multicast

Fundamentos de Redes de Comunicação - Cap 5 Transporte

8

8

Formato do datagrama UDP



- **Source Port (16b):** número do porto que identifica o processo de aplicação remetente
- **Destination port (16b):** número do porto que identifica o processo de aplicação destinatário (remoto)
- **UDP length (16b):** N° de bytes de todo o datagrama
- **UDP checksum (16b):** (opcional) verifica integridade de todo o datagrama - cabeçalho+dados+pseudo cabeçalho (IP Origem, IP Destino, Protocolo, UDP length)
- **Data:** dados da aplicação a transportar (recebido da camada de aplicação)

Fundamentos de Redes de Comunicação - Cap 5 Transporte

9

9

Protocolo UDP

Quando uma comunicação utiliza o protocolo de transporte UDP, o UDP apenas identifica as aplicações que correm numa máquina através dos sockets (conjunto de endereço IP e porto).

- Não há contato prévio entre emissor e recetor para confirmar a disponibilidade de comunicação
- Os pacotes são enviados e o UDP não dará a indicação se chegaram ou não
- Se os pacotes chegarem fora de ordem, o UDP também não tem forma de identificá-los para poderem ser ordenados
- Se for necessário garantir entrega de pacotes e a sua ordem, terá de ser a aplicação a implementar algum mecanismo de controlo

Exemplos de protocolos de aplicação que usam UDP:

Domain Name System (DNS), Network File System (NFS), Simple Network Management Protocol (SNMP), Lightweight Directory Access Protocol (LDAP), NetBIOS, Trivial File Transfer Protocol (TFTP)

Fundamentos de Redes de Comunicação - Cap 5 Transporte

10

10

Protocolo UDP

Exemplo de uma aplicação que usa UDP: DNS

Com o wireshark ativo executem o seguinte comando: `nslookup www.sapo.pt`

Analistem o primeiro pacote do protocolo DNS:

No.	Time	Source	Destination	Protocol	Info
2246	409.975051	192.168.2.4	192.168.2.1	DNS	Standard query A ww.sapo.pt
2247	410.092531	192.168.2.1	192.168.2.4	DNS	Standard query response CNAME ww.sapo.pt A

Frame 2246: 70 bytes on wire (560 bits), 70 bytes captured (560 bits)
Ethernet II, Src: IntelCor_3e:51:16 (00:1b:77:3e:51:16), Dst: Belkin_81:a0:20 (00:11:50:81:a0:20)
Internet Protocol, Src: 192.168.2.4 (192.168.2.4), Dst: 192.168.2.1 (192.168.2.1)
User Datagram Protocol, Src Port: 53317 (53317), Dst Port: domain (53)
Source port: 53317 (53317)
Destination port: domain (53)
Length: 36
Checksum: 0xd511 [validation disabled]
Domain Name System (query)

Deverão verificar:

- Utilização do UDP (User Datagram Protocol)
- Não há estabelecimento de sessão (o pedido é enviado sem contacto prévio com o destino)
- O pedido é enviado de um porto efêmero (no caso acima foi o 53317) para um well-known port (o porto 53 está reservado para a aplicação DNS)

Fundamentos de Redes de Comunicação - Cap 5 Transporte

11

11

Protocolo UDP

Exemplo de uma aplicação que usa UDP: DNS

O segundo pacote DNS é a resposta do servidor de DNS ao nosso pedido:

No.	Time	Source	Destination	Protocol	Info
2246	409.975051	192.168.2.4	192.168.2.1	DNS	Standard query A ww.sapo.pt
2247	410.092531	192.168.2.1	192.168.2.4	DNS	Standard query response CNAME ww.sapo.pt A

Frame 2247: 299 bytes on wire (2392 bits), 299 bytes captured (2392 bits)
Ethernet II, Src: Belkin_81:a0:20 (00:11:50:81:a0:20), Dst: IntelCor_3e:51:16 (00:1b:77:3e:51:16)
Internet Protocol, Src: 192.168.2.1 (192.168.2.1), Dst: 192.168.2.4 (192.168.2.4)
User Datagram Protocol, Src Port: domain (53), Dst Port: 53317 (53317)
Source port: domain (53)
Destination port: 53317 (53317)
Length: 265
Checksum: 0x76db [validation disabled]
Domain Name System (response)

- Continua a ser via UDP
- O porto de origem é o 53, well-know port do DNS
- O porto de destino é o 53317 que, neste exemplo, foi o porto efêmero que o nosso computador utilizou para identificar o pedido.

Fundamentos de Redes de Comunicação - Cap 5 Transporte

12

12

Protocolo UDP

Outro exemplo de uma aplicação com UDP: Echo (Ping), Porta Well-known 7 UDP

IP(192.168.2.151, 192.168.2.153, (UDP (Porto Origem = 2083, Porto Destino = 7)))

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.2.151	192.168.2.153	ECHO	Request
2	0.039990	192.168.2.153	192.168.2.151	ECHO	Response

Frame 1 (55 bytes on wire, 55 bytes captured) Ethernet II, Src: 00:12:f0:65:0c:21 (00:12:f0:65:0c:21), Dst: 00:80:c8:2f:07:73 (00:80:c8:2f:07:73) Internet Protocol, Src: 192.168.2.151 (192.168.2.151), Dst: 192.168.2.153 (192.168.2.153) User Datagram Protocol, Src Port: 2083 (2083), Dst Port: echo (7) Echo Echo data: 5465737465206465206563686f					
---	--	--	--	--	--

IP(192.168.2.153, 192.168.2.151, (UDP (Porto Origem = 7, Porto Destino = 2083)))

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.2.151	192.168.2.153	ECHO	Request
2	0.039990	192.168.2.153	192.168.2.151	ECHO	Response

Frame 2 (60 bytes on wire, 60 bytes captured) Ethernet II, Src: 00:04:e2:ab:82:c6 (00:04:e2:ab:82:c6), Dst: 00:12:f0:65:0c:21 (00:12:f0:65:0c:21) Internet Protocol, Src: 192.168.2.153 (192.168.2.153), Dst: 192.168.2.151 (192.168.2.151) User Datagram Protocol, Src Port: echo (7), Dst Port: 2083 (2083) Echo Echo data: 5465737465206465206563686f					
---	--	--	--	--	--

Fundamentos de Redes de Comunicação - Cap 5 Transporte

13

13

Protocolo TCP

Transmission Control Protocol

- Orientado à Ligação:
 - Necessidade de estabelecer uma ligação prévia antes do envio de dados. Confirma a disponibilidade do destino e evita o envio de pacotes se não houver forma de comunicar;
 - No fim da transmissão deve ser encerrada a ligação.
- Fiável:
 - Garante que os dados chegam ao destino e identifica a ordem pela qual devem ser colocados na aplicação de destino
 - Necessita de um mecanismo de confirmação de receção e de numeração dos dados
 - Faz o Controlo de erros

Fundamentos de Redes de Comunicação - Cap 5 Transporte

14

14

Protocolo TCP

Transmission Control Protocol

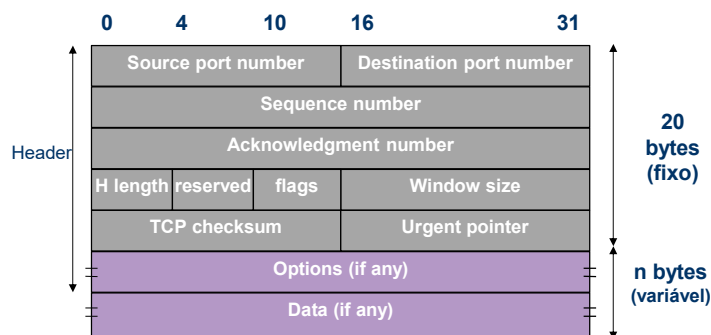
- Controlo do Fluxo:
 - Ambos os extremos do fluxo de dados da ligação utilizam mecanismos de controlo de fluxo
 - Adequação entre débitos de envio e de receção
- Controlo de Congestionamento:
 - Adequação às condições de débito do canal de comunicação
- Full Duplex:
 - Permite o fluxo concorrente de dados em ambos os sentidos da conexão

Fundamentos de Redes de Comunicação - Cap 5 Transporte

15

15

Segmento TCP



- **Source port** (16b) e **Destination port** (16b): números dos porto origem/destino
- **SN – Sequence number** (32b): identifica o número de sequência do primeiro byte do segmento de dados que está a ser enviado
- **AN – Acknowledgement number** (32b): contém o próximo número de sequência do byte que o remetente espera receber.
- **Window size** (16b): número de bytes que o remetente está disponível a receber no próximo segmento (utilizado para o controlo de fluxo)
- **Data**: dados de aplicação a transportar

Fundamentos de Redes de Comunicação - Cap 5 Transporte

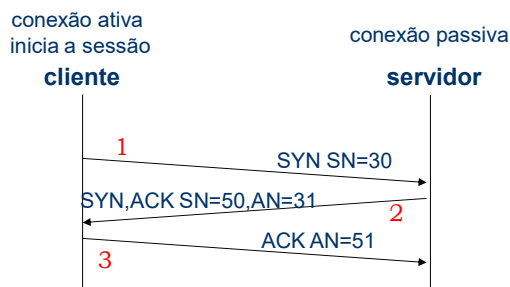
16

16

TCP – Início de Sessão

Antes de qualquer troca de informação numa ligação com TCP, os dois sistemas têm de previamente estabelecer uma ligação.

É feito através do método **three way handshake**:



Fundamentos de Redes de Comunicação - Cap 5 Transporte

17

17

TCP – Início de Sessão

Exemplo de um serviço que usa TCP: HTTP

Com o wireshark ativo, acessem a uma página web no browser.

Antes de ser enviado o pedido é feito o three-way-handshake com o servidor onde está a página:

No.	Time	Source	Destination	Protocol	Info
7	3.943551	192.168.2.4	62.28.183.139	TCP	49969 > http [SYN] Seq=0 win=8192 Len=0 MSS=1460 WS=2
8	3.974151	62.28.183.139	192.168.2.4	TCP	http > 49969 [SYN, ACK] Seq=0 Ack=1 win=8192 Len=0 MSS=1460 WS=2
9	3.974318	192.168.2.4	62.28.183.139	TCP	49968 > http [ACK] Seq=1 Ack=1 win=16968 Len=0
10	3.974497	62.28.183.139	192.168.2.4	TCP	http > 49969 [SYN, ACK] Seq=0 Ack=1 win=8192 Len=0 MSS=1460 WS=2
11	3.974539	192.168.2.4	62.28.183.139	TCP	49969 > http [ACK] Seq=1 Ack=1 win=16968 Len=0
12	3.976071	192.168.2.4	62.28.183.139	HTTP	GET / HTTP/1.1

Frame 7: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0	
Ethernet II, Src: IntelCor_3e:51:16 (00:1b:77:3e:51:16), Dst: Belkin_81:a0:20 (00:11:50:81:a0:20)	
Internet Protocol, Src: 192.168.2.4 (192.168.2.4), Dst: 62.28.183.139 (62.28.183.139)	
Transmission Control Protocol, Src Port: 49969 (49969), Dst Port: http (80), Seq: 0, Len: 0	
Source port: 49969 (49969)	
Destination port: http (80)	
[Stream index: 5]	
Sequence number: 0 (relative sequence number)	
Header length: 32 bytes	
Flags: 0x02 (SYN)	
0000. = Reserved: Not set	
....0. = Nonce: Not set	
....0. = Congestion window reduced (cwr): Not set	
....0. = ECN-Echo: Not set	
....0. = Urgent: Not set	
....0. = Acknowledgement: Not set	
....0. = Push: Not set	
....0. = Reset: Not set	
....0. = SYN: Set	
....0. = FIN: Not set	
Window size: 8192	
Checksum: 0xe795 [validation disabled]	
Options: (12 bytes)	

Fundamentos de Redes de Comunicação - Cap 5 Transporte

18

18

TCP – Início de Sessão

Só depois do three-way-handshake é que o vosso pedido é enviado ao servidor (neste caso, protocolo HTTP).

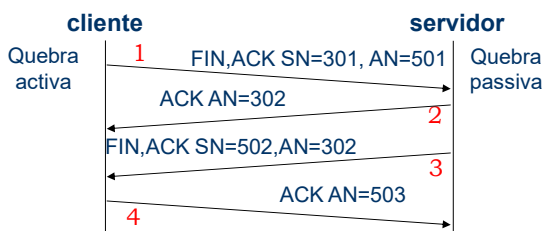
O conteúdo da página Web é enviado também em TCP.

No.	Time	Source	Destination	Protocol	Info
9	3.974318	192.168.2.4	62.28.183.139	TCP	49968 > http [ACK] Seq=1 Ack=1 win=16968 Le
10	3.974497	62.28.183.139	192.168.2.4	TCP	http > 49969 [SYN, ACK] Seq=0 Ack=1 win=819
11	3.974539	192.168.2.4	62.28.183.139	TCP	49969 > http [ACK] Seq=1 Ack=1 win=16968 Le
12	3.976071	192.168.2.4	62.28.183.139	HTTP	GET // HTTP/1.1
13	4.007435	62.28.183.139	192.168.2.4	HTTP	HTTP/1.1 200 OK (text/html)
14	4.049551	fe80::7cf8:a4c3:3b2ff02::c		SSDP	M-SEARCH * HTTP/1.1
15	4.169210	192.168.2.4	62.28.183.139	HTTP	GET /ats/PortalRender.aspx?PageID={f2d80173

Frame 12: 294 bytes on wire (2352 bits), 294 bytes captured (2352 bits)
Ethernet II, Src: Intelcor_3e:51:16 (00:1b:77:3e:51:16), Dst: Belkin_81:a0:20 (00:11:50:81:a0:20)
Internet Protocol, Src: 192.168.2.4 (192.168.2.4), Dst: 62.28.183.139 (62.28.183.139)
Transmission Control Protocol, Src Port: 49969 (49969), Dst Port: http (80), Seq: 1, Ack: 1, Len: 240
Hypertext Transfer Protocol
GET // HTTP/1.1\r\n
Accept: text/html, application/xhtml+xml, */*\r\n
Accept-Language: pt-PT\r\n
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0)\r\n
Accept-Encoding: gzip, deflate\r\n
Host: www.ats.pt\r\n
Connection: Keep-Alive\r\n
\r\n

TCP – Fecho de Sessão

Quando um sistema já não tem dados para enviar faz o half-close, enviando um segmento FIN:



Como a ligação é full-duplex, cada uma das partes necessita de fazer um half-close

Protocolo:

- 1- O cliente envia um segmento FIN com o seu número de sequência (SNcli) atual e com o respetivo número de confirmação
 - 2 -O servidor envia um segmento ACK com AN = SNcli+1 (302)
 - 3 -O servidor envia um segmento FIN com o seu número de sequência (SNSrv) atual e com o respetivo número de confirmação (SN=502, AN = 302)
 - 4 - O cliente envia um segmento ACK com AN = SNSrv+1 (503)
- Entre os pontos 2 e 3, o servidor ainda pode enviar dados.

TCP – Fecho de Sessão

Exemplo de um serviço que usa TCP: HTTP

Depois da página Web ser enviada e recebida (às vezes são milhares de pacotes...) o servidor pede o fecho da sessão, o que significa que não tem mais nada a enviar:

No.	Time	Source	Destination	Protocol	Info
60	2.016290	62.28.183.139	192.168.2.4	HTTP	HTTP/1.1 304 Not Modified
61	2.016721	62.28.183.139	192.168.2.4	TCP	http > 50062 [FIN, ACK] Seq=17933 Ack=1416
62	2.016788	192.168.2.4	62.28.183.139	TCP	50062 > http [ACK] Seq=1416 Ack=17933

<p>Frame 61: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0</p> <p>Ethernet II, Src: Belkin_81:a0:20 (00:11:50:81:a0:20), Dst: IntelCor_3e:51:16 (00:1b:77:3e:51:16)</p> <p>Internet Protocol, Src: 62.28.183.139 (62.28.183.139), Dst: 192.168.2.4 (192.168.2.4)</p> <p>Transmission Control Protocol, Src Port: http (80), Dst Port: 50062 (50062), Seq: 17933, Ack: 1416, Len: 0</p> <p>Source port: http (80)</p> <p>Destination port: 50062 (50062)</p> <p>[Stream index: 1]</p> <p>Sequence number: 17933 (relative sequence number)</p> <p>Acknowledgement number: 1416 (relative ack number)</p> <p>Header length: 20 bytes</p> <p>Flags: 0x11 (FIN, ACK)</p> <p>000. = Reserved: Not set</p> <p>...0 = Nonce: Not set</p> <p>....0. = Congestion window Reduced (CWR): Not set</p> <p>....0. = ECN-Echo: Not set</p> <p>....0. = Urgent: Not set</p> <p>....1. = Acknowledgement: Set</p> <p>....0. = Push: Not set</p> <p>....0. = Reset: Not set</p> <p>....0. = Syn: Not set</p> <p>....1. = Fin: set</p> <p>Window size: 259</p> <p>Checksum: 0x3b18 [validation disabled]</p>

TCP – Fiabilidade

Por cada pacote TCP enviado, o recetor tem de confirmar a sua chegada através de um ACK!

Desta forma, o emissor sabe quais os pacotes que foram entregues.

No.	Time	Source	Destination	Protocol	Info
14	1.817047	62.28.183.139	192.168.2.4	TCP	[TCP segment of a reassembled PDU]
15	1.817413	62.28.183.139	192.168.2.4	TCP	[TCP segment of a reassembled PDU]
16	1.817454	192.168.2.4	62.28.183.139	TCP	50062 > http [ACK] Seq=595 Ack=12727 Win=1768
17	1.817997	62.28.183.139	192.168.2.4	TCP	[TCP segment of a reassembled PDU]
18	1.820780	192.168.2.4	62.28.183.139	TCP	50062 > http [ACK] Seq=595 Ack=14141 Win=2474
19	1.828596	192.168.2.4	62.28.183.139	HTTP	GET /ats/webtemplates/templateATS/stvVies/ver...

<p>Frame 18: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0</p> <p>Ethernet II, Src: IntelCor_3e:51:16 (00:1b:77:3e:51:16), Dst: Belkin_81:a0:20 (00:11:50:81:a0:20)</p> <p>Internet Protocol, Src: 192.168.2.4 (192.168.2.4), Dst: 62.28.183.139 (62.28.183.139)</p> <p>Transmission Control Protocol, Src Port: 50062 (50062), Dst Port: http (80), Seq: 595, Ack: 14141, Len: 0</p> <p>Source port: 50062 (50062)</p> <p>Destination port: http (80)</p> <p>[Stream index: 1]</p> <p>Sequence number: 595 (relative sequence number)</p> <p>Acknowledgement number: 14141 (relative ack number)</p> <p>Header length: 20 bytes</p> <p>Flags: 0x10 (ACK)</p> <p>Window size: 2474</p> <p>Checksum: 0x4477 [validation disabled]</p> <p>[Seq/ACK analysis]</p> <p>[This is an ACK to the segment in frame: 17]</p> <p>[The RTT to ACK the segment was: 0.002783000 seconds]</p>
--

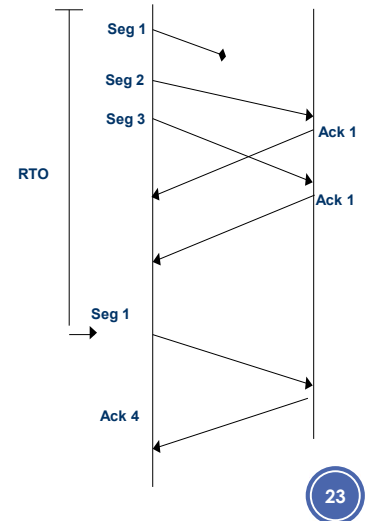
TCP – Controlo de Fluxo

O TCP coloca um **número de sequência (SN)** em todos os pacotes: permite ao recetor saber a ordem pelo qual deverão ser entregues à aplicação de destino.

O recetor tem de confirmar a chegada de cada pacote através de uma mensagem ACK.

No emissor, é ativado um **RTO - Retransmission Timeout** para cada segmento enviado.

Se não existir confirmação da receção (ACK) no período de tempo RTO, o emissor assume que o segmento se perdeu e é retransmitido (ver imagem).



Fundamentos de Redes de Comunicação - Cap 5 Transporte

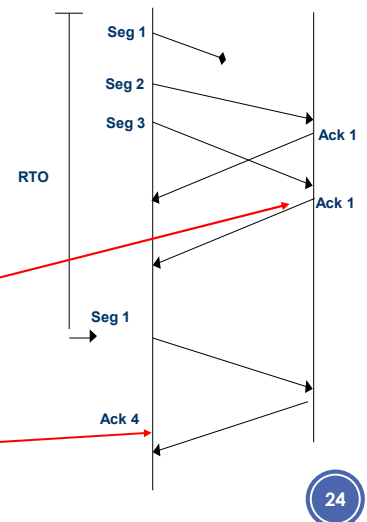
23

23

TCP – Controlo de Fluxo

Os Segmentos podem chegar fora de ordem pois são transportados por pacotes IP e seguir caminhos diferentes ou mesmo serem descartados pelos routers.

- Os ACK de confirmação de receção de segmento apontam para o primeiro segmento em falta (Imagem: recetor envia Ack1 mesmo quando o 2 e 3 já chegaram pois ainda não recebeu o 1). A receção de confirmações duplicadas indicia a ocorrência de problemas na transmissão.
- Um segmento de confirmação valida a correta receção de todos os segmentos enviados anteriormente (Imagem: o ACK 4 confirma que o 2 e 3 já foram recebidos).



Fundamentos de Redes de Comunicação - Cap 5 Transporte

24

24

TCP – Controlo de congestionamento

Objetivos do **controlo de congestionamento**:

- Maximizar o débito
- Adaptação dinâmica às condições de congestionamento
- Aumento da eficiência

O TCP utiliza o tempo que decorre entre o envio de um segmento e a receção do respetivo ACK para perceber se a rede está livre ou congestionada.

O recetor também pode enviar mensagens TCP ao emissor a informar que está sem capacidade de receber mais segmentos (**Window Size = Zero**).

Quando o tempo fica muito elevado, o TCP reduz o ritmo de envio de segmentos.

Este processo é contínuo, ajustando-se durante toda a sessão.

Fundamentos de Redes de Comunicação - Cap 5 Transporte

25

25

TCP vs UDP - Aplicações

Exemplos de **aplicações que usam UDP**: VoIP, Streaming de Vídeo ou de áudio, P2P e DNS

Preferencialmente as que têm mecanismos próprios de controlo da perda de pacotes e que privilegiam uma rede rápida, mesmo que com perda de alguns pacotes, ou seja, precisam de toda a velocidade que a rede possa dar (p.e. o vídeo em IP utiliza MPEG que disfarça as perdas de pacotes).

Também as que necessitam de transmissão para múltiplos recetores, tanto em multicast ou broadcast.

Fundamentos de Redes de Comunicação - Cap 5 Transporte

26

26

TCP vs UDP - Aplicações

Exemplos de **aplicações que utilizam o TCP**: E-mail, Transmissão de ficheiros de forma segura (FTP) ou Acesso remoto a máquinas (SSH ou telnet)

As que necessitam de um transporte de pacotes fiável e garantido mesmo que à custa de velocidade de transmissão.

Quando projetamos uma aplicação em IP temos de tomar uma **decisão acerca do protocolo de transporte**:

- Ou construímos uma aplicação que seja por si só capaz de detetar e colmatar as falhas da rede – usamos UDP
- Ou construímos uma aplicação sem esses mecanismos mas que depende da fiabilidade da rede para garantir a entrega de toda a informação mesmo que tenha de esperar por ela – usamos TCP

Fundamentos de Redes de Comunicação - Cap 5 Transporte

27

27

Protocolo QUIC

Evolução recente nas funções de controlo de sessão. Desenvolvido pela Google para reduzir a latência nas transmissões de média.

- Suportado em UDP para otimizar a velocidade: não há handshake, por exemplo
- Complementa o UDP com mecanismos avançados de deteção e recuperação de perdas de pacotes (em vez de recorrer ao TCP)
- Mantém a sessão quando se alterna de rede (por exemplo, quando se muda de uma rede wifi para outra)
- Melhor segurança: criptografia integrada
- Outras vantagens:
 - Melhor desempenho na consulta a páginas web HTTP.
 - Maior eficiência para serviços como streaming e jogos online.

A maior parte das páginas Web passou a utilizar este protocolo.

Fundamentos de Redes de Comunicação - Cap 5 Transporte

28

28