



Universidade do Minho
Escola de Engenharia

Comunicações por Computador

Trabalho Prático 1

Luís Pedro Oliveira de Castro Vieira A89601

José Pedro de Castro Ferreira A89572

Luís Enes Sousa A89597



A89601



A89572



A89597

Conteúdo

1	Protocolos da Camada de Transporte	1
1.1	Pergunta 1	1
1.2	Pergunta 2	3
1.2.1	FTP	3
1.2.2	TFTP	4
1.3	Pergunta 3	5
1.3.1	Uso da camada de transporte	5
1.3.2	Eficiência na transferência	5
1.3.3	Complexidade	5
1.3.4	Segurança	6
1.4	Pergunta 4	7
2	Conclusão	9

1 Protocolos da Camada de Transporte

1.1 Pergunta 1

Inclua no relatório uma tabela em que identifique, para cada comando executado, qual o protocolo de aplicação, o protocolo de transporte, porta de atendimento e overhead de transporte, como ilustrado no exemplo seguinte:

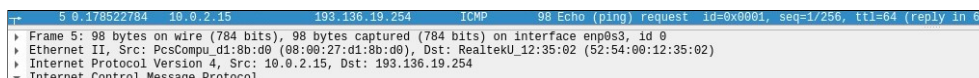
	Protocolo de Aplicação	Protocolo de Transporte	Porta de Atendimento	Overhead de Transporte
Ping	PING	-	-	-
tracert	TRACEROUTE	UDP	33434	8
telnet	TELNET	TCP	23	40
ftp	FTP	TCP	21	20
Tftp	TFTP	UDP	69	8
browser/http	HTTP	TCP	80	20
nslookup	DNS	UDP	53	8
ssh	SSH	TCP	22	40

O comando *ping* não tem um protocolo de transporte correspondente, pois este executa no nível 3 da camada protocolar (Network Layer).

Para preencher a coluna do protocolo de transporte, verificamos qual era o protocolo de nível 4 associado à trama correspondente.

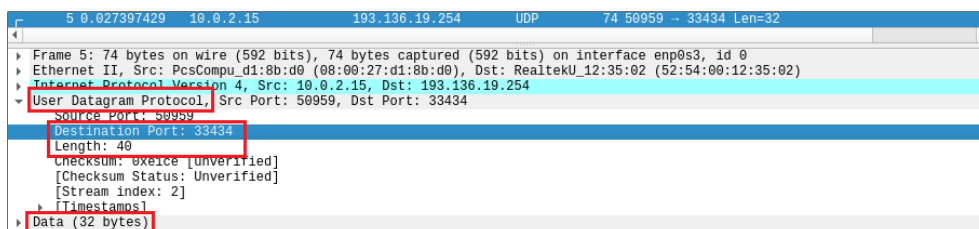
Em relação à coluna da porta de atendimento, esta foi preenchida analisando a porta de destino da trama enviada pelo host.

Quanto à coluna do overhead de transporte, foi preenchida de duas maneiras diferentes: uma para as tramas UDP e outra para as TCP. Como se pode ver na figura 2, o overhead duma trama UDP é calculado pela diferença entre *Total length* e *Data length* ($40 - 32 = 8$ bytes). Isto acontece em todas as tramas do protocolo UDP, logo o overhead de transporte de tramas UDP é sempre 8 bytes. Quanto às tramas TCP, o overhead de transporte está especificado no header.



5	0.178522784	10.0.2.15	193.136.19.254	ICMP	98 Echo (ping) request	id=0x0001, seq=1/256, ttl=64 (reply in 6)
---	-------------	-----------	----------------	------	------------------------	---

Figura 1: Trama ping



5	0.027397429	10.0.2.15	193.136.19.254	UDP	74 50959 → 33434 Len=32
---	-------------	-----------	----------------	-----	-------------------------

Figura 2: Trama traceroute

```

5 0.055547279 10.0.2.15 193.136.9.183 TCP 74 54406 → 23 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
Protocol: TCP (6)
Header checksum: 0xaf64 [validation disabled]
[Header checksum status: Unverified]
Source: 10.0.2.15
Destination: 193.136.9.183
Transmission Control Protocol, Src Port: 54406, Dst Port: 23, Seq: 0, Len: 0
Source Port: 54406
Destination Port: 23
[Stream index: 0]
[TCP Segment Len: 0]
Sequence number: 0 (relative sequence number)
Sequence number (raw): 1791519585
[Next sequence number: 1 (relative sequence number)]
Acknowledgment number: 0
Acknowledgment number (raw): 0
1010 .... = Header Length: 40 bytes (10)
Flags: 0x002 (SYN)

```

Figura 3: Trama telnet

```

5 0.121507261 10.0.2.15 193.136.9.183 TCP 54 48074 → 21 [ACK] Seq=1 Ack=1 Win=64240 Len=0
Frame 5: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface enp0s3, id 0
Ethernet II, Src: PcsCompu_d1:8b:d0 (08:00:27:d1:8b:d0), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 193.136.9.183
Transmission Control Protocol, Src Port: 48074, Dst Port: 21, Seq: 1, Ack: 1, Len: 0
Source Port: 48074
Destination Port: 21
[Stream index: 0]
[TCP Segment Len: 0]
Sequence number: 1 (relative sequence number)
Sequence number (raw): 3894540181
[Next sequence number: 1 (relative sequence number)]
Acknowledgment number: 1 (relative ack number)
Acknowledgment number (raw): 7104002
0101 .... = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)

```

Figura 4: Trama ftp

```

5 0.059799613 10.0.2.15 193.136.9.183 TFTP 86 Read Request, File: file1, Transfer type: octet, tsize
Frame 5: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface enp0s3, id 0
Ethernet II, Src: PcsCompu_d1:8b:d0 (08:00:27:d1:8b:d0), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 193.136.9.183
User Datagram Protocol, Src Port: 36867, Dst Port: 69
Source Port: 36867
Destination Port: 69
Length: 52

```

Figura 5: Trama Tftp

```

22 3.566390795 10.0.2.15 193.136.9.240 HTTP 542 GET /disciplinas/CC-MIEI/ HTTP/1.1
Transmission Control Protocol, Src Port: 35546, Dst Port: 80, Seq: 1, Ack: 1, Len: 488
Source Port: 35546
Destination Port: 80
[Stream index: 1]
[TCP Segment Len: 488]
Sequence number: 1 (relative sequence number)
Sequence number (raw): 543583041
[Next sequence number: 489 (relative sequence number)]
Acknowledgment number: 1 (relative ack number)
Acknowledgment number (raw): 8256002
0101 .... = Header Length: 20 bytes (5)
Flags: 0x010 (PSH, ACK)

```

Figura 6: Trama browser/http

```

1 0.000000000 10.0.2.15 192.168.1.1 DNS 84 Standard query 0xe72f AAAA www.uminho.pt OPT
Ethernet II, Src: PcsCompu_d1:8b:d0 (08:00:27:d1:8b:d0), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 192.168.1.1
User Datagram Protocol, Src Port: 34138, Dst Port: 53
Source Port: 34138
Destination Port: 53
Length: 50

```

Figura 7: Trama nslookup

```

3 0.004649228 10.0.2.15 193.136.9.183 TCP 74 56844 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
Frame 3: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface enp0s3, id 0
Ethernet II, Src: PcsCompu_d1:8b:d0 (08:00:27:d1:8b:d0), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 193.136.9.183
Transmission Control Protocol, Src Port: 56844, Dst Port: 22, Seq: 0, Len: 0
Source Port: 56844
Destination Port: 22
[Stream index: 0]
[TCP Segment Len: 0]
Sequence number: 0 (relative sequence number)
Sequence number (raw): 1070962742
[Next sequence number: 1 (relative sequence number)]
Acknowledgment number: 0
Acknowledgment number (raw): 0
1010 .... = Header Length: 40 bytes (10)
Flags: 0x002 (SYN)

```

Figura 8: Trama ssh

1.2 Pergunta 2

Uma representação num diagrama temporal das transferências da file1 por FTP e TFTP respetivamente. Se for caso disso, identifique as fases de estabelecimento de conexão, transferência de dados e fim de conexão. Identifica também claramente os tipos de segmentos trocados e os números de sequência usados nos dados como nas confirmações.

1.2.1 FTP

O protocolo FTP, *File Transfer Protocol*, pode ser executado em dois modos: passivo e ativo, dependendo de quem começa a conexão entre o cliente e o servidor.

-> FTP Ativo

Se a conexão for inicializada por parte do servidor, esta será ativa, ou seja, tanto o servidor como o cliente necessitarão de disponibilizar e abrir portas para o tráfego circular.

-> FTP Passivo

Por outro lado, se a conexão for inicializada por parte do cliente, esta será passiva, isto é, é indispensável que o servidor abra portas de forma a receber tráfego, mas o mesmo não é necessário para o cliente.

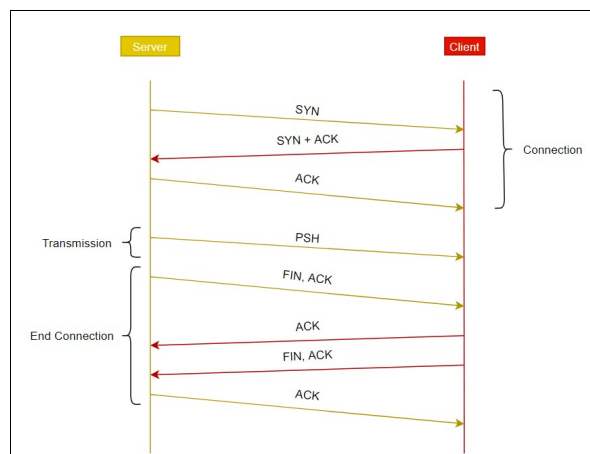


Figura 9: Diagrama Temporal de uma Conexão FTP

O protocolo FTP faz uso de um par de conexões entre o cliente e o servidor, sendo que a primeira conexão a ser estabelecida é com a porta 20 do servidor e a segunda com a porta 21.

Os servidores, ao abrirem a porta 21, começam à escuta por conexões de entrada do cliente, os quais se ligam a esta para iniciar as operações de transferência de ficheiros. No entanto, e algo importante a notar, é a necessidade extrema e estrita da porta 20, a qual possibilita efetivamente estas operações.

Ao analisarmos a figura 1 afirmamos, com certeza, que a conexão se inicia com o servidor a enviar um segmento SYN, indicando que os números de sequência se devem sincronizar de forma a iniciar a conexão. De seguida, o cliente responde com um segmento SYN e uma trama ACK, o qual permite inferir que houve sucesso na receção do segmento SYN previamente enviado pelo servidor.

Da mesma forma que o cliente respondeu ao servidor a afirmar que recebeu com sucesso o segmento SYN, este último efetua o mesmo procedimento enviando uma trama ACK ao cliente.

Desta forma, e estando estabelecida a conexão, o servidor envia para o cliente os dados que são pretendidos, iniciando-se a fase de transmissão.

Posteriormente, e de forma a finalizar a conexão, inicia-se uma nova troca de segmentos e tramas entre o cliente e o servidor. Neste caso, o servidor começa por enviar um segmento FIN ao cliente a informar que terminou a transmissão de dados, ao qual o cliente responde, tal como na fase da conexão, com um segmento FIN e uma trama ACK a informar que recebeu com sucesso o segmento anterior. Por fim, o servidor envia uma trama ACK a confirmar o sucesso na receção do segmento FIN anterior, finalizando, assim, a conexão entre ambos.

1.2.2 TFTP

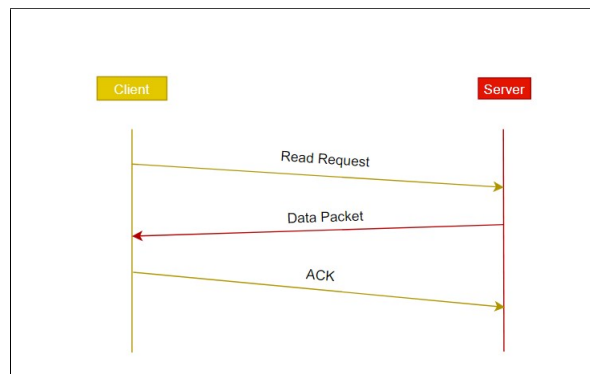


Figura 10: Diagrama Temporal de uma Conexão TFTP

Como podemos confirmar pela figura acima, o cliente começa por enviar um *Read Request* ao servidor, que, posteriormente, irá enviar um pacote de dados e, por fim, o cliente responde com uma trama ACK a informar o sucesso da receção desses dados.

Neste protocolo, todas as transferências começam com um pedido de leitura ou com um pedido de escrita de um ficheiro e, ao mesmo tempo, uma requisição de conexão por parte do cliente. Se o servidor atender ao pedido, a conexão é estabelecida e o ficheiro é enviado em blocos de comprimento fixo. Caso o comprimento do pacote seja superior ao limite, dá-se fragmentação do pacote e são enviados em vários pacotes, com o mesmo comprimento.

1.3 Pergunta 3

Com base nas experiências realizadas, distinga e compare sucinatamente as quatro aplicações de transferência de ficheiros que usou nos seguintes pontos (i) uso da camada de transporte; (ii) eficiência na transferência; (iii) complexidade; (iv) segurança;

1.3.1 Uso da camada de transporte

1. **SFTP:**
Usufrui do protocolo TCP
2. **FTP:**
Usufrui do protocolo TCP
3. **TFTP:**
Usufrui do protocolo UDP
4. **HTTP:**
Usufrui do protocolo TCP

1.3.2 Eficiência na transferência

1. **SFTP:**
Similar ao protocolo FTP, mas utiliza dados encriptados usufruindo do SSH para tal efeito.
2. **FTP:**
Como vimos previamente na questão 2, o protocolo FTP garante, por usufruir do protocolo TCP e através da troca de **ACK**nowledgments, que os segmentos serão efetivamente transmitidos. Isto, porém, peca na eficiência uma vez que é necessário esperar pelo ACK para poder continuar. Comparativamente ao SFTP, por não fazer uso da encriptação, torna possível a interceção dos segmentos por parte de terceiros.
3. **TFTP:**
Desde início, por usufruir do protocolo UDP, torna-se menos viável, pois não utiliza ACKs para confirmar a receção bem sucedida dos pacotes. Como não garante a entrega dos mesmos, é necessário que haja a retransmissão destes em caso de falha. No entanto, quando bem sucedido, é mais rápido que o protocolo FTP.
4. **HTTP:**
Este protocolo permite que, numa só ligação TCP, sejam enviados múltiplos HTTP requests sem ser necessário esperar pelas respetivas respostas.

1.3.3 Complexidade

1. **SFTP:**
Por ser um protocolo bastante fiável, que permite acesso, transferência e gestão de dados, o SFTP acaba por ser também deveras complexo, pois todas estas funcionalidades possuem elevados custos de processamento.

2. **FTP:**

É capaz de suportar diferentes pedidos para transferência de dados em paralelo, no qual cada uma das transferências inicializa uma nova conexão, fazendo com que existam diferentes velocidades de transferência. Devido a este efeito de criação regular de novas conexões, o protocolo FTP releva-se, também, bastante complexo.

3. **TFTP:**

O protocolo TFTP é, tal como o nome indica, *Trivial*, ou seja, uma alternativa simplificada do protocolo FTP. Por consequente, o TFTP suporta um menor número de funcionalidades, o que, juntamente com o facto de se basear no protocolo UDP para transporte, acaba por afetar a complexidade deste protocolo em geral. Podemos, então, concluir que o TFTP não se trata de um protocolo complexo.

4. **HTTP:**

Este último protocolo é capaz de garantir confiança, escalabilidade e desacoplamento de sistemas e, para além disso, oferece um suporte para hipermedia para redes complexas ou redes diferentes e não confiáveis. Por isto, o protocolo HTTP acaba por se tornar complexo.

1.3.4 Segurança

1. **SFTP:**

Como mencionado previamente, o protocolo SFTP faz uso do SSH (*Secure Shell*) permitindo-nos, então, concluir que se trata de um protocolo efetivamente seguro. No entanto, é necessário entender que o SSH faz uso de uma arquitetura em camadas, nomeadamente de uma camada de transporte, uma de autenticação e uma de conexão. No que toca ao assunto abordado, a segurança, ao nível do transporte existe um auxílio por parte do protocolo TCP/IP e fornece encriptação, autenticação do servidor e proteção na íntegra dos dados. Já ao nível da camada de autenticação, esta é responsável por manipular a autenticação dos clientes.

2. **FTP:**

Por não fornecer encriptação de dados, o protocolo FTP tornou-se conhecido pelas suas diversas falhas na segurança. Como as transmissões não se encontram encriptadas, permite que qualquer pessoa seja capaz de efetuar uma captura de pacotes na rede, tendo acesso a diversas informações particulares, como nomes de utilizadores, palavras-passe, etc. Isto acaba por demonstrar que o protocolo é realmente muito pouco seguro.

3. **TFTP:**

Na utilização do protocolo TFTP não é necessário uma autenticação, o que torna, de imediato, evidente que este protocolo não é o mais seguro, visto não proteger os dados que estão a ser transferidos.

4. HTTP:

Na utilização do protocolo HTTP, toda a informação é representada em texto e não se encontra encriptada, logo, os dados poderão ser adulterados muito facilmente, o que o torna muito inseguro.

1.4 Pergunta 4

As características das ligações de rede têm uma enorme influência nos níveis de Transporte e de Aplicação. Discuta, relacionando a resposta com as experiências realizadas, as influências das situações de perda ou duplicação de pacotes IP no desempenho global de Aplicações fiáveis (se possível, relacionando com alguns dos mecanismos de transporte envolvidos).

A perda e duplicação de pacotes IP nos níveis de Transporte e de Aplicação são bastante comuns, tornando-se, cada vez mais, um desafio combater estas situações. Um exemplo desta situação é a LAN3, que se encontra na topologia de rede fornecida junto com o enunciado. Os protocolos de transporte estudados, **TCP** e **UDP**, procuram tentar resolver estes problemas. Como já referimos, o protocolo TCP deteta e corrige erros, pelo que, quando são encontrados pacotes com erros, ocorre uma tentativa de retransmissão do mesmo. No entanto, com esta retransmissão, acontece também uma redução no débito de transmissão e, consequentemente, as filas dos routers vão sendo esvaziadas. Posteriormente, o débito de transmissão é aumentado, sendo este um aumento global. Por outro lado, o protocolo UDP, apesar de detetar erros, não os corrige: os pacotes com erros são simplesmente descartados, uma vez que este protocolo não possui nenhum campo que tenha como função a correção de erros. Sendo assim, não é possível detetar perdas de pacotes. A questão de como resolver este problema é respondida com a dependência de um protocolo que se encontra acima do protocolo UDP, cuja função é identificar pacotes pelo seu ID ou número de sequência, conseguindo, deste modo, saber se todos os pacotes foram recebidos, ou até se algum se perdeu no processo.

```
150 39.771679475 10.3.3.3 10.1.1.1 TCP 82 [TCP Retransmission] 59726 -> 21 [PSH, ACK] Seq=289 Ack=1032 Win=64256 Len=10 TSval=526682392 TSecr=760615126
175 40.238389278 10.3.3.3 10.1.1.1 TCP 66 [TCP Out-Of-Order] 30891 -> 28 [FIN, ACK] Seq=1 Ack=2 Win=65280 Len=0 TSval=526683858 TSecr=760615591
```

Figura 11: Computador a fazer download usando o FTP

```
51 39.443332645 10.3.3.3 10.1.1.1 ICMP 74 Destination unreachable (Port unreachable)
52 39.443333490 10.3.3.3 10.1.1.1 ICMP 74 Destination unreachable (Port unreachable)
```

Figura 12: Computador a fazer download usando o TFTP

As figuras 11 e 12 mostram o download do ficheiro *file1.txt*, usando FTP e TFTP. Durante a captura, em ambas as situações, a maior parte dos envios foram bem sucedidos. No entanto, após várias tentativas, podemos apurar que, por vezes, eram encontrados erros. Quando usado o FTP (como podemos ver pela Figura 11) existiam segmentos ACK que acabavam repetidos, mas, mesmo assim, o ficheiro acabava por conseguir chegar ao seu destino final. Ao usar o TFTP, o ficheiro não chegava ao seu destino final e o host acabava por perder muito tempo até se aperceber que tal tinha acontecido. No entanto, nos

casos em que as transferências era feitas com sucesso, a transferência usando TFTP era mais rápida do que se fosse feita com o FTP.

Assim, podemos concluir que, caso o ficheiro a transferir seja pequeno ou a velocidade seja a prioridade, o TFTP deverá ser a aplicação a utilizar. Caso a prioridade seja garantir que o ficheiro chega ao destino (grandes quantidades de dados) devemos usar o FTP, uma vez que assegura a transferência de todos os dados, ainda que seja um pouco mais demorado.

2 Conclusão

Terminada a realização deste trabalho prático, conseguimos aprofundar e ver em prática os nossos conhecimentos acerca dos diversos Protocolos da Camada de Transporte. Para além destes, foram ainda objetos de estudo múltiplos Protocolos da Camada de Aplicação. Entre os temas explorados encontram-se os seus comportamentos, assim como as vantagens e desvantagens da utilização dos mesmos. Com o intuito de consolidar estes conhecimentos, e com o auxílio da ferramenta *Core* e *Wireshark*, foram realizadas algumas demonstrações, possibilitando a visualização de variados procedimentos e processos de transferência de dados, em diferentes protocolos.

Posto isto, podemos concluir que, se quisermos realizar uma transferência de dados e garantir que todos os dados enviados são recebidos, temos de escolher uma aplicação que utilize o protocolo TCP, apesar deste não ser o protocolo mais eficiente, uma vez que deteta e corrige erros. Por outro lado, se a perda de pacotes não for um problema maior, devemos optar pelo protocolo UDP, aproveitando, deste modo, a velocidade máxima de transferência possível.

Assim, terminamos este trabalho prático afirmando entender melhor os temas abordados e descritos durante a realização do mesmo.