Protocolos da Camada de Transporte

Trabalho realizado por:

Bruno Filipe de Sousa Dias

Francisco Alves Andrade

Paulo Silva Sousa

A89513
Francisco Andrade

A89583
Bruno Dias

A89465
Paulo Sousa

Grupo 14
2020/2021
Universidade do Minho

Índice

1 Questões e Respostas

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.

De forma a poder responder com transparência a esta questão, foi construída uma tabela semelhante à exposta no enunciado. Para teste dos vários comandos foi utilizado o terminal e o Wireshark como programa de análise do tráfego de rede.

Comado usado	**Protocolo de Aplicação**	**Protocolo de Transporte**	**Porta de Atendimento**	**Overhead de transporte em bytes**
Ping	PING	-	-	-
Traceroute	TRACEROUTE	UDP	33446	8
Telnet	TELNET	TCP	23	20
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	20

Table 1: Pergunta 1

Para podermos completar da maneira mais correta a tabela, precisamos de perceber alguns aspetos cruciais. Um deles é o facto de que todos os protocolos que utilizem o protocolo UDP como protocolo de transporte possuierem um *overhead* constante, independentemente da circunstância, de exatamente 8 bytes. O campo TCP possui um *overhead* de normalmente 20 bytes, no entanto, existem casos em que este *overhead* pode ser maior, caso seja utilizado o campo *options* ou não (neste estudo não se verificou o mesmo em nenhum caso).

Para complementar as respostas dadas, seguem seguidamente imagens relativas ao tráfego captado no Wireshark que foi utilizado para obter algumas soluções.

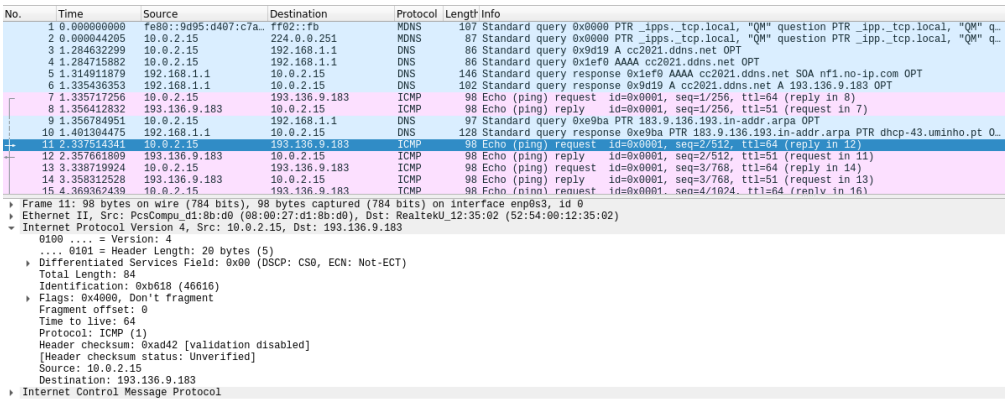

Figure 1: Captura Ping

O Ping trabalha diretamente com a camada de rede, e portanto não é aplicado qualquer protocolo de transporte não existindo dessa forma Porta de Atendimento nem Overhead de Transporte.

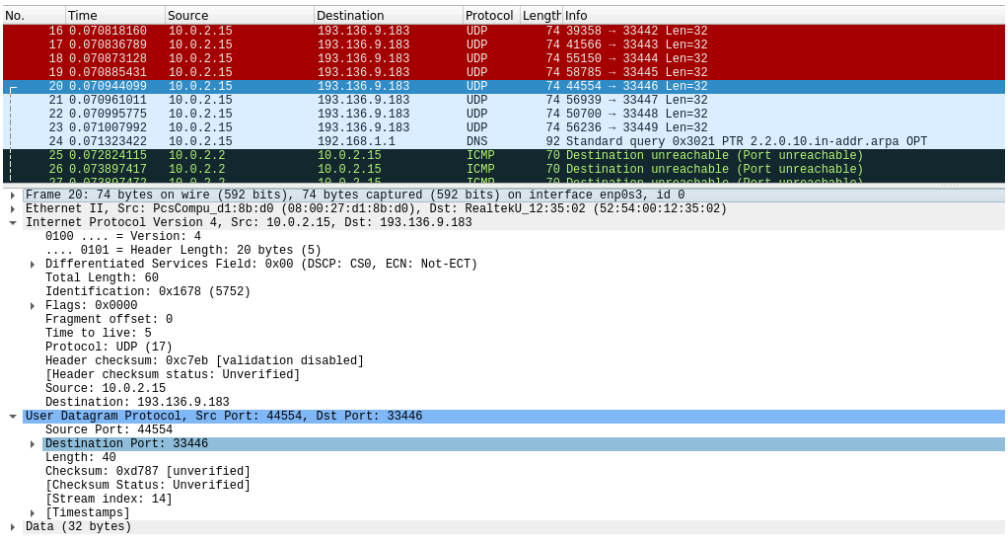

Figure 2: Captura Traceroute

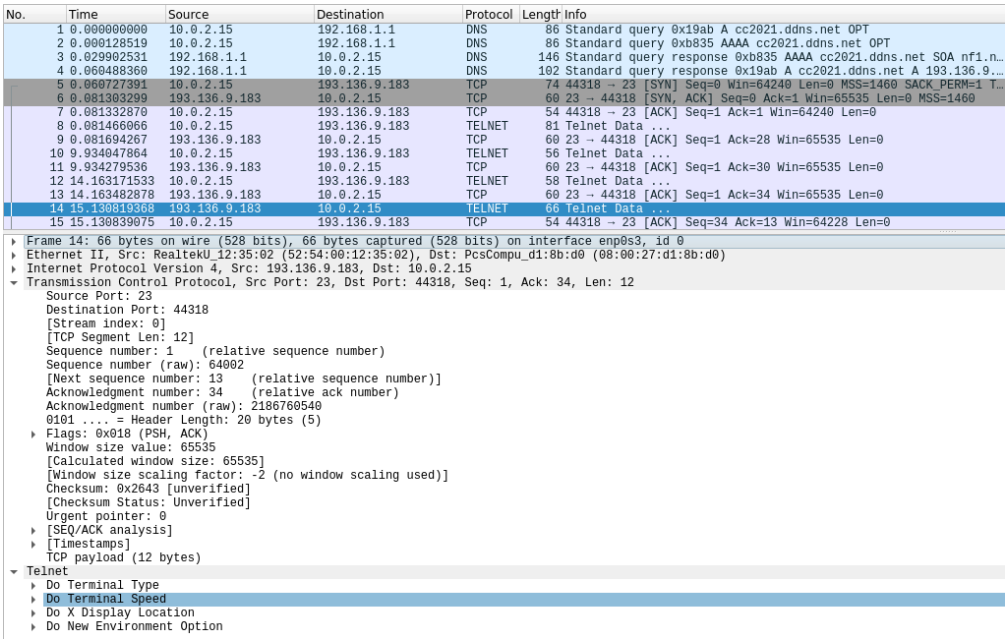

Figure 3: Captura Telnet

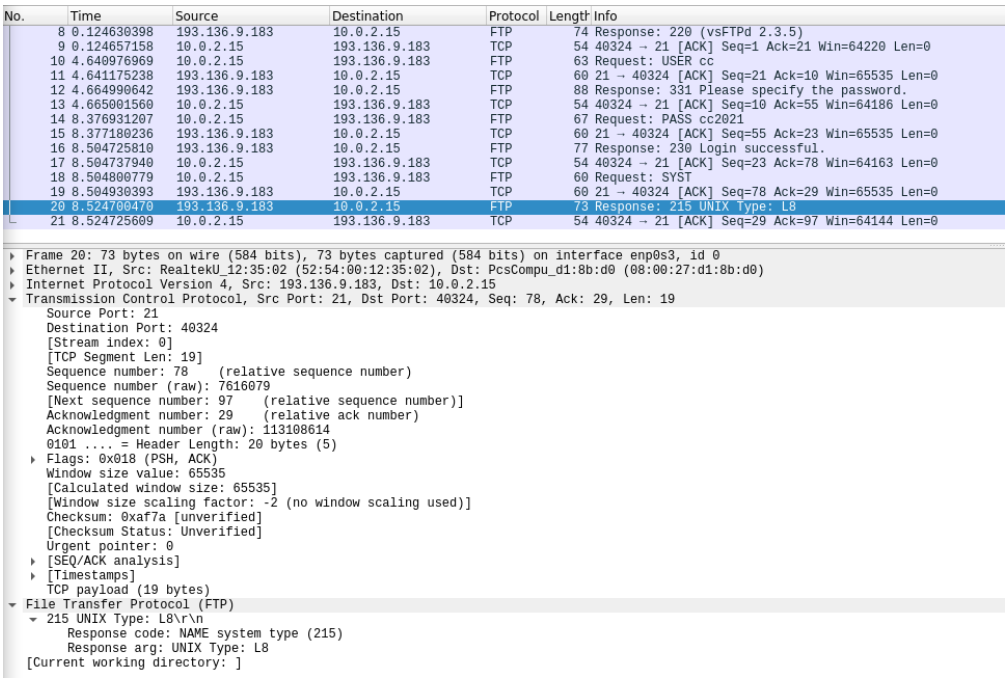

Figure 4: Captura FTP

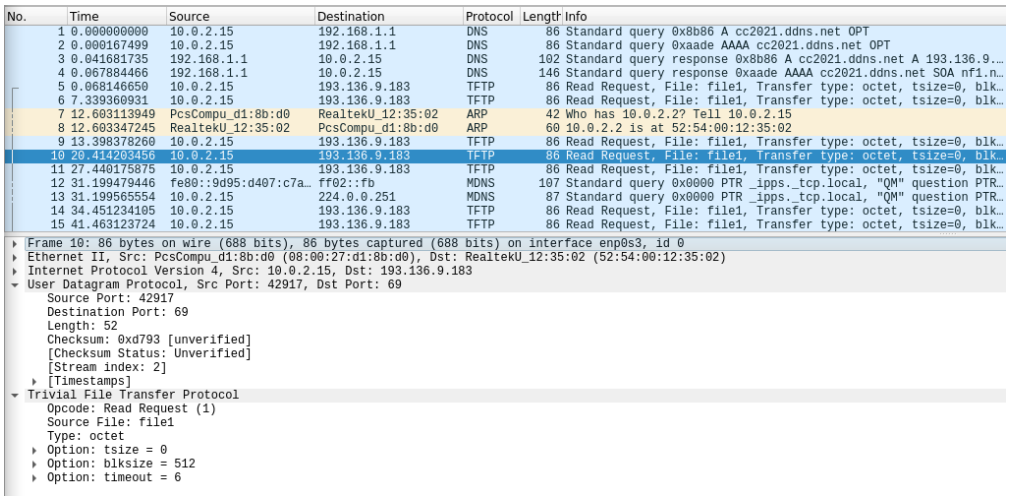

Figure 5: Captura TFTP

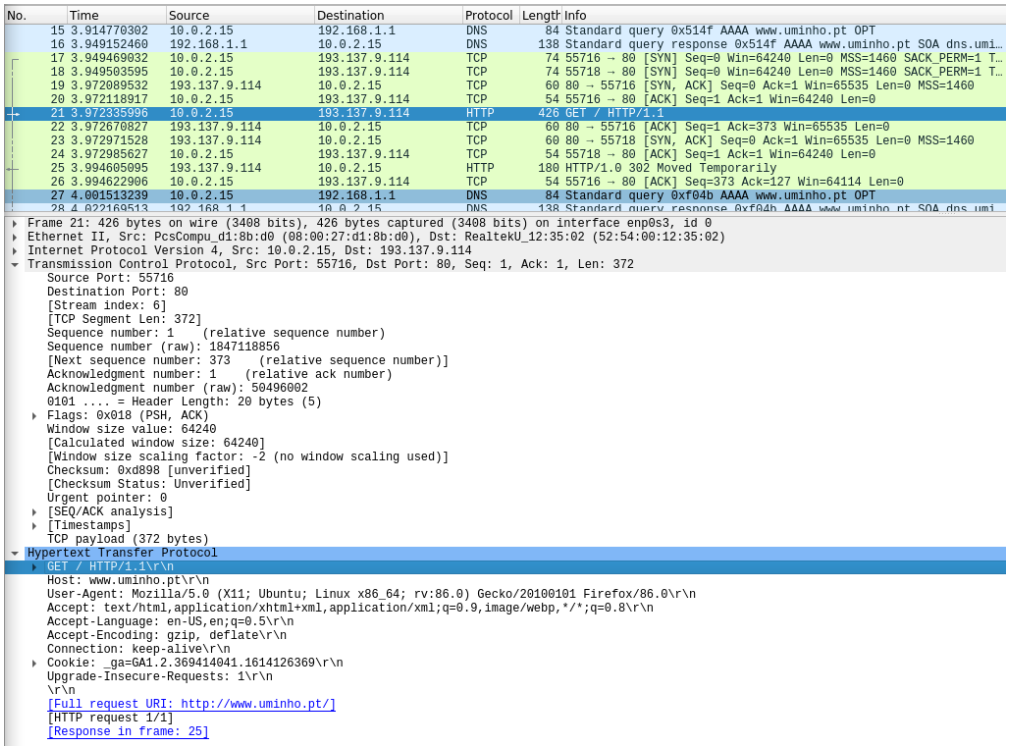

Figure 6: Captura Browser/HTTP

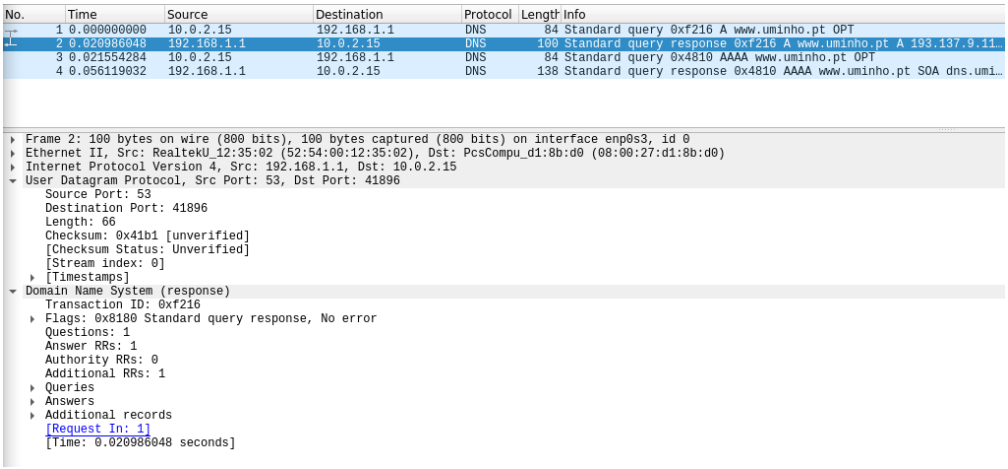

Figure 7: Captura Nslookup

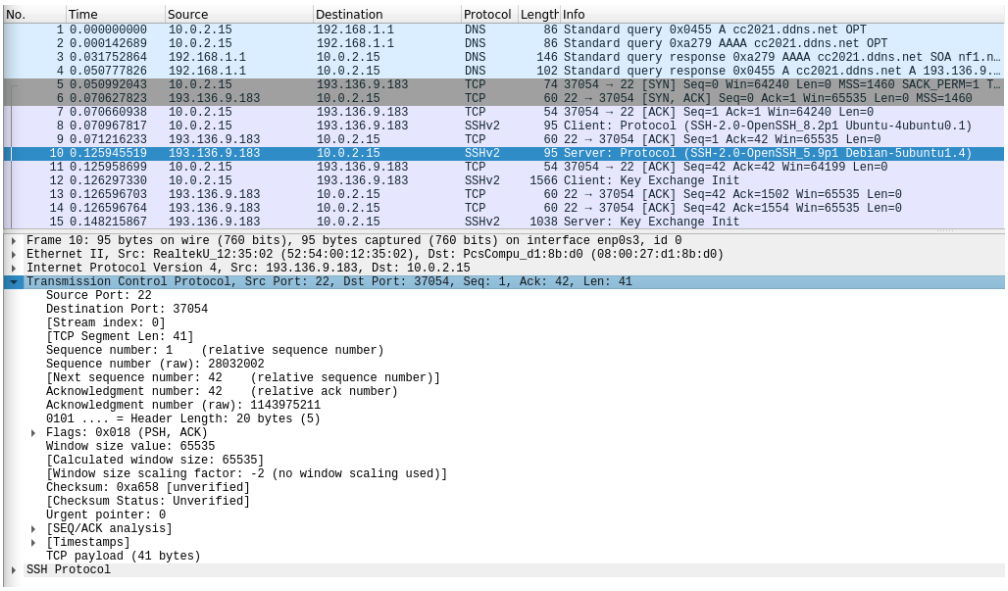

Figure 8: Captura SSH

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 quer nos dados como nas confirmações.

1.2.1 FTP

Um cliente realiza uma conexão TCP para a porta 21 do servidor. Essa conexão, chamada de conexão de controlo, permanece aberta ao longo da sessão enquanto uma segunda conexão, chamada conexão de dados, é estabelecida na porta 20 do servidor e em alguma porta do cliente, como requisitado para a transferência de arquivos. Esta última é extremamente importante no processo de realização da transferência dos ficheiros, apesar dos clientes se conectarem à porta 21 dos servidores remotos para iniciarem a opera

O protocolo FTP pode ser executado em dois modos distintos que determinam como é que a conexão de dados é estabelecida. No modo **Ativo**, o cliente envia para o servidor o endereço IP e o número da porta na qual ele irá ouvir e então o servidor inicia a conexão TCP.

Em situações onde o cliente se encontra atrás de uma *firewall* e inapto para aceitar entradas de conexão, é utilizado o modo **Passivo**. O cliente envia um comando PASV para o servidor e recebe um endereço IP e um número de porta como resposta, os quais o cliente utiliza para abrir a conexão de dados com o servidor.

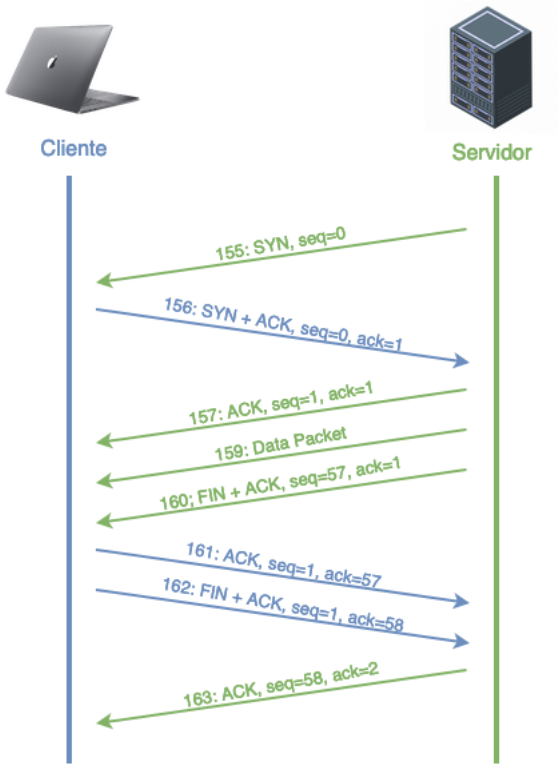

Figure 9: Diagrama FTP

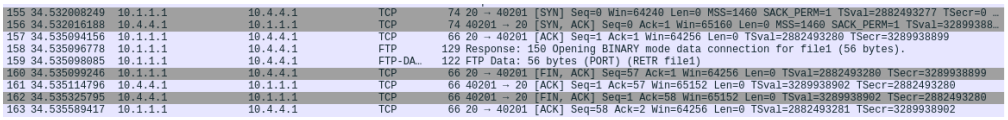

Figure 10: Captura FTP

1.2.2 TFTP

Neste protocolo transferência começa com um de dois pedidos possíveis, de leitura ou de escrita, num arquivo, pedido esse que serve também como solicitação de conexão. Consequentemente, o servidor pode conceder o pedido, e nesse caso a conexão estará aberta e o arquivo é emitido em blocos fixos de 512 bytes. Excecionalmente, se o pacote for superior aos 512 bytes de limite, este será fragmentado em pacotes de um tamaho máximo de 512 bytes, até completar a quantidade de dados a transmitir.

Tal como podemos ver na figura 11 do diagrama temporal, o Cliente inicia o processo com um *Read Request* ao servidor, ao qual este responde com um *Data Packet* com os dados de envio pretendidos. Para finalizar o processo, o Cliente envia um *Acknowledgement (ACK)*, relatando o sucesso da operação realizada, isto é, os dados terem sido recebidos sem problemas.

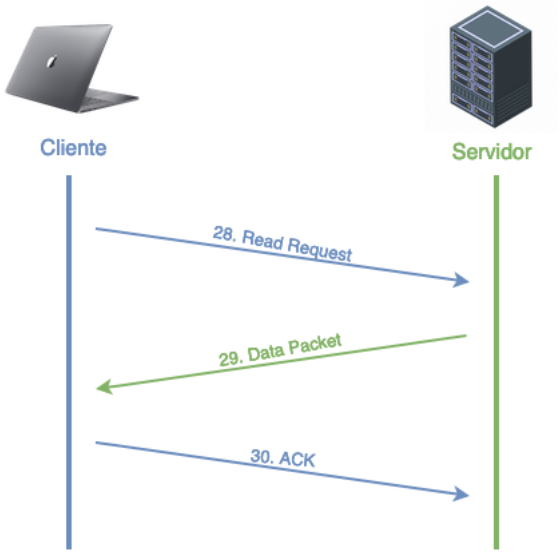

Figure 11: Diagrama TFTP

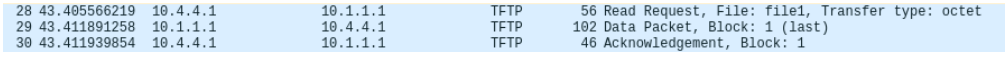

Figure 12: Captura TFTP

1.3 Pergunta 3

Com base nas experiências realizadas, distinga e compare sucintamente 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;

Para respondermos à seguinte pergunta, precisamos de saber as aplicações de transferência de ficheiros que pretendemos estudar. Estas são: **SFTP** (SSH File Transfer Protocol), **FTP** (File Transfer Protocol), **TFTP** (Trivial File Transfer Protocol) e **HTTP** (Hypertext Transfer Protocol).

1.3.1 Uso da camada de transporte

- **SFTP -** Protocolo utilizado é o TCP
- **FTP -** Protocolo utilizado é o TCP
- **TFTP -** Protocolo utilizado é o UDP
- **HTTP -** Protocolo utilizado é o TCP

1.3.2 Eficiência na transferência

- **SFTP -** Comporta-se da mesma maneira que o FTP, sendo entanto um pouco mais forte e ainda mais fiável, dado que os dados se encontram encriptados.
- **FTP -** Bastante fiável no processo de transferências de dados, no entanto, peca no seu custo de eficiência. Esta fiabilidade é adquirida através do uso de *Acknowleges* garantido a transmissão de um segmento de Data, e consequentemente os pacotes irão levar um maior overhead. No entanto, e mesmo por usarmos *Acknowleges*, este processo precisa dos mesmos para conseguir prosseguir corretamente. Os dados não são encontram encriptados, criando uma brecha para possíveis ataques e interceções.
- **TFTP -** Dado que é utilizado o protocolo UDP, o protocolo TFTP perde bastante da sua fiabilidade. Isto acontece, porque ao contrário do que acontece com o SFTP ou o FTP, o protocolo TFTP não faz qualquer uso de *Acknowleges* tornando-se assim impossível perceber e confirmar se um pacote é entregue com sucesso e sem qualquer erro. Como não possui *Acknowleges* os pacotes irão levar um menor overhead. Dado que este protocolo possui este comportamento, é então necessário enviar o mesmo pacote várias vezes, para existir uma certeza de que, pelo menos um pacote, irá chegar ao seu destino corretamente. Além disso, e pelo facto de não utilizar *Acknowleges*, perde fiabilidade, mas coincidentemente não necessita de haver tempos de espera por confirmações de pacotes que chegaram com sucesso e assim este pacote pode ganhar um pouco na sua velocidade, tornando-se mais rápido.

- **HTTP -** O protocolo HTTP permite que vários HTTP requests sejam enviados ao mesmo tempo, para uma mesma ligação TCP (como referido acima) sem ser necessário aguardar qualquer resposta. Isto acontece devido á utilização de pipelining, que para além de ter esse benefício, torna tudo muito mais rápido e mais eficiente.

1.3.3 Complexidade

- **SFTP -** Este protocolo possui uma grande complexidade, devido à vasta amplitude de funcionalidade que oferece. Estas funcionalidades são: acesso a dados, transferência de dados e gestão de dados. Todas esta funcionalidades possuem um grande custo ao nível de processamento e por isso tornam o protocolo em questão bastante complexo.

- **FTP -** A grande fiabilidade deste protocolo, referida anteriormente faz com que, tal como é esperado, este protocolo seja portador de uma grande complexidade. Além da fiabilidade do mesmo, este ainda possui a funcionalidade de tranferir dados em paralelo, havendo a necessidade de estabelecer novas conexões que podem diferir muito na sua taxa e velocidade de transferência de dados e pacotes. Tudo isto faz com que o protocolo seja bastante complexo.

- **TFTP -** Este protocolo passa por ser uma versão mais simplificada do protocolo FTP, sendo mesmo confirmado pelo seu nome Trivial FTP. Uma das diferenças é o facto de ser utilizado o protocolo UDP em substituição do protocolo TCP, o que nos mostra logo que é um processo muito menos complexo. Além disso, todo o protocolo possui um menor leque de funcionalidades, comparando ao protocolo FTP. Assim sendo, este protocolo revela-se pouco complexo.

- **HTTP -** O protocolo HTTP é um protocolo utilizado por sistemas de informação distribuídos e colaborativos, e é também ele um protocolo que se banha de várias (e complexas) funcionalidades. Este permite escalabilidade de sistemas, desacoplamento do mesmo, entre várias outras funcionalidades, o que faz com que este protocolo seja muito complexo.

1.3.4 Segurança

- **SFTP -** O protocolo SFTP é de todos os protocolos em estudo o que providência uma maior segurança. Este utiliza uma arquitetura em camadas, que juntamente fornecem encriptação e proteção da integridade de dados, autenticação do servidor e autenticação e manipulação dos clientes. Assim, podemos perceber o porquê do protocolo fornecer uma segurança tão grande.

- **FTP -** Este protocolo, fornece mecanismos de autenticação, no entanto, é extremamente inseguro. Isto acontece principalmente por não haver encriptação de dados. Dessa maneira, qualquer pessoa capaz de captar os pacotes, quer sendo normalmente, quer sendo através de qualquer tipo de ataque, irá ter acesso a toda a informação, estando englobada aquela que não deveria poder ver, incluindo-se usernames, passwords, detalhes de uma conta bancário, entre outros. Assim, revela uma grande precariedade quanto á sua segurança.

- **TFTP -** Este protocolo não fornece qualquer tipo de mecanismos de autenticação, e para mais, não oferece também nenhum tipo de encriptação de daods. Assim, podemos facilmente concluir que este protocolo possui um nível de segurança muito fraco.

- **HTTP -** Este protocolo, apesar de usufruir e de recorrer à autenticação, não possui mais uma vez, mecanismos de encriptação de dados (dado que se processa maioritariamente para realizar transferências de dados na Internet, estando esta representada sobre a forma de texto). Assim, existe a possibilidade de existirem ataques, que permitam a vizualização de dados confidenciais, ou até mesmo a alteração dos mesmos. Dessa forma, este protocolo, confirma-se como sendo pouco seguro.

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).

No que toca aos níveis de Transporte e de Aplicação existe muitas das vezes perda e/ou duplicação de pacotes. Os protocolos TCP e UDP tentam então combater estes problemas, recorrendo a algumas soluções (reagindo de maneiras diferentes). Estas situações acontecem por exemplo na LAN3 da Topologia fornecida para a realização deste projeto (à qual vamos recorrer para responder a esta questão).

Para poder construir uma resposta sólida a esta perquisição, é primeiramente necessário reforçar as funcionalidades e caraterísticas, tanto do protocolo TCP, como do protocolo UDP, relembrando conceitos acima referidos (ou estudados noutros anos):

- **TCP -** O Protocolo TCP realiza deteção e correção de erros, tentando retransmitir um mesmo pacote caso este não chegue com sucesso e sem anomalias ao seu destino, garantido que todos os pacotes são entregues. Dado que existe todo este mecanismo, quando ocorrem retransmissão de pacotes, o débito de transmissão diminui, dado que estamos a reexpedir um pacote já anteriormente enviado. FTP trabalha à base deste protocolo de transporte.

- **UDP -** O Protocolo UDP realiza também deteção de erros, no entanto, não possui qualquer funcionalidade de correção dos mesmos, e assim, se algum pacote possuir alguma anomalia este protocolo apenas o descarta, dado que não possui mecanismos de recuperação ou correção. Isto faz com que não seja possível reconhecer qualquer perda de pacotes. Na eventualidade de se visar a correção destes erros, ficamos inteiramente dependentes de protocolos que se encontram acima do protocolo UDP. Estes fazem uso do número de seqência do pacote, por exemplo, de forma a verificar se a totalidade dos pacotes foi recebida e se esta não possui qualquer irregularidade. TFTP trabalha à base deste protocolo de transporte.

Ficando compreendidos estes conceitos, são testadas então as transferências de um ficheiro, através de FTP e de TFTP, utilizando Sistemas da LAN3, de modo a compreender o que se sucede na eventualidade de acontecer um erro.

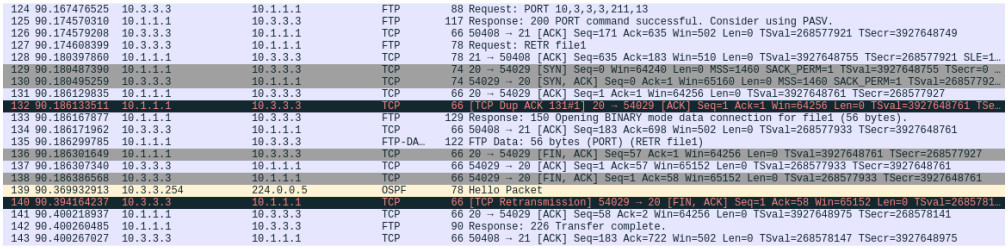

Figure 13: Sistema Corvo faz download do file1 através de FTP

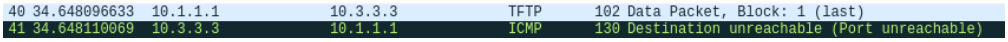

Figure 14: Sistema Corvo faz download do file1 através de TFTP

As figuras apresentadas acima, exibem o tráfego de rede capturado pelo Wireshark em momentos em que foram feitos downloads do file1, numa através de FTP e na outra através de TFTP. Em ambos os casos é utilizado o Sistema Corvo, da LAN3, fazendo download do file1 através do Serevr1. Em ambos os casos aconteceram erros (objeto de estudo), no entanto, é importante realçar que estes foram poucos e que na maioria das vezes todo o processo ocorria corretamente e com sucesso. Analisando então os casos em estudo:

- No caso da utilização do **FTP**, podemos ver na Figura 13 dois erros que acontecem algumas vezes. Por sorte conseguimos captar ambos os erros num mesmo "ciclo". O primeiro erro que podemos visualizar é o facto de existirem seguementos ACK repetidos, mas que não interfere no bom sucesso do processo, chegando o pacote ao seu destino final na mesma. O segundo erro é quando um pacote não chega ao seu destino final corretamente: como podemos ver, o Server1 envia um (FIN+ACK) ao sistema Corvo de modo a encerrar a conecção, no entanto o Corvo não responde com nenhum ACK, e então, depois do Server1 esperar um tempo e não receber um ACK, ele voltar a enviar o mesmo pacote, ficando assim explicada a retransmissão de pacotes. Esta complexidade e tentativa de deteção e correção de erros, faz com que a transmissão de pacotes se torne um pouco mais lenta.

- No caso da utilização do **TFTP**, podemos ver que os dados não conseguem chegar ao seu destino final corretamente, mas como não há qualquer deteção ou correção de erros, o próprio Host demoro demasiado tempo a constatar que o pacote enviado não foi entregue corretamente, perdendo alguma eficácia. Ao contrário do FTP e dado que não existem quaisquer mecanismos de correção de erros, faz com que a transmissão de pacotes se torne bastante mais rápida.

Assim, ficamos a perceber que, nos casos em que todo o processo ocorria de forma perfeita sem existir qualquer cenário de erro, o processo de transferência dos arquivos era mais rápido através de TFTP do que de FTP. No entanto, quando ocorriam erros, o TFTP perdia bastante tempo na perceção do mesmo e não conseguia recuperar nenhuma informação perdida, ao contrário do FTP que conseguia recuperar de quaisquer eventualidades de erros, dado que possui os devidos mecanismos de deteção e correção dos mesmos. Assim, ficamos comprometidos entre um processo mais rápido contudo sem garantias (TFTP), e um processo um pouco mais demorado, mas que nos garante que o ficheiro e os dados chegam ao seu destino corretamente (FTP). Cabe a nós decidir a rota a tomar, face a cada situação e à primazia que damos a cada caso em específico (velocidade ou certeza de que tudo é entregue corretamente).

2 Conclusões

De modo a elaborar este trabalho, utilizamos a máquina virtual disponibilizada pelos docentes para ter acesso às ferramentas utilizadas para o seu desenvolvimento: o Core e o Wireshark.

Após realizar este projeto conseguimos sedimentar bastantes conhecimentos sobre Protocolos da Camada de Transportes e Protocolos da Camada da Aplicação. Além disso, conseguimos explorar o comportamento destes protocolos e as suas vantagens e desvantagens, tendo em conta a sua aplicação.