Mestrado Integrado em Engenharia de Telecomunicações e Informática

**UC de Redes de Computadores I**

**LANs Ethernet e redes TCP/IP usando o CORE**

Ano Letivo 2021/2022



**UNIVERSIDADE DO MINHO**

ESCOLA DE ENGENHARIA

Eva Castro, a93097

Filipe Peixoto, a93096

José Gomes, a93083

Índice

[1. Introdução 3](#_Toc91586759)

[2. Emulação de LANs (*Local Area Networks*) *Ethernet* 4](#_Toc91586760)

[3. Interligação de redes 7](#_Toc91586761)

[4. DHCP 11](#_Toc91586762)

[5. Uso das camadas de rede e transporte por parte das aplicações 13](#_Toc91586763)

[6. Conclusão 17](#_Toc91586764)

[Figura 1 - Topologia em árvore 4](#_Toc91586767)

[Figura 2 - Wireshark n5 5](#_Toc91586768)

[Figura 3 - Wireshark n5 (segunda vez) 6](#_Toc91586769)

[Figura 4 - Wireshark n1 6](#_Toc91586770)

[Figura 5 - Interligação de redes 7](#_Toc91586771)

[Figura 6 - DefaultRoute n14 9](#_Toc91586772)

[Figura 7 - Interface eth2 configuration router1 (n1) 9](#_Toc91586773)

[Figura 8 - IP route router7 (n7) 10](#_Toc91586774)

[Figura 9 - Ping e traceroute n14-n31 10](#_Toc91586775)

[Figura 10 – Traceroute R1.1 para R1.2 10](#_Toc91586776)

[Figura 11 - Rede local R3 com DHCP 11](#_Toc91586777)

[Figura 12 - Configuração servidor 11](#_Toc91586778)

[Figura 13 - Captura de pacotes no servidor - DHCP 12](#_Toc91586779)

[Figura 14 - Topologia de rede com servidores HTTP e FTP 13](#_Toc91586780)

[Figura 15 - StartUp commands FTP e HTTP 14](#_Toc91586781)

[Figura 16 - Ligação ao servidor FTP através do um cliente da rede 1 14](https://uminho365-my.sharepoint.com/personal/a93097_uminho_pt/Documents/Universidade/3_ano/RC1/relatorio_rc1.docx#_Toc91586782)

[Figura 17 - WireShark FTP Server 15](#_Toc91586783)

[Figura 18 - Ligação ao servidor HTTP através do um cliente da rede 3 15](#_Toc91586784)

[Figura 19 - Wireshark HTTP server 16](#_Toc91586785)

[Tabela 1 - Esquema de endereçamento 7](#_Toc91286256)

[Tabela 2 - Esquema de encaminhamento 8](#_Toc91286257)

# Introdução

No âmbito da unidade curricular de Redes de Computadores I, foi-nos proposta a realização de um trabalho prático cujo objetivo é emular vários tipos de redes locais e interligá-las entre si, utilizando a ferramenta CORE. Esta ferramenta disponibiliza meios para desenhar diferentes tipologias de rede, configurar as ligações entre os componentes da rede, endereços e serviços.

Utilizaremos ainda, a ferramenta *Wireshark* para efetuarmos diagnósticos de conetividade e capturas de tráfego, para melhor entendimento do funcionamento dos componentes de cada rede e também comparar o modo de funcionamento das redes. Utilizaremos comandos como *ping* e *traceroute* para podermos executar os testes experimentais.

# Emulação de LANs (*Local Area Networks*) *Ethernet*

Neste primeiro exercício, desenhamos uma rede com tipologia em árvore, que possui 2 HUBs e 1 SWITCH, como se pode observar na figura 1. O objetivo é efetuar capturas de tráfego utilizando o *Wireshark*, de forma a compreender melhor o seu funcionamento, na prática, dos HUBs e SWITCHs e também dos protocolos ARP e ICMP.



Figura 1 - Topologia em árvore

Deste modo, antes de resolvermos o exercício tivemos de entender o funcionamento destas tecnologias.

O HUB, quando recebe informação por uma porta, transmite essa informação para todas as outras portas, este comportamento diminuiu a performance, pois qualquer outro sistema terminal, ao qual a informação não lhe é dirigida, pode ter acesso à mesma. Este tipo de equipamento não possui tecnologia que lhe permita armazenar informação sobre os sistemas terminais que se encontram conectados a ele.

O SWITCH possui um comportamento semelhante ao do HUB, no entanto possui tecnologia que lhe permite armazenar os endereços MAC (*Medium Access Control*) dos sistemas terminais a ele conectados (algo que ele vai “aprendendo” com o decorrer das ligações, na primeira ligação possui um comportamento igual ao do HUB). Esta funcionalidade dos SWITCHs permite-lhes comutar pacotes diretamente para a porta de destino.

O protocolo ARP (*Address Resolution Protocol*) faz a ponte de ligação entre a cama de rede e a camada de ligação de dados, portanto mapeia endereços IP (*Internet Protocol*) em endereços MAC. São enviados dois tipos de mensagens: ARP *request*, onde o dispositivo de origem pede a resolução de um endereço IP em endereço MAC e ARP *reply*, onde o dispositivo de destino envia o endereço físico resolvido. Para reduzir o fluxo e o número de solicitações de resolução de endereços, recorre-se ao uso de uma tabela, cache ARP, em cada *host* cuja funcionalidade é guardar os mapeamentos dos endereços por algum tempo.

O protocolo ICMP (*Internet Control Message Protocol*) é utilizado para enviar mensagens de diagnóstico e relatório de erros sobre a rede para a fonte do pacote. As mensagens podem ser *echo reply*, *echo request*, TTL (*Time To Live*) *expired*, etc. Qualquer dispositivo de rede pode enviar, gerar, receber e processar mensagens de erro ICMP. Este protocolo é utilizado por alguns diagnósticos de utilidade como o *ping* e o *traceroute*.

Agora estamos em condições de avaliar a conetividade entre os sistemas. Para tal, executamos, no PC n8, o comando *arp -a* para verificar as tabelas ARP dos sistemas terminais, de seguida executamos o comando *ping 10.0.0.25* e num primeiro momento, colocamos o sistema n5 à escuta e obtivemos os resultados que se encontram na figura 2. As capturas de tráfego foram efetuadas no *Wireshark*.

Uma imagem com mesa

Descrição gerada automaticamente

Figura 2 - Wireshark n5

Como podemos observar, antes de serem comutados pacotes entre os dois sistemas terminais, é enviado, em *broadcast*, um ARP *request* a perguntar qual o sistema terminal que possui o endereço IP 10.0.0.25, pois é a tentativa do protocolo ARP mapear o endereço IP no endereço MAC e vemos que depois é enviada um ARP *reply*, em *unicast*, com o endereço mapeado. Este sistema (n5) consegue captar todos os pacotes comutados entre o sistema n8 e n7 pois encontra-se ligado ao mesmo HUB que o sistema n7 (destino).

Voltamos a executar o comando *ping 10.0.0.25* no PC n8 e colocamos o PC n5 à escuta e o resultado obtido encontra-se na figura 3.

Uma imagem com mesa

Descrição gerada automaticamente

Figura 3 - Wireshark n5 (segunda vez)

Notamos que, quando ocorre a primeira comutação de pacotes, não é enviado um ARP *request*, isto porque o mapeamento dos endereços já se encontra em cache na ARP *table*.

De seguida, colocamos o sistema terminal n1 à escuta e podemos observar os resultados na figura 4.

Uma imagem com mesa

Descrição gerada automaticamente

Figura 4 - Wireshark n1

Como podemos ver, o sistema n1 apenas recebeu o ARP *request* e não voltou a receber mais nenhum pacote, isto porque o SWITCHarmazenou as informações deste sistema e percebeu que os pacotes não lhe eram destinados.

Além disso, podemos observar mensagens do protocolo ICMP entre os dois sistemas terminais que comutam pacotes. As mensagens observadas (*Echo request* e *Echo Reply*) são mensagens características da execução do comando *ping*.

Assim, concluímos que o HUB comuta os pacotes para todos os sistemas a ele conectados, enquanto o SWITCH faz o mesmo, apenas na primeira vez, depois comuta os pacotes diretamente para a porta de saída correspondente. O protocolo ARP encarrega-se de mapear endereços IP em endereços MAC e o protocolo ICMP preocupa-se em diagnosticar a rede.

# Interligação de redes

Para a resolução do segundo exercício, é-nos pedido que construamos uma rede que permita interligar várias redes locais, utilizando routers com topologia de rede em malha. Os routers permitem que dispositivos de uma rede possam comunicar com dispositivos de outras redes.

Os routers deverão estar conectados por sub-redes com máscaras de 30 bits da rede 196.168.0.0/24. As redes locais devem estar na gama de endereços 10.0.0.0/23. A rede associada ao encaminhador R1 poderá conter até um máximo de 300 computadores, a rede associada ao encaminhador R2 conterá no máximo 100 e as redes associadas aos encaminhadores R3 e R4 20 computadores. A tabela 1 possui a tabela de endereçamento para cada rede local. Para conseguirmos ter endereços IP para todos os sistemas terminais, tivemos de dividir a rede R1 em duas sub-redes (R1.1 e R1.2).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **End de rede** | **Gama de endereços válidos** | **End de difusão** | **Máscara de Rede** |
| **R1.1 - 252 pc** | 10.0.0.0/24 | 10.0.0.1/24-10.0.0.254/24 | 10.0.0.255/24 | 255.255.255.0 |
| **R2 - 100 pc** | 10.0.1.0/25 | 10.0.1.1/25-10.0.1.126/25 | 10.0.1.127/25 | 255.255.255.128 |
| **R1.2 - 48 pc** | 10.0.1.128/26 | 10.0.1.129/26-10.0.1.190/26 | 10.0.1.191/26 | 255.255.255.192 |
| **R3- 20 pc** | 10.0.1.192/27 | 10.0.1.193/27-10.0.1.222/27 | 10.0.1.223/27 | 255.255.255.224 |
| **R4 -20 pc** | 10.0.1.224/27 | 10.0.1.225/27-10.0.1.254/27 | 10.0.1.255/27 | 255.255.255.224 |

Tabela 1 - Esquema de endereçamento

Na figura 5 é possível observar a rede desenhada no CORE, bem como os endereços IP de cada elemento constituinte das redes.

Uma imagem com texto, mapa, diferente, lago

Descrição gerada automaticamente

Figura 5 - Interligação de redes

Também nos é pedido para estabelecermos um esquema de encaminhamento, demonstrado na tabela 2, e configurar esse mesmo esquema no CORE.

|  |  |  |  |
| --- | --- | --- | --- |
| Router | Rede de Destino | Interface de Saída | Próximo Nó |
| N1 | 10.0.0.0/24 | 10.0.0.254/24 | Diretamente |
| 10.0.1.128/26 | 10.0.1.190/26 | Diretamente |
| 10.0.1.0/25 | 196.168.0.1/30 | 196.168.0.2/30 |
| 10.0.1.192/27 | 196.168.0.21/30 | 196.168.0.22/30 |
| 10.0.1.224/27 | 196.168.0.21/30 | 196.168.0.22/30 |
| N2 | 10.0.0.0/24 | 196.168.0.2/30 | 196.168.0.1/30 |
| 10.0.1.128/26 | 196.168.0.2/30 | 196.168.0.1/30 |
| 10.0.1.0/25 | 10.0.1.126/25 | Diretamente |
| 10.0.1.192/27 | 196.168.0.25/30 | 196.168.0.26/30 |
| 10.0.1.224/27 | 196.168.0.5/30 | 196.168.0.6/30 |
| N3 | 10.0.0.0/24 | 196.168.0.17/30 | 196.168.0.18/30 |
| 10.0.1.128/26 | 196.168.0.17/30 | 196.168.0.18/30 |
| 10.0.1.0/25 | 196.168.0.30/30 | 196.168.0.29/30 |
| 10.0.1.192/27 | 10.0.1.222/27 | Diretamente |
| 10.0.1.224/27 | 196.168.0.14/30 | 196.168.0.13/30 |
| N4 | 10.0.0.0/24 | 196.168.0.13/30 | 196.168.0.14/30 |
| 10.0.1.128/26 | 196.168.0.13/30 | 196.168.0.14/30 |
| 10.0.1.0/25 | 196.168.0.10/30 | 196.168.0.9/30 |
| 10.0.1.192/27 | 196.168.0.13/30 | 196.168.0.14/30 |
| 10.0.1.224/27 | 10.0.1.254/27 | Diretamente |
| N5 | 10.0.0.0/24 | 196.168.0.22/30 | 196.168.0.21/30 |
| 10.0.1.128/26 | 196.168.0.22/30 | 196.168.0.21/30 |
| 10.0.1.0/25 | 196.168.0.22/30 | 196.168.0.21/30 |
| 10.0.1.192/27 | 196.168.0.18/30 | 196.168.0.17/30 |
| 10.0.1.224/27 | 196.168.0.18/30 | 196.168.0.17/30 |
| N6 | 10.0.0.0/24 | 196.168.0.26/30 | 196.168.0.25/30 |
| 10.0.1.128/26 | 196.168.0.26/30 | 196.168.0.25/30 |
| 10.0.1.0/25 | 196.168.0.26/30 | 196.168.0.25/30 |
| 10.0.1.192/27 | 196.168.0.29/30 | 196.168.0.30/30 |
| 10.0.1.224/27 | 196.168.0.29/30 | 196.168.0.30/30 |
| N7 | 10.0.0.0/24 | 196.168.0.6/30 | 196.168.0.5/30 |
| 10.0.1.128/26 | 196.168.0.6/30 | 196.168.0.5/30 |
| 10.0.1.0/25 | 196.168.0.6/30 | 196.168.0.5/30 |
| 10.0.1.192/27 | 196.168.0.9/30 | 196.168.0.10/30 |
| 10.0.1.224/27 | 196.168.0.9/30 | 196.168.0.10/30 |

Tabela 2 - Esquema de encaminhamento

Para configurar o esquema concebido anteriormente no CORE, desativamos o encaminhamento dinâmico nos routers (desativar protocolo OSPF) e adicionamos, manualmente, as rotas estabelecidas no nosso esquema de encaminhamento (encaminhamento estático). Para tal, começamos por desativar o parâmetro *DefaultRoute*, acedido em *Services*, em cada router. Nos sistemas terminais, alteramos o script desse parâmetro e colocamos o *IP default* correspondente à interface do router associada à rede em que se encontra o sistema terminal em questão, como exemplificado na figura 6.

Uma imagem com texto

Descrição gerada automaticamente

Figura 6 - DefaultRoute n14

Uma vez que o router 1 (n1), tem associado a ele duas sub-redes na mesma interface, tivemos de configurar no CORE essa mesma interface. Para isso, utilizamos o software Quagga, que nos permite configurar os routers. Uma alternativa seria, em modo execução acedemos ao terminal vtysh do router em questão e corremos os comandos *configure terminal* > *interface eth2* > *ip add [ip da subrede]*, para verificar as alterações, utilizávamos o comando *show running-config*. Tais passos podem ser confirmados na figura 7. No entanto, nós optamos por aceder a *Services* > *Zebra* e alterar diretamente o *script* e adicionar mais um *ip adress* na interface eth2, como se pode observar também na figura 7.

*Uma imagem com texto

Descrição gerada automaticamente*Uma imagem com texto

Descrição gerada automaticamente

Figura 7 - Interface eth2 configuration router1 (n1)

Neste ponto, estamos em condições de configurar os encaminhamentos estáticos no CORE. Para configurá-los, em modo execução, acedemos ao terminal vtysh do router em questão e corremos os comandos *configure terminal* > *ip route [rede destino] [próximo nó]*, como exemplificado na figura 8 para o router n7. Repetimos o processo para todos os routers, de acordo com o nosso esquema de encaminhamento. Para guardarmos as configurações do router editamos o script acedido em *Services* > *Zebra,* como também se pode verificar na figura 8.

Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

Figura 8 - IP route router7 (n7)

Finalmente, podemos executar os comandos *ping* e *traceroute.* Efetuamos os testes do sistema terminal n14, pertencente à rede 1.2, para o sistema terminal n31, pertencente à rede 4, e como é possível verificar na figura 9, a rota entre os dois sistemas é a rota que estabelecemos anteriormente.

Uma imagem com texto

Descrição gerada automaticamente

Figura 9 - Ping e traceroute n14-n31

Algo interessante e expectável a reparar é que quando fazemos *traceroute* de um sistema terminal da rede R1.1 para a rede R1.2, o pacote passa pela interface do router associada à rede de origem, e só depois é encaminhado para o destino, como se pode averiguar na figura 10.

Uma imagem com texto

Descrição gerada automaticamente

Figura 10 – Traceroute R1.1 para R1.2

# DHCP

Até agora, todos os endereços IP dos sistemas constituintes da rede foram atribuídos de forma estática e manual. Neste exercício pretende-se que utilizemos o protocolo DHCP (*Dynamic Host Configuration Protocol*) que permite a atribuição de endereços IP de forma automática, por parte de um servidor. Esta atribuição ocorre dentro de uma gama configurada no servidor. Este protocolo é bastante útil pois faz uma gestão dos endereços IP, i.e., adiciona e liberta endereços consoante as necessidades da rede local, se é adicionado algum sistema ou removido. Também previne que haja erros na inserção dos endereços.

Selecionámos a rede local R3 e alterámos a sua topologia para que os endereços IP fossem atribuídos dinamicamente. Acrescentamos um servidor, que tivemos de configurar. Selecionamos, no servidor, o parâmetro DHCP e nos outros sistemas terminais, o parâmetro DHCPClient. A figura 11 mostra a nova topologia da rede local R3 e a figura 12 mostra a configuração efetuada no servidor.

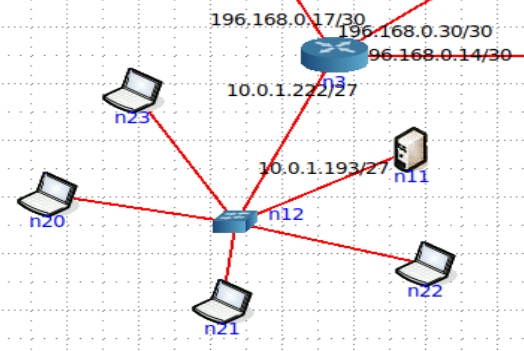


Figura 11 - Rede local R3 com DHCP

Uma imagem com texto

Descrição gerada automaticamente

Figura 12 - Configuração servidor

Com o auxílio da ferramenta *Wireshark,* efetuamos capturas de pacotes no servidor como mostra a figura 13. Como podemos ver, o servidor recebe várias mensagens *DHCP Discovery* que são enviadas pelos clientes, em modo Broadcast, para descobrirem onde está o servidor DHCP. Por sua vez, o servidor responde aos clientes com *DHCP Offer*, que possui o endereço IP a oferecer ao cliente. O cliente responde a esta mensagem com *DHCP Request* com o endereço IP escolhido. Esta mensagem é enviada em modo Broadcast, isto porque o cliente só pode escolher um endereço e podem existir mais do que um servidor DHCP que já podem ter reservado um endereço IP para o cliente em questão, assim sabem qual endereço escolhido pelo cliente e anulam as reservas efetuadas. Finalmente, o servidor responde com *DHCP ACK*, para o cliente saber que a partir de agora está livre para usar o endereço IP escolhido.

Uma imagem com mesa

Descrição gerada automaticamente

Figura 13 - Captura de pacotes no servidor - DHCP

# Uso das camadas de rede e transporte por parte das aplicações

Nesta fase, é pretendido que se ative, pelo menos um servidor HTTP ou um servidor FTP numa das redes locais da rede implementada. Nós optamos por ativar um servidor HTTP na rede R4 e um servidor FTP na rede R2, conforme se pode ver na figura 14.

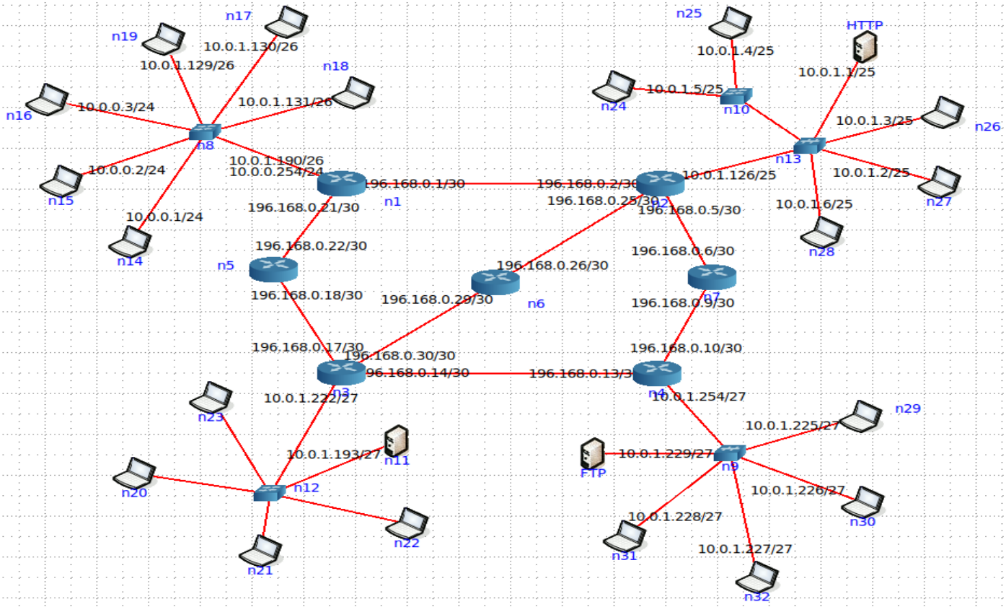


Figura 14 - Topologia de rede com servidores HTTP e FTP

Um servidor FTP é um servidor que faculta, após a autentificação de um utilizador, um serviço de acesso a um disco rígido ou servidor de ficheiros e utiliza o *File Transfer Protocol* (FTP).

Um servidor HTTP é um servidor que utiliza o protocolo HTTP (*Hypertext Transfer Protocol*) e que responde a pedidos de um cliente para páginas *web*.

Quer o protocolo FTP quer o protocolo HTTP são protocolos usados para a transferência de informação entre cliente e servidor e que ocorrem ao nível da camada de aplicação. O que os distingue é que o HTTP fornece uma página da web do servidor ao cliente, enquanto o FTP é utilizado para o download de ficheiros entre cliente e servidor. Ambos os protocolos quando conjugados com o protocolo TCP (*Transmission Control Protocol*), que permite a transmissão de dados entre redes, conseguem enviar dados de forma confiável entre servidores e clientes de diferentes redes, e que ocorre ao nível da camada de transporte. Portanto, é expectável que nas capturas de tráfego que façamos, vejamos os protocolos da camada de aplicação, juntamente com os protocolos das camadas de transporte e de rede.

Para configurarmos o servidor FTP e HTTP verificamos se o software está instalado na máquina onde está a executar o CORE, com os comandos *sudo apt-get install vsftpd*, para o FTP, e *sudo apt-get install apache2*, para o HTTP, e acrescentamos *StartUp* *commands* em cada servidor*,* como mostra a figura 15.

Uma imagem com texto

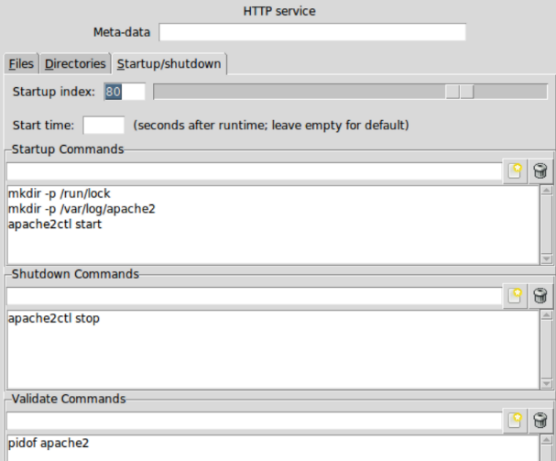
Descrição gerada automaticamente

Figura 15 - StartUp commands FTP e HTTP

Para conectarmos o sistema terminal n14 ao servidor FTP da rede R4, executamos o comando *ftp [endreço ip do servidor]*, na bash do sistema terminal n14, que pertence a uma rede local distinta. Após a execução deste comando é solicitado que se efetue o login. Depois do login ser bem-sucedido, executamos o comando *ls* para listar as diretorias do servidor. Isto pode ser comprovado na figura 16.

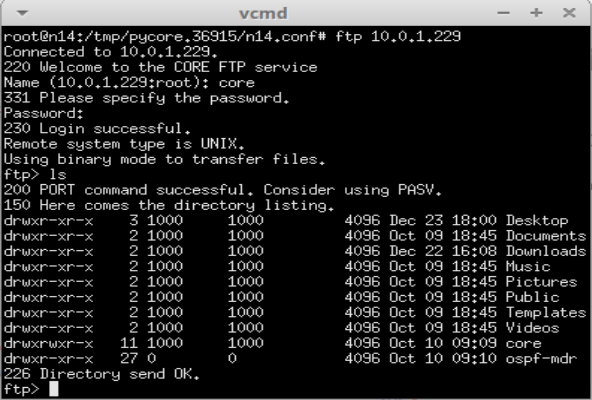


Figura 16 - Ligação ao servidor FTP através do um cliente da rede 1

Na figura 17, é possível observar os pacotes comutados entre o servidor e o sistema terminal n14 quando corremos os comandos supramencionados. Como podemos analisar, um dos protocolos envolvidos na comutação destes pacotes é o protocolo TCP, já abordado anteriormente, que transporta os dados entre o cliente e o servidor. O outro protocolo envolvido é o FTP, como seria de esperar, que transfere os dados entre os dois sistemas. Podemos ver que este protocolo envia mensagens *request* ao cliente, para a introdução dos dados de login; quando o login é efetuado, a mensagem enviada é uma *response* e quando o cliente executa o comando *ls* a pedir a listagem das diretorias, envia uma *response* a dizer que a listagem “vai a caminho”. Por sua vez, o TPC transporta a informação para o cliente.

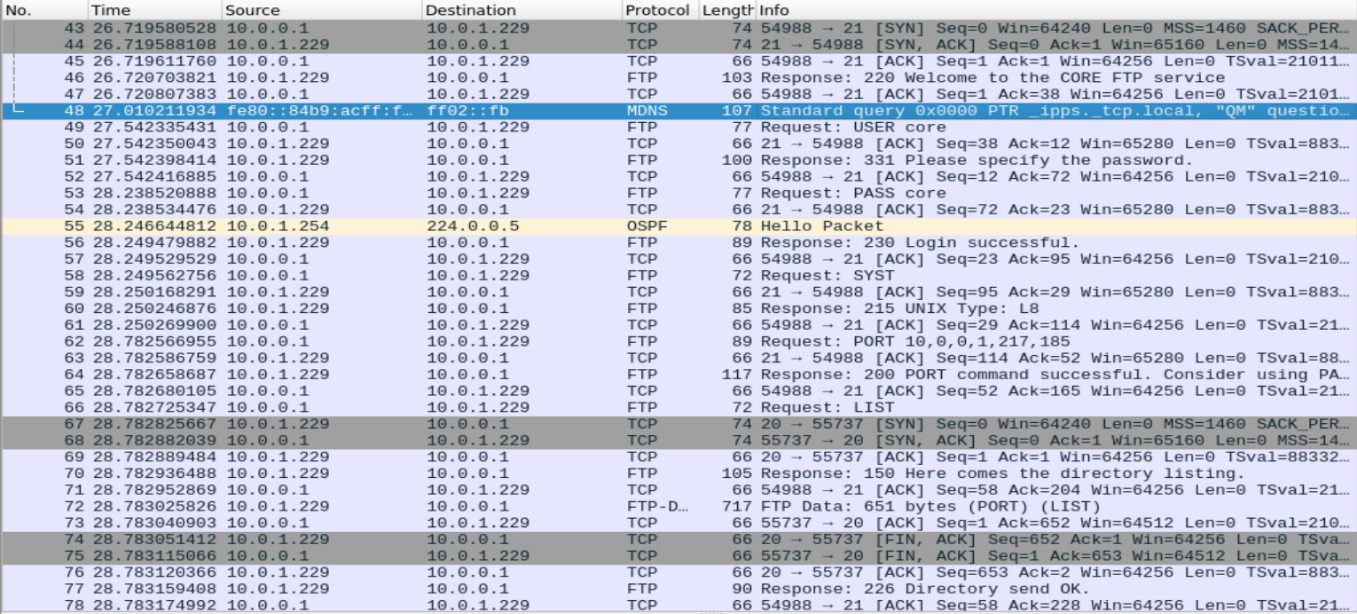


Figura 17 - WireShark FTP Server

Para conectarmos o sistema terminal n23 ao servidor HTTP da rede R2, executamos o comando *wget -S [endreço ip do servidor],* na bash do sistema terminal n23, pertencente à rede R3. O comando *wget* favorece o download de dados da web, conjugado com a opção *-S*, imprime os cabeçalhos enviados pelo servidor HTTP.

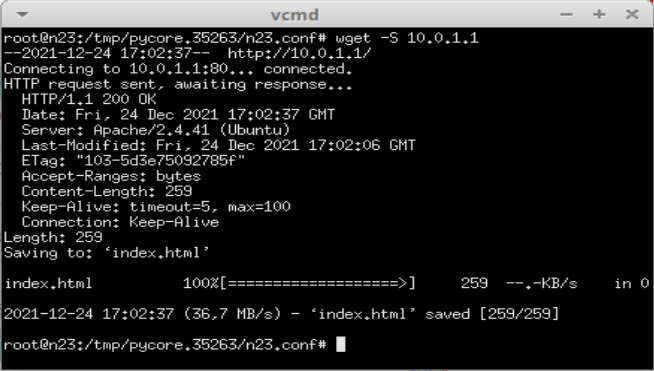


Figura 18 - Ligação ao servidor HTTP através do um cliente da rede 3

Na figura 19, estão presentes os pacotes recebidos e enviados pelo servidor HTTP após a execução do comando acima mencionado. Mais uma vez, vemos que o TPC é um protocolo envolvido na comutação dos pacotes, além do protocolo HTTP. Também pode-se reparar que o protocolo ARP está envolvido, pois como a informação enviada pelo servidor tem de passar pelo router da rede a que pertence, o router tenta descobrir qual o endereço MAC do servidor, para tal usa o protocolo ARP, que como já vimos anteriormente, mapeia endereços IP em endereços MAC. Se fizemos mais capturas de tráfego, iriamos reparar que outros protocolos da pilha protocolar TCP/IP estavam envolvidos.

Uma imagem com texto

Descrição gerada automaticamente

Figura - Wireshark HTTP server

# 6. Conclusão

A concretização deste projeto foi bastante útil para aprofundarmos e aplicarmos os conhecimentos adquiridos nas aulas ao longo do semestre.

Numa primeira fase, tivemos dificuldades em entender como trabalhar no software CORE e quais as suas funcionalidades, mas com a ajuda da docente foi possível atingirmos todos os objetivos propostos na realização deste trabalho.

Conseguimos aplicar conceitos tais como endereçar os sistemas terminais de uma rede, poder avaliar o comportamento de tecnologias como HUBs e SWITCHs, também podemos ver o comportamento dos protocolos da pilha protocolar TCP/IP, lecionados nas aulas teóricas.

Deste modo, é colossal o conhecimento que levamos connosco após a realização deste projeto.