

EJERCICIO 1: INTRODUCCIÓN

- Arranca el Servidor de Windows
- Lanza el programa *Wireshark* en el cliente Windows (conectado en red Interna para limitar el tráfico del entorno)

Comprobamos que estamos en red interna y conectado a través del servidor visualizando la Configuración de red.

```
C:\Users\administrador-204>ipconfig

Configuración IP de Windows

Adaptador de Ethernet Ethernet 2:

    Sufijo DNS específico para la conexión. . . : empresa204.local
    Vínculo: dirección IPv6 local. . . . . : fe80::f460:f2f8:8cf1:6c8a%14
    Dirección IPv4. . . . . : 172.16.204.22
    Máscara de subred . . . . . : 255.255.255.0
    Puerta de enlace predeterminada . . . . : 172.16.204.201
                                                192.168.0.200
```

- Borra todas las entradas de tu cache de ARP: **arp -d ***
- Desde una consola ejecuta la orden **ping www.google.es** y captura los paquetes que lleguen a tu tarjeta de red (en modo NO promiscuo) durante 10 segundos. Observa los paquetes que has capturado.

The screenshot shows two windows. On the left is the Wireshark interface with a packet capture filter set to 'eth 0'. The packet list shows several ARP requests (No. 1, 2, 3, 5, 6, 7) and a DNS query (No. 4). On the right is a Windows command prompt window showing the execution of 'arp -d *' and 'ping www.google.es'. The ping command results show that the connection to 216.239.38.120 was successful, with a response time of 29ms.

Primero observamos que se nos generan paquetes ARP para preguntar por el equipo que tenga la IP que el cliente tiene asignado como nuestro servidor DNS.

En segundo lugar vemos que hay paquetes DNS hacia nuestro servidor Windows Server, el cual habría primeramente resuelto externamente esa solicitud DNS, y luego se la devolvería al cliente.

Posteriormente para acceder a esa IP de Google a la que hacer el PING tenía que encaminarse por su puerta de enlace de la cual también desconocía su MAC, así que se generan nuevos paquetes ARP con ese destino. En todo este tardío proceso observamos que el primer paquete del Ping resultó infructuoso por superar el TTL.

Finalmente observamos que el resto de paquetes ping sí pudieron realizarse correctamente, esto lo vemos con los 6 paquetes ICMP generados, que se corresponden con los pings exitosos observados a la derecha.

EJERCICIO 3: LA ORDEN PING

Para comenzar, estudiaremos la orden **ping**. Como sabemos, la orden **ping** genera paquetes ICMP de tipo **echo request** y **echo reply**. A continuación, comprobaremos el funcionamiento de la misma.

Realiza una captura de los paquetes ICMP generados tras la ejecución de la orden **ping www.elpais.com** y aplica un filtro que te permita capturar únicamente los paquetes que contengan mensajes del protocolo ICMP.

Detén la captura cuando terminen los cuatro intentos y observa cuántos mensajes ICMP se producen, prestando especial atención a los campos **tipo**, **código**, y **bytes de datos**.

Asimismo, analiza las cabeceras IP de cada uno de ellos, y en concreto los campos **longitud de la cabecera**, **longitud total** y **bytes de datos**.

4	0.064236	172.16.204.22	151.101.134.133	ICMP	74 Echo (ping) request	id=0x0001, seq=9/2304, ttl=128 (reply in 5)
5	0.098256	151.101.134.133	172.16.204.22	ICMP	74 Echo (ping) reply	id=0x0001, seq=9/2304, ttl=48 (request in 4)
6	1.105718	172.16.204.22	151.101.134.133	ICMP	74 Echo (ping) request	id=0x0001, seq=10/2560, ttl=128 (reply in 7)
7	1.140051	151.101.134.133	172.16.204.22	ICMP	74 Echo (ping) reply	id=0x0001, seq=10/2560, ttl=48 (request in 6)
8	2.123004	172.16.204.22	151.101.134.133	ICMP	74 Echo (ping) request	id=0x0001, seq=11/2816, ttl=128 (reply in 9)
9	2.157859	151.101.134.133	172.16.204.22	ICMP	74 Echo (ping) reply	id=0x0001, seq=11/2816, ttl=48 (request in 8)
10	3.153964	172.16.204.22	151.101.134.133	ICMP	74 Echo (ping) request	id=0x0001, seq=12/3072, ttl=128 (reply in 11)
11	3.189662	151.101.134.133	172.16.204.22	ICMP	74 Echo (ping) reply	id=0x0001, seq=12/3072, ttl=48 (request in 10)

```
> Frame 4: 74 bytes on wire (592 bits), 7
> Ethernet II, Src: PCSSystemtec_76:75:44
> Internet Protocol Version 4, Src: 172.1
▼ Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0x4d52 [correct]
  [Checksum Status: Good]
  Identifier (BE): 1 (0x0001)
  Identifier (LE): 256 (0x0100)
  Sequence Number (BE): 9 (0x0009)
  Sequence Number (LE): 2304 (0x0900)
  [Response frame: 5]
▼ Data (32 bytes)
  Data: 6162636465666768696a6b6c6d6e6f
  [Length: 32]
```

Se han generado 8 paquetes ICMP correspondientes a los 4 paquetes ejecutados por el Ping.

Si desplegamos la información del cuarto paquete ICMP, vemos que:

El tipo de paquete ICMP es el 8 que es Echo Request

Tiene código 0, que indica que la red destino es inalcanzable

Envía un paquete de longitud 32 bytes que es el que genera por defecto la operación del Ping.

Si desplegamos los datos de la capa de red, nos indica que estamos usando el protocolo IPv4 nos informa de las IP origen y destino.

Por otro lado vemos que la longitud del total de la capa IP del paquete 4 es de 60 bytes, de los cuales 20 bytes son los de la cabecera de ésta capa y los otros 40 los encapsulados para el ICMP.

```
▼ Internet Protocol Version 4, Src: 172.16.204.22, Dst: 151.101.134.133
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 60
  Identification: 0xd33e (54078)
  > 000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 128
  Protocol: ICMP (1)
  Header Checksum: 0x0000 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.16.204.22
  Destination Address: 151.101.134.133
```

Si decidimos analizar el quinto paquete vemos que:

```
> Frame 5: 74 bytes on wire (592 bits), 74
> Ethernet II, Src: PCSSystemtec_44:42:9b
> Internet Protocol Version 4, Src: 151.101.134.133
▼ Internet Control Message Protocol
  Type: 0 (Echo (ping) reply)
  Code: 0
  Checksum: 0x5552 [correct]
  [Checksum Status: Good]
  Identifier (BE): 1 (0x0001)
  Identifier (LE): 256 (0x0100)
  Sequence Number (BE): 9 (0x0009)
  Sequence Number (LE): 2304 (0x0900)
  [Request frame: 4]
  [Response time: 34,020 ms]
▼ Data (32 bytes)
  Data: 6162636465666768696a6b6c6d6e6f
  [Length: 32]
```

El tipo de paquete ICMP es el 0 que es Echo Reply

Tiene código 0, que indica que la red destino (la nuestra) es inalcanzable para el origen.

Envía un paquete de longitud 32 bytes que es el que genera por defecto la operación del Ping.

Vemos además que Wireshark nos añade como dato que el paquete Request al que se corresponde dicha respuesta, que es al 4, el cual analizamos previamente.

EJERCICIO 4: EL PROTOCOLO DNS

El protocolo DNS, que emplea habitualmente el puerto 53 de UDP, es imprescindible para poder denominar a los computadores mediante nombres simbólicos, sin tener que recordar las direcciones IP correspondientes a cada computador. A continuación, observaremos el funcionamiento del DNS, así como su utilización por parte de la orden **ping**.

Modifica el filtro para que encuentre la consulta DNS y toma una nueva captura, escribiendo la orden **ping www.nba.com**. Ten en cuenta que no será posible filtrar paquetes en función del protocolo de aplicación DNS y deberás, por tanto, pensar en otra alternativa.

Examina el contenido del datagrama UDP y fíjate en los números de puerto implicados en la transmisión.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.00...	172.16.204.22	172.16.204.201	DNS	71	Standard query 0x6cdf A www.nba.com
2	0.01...	172.16.204.201	172.16.204.22	DNS	163	Standard query response 0x6cdf A www.nb

> Frame 1: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface \Device\NPF

> Ethernet II, Src: PCSSystemtec_76:75:44 (08:00:27:76:75:44), Dst: PCSSystemtec_44:42:9b (08

> Internet Protocol Version 4, Src: 172.16.204.22, Dst: 172.16.204.201

> User Datagram Protocol, Src Port: 55101, Dst Port: 53

Source Port: 55101

Destination Port: 53

Sí es posible hacer el filtrado para que nos muestre solo los paquetes del protocolo DNS, así observamos que tenemos la consulta y respuesta DNS. Y que estos se han realizado como puerto origen el 55101 (que es un puerto privado multipropósito) y de puerto destino 53 (que es un puerto dedicado por parte del servidor para las consultas DNS).

Captura el resultado del comando **nslookup www.empresa204.local**

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.204.22	172.16.204.201	DNS	87	Standard query 0x0001 PTR 201.204.16.172.in-addr.arpa
2	0.000542	172.16.204.201	172.16.204.22	DNS	198	Standard query response 0x0001 PTR 201.204.16.172.in-addr.arpa PTR WS-204.empresa204.local PTR ww
3	0.004132	172.16.204.22	172.16.204.201	DNS	97	Standard query 0x0002 A www.empresa204.local.empresa204.local
4	0.004608	172.16.204.201	172.16.204.22	DNS	156	Standard query response 0x0002 No such name A www.empresa204.local.empresa204.local SOA WS-204.em
5	0.004860	172.16.204.22	172.16.204.201	DNS	97	Standard query 0x0003 AAAA www.empresa204.local.empresa204.local
6	0.005209	172.16.204.201	172.16.204.22	DNS	156	Standard query response 0x0003 No such name AAAA www.empresa204.local.empresa204.local SOA WS-204
7	0.005374	172.16.204.22	172.16.204.201	DNS	80	Standard query 0x0004 A www.empresa204.local
8	0.005759	172.16.204.201	172.16.204.22	DNS	96	Standard query response 0x0004 A www.empresa204.local A 172.16.204.201
9	0.006002	172.16.204.22	172.16.204.201	DNS	80	Standard query 0x0005 AAAA www.empresa204.local
10	0.006307	172.16.204.201	172.16.204.22	DNS	123	Standard query response 0x0005 AAAA www.empresa204.local SOA WS-204.empresa204.local

< Domain Name System (query)

Transaction ID: 0x0004

> Flags: 0x0100 Standard query

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 0

> Queries

> www.empresa204.local: type A, class IN

Name: www.empresa204.local

[Name Length: 20]

[Label Count: 3]

Type: A (1) (Host Address)

Class: IN (0x0001)

0000 08 00 27

0010 00 42 b6

0020 cc c9 c9

0030 00 00 00

0040 73 61 32

Vemos que tras realizar el ping, los primeros paquetes. 1 y 2, preguntan por el nombre que tiene el servidor para el cual tenemos la IP.

Posteriormente dada su configuración realiza la consulta con distintas combinaciones, hasta que con los paquetes 7 y 8 corresponde realmente con nuestra consulta y es para el que nos responde que ha encontrado un registro, y ese es el que se nos muestra.

EJERCICIO 5: EL PROTOCOLO ARP

Para que un datagrama llegue a su destino es necesario especificar, además de la dirección IP destino, la dirección física del adaptador de red que debe recibir la trama en la que viaja el datagrama. Este adaptador de red puede ser el del *host* destino o bien el de un *router* intermedio.

Precisamente, para averiguar la dirección física que corresponde a una dirección IP determinada se creó el protocolo ARP, del que ya hemos hablado en clase (y habéis *disfrutado* con él haciendo los problemas de encaminamiento IP). Vamos a retomar de nuevo el estudio de este protocolo.

Borra las entradas de tu cache de ARP (arp -d *).

Captura el tráfico generado por la ejecución de la orden **ping www.barrapunto.com**. Modificando el filtro de la captura anterior para que incluya los paquetes del protocolo ARP (sólo los de tu máquina). ¿Se ha generado algún paquete ARP para enviar la trama que contiene la petición a barrapunto?

Vuelve a realizar de nuevo la captura (no dejes pasar mucho tiempo). Y ahora, ¿se han vuelto a generar paquetes ARP? ¿Por qué?

1	0.000000	PCSSystemtec_76:75:44	Broadcast	ARP	42	Who has 192.168.0.200? Tell 172.16.204.22
2	0.757275	PCSSystemtec_76:75:44	Broadcast	ARP	42	Who has 192.168.0.200? Tell 172.16.204.22
3	1.772504	PCSSystemtec_76:75:44	Broadcast	ARP	42	Who has 192.168.0.200? Tell 172.16.204.22
4	2.784752	PCSSystemtec_76:75:44	Broadcast	ARP	42	Who has 172.16.204.201? Tell 172.16.204.22
5	2.785042	PCSSystemtec_44:42:9b	PCSSystemtec_76:75:44	ARP	60	172.16.204.201 is at 08:00:27:44:42:9b
6	2.785053	172.16.204.22	5.9.42.126	ICMP	74	Echo (ping) request id=0x0001, seq=61/15616, ttl=128 (reply in 7)
7	2.860080	5.9.42.126	172.16.204.22	ICMP	74	Echo (ping) reply id=0x0001, seq=61/15616, ttl=41 (request in 6)
8	3.815366	172.16.204.22	5.9.42.126	ICMP	74	Echo (ping) request id=0x0001, seq=62/15872, ttl=128 (reply in 9)
9	3.886366	5.9.42.126	172.16.204.22	ICMP	74	Echo (ping) reply id=0x0001, seq=62/15872, ttl=41 (request in 8)
10	4.835047	172.16.204.22	5.9.42.126	ICMP	74	Echo (ping) request id=0x0001, seq=63/16128, ttl=128 (reply in 11)
11	4.911940	5.9.42.126	172.16.204.22	ICMP	74	Echo (ping) reply id=0x0001, seq=63/16128, ttl=41 (request in 10)
12	8.584636	172.16.204.22	5.9.42.126	ICMP	74	Echo (ping) request id=0x0001, seq=64/16384, ttl=128 (reply in 13)
13	8.663459	5.9.42.126	172.16.204.22	ICMP	74	Echo (ping) reply id=0x0001, seq=64/16384, ttl=41 (request in 12)
14	9.616015	172.16.204.22	5.9.42.126	ICMP	74	Echo (ping) request id=0x0001, seq=65/16640, ttl=128 (reply in 15)
15	9.689172	5.9.42.126	172.16.204.22	ICMP	74	Echo (ping) reply id=0x0001, seq=65/16640, ttl=41 (request in 14)
16	10.633...	172.16.204.22	5.9.42.126	ICMP	74	Echo (ping) request id=0x0001, seq=66/16896, ttl=128 (reply in 17)
17	10.707...	5.9.42.126	172.16.204.22	ICMP	74	Echo (ping) reply id=0x0001, seq=66/16896, ttl=41 (request in 16)
18	11.647...	172.16.204.22	5.9.42.126	ICMP	74	Echo (ping) request id=0x0001, seq=67/17152, ttl=128 (reply in 19)
19	11.725...	5.9.42.126	172.16.204.22	ICMP	74	Echo (ping) reply id=0x0001, seq=67/17152, ttl=41 (request in 18)

Los paquetes correspondientes al primer ping son los del cuadro verde (paquetes 1 al 11) y los del segundo ping los del cuadro naranja (paquetes del 12 al 19). Tal y como explicamos en la [Introducción](#) se generan consultas ARP a nuestro DNS de Windows (para realizar la consulta DNS) y al de la clase puesto que debe encaminar por él, como todo el proceso es tardío el primer paquete del primer ping resultó infructuoso.

Pero no tuvo dificultades para resolver los siguientes 4 del segundo ping, ya que no tuvo que usar paquetes ARP para elaborar su tabla de encaminamiento.

Nota: como sabes, puedes conocer la dirección física de la tarjeta de red de tu ordenador mediante la orden **ipconfig**. Además, recuerda que puedes consultar la tabla ARP con **arp -a**.

EJERCICIO 6: DHCP

El protocolo DHCP permite la asignación automática de direcciones IP

- Libera la IP de la máquina cliente con ipconfig /release.
- Pon el programa Wireshark a escuchar por la interfaz de red en el Servidor Windows Server
- Renovar la IP del equipo cliente con ipconfig /renew.
- Capturar los paquetes DHCP con Wireshark.

Analiza los mensajes de intercambio entre el servidor y el cliente que se han generado para asignar una IP al equipo.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0xadab3072
2	0.002175	172.16.204.201	255.255.255.255	DHCP	351	DHCP Offer - Transaction ID 0xadab3072
3	0.005147	0.0.0.0	255.255.255.255	DHCP	356	DHCP Request - Transaction ID 0xadab3072
4	0.012744	172.16.204.201	255.255.255.255	DHCP	356	DHCP ACK - Transaction ID 0xadab3072

▶ Frame 1: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface \Device\NPF_{E4...}
 ▶ Ethernet II, Src: PCSSystemtec_97:0c:18 (08:00:27:97:0c:18), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
 ▶ Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
 ▶ User Datagram Protocol, Src Port: 68, Dst Port: 67
 ▶ Dynamic Host Configuration Protocol (Discover)
 Message type: Boot Request (1)
 Hardware type: Ethernet (0x01)
 Hardware address length: 6
 Hops: 0
 Transaction ID: 0xadab3072
 Seconds elapsed: 0
 ▶ Bootp flags: 0x8000, Broadcast flag (Broadcast)
 Client IP address: 0.0.0.0
 Your (client) IP address: 0.0.0.0
 Next server IP address: 0.0.0.0

Podemos observar primeramente que se han generado 4 paquetes para la operación DHCP, siendo estos: DHCP Discover, DHCP Offer, DHCP Request, DHCP ACK.

Estos corresponden a la comunicación DHCP con la que estamos familiarizados. El primero de tipo broadcast lo generó el cliente, con intención de obtener una configuración IP válida, vemos que su IP de origen no está configurada, no tiene IP de cliente. El segundo paquete es el del servidor ofreciendo una configuración IP al cliente, la cual posteriormente acepta y solicita el cliente, y el servidor se la termina de transmitir en el último paquete veamos más detalladamente este paquete.

No.	Time	Source	Destination
1	0.000000	0.0.0.0	255.255.255.255
2	0.002175	172.16.204.201	255.255.255.255
3	0.005147	0.0.0.0	255.255.255.255
4	0.012744	172.16.204.201	255.255.255.255

▶ Frame 4: 356 bytes on wire (2848 bits), 356 bytes captured (2848 bits) on in
 ▶ Ethernet II, Src: PCSSystemtec_af:b4:3e (08:00:27:af:b4:3e), Dst: Broadcast
 ▶ Internet Protocol Version 4, Src: 172.16.204.201, Dst: 255.255.255.255
 ▶ User Datagram Protocol, Src Port: 67, Dst Port: 68
 ▶ Dynamic Host Configuration Protocol (ACK)
 Message type: Boot Reply (2)
 Hardware type: Ethernet (0x01)
 Hardware address length: 6
 Hops: 0
 Transaction ID: 0xadab3072
 Seconds elapsed: 0
 ▶ Bootp flags: 0x0000 (Unicast)
 Client IP address: 0.0.0.0
 Your (client) IP address: 172.16.204.21
 Next server IP address: 0.0.0.0
 Relay agent IP address: 0.0.0.0
 Client MAC address: PCSSystemtec_97:0c:18 (08:00:27:97:0c:18)
 Client hardware address padding: 00000000000000000000
 Server host name not given
 Boot file name not given
 Magic cookie: DHCP
 ▶ Option: (53) DHCP Message Type (ACK)
 Length: 1
 DHCP: ACK (5)
 ▶ Option: (58) Renewal Time Value
 Length: 4
 Renewal Time Value: 2 days (172800)

Renewal Time Value: 2 days (172800)
 ▶ Option: (59) Rebinding Time Value
 Length: 4
 Rebinding Time Value: 3 days, 12 hours (302400)
 ▶ Option: (51) IP Address Lease Time
 Length: 4
 IP Address Lease Time: 4 days (345600)
 ▶ Option: (54) DHCP Server Identifier (172.16.204.201)
 Length: 4
 DHCP Server Identifier: 172.16.204.201
 ▶ Option: (1) Subnet Mask (255.255.255.0)
 Length: 4
 Subnet Mask: 255.255.255.0
 ▶ Option: (81) Client Fully Qualified Domain Name
 Length: 3
 Flags: 0x03, Server overrides, Server
 A-RR result: 255
 PTR-RR result: 0
 ▶ Option: (3) Router
 Length: 8
 Router: 172.16.204.201
 Router: 192.168.0.200
 ▶ Option: (6) Domain Name Server
 Length: 4
 Domain Name Server: 172.16.204.201
 ▶ Option: (15) Domain Name
 Length: 17
 Domain Name: empresa204.local
 ▶ Option: (255) End

En él nos encontramos toda la información que se transmite en ese último paquete, siendo los más destacables aparte de la IP y máscara de red: el tiempo de renovación, el tiempo de concesión, el servidor DNS, Router, y nombre del dominio.

EJERCICIO 7: HTTP

Inicia Wireshark filtrando el protocolo http. En el navegador visita la página <http://www.llegarasalto.com>
Captura y analiza un mensaje de petición y uno de respuesta indicando los parámetros del mensaje.

Primeramente observamos que los paquetes relacionados con esta comunicación HTTP no son implícitamente usando el protocolo **http**, sino que usan el protocolo Transport Layer Security en su versión 1.3 el cual cifra las comunicaciones entre cliente servidor, se generan distintos paquetes en ésta comunicación de los cuales estaremos atentos a los que observamos que tienen que ver con la solicitud de la página como estamos acostumbrados.

Vemos en éste paquete, el **22** que primero se establece una comunicación cliente servidor, con un mensaje Handshake - Client Hello que tiene como Server Name Indication el URL que hemos ingresado.

No.	Time	Source	Destination	Protocol	Length	Info
22	1.554398	172.16.204.21	104.21.49.165	TLSv1.3	626	Client Hello (SNI=www.llegarasalto.com)
23	1.555701	172.16.204.21	104.21.49.165	TLSv1.3	626	Client Hello (SNI=www.llegarasalto.com)
24	1.557962	104.21.49.165	172.16.204.21	TCP	60	443 → 61841 [ACK] Seq=1 Ack=573 Win=65535 Len=0

> Frame 22: 626 bytes on wire (5008 bits), 626 bytes captured (5008 bits) on interface \Device\NPF_{CE37F9D2-B0D4-4642-84DE-...}

> Ethernet II, Src: PCSSystemtec_97:0c:18 (08:00:27:97:0c:18), Dst: PCSSystemtec_af:b4:3e (08:00:27:af:b4:3e)

> Internet Protocol Version 4, Src: 172.16.204.21, Dst: 104.21.49.165

> Transmission Control Protocol, Src Port: 61841, Dst Port: 443, Seq: 1, Ack: 1, Len: 572

▼ Transport Layer Security

▼ TLSv1.3 Record Layer: Handshake Protocol: Client Hello

Content Type: Handshake (22)

Version: TLS 1.0 (0x0301)

Length: 567

▼ Handshake Protocol: Client Hello

Handshake Type: Client Hello (1)

Length: 563

Version: TLS 1.2 (0x0303)

Random: 351e6812badf2db8bf4be334e373f944472dbf5eadf14429da22952d43b83d7c

Session ID Length: 32

Session ID: 4b734f42ae7da5121e6747cc8f1ca18ef8bda4241a46c214c2bc7151494e83d2

34	1.810352	104.21.49.165	172.16.204.21	TLSv1.3	1506	Server Hello, Change Cipher Spec
----	----------	---------------	---------------	---------	------	----------------------------------

> Ethernet II, Src: PCSSystemtec_af:b4:3e (08:00:27:af:b4:3e), Dst: PCSSystemtec_97:0c:18 (08:00:27:97:0c:18)

> Internet Protocol Version 4, Src: 104.21.49.165, Dst: 172.16.204.21

> Transmission Control Protocol, Src Port: 443, Dst Port: 61841, Seq: 1, Ack: 573, Len: 1506

▼ Transport Layer Security

▼ TLSv1.3 Record Layer: Handshake Protocol: Server Hello

Content Type: Handshake (22)

Version: TLS 1.2 (0x0303)

Length: 122

▼ Handshake Protocol: Server Hello

Handshake Type: Server Hello (2)

Length: 118

Version: TLS 1.2 (0x0303)

Random: c56c54fd9fe433b9df4a01f6a1180e6bef62ec95238674ce41461a3a986d1d14

Session ID Length: 32

Session ID: 4b734f42ae7da5121e6747cc8f1ca18ef8bda4241a46c214c2bc7151494e83d2

Cipher Suite: TLS AES 128 GCM SHA256 (0x1301)

En este paquete el 34 observamos que el servidor corresponde el handshake, y podemos relacionar ambos mensajes gracias al Session ID.

Posteriormente, varios mensajes más adelante se producen 2 mensajes TLS, 50 y 58, que informan que la incluyen datos de la capa de aplicación de tipo HTTP más el contenido está encriptado.

50	1.859597	172.16.204.21	104.21.49.165	TLSv1.3	555	Application Data
----	----------	---------------	---------------	---------	-----	------------------

> Ethernet II, Src: PCSSystemtec_af:b4:3e (08:00:27:af:b4:3e), Dst: PCSSystemtec_97:0c:18 (08:00:27:97:0c:18)

> Internet Protocol Version 4, Src: 172.16.204.21, Dst: 104.21.49.165

> Transmission Control Protocol, Src Port: 443, Dst Port: 61841, Seq: 1, Ack: 573, Len: 555

▼ Transport Layer Security

▼ TLSv1.3 Record Layer: Application Data

Opaque Type: Application Data

Version: TLS 1.2 (0x0303)

Length: 496

Encrypted Application Data [truncated]: [Application Data Protocol: Hypertext Transfer Protocol]

58 1.873197 104.21.49.165 172.16.204.21 TLSv1.3 390 Application Data

60 1.875695 172.16.204.21 104.21.49.165 TLSv1.3 118 Change Cipher Spec, Application Data

> Ethernet II, Src: PCSSystemtec_af:b4:3e (08:00:27:af:b4:3e), Dst: PCSSystemtec_97:0c:18 (08:00:27:97:0c:18)

> Internet Protocol Version 4, Src: 104.21.49.165, Dst: 172.16.204.21

> Transmission Control Protocol, Src Port: 443, Dst Port: 61841, Seq: 4357, Ack: 573, Len: 336

[4 Reassembled TCP Segments (4559 bytes): #34(1319), #35(1452), #37(1452), #58(336)]

▼ Transport Layer Security

▼ TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol

Opaque Type: Application Data (23)

Version: TLS 1.2 (0x0303)

Length: 4554

Encrypted Application Data [truncated]: 496a43e8b79c5d37a6cc3bcf748174e8b2c870cd3efa820f28c

[Application Data Protocol: Hypertext Transfer Protocol]

Captura los paquetes generados para las consultas:

· <http://www.empresa204.local>

No.	Time	Source	Destination	Protocol	Length	Info
4	0.015658	172.16.204.21	172.16.204.201	HTTP	480	GET / HTTP/1.1
5	0.016596	172.16.204.201	172.16.204.21	HTTP	792	HTTP/1.1 200 OK (text/html)

Al ser una consulta http corriente se generan 2 paquetes HTTP, se solicita la página y ésta es servida exitosamente sin inconvenientes, ni reenviado, ni cifrado y ni autenticado.

· <http://www.empresa204.local:81>

Time	Source	Destination	Protocol	Length	Info
6	0.504583	172.16.204.21	172.16.204.201	HTTP	483 GET / HTTP/1.1
8	0.505192	172.16.204.201	172.16.204.21	HTTP	88 HTTP/1.1 401 Unauthorized (text/html)
16	5.074665	172.16.204.21	172.16.204.201	HTTP	548 GET / HTTP/1.1
17	5.075451	172.16.204.201	172.16.204.21	HTTP	697 HTTP/1.1 200 OK (text/html)

Transmission Control Protocol, Src Port: 62825, Dst Port: 81, Seq: 1, Ack: 1, Len: 494

Hypertext Transfer Protocol

GET / HTTP/1.1\r\n

Host: www.empresa204.local:81\r\n

Connection: keep-alive\r\n

Cache-Control: max-age=0\r\n

Authorization: Basic dXN1MTokM3J2M3Ix\r\n

Credentials: usu1:\$rv3r1

Upgrade-Insecure-Requests: 1\r\n

Al ser una consulta http que requiere autenticación se generan 4 paquetes HTTP, se solicita la página y ésta responde con un mensaje de que requiere autenticación por parte del cliente.

Posteriormente en el mensaje que estamos viendo, el cliente vuelve a solicitar la página introduciendo sus credenciales, las cuales son visibles al tener una conexión no cifrada.

Finalmente la conexión es servida exitosamente.

· <https://www.empresa204.local:443>

21	0.023479	172.16.204.21	172.16.204.201	TLSv1.3	571	Client Hello (SNI=www.empresa204.local)
22	0.026001	172.16.204.201	172.16.204.21	TLSv1.3	1296	Server Hello, Change Cipher Spec, Application Data
23	0.026355	172.16.204.21	172.16.204.201	TLSv1.3	134	Change Cipher Spec, Application Data
24	0.026598	172.16.204.21	172.16.204.201	TLSv1.3	146	Application Data
25	0.026806	172.16.204.201	172.16.204.21	TCP	60	443 → 62872 [ACK] Seq=1243 Ack=690 Win=2097152 Len=0
26	0.026884	172.16.204.21	172.16.204.201	TLSv1.3	546	Application Data
27	0.027207	172.16.204.201	172.16.204.21	TLSv1.3	157	Application Data
28	0.027335	172.16.204.201	172.16.204.21	TLSv1.3	116	Application Data

Frame 24: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits) on interface \Device\NPF_{CE37F9D2-B0D4-4642-84DE-7} Ethernet II, Src: PCSSystemtec_76:75:44 (08:00:27:76:75:44), Dst: PCSSystemtec_44:42:9b (08:00:27:44:42:9b)

Internet Protocol Version 4, Src: 172.16.204.21, Dst: 172.16.204.201

Transmission Control Protocol, Src Port: 62872, Dst Port: 443, Seq: 598, Ack: 1243, Len: 92

Transport Layer Security

▼ TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol

Opaque Type: Application Data (23)

Version: TLS 1.2 (0x0303)

Length: 87

Encrypted Application Data: b0973ac0fa87afc17cee622d092233b1162a6ed8343778b2530ed6af8a1ae880b1e0adc14f4c6c4cd4125c9a395

[Application Data Protocol: Hypertext Transfer Protocol]

Al ser una consulta https, esta es cifrada, y al igual que [el caso inicial](#) podemos observar 2 paquetes de Handshake entre cliente y servidor, para proceder a una conexión cifrada, el 3^{er} paquete es para consensuar esa información de cifrado.

Finalmente, el 4^o paquete (el desplegado) y el 5^o corresponden a la transferencia encriptada de la página HTML.

Debes capturar los paquetes que se generan con las siguientes

· <http://www.dominio204.local:82>

Time	Source	Destination	Protocol	Length	Info
32 0.169508	172.16.204.151	172.16.204.202	HTTP	483	GET / HTTP/1.1
34 0.170176	172.16.204.202	172.16.204.151	HTTP	801	HTTP/1.1 401 Unauthorized (text/html)
47 5.298477	172.16.204.151	172.16.204.202	HTTP	552	GET / HTTP/1.1
49 5.320697	172.16.204.202	172.16.204.151	HTTP	658	HTTP/1.1 200 OK (text/html)

Frame 47: 552 bytes on wire (4416 bits), 552 bytes captured (4416 bits) on interface \Device\NPF_{...} Ethernet II, Src: PCSSystemtec_76:75:44 (08:00:27:76:75:44), Dst: PCSSystemtec_c1:96:0d (08:00:27:76:75:44) Internet Protocol Version 4, Src: 172.16.204.151, Dst: 172.16.204.202 Transmission Control Protocol, Src Port: 53713, Dst Port: 82, Seq: 1, Ack: 1, Len: 498

Hypertext Transfer Protocol

GET / HTTP/1.1\r\n

Host: www.dominio204.local:82\r\n

Connection: keep-alive\r\n

Cache-Control: max-age=0\r\n

Authorization: Basic ZW1wbGVhZG8yOmFiYw==\r\n

Credentials: empleado2:abc

Upgrade-Insecure-Requests: 1\r\n

Se repite el caso de la carga de [página](#) con autenticación de Windows Server, y nuevamente podemos ver las credenciales usadas.

Comprobar web segura (encriptado)

· <https://www.dominio204.local:444>

Time	Source	Destination	Protocol	Length	Info
16 0.005604	172.16.204.151	172.16.204.202	TLSv1.3	945	Client Hello (SNI=www.dominio204.local)
17 0.006035	172.16.204.202	172.16.204.151	TCP	60	10000 → 53744 [ACK] Seq=1 Ack=892 Win=64128 Len=0
18 0.008023	172.16.204.202	172.16.204.151	TLSv1.3	299	Server Hello, Change Cipher Spec, Application Data, Application Data
19 0.012585	172.16.204.151	172.16.204.202	TLSv1.3	134	Change Cipher Spec, Application Data
20 0.012833	172.16.204.151	172.16.204.202	TLSv1.3	911	Application Data
21 0.012945	172.16.204.202	172.16.204.151	TCP	60	10000 → 53744 [ACK] Seq=246 Ack=972 Win=64128 Len=0
22 0.013074	172.16.204.202	172.16.204.151	TCP	60	10000 → 53744 [ACK] Seq=246 Ack=1829 Win=64000 Len=0
23 0.013303	172.16.204.202	172.16.204.151	TLSv1.3	325	Application Data

· <https://web1.dominio204.local:445>

No.	Time	Source	Destination	Protocol	Length	Info
84	3.242796	172.16.204.151	20.103.180.120	TLSv1.2	644	Client Hello (SNI=postnav-edge.smartscreen.microsoft.com)
85	3.244110	20.103.180.120	172.16.204.151	TCP	60	443 → 53792 [ACK] Seq=1 Ack=591 Win=65535 Len=0
86	3.297435	20.103.180.120	172.16.204.151	TCP	1514	443 → 53792 [ACK] Seq=1 Ack=591 Win=65535 Len=1460 [TCP segment of a reassembled PDU]
87	3.297435	20.103.180.120	172.16.204.151	TCP	1514	443 → 53792 [ACK] Seq=1461 Ack=591 Win=65535 Len=1460 [TCP segment of a reassembled PDU]
88	3.297495	172.16.204.151	20.103.180.120	TCP	54	53792 → 443 [ACK] Seq=591 Ack=2921 Win=64240 Len=0
89	3.298023	20.103.180.120	172.16.204.151	TCP	1514	443 → 53792 [ACK] Seq=2921 Ack=591 Win=65535 Len=1460 [TCP segment of a reassembled PDU]
90	3.298023	20.103.180.120	172.16.204.151	TCP	1514	443 → 53792 [ACK] Seq=4381 Ack=591 Win=65535 Len=1460 [TCP segment of a reassembled PDU]
91	3.298023	20.103.180.120	172.16.204.151	TLSv1.2	1308	Server Hello, Certificate, Certificate Status, Server Key Exchange, Server Hello Done
92	3.298058	172.16.204.151	20.103.180.120	TCP	54	53792 → 443 [ACK] Seq=591 Ack=7095 Win=64240 Len=0
93	3.301708	172.16.204.151	20.103.180.120	TLSv1.2	212	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
94	3.302801	20.103.180.120	172.16.204.151	TCP	60	443 → 53792 [ACK] Seq=7095 Ack=749 Win=65535 Len=0
95	3.357893	20.103.180.120	172.16.204.151	TLSv1.2	105	Change Cipher Spec, Encrypted Handshake Message
96	3.358197	172.16.204.151	20.103.180.120	TLSv1.2	700	Application Data
97	3.358324	172.16.204.151	20.103.180.120	TLSv1.2	2055	Application Data
98	3.358724	20.103.180.120	172.16.204.151	TCP	60	443 → 53792 [ACK] Seq=7146 Ack=1395 Win=65535 Len=0
99	3.358724	20.103.180.120	172.16.204.151	TCP	60	443 → 53792 [ACK] Seq=7146 Ack=2855 Win=65535 Len=0
100	3.358724	20.103.180.120	172.16.204.151	TCP	60	443 → 53792 [ACK] Seq=7146 Ack=3396 Win=65535 Len=0
101	3.372096	172.16.204.202	172.16.204.151	TCP	60	[TCP Retransmission] 445 → 53790 [FIN, ACK] Seq=1348 Ack=549 Win=64128 Len=0
102	3.372125	172.16.204.151	172.16.204.202	TCP	54	[TCP ZeroWindow] 53790 → 445 [ACK] Seq=549 Ack=1349 Win=0 Len=0
103	3.420764	20.103.180.120	172.16.204.151	TLSv1.2	1509	Application Data

Frame 103: 1509 bytes on wire (12072 bits), 1509 bytes captured (12072 bits) on interface \Device\NPF_{CE37F9D2-B0D4-4642-84DE-A14D496317A5}, id 0

Ethernet II, Src: PCSSystemtec_c1:96:0d (08:00:27:c1:96:0d), Dst: PCSSystemtec_76:75:44 (08:00:27:76:75:44)

Internet Protocol Version 4, Src: 20.103.180.120, Dst: 172.16.204.151

Transmission Control Protocol, Src Port: 443, Dst Port: 53792, Seq: 7146, Ack: 3396, Len: 1455

Transport Layer Security

TLsv1.2 Record Layer: Application Data Protocol: Hypertext Transfer Protocol

Content Type: Application Data (23)

Version: TLS 1.2 (0x0303)

Length: 1450

Encrypted Application Data [truncated]: 0000000000000001d9de20b8965b4893a0312866e4acfaeccc5a5fd8f2ac8229fc9c9f0f95b7889ee645351f75ce164d0087ec1f45a405d43ad01bd76361b8cf4b6b386501

[Application Data Protocol: Hypertext Transfer Protocol]

En ambos casos se repite la situación del [caso inicial](#) y la [página cifrada](#) de Windows Server. Con los mismos 5 paquetes consensuando el cifrado y transmitiendo la página encriptada. Y en el caso de web1 vemos además que hay un intercambio de credenciales pero ésta está cifrada

Comprobar la redirección

· <http://web1.dominio204.local>

→	32	0.188659	172.16.204.151	172.16.204.202	HTTP	484	GET / HTTP/1.1
→	34	0.189199	172.16.204.202	172.16.204.151	HTTP	616	HTTP/1.1 302 Found (text/html)

Frame 34: 616 bytes on wire (4928 bits), 616 bytes captured (4928 bits) on interface \Device\NPF_{CE37F9D2-Ethernet II, Src: PCSSystemtec_c1:96:0d (08:00:27:c1:96:0d), Dst: PCSSystemtec_76:75:44 (08:00:27:76:75:44) Internet Protocol Version 4, Src: 172.16.204.202, Dst: 172.16.204.151 Transmission Control Protocol, Src Port: 85, Dst Port: 53948, Seq: 1, Ack: 431, Len: 562 Hypertext Transfer Protocol Line-based text data: text/html (9 lines) <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">\n <html><head>\n <title>302 Found</title>\n </head><body>\n <h1>Found</h1>\n <p>The document has moved here.</p>\n <hr>\n <address>Apache/2.4.41 (Ubuntu) Server at web1.dominio204.local Port 85</address>\n </body></html>\n
--

Vemos que como respuesta a la solicitud de la página se devuelve un código 302, que nos indica que seremos redireccionados, si vemos detalladamente ese mensaje podemos ver las páginas origen y destino de la redirección, y sus puertos, posteriormente se procede a la carga de la página de manera cifrada, como ya estamos familiarizados.

Comprobar página de error

· <http://www.dominio204.local/noexiste.html>

	Time	Source	Destination	Protocol	Length	Info
	4	0.017303	172.16.204.151	172.16.204.202	HTTP	493 GET /noexiste.html HTTP/1.1
	6	0.018352	172.16.204.202	172.16.204.151	HTTP	647 HTTP/1.1 404 Not Found (text/html)

Frame 6: 647 bytes on wire (5176 bits), 647 bytes captured (5176 bits) on interface \Device\NPF_{CE37F9D2-B0D4-46-Ethernet II, Src: PCSSystemtec_c1:96:0d (08:00:27:c1:96:0d), Dst: PCSSystemtec_76:75:44 (08:00:27:76:75:44) Internet Protocol Version 4, Src: 172.16.204.202, Dst: 172.16.204.151 Transmission Control Protocol, Src Port: 80, Dst Port: 54014, Seq: 1, Ack: 440, Len: 593 Hypertext Transfer Protocol Line-based text data: text/html (5 lines) <!DOCTYPE html>\n <html><head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>\n <title>ERROR 404</title>\n <style>body {color: #454851; font-family:sans-serif; background-color:#EE6352; margin:20px;}</style>\n </head><body><h1>ERROR 404: Página no encontrada 😢</h1></body></html>\n

Vemos que como respuesta a la solicitud de la página se devuelve un código 404, que nos indica que la página no pudo ser encontrada, el servidor procede entonces a devolvernos la página de error que habíamos configurado en un ejercicio anterior. Cuyo contenido íntegro podemos visualizar si desplegamos los datos del paquete que devuelve el servidor.