



# AEC3 – Redes de Ordenadores

ISMAEL HERNÁNDEZ CLEMENTE

# Descripción

## Objetivo:

Instalar un servidor WEB y un servidor DNS en la misma máquina servidora, realizar una navegación WEB desde una máquina cliente mediante Firefox y analizar, con Wireshark, las cabeceras TCP, UDP e IP de las consultas DNS y del tráfico WEB.

## Requisitos previos:

- Dos máquinas virtuales con **Ubuntu** correctamente configuradas.
- Configuración de red de las máquinas virtuales como **adaptador puente**.

## Pasos para la ejecución de la práctica:

1. **Asignar roles a las máquinas virtuales:**
  - Elegir una máquina como cliente (**MV\_CLIENTE**) y otra como servidora (**MV\_SERVIDOR**).
2. **Identificar las IPs:**
  - Obtener las direcciones IP de **MV\_CLIENTE** y **MV\_SERVIDOR**.
3. **Instalar paquetes en MV\_SERVIDOR:**
  - **Apache2:** Usar la página web de prueba predeterminada.
  - **Bind9:** Configurar el servidor DNS con las siguientes características:
    - Dominio: internet.wrl.
    - Entrada en la base de datos:
      - Subdominio www.internet.wrl debe apuntar a la IP de **MV\_SERVIDOR**.
    - Configuración de TTL en 5 segundos para forzar nuevas consultas DNS.
    - Usar como base los archivos proporcionados:
      - named.conf.options,
      - named.conf.local,
      - db.u-tad.uni.
      - Modificar adecuadamente para reflejar el dominio internet.wrl.
  - Verificar la sintaxis de los archivos y recargar la configuración del servidor DNS.
4. **Captura 1:**
  - Tomar un pantallazo del archivo db.internet.wrl.
5. **Configurar DNS en MV\_CLIENTE:**
  - Establecer como DNS principal la IP de **MV\_SERVIDOR**.

## 6. Iniciar Wireshark y Firefox:

- En **MV\_CLIENTE**, iniciar Wireshark y navegar a <http://www.internet.wrl> en Firefox.
- **Captura 2:** Tomar un pantallazo mostrando la URL ingresada en el navegador.

## 7. Análisis del tráfico DNS con Wireshark:

- Filtrar el tráfico DNS y recargar varias veces la página para observar el tráfico.
- Analizar el paquete de consulta DNS y registrar:
  - **Nivel UDP:**
    - Puerto Origen.
    - Puerto Destino.
    - Protocolo de transporte.
  - **Nivel de aplicación:**
    - Query DNS.
- **Captura 3:** Incluir los pantallazos correspondientes.

## 8. Análisis del tráfico HTTP con Wireshark:

- Filtrar el tráfico HTTP y analizar el paquete de conexión inicial (**SYN**).
- Registrar:
  - **Nivel TCP:**
    - Puerto Origen.
    - Puerto Destino.
    - SeqN, AckN, Checksum.
    - Flags activas.
  - **Nivel IP:**
    - IP Origen.
    - IP Destino.
    - TTL.
- **Captura 4:** Incluir los pantallazos correspondientes.

# Realización de la Actividad Practica

## 1. Preparación inicial

- Inicialmente igual que en la AEC2 definí qué máquina virtual será cliente (MV\_CLIENTE) y cuál será servidor (MV\_SERVIDOR). Asegurando de que ambas máquinas están configuradas como Adaptador puente en las opciones de red para que se conecten a la red local y verifiqué las IPs de ambas máquinas utilizando el comando:

```
ip addr
```

## 2. Configuración de MV\_SERVIDOR

### Instalación de Apache2

Instale Apache2 en MV\_SERVIDOR:

```
sudo apt update
```

```
sudo apt install apache2 -y
```

Verifique que Apache esté funcionando accediendo a la página de prueba:

```
curl http://localhost
```

### Instalación de Bind9

1. Instale Bind9:

```
sudo apt install bind9 -y
```

2. Configure los siguientes archivos para Bind9:

**Archivo: named.conf.options**

```
sudo nano /etc/bind/named.conf.options
```

Actualice o confirme que tenga las siguientes configuraciones:

```
options {
```

```
    directory "/var/cache/bind";
```

```
    listen-on { any; };
```

```
    listen-on-v6 { none; };
```

```
    forwarders {
```

```
        8.8.8.8; // DNS de Google
```

```
    };
```

```
    dnssec-validation auto;
```

```
    auth-nxdomain no;
```

```
    allow-query { any; };
```

```
};
```

**Archivo: named.conf.local**

```
sudo nano /etc/bind/named.conf.local
```

Incluí la definición del dominio:

```
zone "internet.wrl" {  
  
    type master;  
  
    file "/etc/bind/db.internet.wrl";  
  
};
```

**Archivo: db.internet.wrl**

Cree el archivo y lo configuré con:

```
sudo nano /etc/bind/db.internet.wrl
```

Contenido del archivo (reemplacé 192.168.x.x con la IP de MV\_SERVIDOR):

```
$TTL 5  
  
@ IN SOA ns.internet.wrl. admin.internet.wrl. (  
    1 ; Serial  
    604800 ; Refresh  
    86400 ; Retry  
    2419200 ; Expire  
    5 ) ; TTL  
  
@ IN NS ns.internet.wrl.  
  
ns IN A 192.168.1.138  
  
www IN A 192.168.1.138
```

### 3. Verificación de configuración

- Comprobé la sintaxis de los archivos mediante:

```
sudo named-checkconf
```

```
sudo named-checkzone internet.wrl /etc/bind/db.internet.wrl
```

- Reinicié Bind9:

```
sudo systemctl restart bind9
```

#### 4. Configuración de MV\_CLIENTE

- Configuré MV\_CLIENTE para usar el DNS de MV\_SERVIDOR:

```
sudo nano /etc/resolv.conf
```

Añadí la línea:

```
nameserver 192.168.x.x
```

(Sustituyendo 192.168.x.x por la IP de MV\_SERVIDOR).

#### 5. Análisis de tráfico DNS

1. Inicié **Wireshark** en MV\_CLIENTE y empecé a capturar tráfico en la interfaz activa y en Firefox, accedí a:

```
http://www.internet.wrl
```

Filtros en Wireshark para capturar tráfico DNS:

```
udp.port == 53
```

3. Identifiqué y analicé los paquetes DNS:
  - **UDP:** Puerto origen, puerto destino, protocolo.
  - **Aplicación:** Query DNS.

#### 6. Análisis de tráfico HTTP

1. En Wireshark, apliqué el filtro para tráfico HTTP:

```
tcp.port == 80
```

2. Recargué la página en Firefox.
3. Por último tuve que analizar el paquete SYN inicial:
  - **TCP:** Puerto origen, puerto destino, SeqN, AckN, Checksum, flags activas.
  - **IP:** IP origen, IP destino, TTL.

## CAPTURA\_1

```
server@MV-server:~$ sudo systemctl restart bind9
server@MV-server:~$ cat /etc/bind/db.internet.wrl
$TTL      5
@         IN      SOA      ns.internet.wrl. admin.internet.wrl. (
                                1 ; Serial
                                604800 ; Refresh
                                86400 ; Retry
                                2419200 ; Expire
                                5 ) ; TTL
@         IN      NS       ns.internet.wrl.
ns        IN      A        192.168.1.138
www       IN      A        192.168.1.138
server@MV-server:~$ S
```

The image is a composite of two screenshots. The left screenshot shows a terminal window with a tmux interface. The terminal displays the command 'sudo apt-get install apache2' and its output, which includes the installation of various dependencies and the Apache2 package. The right screenshot shows a web browser window displaying the Apache2 Default Page. The page features the Ubuntu logo, the text 'Apache2 Default Page', and a large orange button that says 'It works!'. Below this, there is a 'Configuration Overview' section that explains the default configuration of Apache2 on Ubuntu, mentioning the 'apache2.conf' file and the 'ports.conf' file. The terminal window also shows the command 'sudo systemctl status apache2' and its output, which indicates that the service is active and running.

The screenshot displays the Wireshark application window titled "Capturing from enp0s3". The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help) and a toolbar with various icons for file operations, packet selection, and analysis.

A filter bar at the top shows the active filter: `udp.port == 53`. Below it, a list of captured packets is displayed:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.1.135	192.168.1.255	UDP	86	57621 → 57621 Len=44
2	0.197557457	2a0c:5a81:b303:b600::	2606:4700:4400::ac4...	TLSv1.2	125	Application Data
3	0.201898430	2606:4700:4400::ac4...	2a0c:5a81:b303:b600::	TLSv1.2	125	Application Data
4	0.201918931	2a0c:5a81:b303:b600::	2606:4700:4400::ac4...	TCP	86	46304 → 443 [ACK] Seq=40 Ac...
5	5.036950391	192.168.1.135	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
6	5.203759058	fe80::1	2a0c:5a81:b303:b600::	ICMPv6	86	Neighbor Solicitation for 2...
7	5.203802387	2a0c:5a81:b303:b600::	fe80::1	ICMPv6	78	Neighbor Advertisement 2a0c...
8	5.705311563	fe80::a00:27ff:feeb...	fe80::1	ICMPv6	86	Neighbor Solicitation for f...
9	5.706453072	fe80::1	fe80::a00:27ff:feeb...	ICMPv6	78	Neighbor Advertisement fe80...
10	6.037932602	192.168.1.135	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
11	7.037687327	192.168.1.135	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
12	8.037854448	192.168.1.135	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1

The bottom pane shows the details of the selected packet (Frame 1):

- Frame 1: 86 bytes on wire (688 bits), 86 bytes captured on interface enp0s3
- Ethernet II, Src: MicroStarINT\_d8:d6:fa (00:d8:61:f2:d8:d6:fa), Dst: Multicast (01:00:5e:00:00:01)
- Internet Protocol Version 4, Src: 192.168.1.135, Destination: 239.255.255.250
- User Datagram Protocol, Src Port: 57621, Dst Port: 57621
- Data (44 bytes)

The hex dump view shows the raw data of the packet:

```

0000 ff ff ff ff ff ff 00 d8 61 d8 d6 fa 08 00 45
0010 00 48 a0 29 00 00 80 11 15 a5 c0 a8 01 87 c0
0020 01 ff e1 15 e1 15 00 34 ba 71 53 70 6f 74 55
0030 70 30 a9 eb df cd 36 af 56 fc 00 01 00 04 48
0040 c2 03 3f 65 57 e6 8f ba c0 1b e2 75 f4 93 50
0050 5b 88 82 38 66 f9
  
```

# CAPTURA\_4

Start capturing packets

No.	Time	Source	Destination	Protocol	Length	Info
1235	3.617058242	2a02:26f0:1380:1f::...	2a0c:5a81:b303:b600...	TCP	86	[TCP ACKed unseen segment] 80 → 52626 [ACK] Seq=1 Ack=...
1236	3.617058482	2a02:26f0:1380:1f::...	2a0c:5a81:b303:b600...	TCP	86	[TCP ACKed unseen segment] 80 → 52618 [ACK] Seq=1 Ack=...
1237	3.617058542	2a02:26f0:1380:1f::...	2a0c:5a81:b303:b600...	TCP	86	[TCP ACKed unseen segment] 80 → 52606 [ACK] Seq=1 Ack=...
1238	3.617535126	2a02:26f0:1380:1f::...	2a0c:5a81:b303:b600...	TCP	86	[TCP ACKed unseen segment] 80 → 52614 [ACK] Seq=1 Ack=...
1239	3.644971244	192.168.1.137	192.168.1.138	TCP	74	38674 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_P...
1240	3.649930283	192.168.1.138	192.168.1.137	TCP	74	80 → 38674 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=...
1241	3.652266267	192.168.1.137	192.168.1.138	TCP	66	38674 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=196...
1242	3.654197275	192.168.1.137	192.168.1.138	HTTP	548	GET / HTTP/1.1
1243	3.655088502	192.168.1.138	192.168.1.137	TCP	66	80 → 38674 [ACK] Seq=1 Ack=483 Win=64768 Len=0 TSval=3...
1244	3.657012822	192.168.1.138	192.168.1.137	TCP	2962	80 → 38674 [PSH, ACK] Seq=1 Ack=483 Win=64768 Len=2896...
1245	3.657013012	192.168.1.138	192.168.1.137	HTTP	630	HTTP/1.1 200 OK (text/html)
1246	3.657039842	192.168.1.137	192.168.1.138	TCP	66	38674 → 80 [ACK] Seq=483 Ack=2897 Win=64128 Len=0 TSva...
1247	3.657118882	192.168.1.137	192.168.1.138	TCP	66	38674 → 80 [ACK] Seq=483 Ack=3461 Win=64128 Len=0 TSva...
1248	8.662934648	192.168.1.138	192.168.1.137	TCP	66	80 → 38674 [FIN, ACK] Seq=3461 Ack=483 Win=64768 Len=0...
1249	8.663305743	192.168.1.137	192.168.1.138	TCP	66	38674 → 80 [FIN, ACK] Seq=483 Ack=3462 Win=64128 Len=0...
1250	8.664484711	192.168.1.138	192.168.1.137	TCP	66	80 → 38674 [ACK] Seq=3462 Ack=484 Win=64768 Len=0 TSva...

40: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface enp0s3, id...  
II, Src: PCSSystemtec\_9b:fc:77 (08:00:27:9b:fc:77), Dst: PCSSystemtec\_eb:ad:65 (08:00:27:9b:fc:77)  
Protocol Version 4, Src: 192.168.1.138, Dst: 192.168.1.137  
sion Control Protocol, Src Port: 80, Dst Port: 38674, Seq: 0, Ack: 1, Len: 0

0000 08 00 27 eb ad 65 08 00 27 9b fc 77 08 00  
0010 00 3c 00 00 40 00 40 06 b6 58 c0 a8 01 8a  
0020 01 89 00 50 97 12 75 b2 5a ee f1 e6 99 48  
0030 fe 88 56 ac 00 00 02 04 05 b4 04 02 08 0a  
0040 e8 33 74 d5 4b 1e 01 03 03 07