Guía completa: Montar un servidor DNS local con CoreDNS en Docker (Windows)

# 1. Crear directorio principal y archivos base

Desde PowerShell, crear la estructura del proyecto para CoreDNS:

```powershell  
mkdir coredns-dns-local  
cd coredns-dns-local  
mkdir zones  
New-Item -ItemType File -Name "Corefile"  
New-Item -ItemType File -Name "zones\buenosdias.zone"  
```

Esto crea el entorno de trabajo necesario con una subcarpeta `zones` donde irá la zona DNS personalizada.

# 2. Contenido del archivo Corefile

Archivo `Corefile`:

```text  
. {  
 forward . 8.8.8.8  
 log  
 errors  
}

buenosdias.com {  
 file /zones/buenosdias.zone buenosdias.com  
 log  
}  
```

Este archivo indica a CoreDNS que reenvíe cualquier dominio no gestionado a Google DNS y que procese `buenosdias.com` con un archivo local.

# 3. Contenido del archivo de zona DNS (buenosdias.zone)

Archivo `zones\buenosdias.zone`:

```text  
$ORIGIN buenosdias.com.  
$TTL 3600

@ IN SOA ns1.buenosdias.com. admin.buenosdias.com. (  
 2025061201 ; serial  
 3600 ; refresh  
 1800 ; retry  
 604800 ; expire  
 86400 ) ; minimum

IN NS ns1.buenosdias.com.

ns1 IN A 192.168.8.149  
mail IN A 192.168.8.149  
api IN CNAME mail  
@ IN MX 10 mail.buenosdias.com.  
@ IN TXT "v=spf1 a mx ~all"  
```

Este archivo define el dominio, el servidor principal (SOA), y los registros A, MX, TXT y CNAME.

# 4. Eliminar BOM UTF-8 del archivo de zona

Windows a veces guarda archivos en UTF-8 con BOM, lo que impide que CoreDNS los lea correctamente.  
Este comando reescribe el archivo en UTF-8 sin BOM:

```powershell  
$utf8NoBom = New-Object System.Text.UTF8Encoding($False)  
$path = Join-Path -Path $PWD -ChildPath "zones\buenosdias.zone"  
[System.IO.File]::WriteAllLines($path, (Get-Content $path), $utf8NoBom)  
```

# 5. Crear una imagen de CoreDNS con depuración (Dockerfile-debug)

Crear un archivo `Dockerfile-debug` con este contenido:

```Dockerfile  
FROM alpine

RUN apk add --no-cache curl bind-tools coreutils

ADD https://github.com/coredns/coredns/releases/download/v1.11.1/coredns\_1.11.1\_linux\_amd64.tgz /tmp/  
RUN tar -xzf /tmp/coredns\_1.11.1\_linux\_amd64.tgz -C /usr/bin && chmod +x /usr/bin/coredns

COPY Corefile /Corefile  
COPY zones /zones

ENTRYPOINT ["/usr/bin/coredns"]  
CMD ["-conf", "/Corefile"]  
```

Esta imagen permite entrar al contenedor, ejecutar comandos como `dig` y verificar los archivos directamente.

# 6. Construir la imagen Docker personalizada

```powershell  
docker build -f Dockerfile-debug -t coredns-debug .  
```

# 7. Crear docker-compose.yml con la imagen personalizada

```yaml  
version: '3.9'

services:  
 coredns:  
 image: coredns-debug  
 container\_name: coredns  
 ports:  
 - "53:53/udp"  
 - "53:53/tcp"  
 command: -conf /Corefile  
```

Esto configura un servicio que expone CoreDNS por UDP y TCP en el puerto 53.

# 8. Comandos finales para levantar el servidor y ver logs

Levantar el servicio:

```powershell  
docker compose up -d  
```

Verificar que está corriendo:

```powershell  
docker ps  
```

Ver logs del contenedor:

```powershell  
docker logs coredns  
```

**9. Probar los registros DNS desde un contenedor auxiliar (dig)**

Una vez construido el sistema con la imagen coredns-debug y levantado el servicio con docker compose, es posible verificar el correcto funcionamiento de los registros DNS utilizando dig desde un contenedor auxiliar.

Esto permite hacer pruebas reales sin instalar herramientas externas en el sistema operativo anfitrión.

**🔹 Paso 1: Levantar CoreDNS con Docker Compose**

Desde la carpeta del proyecto:

powershell

CopiarEditar

docker compose up -d

Esto lanza el contenedor coredns con CoreDNS en marcha, escuchando en el puerto 53 TCP/UDP de localhost.

**🔹 Paso 2: Lanzar un contenedor de pruebas con shell (sh) y dig**

Como el contenedor coredns-debug incluye herramientas como sh y dig, se puede reutilizar para realizar consultas DNS. Se lanza con la red compartida del contenedor principal para que pueda comunicarse con el CoreDNS activo.

powershell

CopiarEditar

docker run --rm -it --entrypoint sh --network container:coredns coredns-debug

Este comando abre una terminal dentro del contenedor auxiliar, en la misma red que coredns, lo que permite hacer consultas a 127.0.0.1.

**🔹 Paso 3: Consultar registros con dig**

Desde la shell del contenedor, puedes probar registros como:

sh

CopiarEditar

dig @127.0.0.1 buenosdias.com MX

dig @127.0.0.1 mail.buenosdias.com A

dig @127.0.0.1 api.buenosdias.com CNAME

dig @127.0.0.1 buenosdias.com TXT

dig @127.0.0.1 buenosdias.com SOA

Si todo está correctamente configurado, las respuestas mostrarán los registros definidos en buenosdias.zone.

## 10. Configurar Redirección DNS en Windows

* **Al empezar:**
  + Powershell (admin): Set-DnsClientServerAddress -InterfaceAlias "Wi-Fi" -ServerAddresses ("127.0.0.1", "1.1.1.1")
* **Al acabar:**
  + Set-DnsClientServerAddress -InterfaceAlias "Wi-Fi" -ResetServerAddresses
* **Comprobar configuración:**
  + Get-DnsClientServerAddress -AddressFamily IPv4 | Where-Object {$\_.InterfaceAlias -eq "Wi-Fi"}
* **Comprobar redirección:**
  + nslookup buenosdias.com
* Si falla nslookup (a veces ignora configuraciones por defecto del SO):
  + nslookup buenosdias.com 127.0.0.1