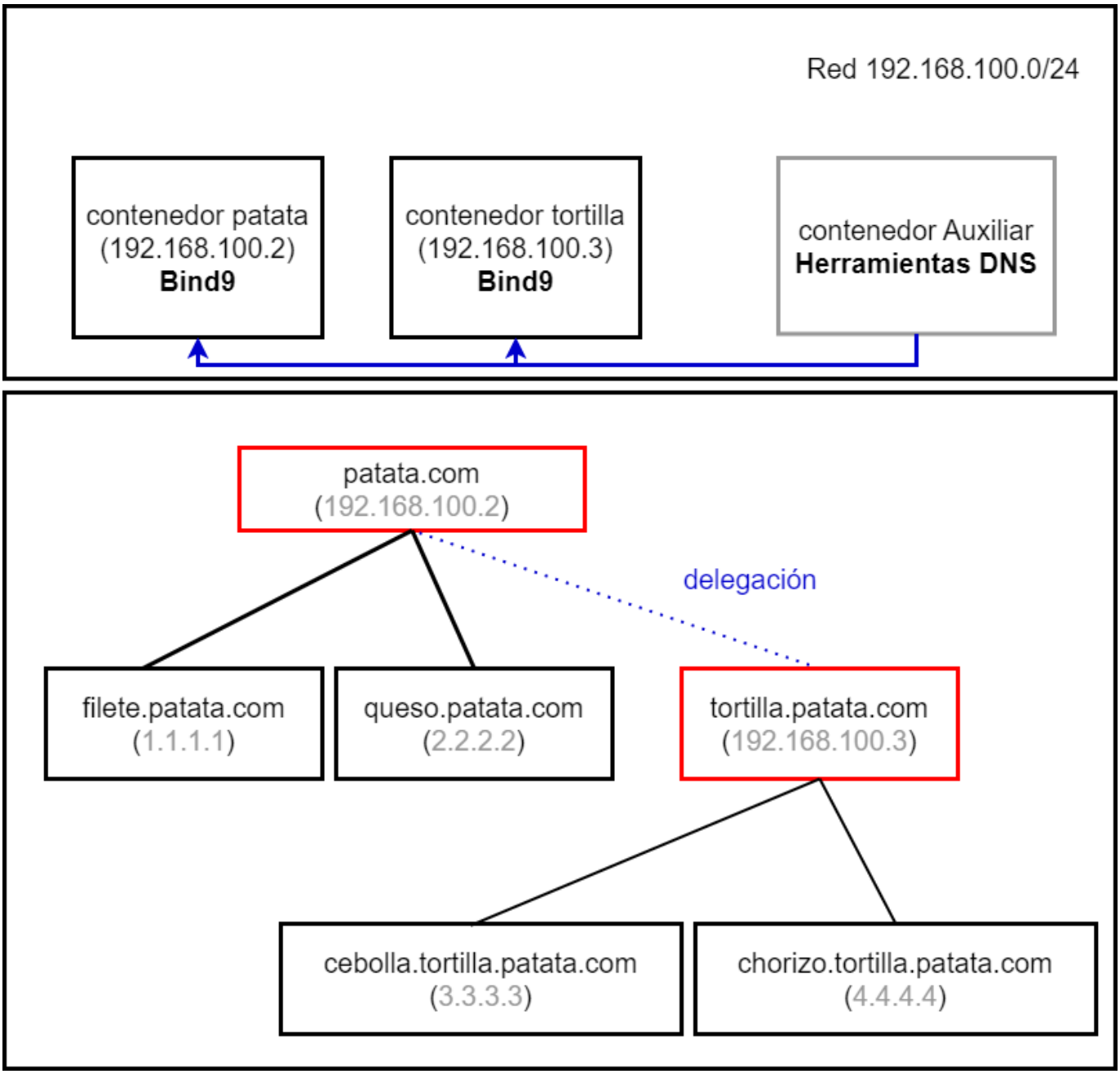
Delegacion de Zona

Configuración



Para ilustrar la **delegación de una zona** vamos a crear dos contenedores de **docker** con **Bind9**.

Al primer contenedor lo llamaremos "**patata**" y al segundo "**tortilla**".

El servidor **patata** será el servidor primario de la zona **patata.com** y delegará la zona **tortilla.patata.com** al servidor **tortilla**.

- **Servidor patata:**
  - patata.com resolverá a la IP del **contenedor patata** que será **192.168.100.2**.
  - filete.patata.com resolverá a la IP 1.1.1.1.
  - queso.patata.com resolverá a la IP 2.2.2.2.
  - tortilla.patata.com estará delegado al servidor **tortilla**.
- **Servidor tortilla:**
  - tortilla.patata.com resolverá a la IP del **contenedor tortilla** que será **192.168.100.3**.
  - cebolla.tortilla.patata.com resolverá a la IP 3.3.3.3.

- chorizo.tortilla.patata.com resolverá a la IP 4.4.4.4.

Las IPs a las que resuelven los subdominios no son relevantes, están puestas a modo de ejemplo y para comprobar con las **herramientas dns** como **dig** que los servidores están haciendo correctamente su trabajo.

¡Comenzamos!

Partimos de una máquina virtual **Ubuntu 22.04LTS** a la que le hemos instalado **docker** siguiendo esta guía.

Nos identificamos como **super usuario** (root) para evitar tener que escribir *sudo* delante de cada comando.

```
sudo -i
```

Imagen de Bind9 y documentación

Si en la web de Docker Hub buscamos por **Bind9**, podemos encontrar una imagen oficial de Bind9 con Ubuntu creada por Canonical. Utilizaremos esta imagen para nuestros contenedores.

Es interesante echar un vistazo a la documentación de la imagen, ya que en ella encontraremos ejemplos de uso. En especial, a la sección **parameters**.

Obteniendo los archivos de configuración

Vamos a crear un contenedor del cual obtendremos los archivos de configuración.

```
docker run -d --name bind9 ubuntu/bind9
```

Podemos ver que el contenedor está ejecutándose con el comando:

```
docker ps
```

Ahora copiamos la carpeta con los archivos de configuración del contenedor a nuestro equipo host. La carpeta en el contenedor se encuentra en la ruta **/etc/bind**.

```
docker cp bind9:/etc/bind/ config
```

En nuestro equipo host aparecerá un directorio llamado **config** con los archivos de configuración de bind.

Comando:

```
ls config
```

Salida:

```
bind.keys  db.0  db.127  db.255  db.empty  db.local  named.conf
named.conf.default-zones  named.conf.local  named.conf.options  rndc.key
zones.rfc1918
```

Estos archivos los utilizaremos como base para configurar nuestros contenedores.

Así que creamos dos carpetas **patata-config** y **tortilla-config** y copiamos los archivos del directorio config a cada una de ellas:

```
cp -r config patata-config
cp -r config tortilla-config
```

Configuración de zonas y RR

Contenedor patata

Editamos con **nano** el archivo **patata-config/named.conf.local** y declaramos la zona **patata.com**:

```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "patata.com" {
        type master;
        file "/etc/bind/db.patata.com";
};
```

Y ahora tenemos que crear el archivo de zona **db.patata.com** en la carpeta **patata-config** con el contenido:

```
;
; patata.com
;

$TTL    604800
$ORIGIN patata.com.
@       IN      SOA     ns.patata.com. root.patata.com. (
                     2022112201         ; Serial
                         604800         ; Refresh
```

```
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL

@       IN      NS      ns.patata.com.
@       IN      A       192.168.100.2
ns      IN      A       192.168.100.2

filete  IN      A       1.1.1.1
queso   IN      A       2.2.2.2

; ----------------------------------------
; Subdominio delegado: tortilla.patata.com
; ----------------------------------------

$ORIGIN tortilla.patata.com.

@       IN      NS      ns.tortilla.patata.com.
ns      IN      A       192.168.100.3
```

Contenedor tortilla

Editamos con **nano** el archivo **tortilla-config/named.conf.local** y declaramos la zona **tortilla.patata.com**:

```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "tortilla.patata.com" {
        type master;
        file "/etc/bind/db.tortilla.patata.com";
};
```

Y ahora tenemos que crear el archivo de zona **db.tortilla.patata.com** en la carpeta **tortilla-config** con el contenido:

```
 tortilla.patata.com
;

$TTL    604800
$ORIGIN tortilla.patata.com.
@       IN      SOA     ns.tortilla.patata.com. root.tortilla.patata.com. (
```

```
                      2022112201         ; Serial
                          604800         ; Refresh
                           86400         ; Retry
                         2419200         ; Expire
                          604800 )       ; Negative Cache TTL

@       IN      NS      ns.tortilla.patata.com.
@       IN      A       192.168.100.3
ns      IN      A       192.168.100.3

cebolla IN      A       3.3.3.3
chorizo IN      A       4.4.4.4
```

Creando una red para nuestros contenedores

Necesitamos **crear una red en Docker** para asignar **IPs** a nuestros contenedores. Ya que estamos hablando de patatas y tortilla, le llamaremos **red-cocina**.

```
docker network create --subnet=192.168.100.0/24 red-cocina
```

Arrancando nuestros contenedores

Ahora podemos arrancar nuestros contenedores con los comandos:

```
docker run -d --name contenedor-patata --net red-cocina --ip 192.168.100.2 -v
$PWD/patata-config:/etc/bind/ ubuntu/bind9
docker run -d --name contenedor-tortilla --net red-cocina --ip 192.168.100.3 -v
$PWD/tortilla-config:/etc/bind/ ubuntu/bind9
```

Podemos comprobar que nuestros contenedores están corriendo con el comando:

```
docker ps
```

En el caso de que alguno no esté corriendo, probablemente se deba a un **error en algún fichero de configuración**.

En ese caso, debremos borrar el contenedor, corregir el error, y volver a lanzarlo.

```
docker rm <nombre-contenedor>
```

Consultando a nuestros servidores DNS

Consultando desde otro contenedor

Para realizar consultas a nuestros servidores DNS utilizaremos contenedor de Docker basado en una imagen que cuenta con las **herramientas de consulta**.

En **Docker Hub** encontré la imagen **tutum/dnsutils** que incluye las herramientas **dig** y **nslookup**.

Consultas básicas

- Consultamos a nuestro servidor **patata** por **patata.com**:

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup patata.com 192.168.100.2
```

Salida:

```
Server:     192.168.100.2
Address:    192.168.100.2#53

Name:   patata.com
Address: 192.168.100.2
```

Vemos que está resolviendo correctamente el nombre y que además es una respuesta autoritativa, ya que le hemos preguntado por su zona.

- Le preguntamos ahora por **filete.patata.com**.

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup filete.patata.com
192.168.100.2
```

Salida:

```
Server:     192.168.100.2
Address:    192.168.100.2#53

Name:   filete.patata.com
Address: 1.1.1.1
```

Y vemos que nos la resuelve a **1.1.1.1** como especificamos en su archivo de configuración.

- Le preguntamos ahora por **queso.patata.com**.

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup queso.patata.com
192.168.100.2
```

Salida:

```
Server:     192.168.100.2
Address:    192.168.100.2#53

Name:   queso.patata.com
Address: 2.2.2.2
```

Y vemos que está resolviéndola a **2.2.2.2** como era esperado.

- Si le preguntamos al **contenedor-tortilla** por **tortilla.patata.com**, **cebolla.tortilla.patata.com** y **chorizo.tortilla.patata.com** nos responderá con las IPs **192.168.100.3**, **3.3.3.3** y **4.4.4.4** respectivamente:

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup tortilla.patata.com
192.168.100.3
```

Salida:

```
Server:     192.168.100.3
Address:    192.168.100.3#53

Name:   tortilla.patata.com
Address: 192.168.100.3
```

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup
cebolla.tortilla.patata.com 192.168.100.3
```

Salida:

```
Server:     192.168.100.3
Address:    192.168.100.3#53
```

```
Name:   cebolla.tortilla.patata.com
Address: 3.3.3.3
```

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup
chorizo.tortilla.patata.com 192.168.100.3
```

Salida:

```
Server:     192.168.100.3
Address:    192.168.100.3#53

Name:   chorizo.tortilla.patata.com
Address: 4.4.4.4
```

Comprobando la resolución desde el contenedor patata

Ahora si preguntamos a nuestro servidor del **contenedor patata** por el nombre **tortilla.patata.com** debería ser capaz de resolverlo a **192.168.100.3**, gracias al registro **de pegamento** o registro **glue**.

Este **registro A/AAAA adicional** permite la correcta resolución en la dirección IP del servidor DNS.

```
... (Archivo de zona del servidor patata.com: db.patata.com)

$ORIGIN tortilla.patata.com.
@       IN      NS      ns.tortilla.patata.com. ; Servidor DNS de la zona
tortilla.patata.com.
ns      IN      A       192.168.100.3           ; Este es el registro de pegamento
```

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup tortilla.patata.com
192.168.100.2
```

Salida:

```
Server:     192.168.100.2
Address:    192.168.100.2#53

Non-authoritative answer:
Name:   tortilla.patata.com
Address: 192.168.100.3
```

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup
chorizo.tortilla.patata.com 192.168.100.2
```

Salida:

```
Server:     192.168.100.2
Address:    192.168.100.2#53

Non-authoritative answer:
Name:   chorizo.tortilla.patata.com
Address: 4.4.4.4
```

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup
cebolla.tortilla.patata.com 192.168.100.2
```

Salida:

```
Server:     192.168.100.2
Address:    192.168.100.2#53

Non-authoritative answer:
Name:   cebolla.tortilla.patata.com
Address: 3.3.3.3
```

Vemos que el servidor **patata** es capaz de resolver preguntas sobre el dominio **tortilla.patata.com**.

Comprobando la resolución desde el contenedor tortilla

Si intentamos resolver **patata.com** desde el contenedor **tortilla** veremos un resultado inesperado:

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup patata.com 192.168.100.3
```

Salida:

```
Server:     192.168.100.3
Address:    192.168.100.3#53

Non-authoritative answer:
Name:   patata.com
Address: 207.194.137.117
```

Sería una respuesta parecida a poner en nuestro equipo host:

Comando:

```
nslookup patata.com
```

Salida:

```
Server:     127.0.0.53
Address:    127.0.0.53#53

Non-authoritative answer:
Name:   patata.com
Address: 207.194.137.117
```

El motivo es que el servidor tortilla no tiene conocimiento del servidor patata, por lo que lo resuelve desde los **servidores raíz**. Y por eso obtenemos la resolución del dominio existente **patata.com** y no de la zona de nuestro **contenedor patata** con IP **192.168.100.2**.

Esto podemos comprobarlo poniendo:

Comando:

```
docker run -it --net red-cocina tutum/dnsutils nslookup queso.patata.com
192.168.100.3
```

Salida:

```
Server:     192.168.100.3
Address:    192.168.100.3#53

** server can't find queso.patata.com: NXDOMAIN
```

Vemos que el servidor responsable de tortilla.patata.com no sabe cómo resolver el subdominio **queso.patata.com**, ya que nuestra zona **patata.com** no cuelga de los servidores raíz y además está consultando al servidor responsable de la zona **patata.com** de Internet.