

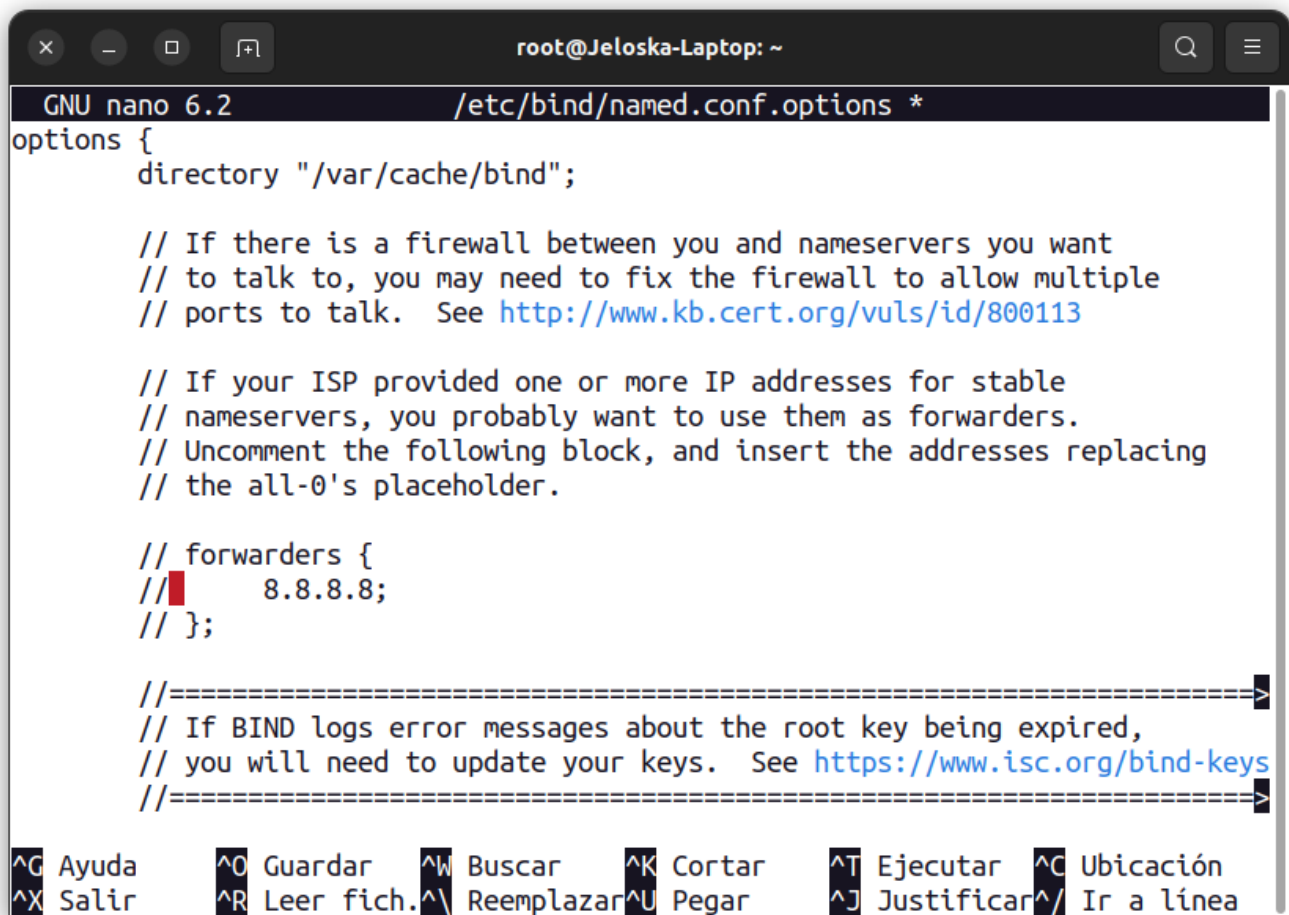
Laboratorio 5

Jeloska Isabel Chavez Paredes

Item 1: Configuración de DNS como servidor caché

Para establecer el DNS como servidor caché, primero modificamos el archivo

`/etc/bind/named.conf.options`. Agregamos el siguiente bloque de código, en este caso utilizando el DNS de Google: `plaintext forwarders { 8.8.8.8; };`



```
root@Jeloska-Laptop: ~
GNU nano 6.2 /etc/bind/named.conf.options *
options {
    directory "/var/cache/bind";

    // If there is a firewall between you and nameservers you want
    // to talk to, you may need to fix the firewall to allow multiple
    // ports to talk. See http://www.kb.cert.org/vuls/id/800113

    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.

    // forwarders {
    //      8.8.8.8;
    // };

    //=====
    // If BIND logs error messages about the root key being expired,
    // you will need to update your keys. See https://www.isc.org/bind-keys
    //=====
^G Ayuda      ^O Guardar   ^W Buscar    ^K Cortar    ^T Ejecutar  ^C Ubicación
^X Salir      ^R Leer fich.^_ Reemplazar  ^U Pegar     ^J Justificar^_/ Ir a línea
```

Una vez realizada esta configuración, es necesario reiniciar el servicio DNS para aplicar los cambios. Esto se logra ejecutando el siguiente comando:

```
sudo systemctl restart bind9
```

Para verificar la efectividad de esta configuración y medir el tiempo de consulta, utilizamos el comando `dig`, el cual se instala con el paquete `dnsutils`. A continuación, se muestran los resultados de dos consultas de tiempo, donde se puede observar que inicialmente el tiempo de consulta era de 39ms y posteriormente se redujo a 0ms.

```
root@Jeloska-Laptop:~# nano /etc/bind/named.conf.options
root@Jeloska-Laptop:~# systemctl restart bind9
root@Jeloska-Laptop:~# dig google.com
```

```
root@Jeloska-Laptop:~# dig google.com
; <>> DiG 9.18.18-0ubuntu0.22.04.2-Ubuntu <>> google.com
```

```

; <<>> DiG 9.18.18-0ubuntu0.22.04.2-Ubuntu <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 17742
;; flags: qr rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;google.com.                IN      A
;; ANSWER SECTION:
google.com. 99      IN      A      64.233.186.102
google.com. 99      IN      A      64.233.186.113
google.com. 99      IN      A      64.233.186.138
google.com. 99      IN      A      64.233.186.100
google.com. 99      IN      A      64.233.186.139
google.com. 99      IN      A      64.233.186.101
;; Query time: 39 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Thu Mar 28 03:22:42 -04 2024
;; MSG SIZE rcvd: 135

;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 17742
;; flags: qr rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;google.com.                IN      A
;; ANSWER SECTION:
google.com. 95      IN      A      64.233.186.138
google.com. 95      IN      A      64.233.186.102
google.com. 95      IN      A      64.233.186.100
google.com. 95      IN      A      64.233.186.101
google.com. 95      IN      A      64.233.186.139
google.com. 95      IN      A      64.233.186.113
;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53) (UDP)
;; WHEN: Thu Mar 28 03:22:46 -04 2024
;; MSG SIZE rcvd: 135

```

Este cambio demuestra cómo la configuración del servidor DNS como caché puede mejorar significativamente los tiempos de consulta, optimizando el rendimiento y la eficiencia del sistema de nombres de dominio.

Item 2: Configuración DNS primario

1. Verificación y Validación del Archivo de Configuración Bind9:

- **Paso 1:** Abrimos el archivo `/etc/bind/named.conf` utilizando el editor nano para verificar que las líneas `include "/etc/bind/named.conf.options";`, `include "/etc/bind/named.conf.local";` y `include "/etc/bind/named.conf.default-zones";` no estén comentadas, lo que asegura que las configuraciones en esos archivos se estén incluyendo correctamente en la configuración global de Bind9.

```

GNU nano 6.2 /etc/bind/named.conf
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind9/README.Debian.gz for information on the
// structure of BIND configuration files in Debian, *BEFORE* you customize
// this configuration file.
//
// If you are just adding zones, please do that in /etc/bind/named.conf.local

include "/etc/bind/named.conf.options";
include "/etc/bind/named.conf.local";
include "/etc/bind/named.conf.default-zones";

```

Se verificó que las líneas mencionadas no están comentadas en el archivo de configuración `named.conf`.

2. Agregando una Entrada de Zona en el Archivo `named.conf.local`:

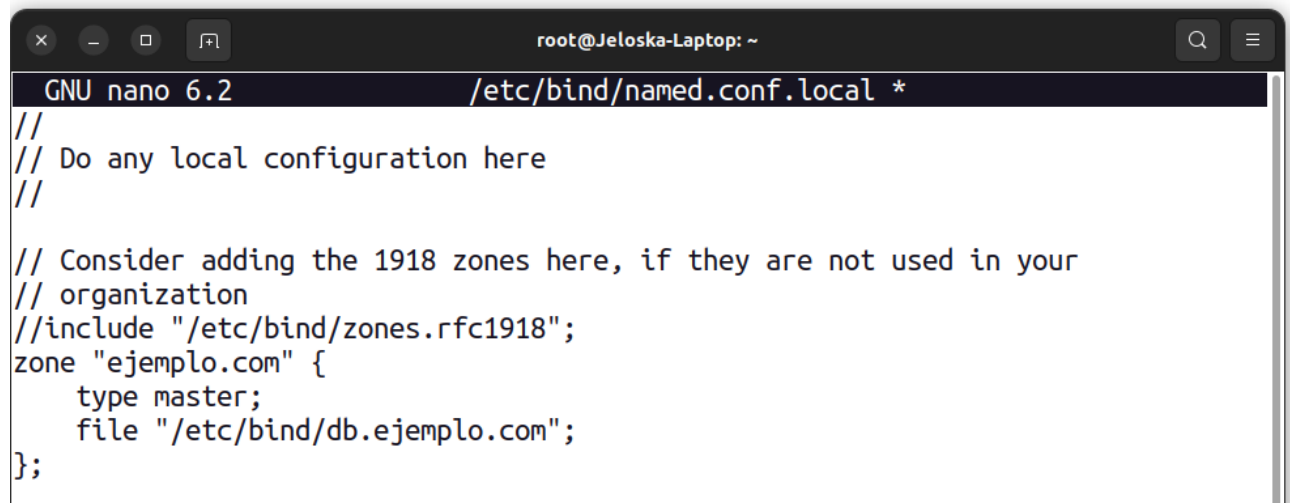
Agregamos una entrada de zona para nuestro dominio `ejemplo.com` en el archivo `/etc/bind/named.conf.local`. Para ello, añadimos las siguientes líneas:

```

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

```

Estas líneas indican que la zona `ejemplo.com` será manejada como una zona maestra (`type master`) y que los datos de esa zona se encuentran en el archivo `/etc/bind/db.ejemplo.com`. De esta manera, Bind9 reconocerá y gestionará la zona `ejemplo.com` de acuerdo con la configuración en el archivo de zona correspondiente.

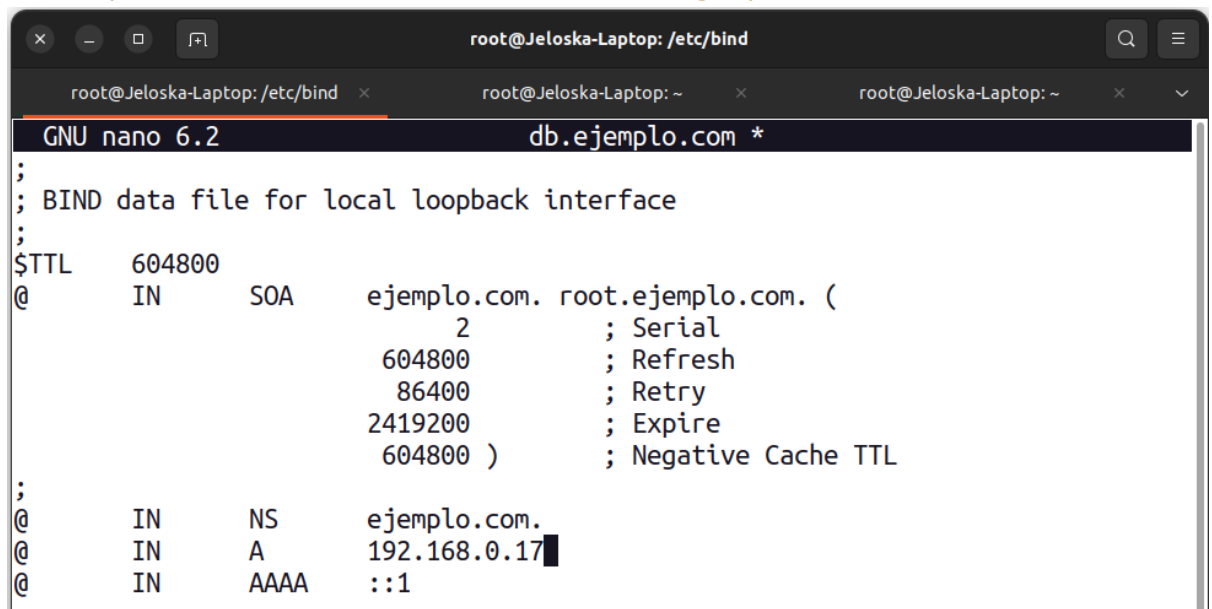


```
root@Jeloska-Laptop: ~
GNU nano 6.2 /etc/bind/named.conf.local *
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
zone "ejemplo.com" {
    type master;
    file "/etc/bind/db.ejemplo.com";
};
```

3. Creación y Edición del Archivo de Zona para ejemplo.com:

- **Paso 1:** Copiamos la plantilla de archivo de zona `db.local` para crear el archivo de zona `db.ejemplo.com` para el dominio `ejemplo.com` de esta manera preparamos el archivo para ser editado con los registros de recursos necesarios para la resolución DNS de ese dominio
`sudo cp /etc/bind/db.local /etc/bind/db.ejemplo.com`



```
root@Jeloska-Laptop: /etc/bind
GNU nano 6.2 db.ejemplo.com *
;
; BIND data file for local loopback interface
;
$TTL      604800
@         IN      SOA     ejemplo.com. root.ejemplo.com. (
                        2      ; Serial
                        604800 ; Refresh
                        86400  ; Retry
                        2419200 ; Expire
                        604800 ) ; Negative Cache TTL
;
@         IN      NS      ejemplo.com.
@         IN      A       192.168.0.17
@         IN      AAAA    ::1
```

- **Paso 2:** Abrimos el archivo `db.ejemplo.com` con Nano y efectuamos las siguientes configuraciones:

Registro SOA (Start of Authority): - Establecimos el registro SOA para definir la autoridad de la zona `ejemplo.com`. - `@ IN SOA ns1.ejemplo.com. admin.ejemplo.com. (2022032701 ...)`: - `@` representa el dominio principal `ejemplo.com`. - `IN` indica la clase de datos, que en este caso es Internet. - `SOA` especifica el tipo de registro, que es el inicio de autoridad. - `ns1.ejemplo.com.` es el nombre del servidor de nombres autoritario para la

zona. - `admin.ejemplo.com.` es la dirección de correo electrónico del administrador de la zona. - `2022032701` es el número de serie que indica la versión actual del archivo de zona. - `604800`, `86400`, `2419200` y `604800` son los parámetros de tiempo para refrescar, reintento, expiración y TTL negativo, respectivamente.

Registro NS (Name Server): - Agregamos un registro NS para designar `ns1.ejemplo.com.` como el servidor de nombres principal de la zona. - `@ IN NS ns1.ejemplo.com.`: Indica que `ns1.ejemplo.com.` es el servidor de nombres autoritario para la zona `ejemplo.com.`

Registros de Dirección (A): - Configuramos registros de dirección (A) para asignar direcciones IP a los nombres de host en la zona. - `@ IN A 192.168.1.10`: Asigna la dirección IP `192.168.1.10` al nombre de host principal `ejemplo.com.` - `www IN A 192.168.1.10`: Asigna la misma dirección IP al subdominio `www.ejemplo.com.`

4. Verificación de la Configuración de la Zona y Reinicio del Servidor DNS:

- **Paso 1:** Verificamos la sintaxis del archivo de zona `db.ejemplo.com` utilizando el comando `named-checkzone`:

```
sudo named-checkzone ejemplo.com /etc/bind/db.ejemplo.com
```

La salida del comando indica si la zona `ejemplo.com` se cargó correctamente y no presenta errores sintácticos.

- **Paso 2:** Reiniciamos el servicio Bind9 para que los cambios realizados en la configuración, como la adición de la nueva zona `ejemplo.com`, surtan efecto y se apliquen correctamente en el servidor DNS.

```
sudo systemctl restart bind9
```

```
root@Jeloska-Laptop:/etc/bind# named-checkzone ejemplo.com db.ejemplo.com
zone ejemplo.com/IN: loaded serial 2
OK
root@Jeloska-Laptop:/etc/bind# systemctl restart bind9
root@Jeloska-Laptop:/etc/bind# █
```

5. Configuración del archivo `/etc/resolv.conf`:

- **Paso 1:** Editamos el archivo `/etc/resolv.conf` para agregar la línea `nameserver` con la dirección IP del servidor DNS actual para que el sistema utilice el servidor DNS local.

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

```
GNU nano 6.2 /etc/resolv.conf *
# This is /run/systemd/resolve/stub-resolv.conf managed by man:systemd-resolved(8).
# Do not edit.
#
# This file might be symlinked as /etc/resolv.conf. If you're looking at
# /etc/resolv.conf and seeing this text, you have followed the symlink.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "resolvectl status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs should typically not access this file directly, but only
# through the symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a
# different way, replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

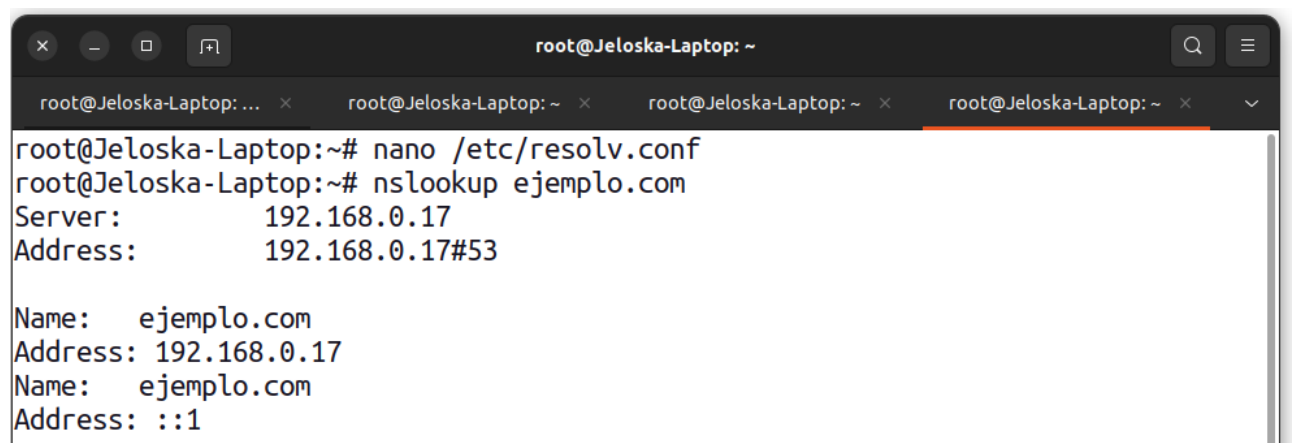
#nameserver 127.0.0.53
#options edns0 trust-ad
#search .
nameserver 192.168.0.17
```

6. Prueba de Resolución DNS utilizando nslookup:

- **Paso 1:** Ejecutamos el comando `nslookup` para verificar que la resolución DNS del dominio `ejemplo.com` se esté realizando correctamente utilizando el servidor DNS local.

```
nslookup ejemplo.com
```

La salida del comando `nslookup` muestra el servidor DNS utilizado y la dirección IP resuelta para el dominio `ejemplo.com`, confirmando que la configuración del servidor DNS primario para el dominio `ejemplo.com` es exitosa.



```
root@Jeloska-Laptop: ~
root@Jeloska-Laptop:~# nano /etc/resolv.conf
root@Jeloska-Laptop:~# nslookup ejemplo.com
Server:      192.168.0.17
Address:     192.168.0.17#53

Name:   ejemplo.com
Address: 192.168.0.17
Name:   ejemplo.com
Address: ::1
```

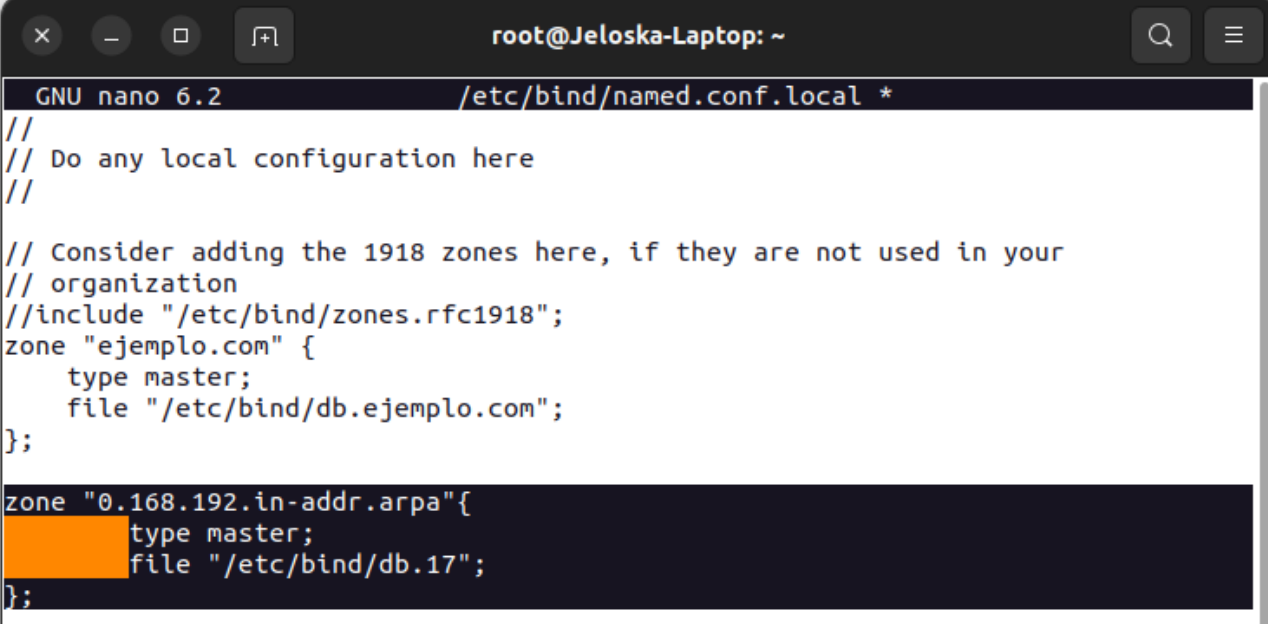
Item 3: Procedimiento para Archivo de Zona Inversa

1. Agregando una Entrada de Zona Inversa en el Archivo `named.conf.local`:

Para configurar la zona inversa en el servidor DNS Bind9, agregamos la siguiente entrada en el archivo `/etc/bind/named.conf.local`:

```
zone "0.168.192.in-addr.arpa" {  
    type master;  
    file "/etc/bind/db.17";  
};
```

Esta configuración define que la zona inversa correspondiente a la subred `192.168.0.0/24` será manejada como una zona maestra (`type master`) y que los datos de la zona inversa se encuentran en el archivo `/etc/bind/db.17`.



The screenshot shows a terminal window titled 'root@Jeloska-Laptop: ~'. Inside, the GNU nano 6.2 editor is open, editing the file '/etc/bind/named.conf.local'. The file content is as follows:

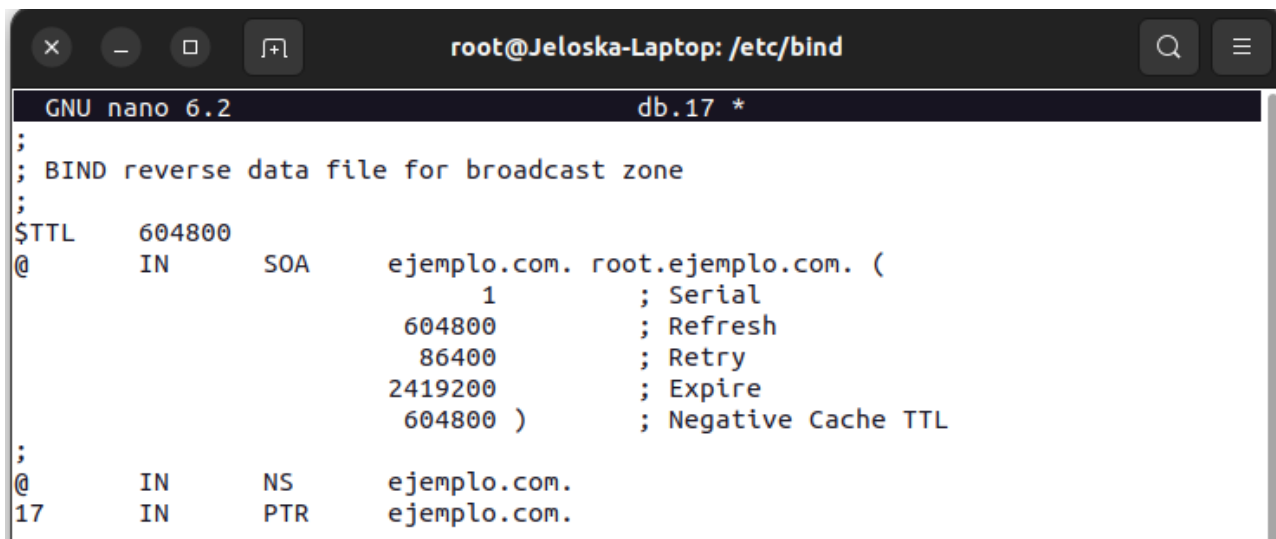
```
//  
// Do any local configuration here  
//  
  
// Consider adding the 1918 zones here, if they are not used in your  
// organization  
//include "/etc/bind/zones.rfc1918";  
zone "ejemplo.com" {  
    type master;  
    file "/etc/bind/db.ejemplo.com";  
};  
  
zone "0.168.192.in-addr.arpa"{  
    type master;  
    file "/etc/bind/db.17";  
};
```

2. Creación y Edición del Archivo de Zona Inversa `db.17`:

- **Paso 1:** Copiamos la plantilla de archivo de zona `db.255` para crear el archivo de zona inversa `db.17` correspondiente a la subred `192.168.0.0/24`:

```
sudo cp /etc/bind/db.255 /etc/bind/db.17  
![Captura de pantalla de la edición del archivo db.17](image-27.png)
```

- **Paso 2:** Abrimos el archivo `db.17` con el editor Nano para agregar los registros de recursos inversos necesarios:



```
root@Jeloska-Laptop: /etc/bind
GNU nano 6.2 db.17 *
;
; BIND reverse data file for broadcast zone
;
$TTL      604800
@         IN      SOA      ejemplo.com. root.ejemplo.com. (
                                1          ; Serial
                                604800     ; Refresh
                                86400      ; Retry
                                2419200    ; Expire
                                604800 )   ; Negative Cache TTL
;
@         IN      NS       ejemplo.com.
17        IN      PTR      ejemplo.com.
```

1. **Añadimos el Registro SOA:** Agregamos el registro **SOA** (Start of Authority) al principio del archivo para establecer la autoridad de la zona inversa. Este registro incluye información como el nombre del servidor DNS autoritario, el contacto del administrador, y otros parámetros. En nuestro caso, el registro **SOA** se ve así:

```
@         IN      SOA      ejemplo.com. root.ejemplo.com. (
                                1          ; Serial
                                604800     ; Refresh
                                86400      ; Retry
                                2419200    ; Expire
                                604800 )   ; Negative Cache TTL
```

- **@:** Representa el dominio principal de la zona inversa **0.168.192.in-addr.arpa**.
- **IN:** Indica la clase de datos, en este caso, Internet.
- **SOA:** Tipo de registro que define la autoridad de la zona.
- **ejemplo.com.:** Nombre del servidor DNS autoritario.
- **root.ejemplo.com.:** Dirección de correo electrónico del administrador de la zona.

2. **Agregamos el Registro NS:** Añadimos el registro **NS** (Name Server) para designar el servidor de nombres principal de la zona inversa. Esto se hace con la siguiente línea:

```
@         IN      NS       ejemplo.com.
```

- **@:** Representa el dominio principal de la zona inversa **0.168.192.in-addr.arpa**.
- **IN:** Indica la clase de datos.
- **NS:** Tipo de registro que especifica el servidor de nombres principal.
- **ejemplo.com.:** Nombre del servidor de nombres principal.

3. **Agregamos el Registro PTR:** Por último, añadimos un registro **PTR** (Pointer) que asocia una dirección IP con un nombre de host. En nuestro caso, la dirección IP **192.168.0.17** se asigna al nombre de host **ejemplo.com.:**


```
17      IN      PTR      ejemplo.com.
```

- **17**: El último octeto de la dirección IP invertida (**192.168.0.17** se convierte en **17.0.168.192.in-addr.arpa**).
- **IN**: Indica la clase de datos.
- **PTR**: Tipo de registro que establece una relación de puntero entre la dirección IP y el nombre de host.
- **ejemplo.com.**: Nombre de host asociado con la dirección IP **192.168.0.17**.

Estos cambios son importantes para la configuración correcta de la zona inversa en el servidor DNS Bind9, ya que definen la autoridad de la zona, el servidor de nombres principal y las asociaciones de direcciones IP con nombres de host.

3. Verificación de la Configuración de la Zona Inversa y Reinicio del Servidor DNS:

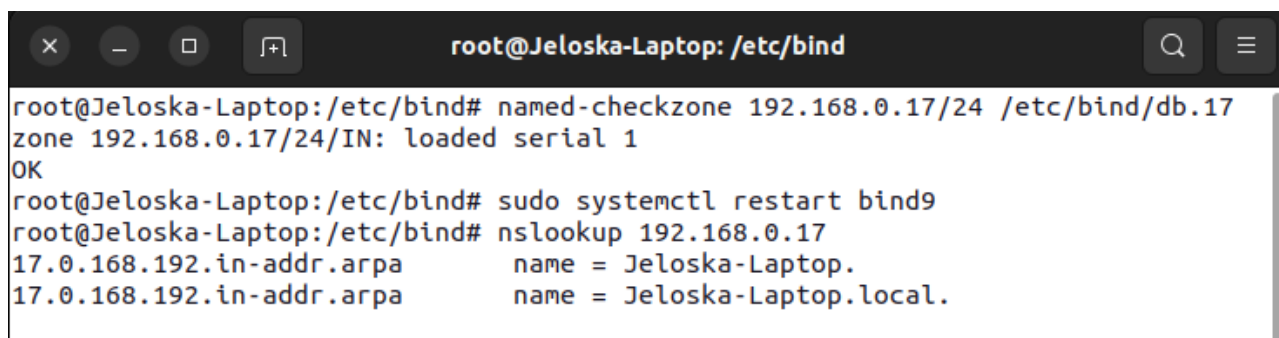
- **Paso 1:** Verificamos la sintaxis del archivo de zona inversa **db.17** utilizando el comando **named-checkzone**:

```
named-checkzone 0.168.192.in-addr.arpa /etc/bind/db.17
```

Este comando comprueba la sintaxis del archivo y nos indica si la zona inversa se cargó correctamente sin errores sintácticos.

- **Paso 2:** Reiniciamos el servicio Bind9 para aplicar los cambios en la configuración del servidor DNS:

```
sudo systemctl restart bind9
```



```
root@Jeloska-Laptop: /etc/bind
root@Jeloska-Laptop:/etc/bind# named-checkzone 192.168.0.17/24 /etc/bind/db.17
zone 192.168.0.17/24/IN: loaded serial 1
OK
root@Jeloska-Laptop:/etc/bind# sudo systemctl restart bind9
root@Jeloska-Laptop:/etc/bind# nslookup 192.168.0.17
17.0.168.192.in-addr.arpa      name = Jeloska-Laptop.
17.0.168.192.in-addr.arpa      name = Jeloska-Laptop.local.
```

4. Prueba de Resolución DNS utilizando **nslookup**:

Para verificar la resolución inversa de direcciones IP, ejecutamos el comando **nslookup** seguido de la dirección IP que queremos resolver. En nuestro caso, utilizamos la dirección IP local **192.168.0.17**.

```
nslookup 192.168.0.17
```


Este comando envía una consulta DNS al servidor DNS configurado en el sistema para obtener información inversa de la dirección IP especificada.

La salida esperada de este comando es un registro que muestra la dirección IP invertida y su correspondiente nombre de host:

```
jeloskaisabel@Jeloska-Laptop:~$ su -  
Password:  
root@Jeloska-Laptop:~# nslookup 192.168.0.17  
17.0.168.192.in-addr.arpa      name = Jeloska-Laptop.  
17.0.168.192.in-addr.arpa      name = Jeloska-Laptop.local.  
-
```

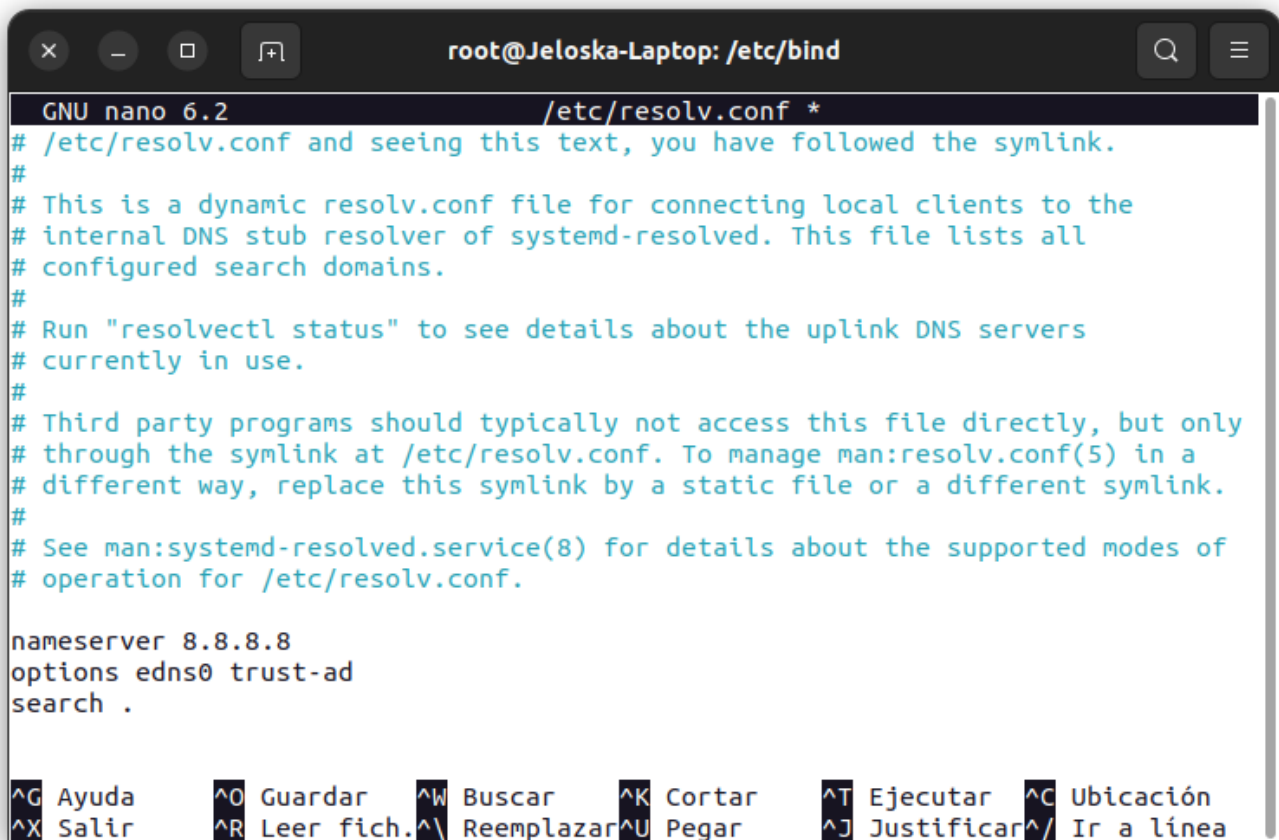
Aquí, la parte `17.0.168.192.in-addr.arpa` representa la dirección IP invertida, y `ejemplo.com` es el nombre de host asociado con la dirección IP `192.168.0.17` en la zona inversa configurada en el servidor DNS.

La captura de pantalla adjunta muestra un ejemplo de la salida obtenida al realizar esta prueba de resolución DNS utilizando `nslookup`. Esta salida confirma que la configuración de la zona inversa en el servidor DNS está funcionando correctamente al asociar la dirección IP con el nombre de host correspondiente.

Item 4: Cambiar el DNS de ISP por DNS público

1. Cambiar DNS a Google DNS:

Para cambiar la configuración del servidor DNS en Linux, comenzamos editando el archivo de configuración de red `/etc/resolv.conf`. Dentro de este archivo, localizamos la línea que comienza con `nameserver` y reemplazamos la dirección IP del servidor DNS actual con la dirección de Google DNS (`8.8.8.8`). El cambio en el archivo se reflejará de la siguiente manera:

A screenshot of a terminal window titled 'root@Jeloska-Laptop: /etc/bind'. The terminal shows the GNU nano 6.2 editor editing the file /etc/resolv.conf. The file content includes several comments explaining its purpose as a dynamic resolv.conf file for connecting local clients to the internal DNS stub resolver of systemd-resolved. It lists all configured search domains and provides instructions on how to manage it. The configuration at the bottom specifies the nameserver 8.8.8.8, options edns0 trust-ad, and a search domain of . The bottom of the terminal shows a row of keyboard shortcuts for nano: ^G Ayuda, ^O Guardar, ^W Buscar, ^K Cortar, ^T Ejecutar, ^C Ubicación, ^X Salir, ^R Leer fich., ^\ Reemplazar, ^U Pegar, ^J Justificar, ^_/ Ir a línea.

```
root@Jeloska-Laptop: /etc/bind
GNU nano 6.2 /etc/resolv.conf *
# /etc/resolv.conf and seeing this text, you have followed the symlink.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "resolvectl status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs should typically not access this file directly, but only
# through the symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a
# different way, replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

nameserver 8.8.8.8
options edns0 trust-ad
search .

^G Ayuda  ^O Guardar  ^W Buscar  ^K Cortar  ^T Ejecutar  ^C Ubicación
^X Salir  ^R Leer fich.  ^\ Reemplazar  ^U Pegar  ^J Justificar  ^_/ Ir a línea
```

1.2. Pruebas de velocidad de acceso:

Se utilizó un script en bash del apéndice A para registrar los tiempos de consulta DNS. Este script está diseñado para realizar mediciones de tiempo utilizando el comando `dig` en un entorno Linux. En primer lugar, se configura el número de veces que se ejecutará el comando `dig`, estableciendo en este caso 10 iteraciones. También se define el nombre del archivo de salida, `tiempos_consulta_dns_google.txt`, donde se almacenarán los tiempos de consulta DNS.

Una vez configuradas estas variables, el script realiza una limpieza del archivo de salida en caso de que ya exista, asegurando que los datos se guarden de manera organizada y sin contenido previo. Luego, se inicia un bucle `for` que se repetirá las 10 veces definidas anteriormente.

En cada iteración del bucle, el script ejecuta el comando `dig` para realizar una consulta DNS al dominio `google.com`, utilizando el servidor DNS de Google con la dirección IP `8.8.8.8`. El tiempo de respuesta de cada consulta se extrae utilizando herramientas como `grep` y `awk`, y se guarda en el archivo de salida en milisegundos.

Además, el script incluye una pausa de 1 segundo entre cada ejecución del comando `dig` para evitar sobrecargar el servidor DNS y obtener mediciones más precisas. Una vez finalizadas todas las iteraciones, se muestra un mensaje indicando la finalización del proceso y se informa al usuario sobre la ubicación del archivo donde se han guardado los tiempos de consulta, facilitando así su análisis posterior.

```
root@Jeloska-Laptop: /etc/bind
GNU nano 6.2                                tiempos_consulta_google.sh *
#!/bin/bash

# Número de veces que se ejecutará el comando dig
ITERACIONES=20

# Nombre del archivo de salida para guardar los tiempos de consulta
ARCHIVO_SALIDA=tiempos_consulta_dns_google.txt

# Limpiar el archivo de salida si existe previamente
> $ARCHIVO_SALIDA

# Bucle para ejecutar el comando dig y guardar los tiempos de consulta en el archivo
for ((i=1; i<=$ITERACIONES; i++)); do
    echo "Ejecución $i:"
    tiempo=$(dig google.com @8.8.8.8 | grep "Query time:" | awk '{print $4}')
    echo "Tiempo de consulta: $tiempo ms"
    echo "$tiempo" >> $ARCHIVO_SALIDA
    sleep 1 # Esperar 1 segundo entre cada ejecución
done

echo "Proceso completado. Los tiempos de consulta se han guardado en $ARCHIVO_SALIDA"
```

Los resultados obtenidos al ejecutar el script se presentan en la captura de pantalla adjunta.

```
root@Jeloska-Laptop:~# ./tiempos_consulta_google.sh
```

```
Ejecución 1:
Tiempo de consulta: 48 ms
Ejecución 2:
Tiempo de consulta: 44 ms
Ejecución 3:
Tiempo de consulta: 68 ms
Ejecución 4:
Tiempo de consulta: 120 ms
Ejecución 5:
Tiempo de consulta: 44 ms
Ejecución 6:
Tiempo de consulta: 44 ms
```

2. Cambiar DNS a Cloudflare DNS:

Para modificar la configuración del servidor DNS, primero editamos el archivo de configuración de red `/etc/resolv.conf`. En este archivo, ubicamos la línea que comienza con `nameserver` y sustituimos la dirección IP del servidor DNS actual por la dirección de Google DNS (`1.1.1.1`).

```
GNU nano 6.2                                /etc/resolv.conf *
# Run "resolvectl status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs should typically not access this file directly, but only
# through the symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a
# different way, replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

nameserver 1.1.1.1
options edns0 trust-ad
search .
```

2.1. Pruebas de velocidad de acceso:

Al igual que en la instancia anterior, utilizamos el script del apéndice A para realizar mediciones de tiempo de consulta DNS. Esta vez, el archivo de salida se denominó `tiempos_consulta_dns_cloudflare.txt`.

La línea de código empleada en el script para esta medición fue la siguiente:

```
tiempo=$(dig cloudflare.com @1.1.1.1 | grep "Query time:" | awk '{print $4}')
```

Este comando ejecuta una consulta DNS al dominio `cloudflare.com` utilizando el servidor DNS de Cloudflare (`1.1.1.1`). Luego, se extrae el tiempo de respuesta de la consulta y se guarda en el archivo de salida mencionado anteriormente.

```
GNU nano 6.2                                tiempos_consulta_cloudflare.sh *
#!/bin/bash

# Número de veces que se ejecutará el comando dig
ITERACIONES=20

# Nombre del archivo de salida para guardar los tiempos de consulta
ARCHIVO_SALIDA=tiempos_consulta_dns_cloudflare.txt

# Limpiar el archivo de salida si existe previamente
> $ARCHIVO_SALIDA

# Bucle para ejecutar el comando dig y guardar los tiempos de consulta en el archivo
for ((i=1; i<=$ITERACIONES; i++)); do
    echo "Ejecución $i:"
    tiempo=$(dig cloudflare.com @1.1.1.1 | grep "Query time:" | awk '{print $4}')
    echo "Tiempo de consulta: $tiempo ms"
    echo "$tiempo" >> $ARCHIVO_SALIDA
    sleep 1 # Esperar 1 segundo entre cada ejecución
done

echo "Proceso completado. Los tiempos de consulta se han guardado en $ARCHIVO_SALIDA"
```

En la siguiente captura se muestran los resultados obtenidos:

```
root@Jeloska-Laptop:~# ./tiempos_consulta_cloudflare.sh
Ejecución 1:
Tiempo de consulta: 44 ms
Ejecución 2:
Tiempo de consulta: 52 ms
Ejecución 3:
Tiempo de consulta: 40 ms
Ejecución 4:
Tiempo de consulta: 24 ms
```

Comparación y análisis de los resultados:

Tiempo de Respuesta de los Servidores DNS (en milisegundos)

Intento	Cloudflare	Google
1	44	48
2	52	44
3	40	68
4	24	120
5	36	44
6	24	44

Intento	Cloudflare	Google
7	28	64
8	28	76
9	44	68
10	32	44
11	28	44
12	40	48
13	28	44
14	40	48
15	44	108
16	44	60
17	40	68
18	44	72
19	44	44
20	48	44
Promedio	37.6	58.8

Análisis de los Resultados:

1. Tiempo Promedio:

- Cloudflare: 37.6 ms
- Google: 58.8 ms
- Podemos observar que, en promedio, Cloudflare tiene tiempos de respuesta más bajos en comparación con Google.

2. Variabilidad:

- Cloudflare muestra una menor variabilidad en los tiempos de respuesta, con valores que oscilan entre 24 ms y 52 ms.
- Google presenta una mayor variabilidad, con tiempos que van desde 44 ms hasta 120 ms.

3. Rendimiento Global:

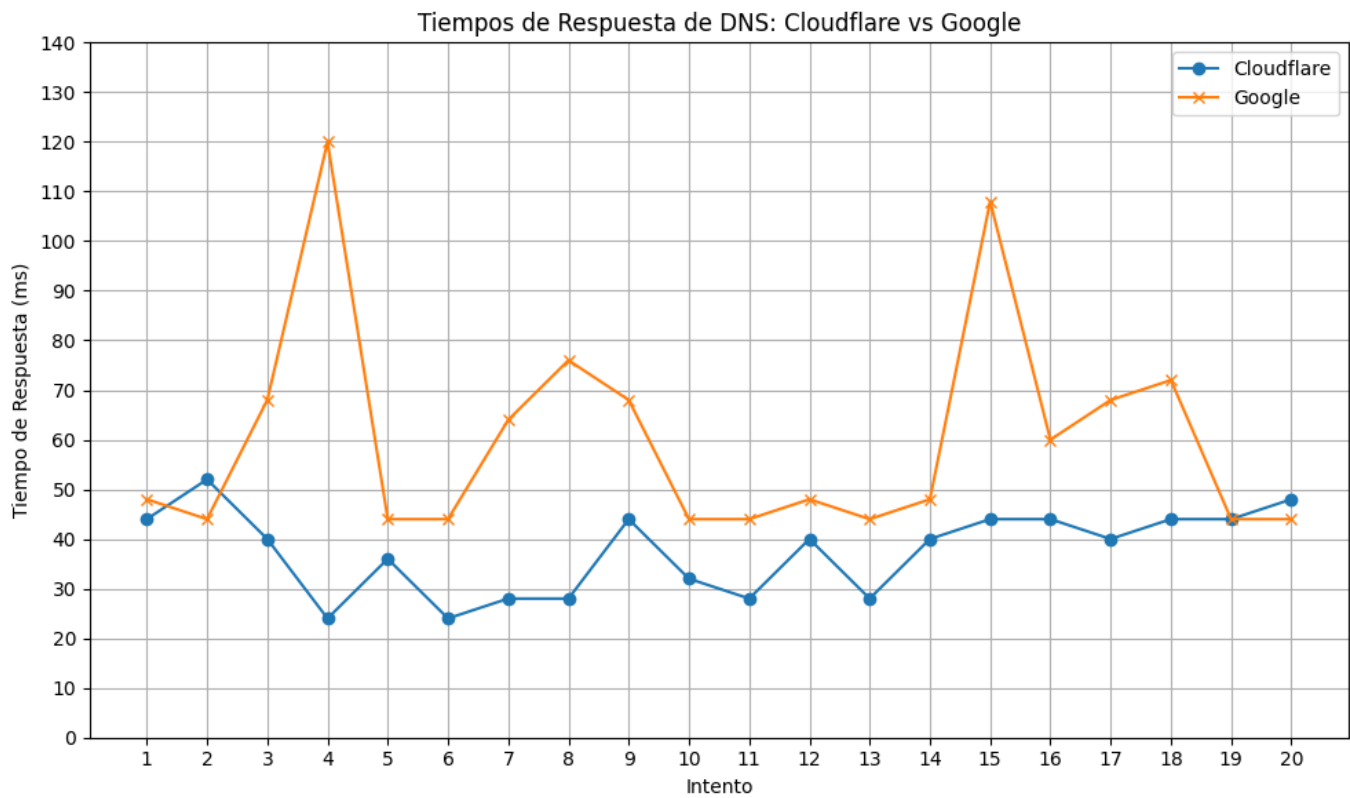
- Basándonos en estos datos, Cloudflare parece ofrecer un mejor rendimiento en términos de tiempo de respuesta promedio y estabilidad en comparación con Google.

4. Consideraciones Adicionales:

- Es importante considerar que estos tiempos pueden variar dependiendo de la ubicación geográfica y las condiciones de red en el momento de la medición.

- Para un análisis más completo y preciso, se recomienda realizar múltiples mediciones en diferentes momentos y ubicaciones.

Gráfica comparativa



El análisis del gráfico comparativo de los tiempos de respuesta entre los servidores DNS de Cloudflare y Google revela importantes conclusiones. Cloudflare muestra una mayor estabilidad en sus tiempos de respuesta a lo largo de múltiples intentos de medición, evidenciado por una curva más suave y consistente en el gráfico, con variaciones mínimas entre los puntos de datos.

Por otro lado, los tiempos de respuesta del servidor DNS de Google presentan una variabilidad más marcada, con picos de tiempo en ciertos intentos que generan una curva más irregular en el gráfico. Esta variabilidad sugiere que el rendimiento del servidor DNS de Google puede fluctuar más significativamente en diferentes momentos o condiciones de red.

Apendice A: Script de Medición de Tiempos de Consulta DNS

```
#!/bin/bash

# Nombre del script: medir_tiempos_dns.sh

# Número de veces que se ejecutará el comando dig
ITERACIONES=20

# Nombre del archivo de salida para guardar los tiempos de consulta
ARCHIVO_SALIDA=tiempos_consulta_dns.txt

# Limpiar el archivo de salida si existe previamente
> $ARCHIVO_SALIDA
```

```
# Bucle para ejecutar el comando dig y guardar los tiempos de consulta en
el archivo
for ((i=1; i<=$ITERACIONES; i++)); do
    echo "Ejecución $i:"
    tiempo=$(dig google.com @8.8.8.8 | grep "Query time:" | awk '{print
$4}')
    echo "Tiempo de consulta: $tiempo ms"
    echo "$tiempo" >> $ARCHIVO_SALIDA
    sleep 1 # Esperar 1 segundo entre cada ejecución
done

echo "Proceso completado. Los tiempos de consulta se han guardado en
$ARCHIVO_SALIDA"
```