DNS de alto rendimiento

Francisco Javier Álvarez Calvo

DNS de alto rendimiento

Francisco Javier Álvarez Calvo

Trabajo de Fin de Grado para la obtención del título de Grado en Informática por la UOC en el área de administración de redes y sistemas operativos

Tutorizada por

Prof. Joaquin Lopez Sanchez-Montañes
Prof. David Bañeres Besora
Prof. Montse Serra Vizern

Agradecimientos

Dedicado a mi familia, y en especial a Mila.
Sin ti esta aventura no tendría un final feliz.

FICHA DEL TRABAJO FINAL

Título del trabajo:	DNS de alto rendimiento
Nombre del autor:	Francisco Javier Álvarez Calvo
Nombre del consultor/a:	Joaquín López Sánchez-Montañés
Nombre del PRA:	David Bañeres Besora
Fecha de entrega (mm/aaaa):	06/2023
Titulación:	Grado de Ingeniería Informática
Área del Trabajo Final:	Administración de Redes y Sistemas Operativos
Idioma del trabajo:	castellano
Palabras clave:	DNS, servidor raiz

Resumen del Trabajo

La finalidad de este trabajo es estudiar la configuración ideal para ofrecer un servicio de resolución de nombre de forma segura y en alta disponibilidad. Las principales cualidades que se esperan obtener serán por un lado una gran tasa de respuestas por segundo y por el otro la securización y adecuado bastionado del servicio DNS, lo que le conferirán las competencias necesarias para su utilización como servidor de nombres raíz o equivalente.

Se analizarán medidas tanto pasivas como activas de securización ante posibles ataques, y se analizará la manera ideal de aislar el tráfico de gestión del tráfico de servicio con el fin de disminuir posibles vulnerabilidades. El servicio será balanceado y tendrá la capacidad de ofrecer respuestas diferentes en función de del origen de las peticiones.

La gestión del servicio será realizada de forma remota y la actualización de los registros ser realizará de forma dinámica a través de protocolos conocidos aunque no comúnmente utilizados, dotando a la solución de la robusted y versatilidad necesarias.

Abstract

The purpose of this work is to study the ideal configuration to offer a name resolution service in a secure and highly available way. The main qualities expected will be on the one hand a high response rate per second and on the other hand the securitization and adequate bastioning of the DNS service, which will give it the necessary competences to be used as a root name server or equivalent.

Both passive and active measures for securing against possible attacks will be analyzed, and the ideal way to isolate the management traffic from the service traffic will be analyzed in order to traffic from the service traffic in order to reduce possible vulnerabilities. The service will be balanced and will have the capacity to offer different responses depending on the different responses depending on the origin of the requests.

The service will be managed remotely and the update of the logs will be done dynamically through known but not commonly used protocols. protocols, providing the solution with the necessary robustness and versatility.

Índice general

Índice de figuras

1 Introducción

1.1 Marco actual y justificación

Actualmente el consumo de todo tipo de servicios web es muy elevado y sigue una clara tendencia de crecimiento constante y paulatino de la mano de las nuevas tecnologías, tanto en el ámbito público como en el privado. La transformación digital ha supuesto un cambio de paradigma tanto el modo de consumo de productos y servicios como en la gestión y explotación de la información. La proliferación de aplicativos que ofrecen soluciones a las necesidades de las diferentes áreas de negocio ha provocado un aumento exponencial en el número de servicios consumidos. Todos y cada uno de estos servicios necesitan ser identificados a través de un nombre que se almacena en un servidor de nombres o *DNS (Domain Name Server)* bajo el paraguas de un dominio, con independencia de que su consumo sea local o global.

En la figura 1.1 se muestra el incremento del tráfico generado por las consultas DNS a lo largo de 7 años. Existe un pico provocado por un comportamiento del navegador *Chromium* y sus derivados [5], que incrementó el tráfico de manera puntual y que, posteriormente, fue corregido. En cómputo global la tendencia es evidente: el tráfico ha ido aumentando paulatinamente con el paso de los años.

Es tal la dependencia del servicio de nombres de dominio que, ante un posible fallo, todos y cada uno de los servicios dejarían de estar disponibles, por resultar imposible alcanzar el destino final de las peticiones. Esta dependencia se manifiesta en todos los ámbitos tecnológicos de las organizaciones: la indisponibilidad del servicio DNS a nivel interno redunda en la falta de acceso a los servicios propios y, a nivel externo, a todos los dominios gestionados como servidor autoritativo. En caso de tratarse del servicio de los servidores raíz, estaríamos hablando de una total indisponibilidad de acceso a Internet a nivel mundial, de ahí que existan varios servidores situados en distintos puntos del globo [1], en total unas 1707 instancias en 12 servidores raíz independientes. Este número ha ido aumentando en los últimos años por necesidades geográficas y, también, debido a que varios ataques de denegación de servicio distribuído [2].

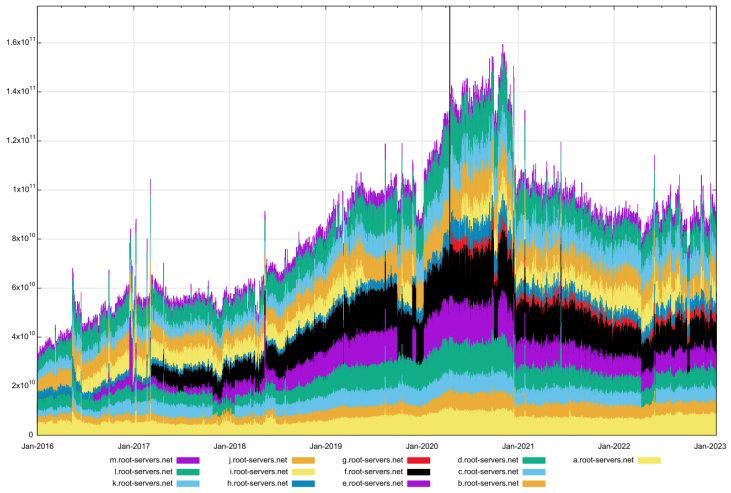

Figura 1.1: Consultas por segundo a lo largo del tiempo en los root servers [4]

Dada la criticidad, se pretende proporcionar una procedimentación para su correcta instalación, configuración, disponibilidad, gestión y securización, con el fin de asegurarnos una disponibilidad del DNS superior al 99,9 %.

1.2 Objetivos

El objetivo principal de este proyecto es dar respuesta a los múltiples problemas que presenta una solución de este tipo, en la que se requiere una confiabilidad, estabilidad, y capacidad de respuesta, más, si cabe, que para el resto de aplicativos y servicios de la organización.

Atendiendo a lo expuesto, dispongo las características de las que se pretende dotar a la solución a través de este proyecto de fin de grado para la configuración de un servidor DNS autoritativo con varios niveles de protección ante ataques, alta disponibilidad y que sea capaz de actualizar registros en caliente con ciertas garantías.

Entre otras debería disponer de las siguientes especificaciones:

- Oferta simultánea de servicio de resolución de nombres a uno o varios dominios.
- Separación a nivel de red entre la parte de gestión y la de servicio.
- Alta disponibilidad: disponer más de un servicio DNS simultáneamente
- Balanceo de carga para dar respuestas a un elevado número de peticiones
- Actualizaciones en caliente y atómicas emulando, en la medida de lo posible, las transacciones que ha de cumplir una BBDD estándar: atomicidad,

consistencia, aislamiento y durabilidad.
- Disposición de una utilidad o envoltorio que facilite la gestión de los registros
- Exposición a Internet del servicio en modo sólo lectura, ocultando el servidor maestro donde se modifican las zonas DNS.
- Oferta de protección frente a ataques de denegación de servicio por exceso de peticiones
- Limitación del acceso a la configuración a una serie de clientes conocidos
- Respuestas condicionadas: se podrá ofrecer una respuesta u otra en función de quién nos haga la petición.
- El servicio correrá preferentemente bajo código abierto
- Utilización de un servicio estable, conocido y de probada fiabilidad, pero que cumpla como mínimo con todas las características previamente expuestas.

2 Arquitectura de la solución

El servidor DNS dispone, de serie, de una configuración en alta disponibilidad denominada *maestro-esclavo*. En esta configuración, uno de los nodos permite la modificación de las zonas (*maestro*), mientras que los nodos restantes, denominados *esclavos*, son de sólo lectura, y únicamente permiten consultas.

Cabe destacar que el servidor maestro será único y en caso de indisponibilidad no podremos realizar modificaciones en las zonas DNS, ni crear nuevas zonas. En lo que respecta a la consulta de registros DNS, deberemos disponer de más de un servidor, a poder ser en sólo lectura. Esto no significa que no se puedan ejecutar consultas en el nodo maestro, pero en la configuración de *maestro oculto*, lo que hacemos es esconder el nodo maestro de tal forma que no se publique su IP sino, únicamente, la de los nodos esclavos. De esta forma la asignación de roles queda claramente diferenciada:

- Nodo Maestro -> Un único nodo en modo lectura/escritura que no atiende consultas dns
- Nodo Esclavo -> Varios nodos en modo sólo lectura que sólo atienden consultas dns

A efectos prácticos esto se traduce en que la interfaz de consulta del nodo maestro no será accesible desde los equipos clientes, aunque podemos presentarla en otra interfaz a modo de gestión, soporte o *troubleshooting*.

Esta configuración es muy común en entornos de cierto tamaño y expuestos hacia Internet, aunque sería totalmente válido configurarlo en un DNS *local* con el fin de aportar seguridad extra.

En la figura 2.1 se muestra una configuración típica maestro - esclavo para el caso de un DNS público. La zona verde representa la zona *DMZ (DeMilitarized Zone ó zona desmilitarizada)*, mientras que la parte azul representa Internet.

Dentro de la red interna se encuentra el maestro o *master*, que es gestionado por un equipo de gestión a través de una interfaz específica, aislada del tráfico administrativo. Los cambios son publicados en el maestro desde el equipo de

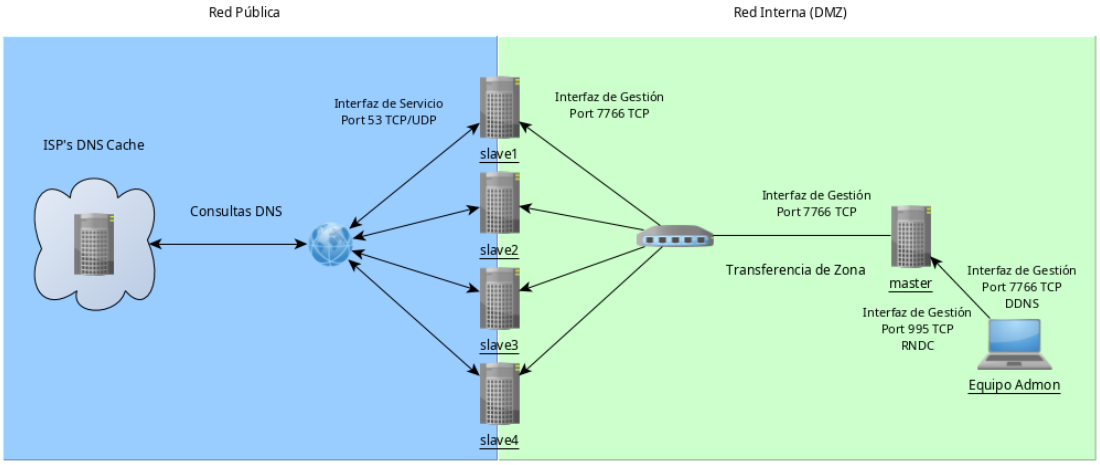

Figura 2.1: DNS en alta disponibilidad con maestro oculto

gestión y éstos se replican en cada uno de los esclavos (*slave[1-4]*) de forma independiente.

3 Instalación y configuración

La instalación se realizará sobre un Sistema Operativo (SO) que soporte el software *BIND (named)* En este caso, seleccionamos el SO RockyLinux en su versión 8, concretamente en la 8.8. El principal motivo que sustenta dicha elección es el hecho de que recoja la filosofía adoptada por CentOS con respecto a *RedHat Enterprise Linux (RHEL)* hasta su versión 7, pero en la siguiente versión (8). Esta elección nos permite disponer de dos opciones idénticas a nivel de versionado de paquetería software, pero claramente diferenciados a nivel de soporte: RockyLinux está disponible como versión comunitario, mientras que *RHEL* posee soporte empresarial.

Los **requisitos mínimos** del servidor para albergar el servidor DNS serán:

- CPU: 2 sockets tipo Xenon x 2 cores
- Memoria: 4 GB
- Almacenamiento: 30GB
- Red: 10Gbps

Estos requisitos serían suficientes para un entorno de desarrollo o pruebas.

Los **requisitos recomendados** para un entorno de producción serán más elevados, pero deberemos tener en cuenta para su correcto dimensionamiento la carga de trabajo, esto es, el número de consultas que el servidor tendrá que responder por segundo.

- CPU: 4 sockets tipo Xenon x 2 cores
- Memoria: 16 GB
- Almacenamiento: 120GB
- Red: 25Gbps

3.1 Instalación y configuración general

3.1.1 Instalación

La instalación del sistema operativo puede realizarse en múltiples entornos. A continuación se describen algunos de ellos:

Bare Metal o máquina virtual

La instalación en *Bare Metal* se realiza en un equipo físico, es decir, sin emplear ningún tipo de virtualiación ni containerización.

En este caso, el procedimiento, a grandes rasgos, será el siguiente:

- Descargar la imagen (*ISO 9660*) ***minimal*** del sistema operativo Rocky-Linux 8 en [11].
- Grabar la imagen en un dispositivo compatible con el hardware - un *pen-drive* o un CD/DVD -.
- Arrancar el servidor desde el medio configurado en el punto anterior

Comenzamos el proceso de instalación del sistema operativo propiamente dicho. Mostramos lo más destacable:

- Formateo del disco mediante particiones: una para arranque, el resto en LVM (*Logical Volume Manager*) según [8]. Empleamos *LVM* para facilitar las tareas de ampliación o cambio de disco en caso de ser necesario.
- La partición de arranque se marcará como tal y se montará adecuadamente en función de si disponemos de *UEFI* o *BIOS*.
- En *LVM* configurar una partición específica para la raíz *root (/)* y seleccionamos el sistema de ficheros *ext4*.
- En *LVM* configurar una partición específica para la carpeta */var* pues esta carpeta almacenará las zonas DNS. Seleccionamos el sistema de ficheros *ext4*.
- Seleccionar instalar únicamente el servicio *sshd*.
- Instalamos el gestor de arranque *GRUB* y reiniciamos

Instancia en la nube

Para el caso que nos ocupa seleccionamos la nube de *Amazon Web Services [9]* como ejemplo. Entre los pasos a destacar, estarían los que a continuación se relacionan:

1. Una vez registrado en *AWS* dirigirse a la sección *EC2*
2. Crear una instancia de tipo medio (*t2.medium*). Las especificaciones las podemos encontrar en [10]
3. Establecer el nombre de la instancia
4. Seleccionar la imagen de RockyLinux 8 desde *AWS Marketplace AMIs*
5. Reservar una *Elastic IP* o IP pública para poder acceder desde el exterior
6. Fijar las políticas de seguridad asociadas a cada interfaz de red.
7. Crear un par de claves y descargar el fichero para acceder a al máquina una vez esté desplegada
8. Para conectar a la máquina utilizar el comando

```
ssh -i clave.pem rocky@<ipPublica>
```

Es importante destacar que las políticas de seguridad en las tarjetas de red de gestión difieren de las de servicio. Las primeras deberán permitir la conexión al servicio de consulta DNS (puerto 53) y las segundas a tareas de administrativas del servicio (puerto 7766).

3.1.2 Configuración

Una vez instalado el sistema operativo y configurada la red, se procede a deshabilitar los siguientes servicios:

- postfix
- sssd
- NetworkManager
- gssproxy

El comando a utilizar para inhabilitar los servicios es el siguiente:

```
~ # systemctl mask --now <nombreServicio>
```

Procede deshabilitar, tambien, *IPv6* en *BIND*, debido a que la configuración se hará sobre IPv4. Para deshabilitarlo es necesario editar el fichero de configuración **/etc/sysconfig/named** añadiendo la siguiente línea:

```
OPTIONS="-4"
```

A continuación se reinicia el servicio mediante el comando:

```
~ # systemctl restart named
```

En caso de no deshabilitar IPv6 los eventos nos mostrarán errores como los siguientes:

```
abr 19 09:37:49 lab-dns-0001 named[28607]: error (network unreachable)
resolving './DNSKEY/IN': 2001:500:2d::d#53
abr 19 09:37:49 lab-dns-0001 named[28607]: error (network unreachable)
resolving './DNSKEY/IN': 2001:503:c27::2:30#53
abr 19 09:37:49 lab-dns-0001 named[28607]: error (network unreachable)
resolving './NS/IN': 2001:503:ba3e::2:30#53
abr 19 09:37:49 lab-dns-0001 named[28607]: error (network unreachable)
resolving './NS/IN': 2001:500:2d::d#53
abr 19 09:37:49 lab-dns-0001 named[28607]: error (network unreachable)
resolving './NS/IN': 2001:503:c27::2:30#53
```

3.1.3 Instalación de BIND

Se comienza instalando los paquetes necesarios para el servicio:

```
~ # yum install bind bind-utils -y
```

Este comando instalará además las dependencias necesarias. La versión en el momento de la instalación es la 9.11.36. En el versionado utilizado por RedHat sería la bind-9.11.36-3.el8 en su versión de 64 bits.

3.1.4 Configuración de BIND

El fichero principal de configuración del BIND es /etc/named.conf. Tiene varias secciones de configuración. Hay que tener en cuenta que el archivo permite la inclusión de configuración en ficheros externos a través de la opción *include*. En ciertos casos y ante una disposición compleja, es aconsejable utilizar esta directiva.

3.2 Instalación y configuración específica de un nodo maestro

A continuación, se muestran diferentes configuraciones específicas para el caso particular en el que el servidor DNS es un servidor DNS público, que, por lo tanto permite, consultas desde cualquier IP de Internet. Aquí, la seguridad será uno de los elementos clave a tener en cuenta.

External Hidden Master Authoritative only: un servidor DNS exclusivamente autoritativo es aquel que posee la autoridad para resolver una o varias zonas DNS definidas en su configuración. En este caso, además, se trata de un servidor maestro, rol que incluye los cambios de zona que se transmiten al resto de servidores esclavos. Este servidor es *External* debido a que está expuesto y atiende peticiones de Internet.

Fichero de configuración **/etc/named.conf**

```
options {
## Por defecto no se abre el puerto 53
listen-on port 7766 { <master-management-IP>; };

    directory               "/var/named";
    dump-file               "/var/named/data/cache_dump.db";
    statistics-file         "/var/named/data/named_stats.txt";
    memstatistics-file      "/var/named/data/named_mem_stats.txt";
    pid-file                "/run/named/named.pid";
    session-keyfile         "/run/named/session.key";
    zone-statistics         yes;

    minimal-responses       yes;
    version                 "not currently available";
    recursion               no;
    dnssec-enable           no;

    allow-query             { none; };
    allow-transfer          { none; };
    notify-source           <master-management-IP> port 7766;
    notify-to-soa           yes; registro SOA
};

...

acl "<acl-slaves>" {
        <Slave-IP1>;
        <Slave-IP2>;
};

...

key "<key-master-slave>" {
        algorithm hmac-sha256;
        secret "secret-key-master-slave";
};

...

include "/etc/named/named.zones.conf";
```

Fichero de configuración específico para las zonas DNS de cada dominio.

/etc/named/named.zones.conf

```
{     A;    B; }   == A is allowed, accept immediately
{  {  A; }; B; }   == A is allowed, accept immediately
{    !A;    B; }   == A is forbidden, reject immediately
{ !{  A; }; B; }   == A is forbidden, reject immediately
{  { !A; }; B; }   == A matched but was negated, try element B
{ !{ !A; }; B; }   == A matched but was negated, try element B
```

Figura 3.1: Combinaciones válidas de dos elementos en *ACL* de *BIND* [21]

```
zone "midominio.es" {type master; file "master/midominio.es.dns";
allow-transfer { !{!"<acl-slaves>";}; key <key-master-slave>; }; };
```

3.2.1 Análisis fichero configuración dns maestro

Se revisan, a continuación, varias de las opciones fijadas para el servidor maestro que difieran de la configuración por defecto.

- *listen-on ...* → indica qué interfaces y puertos se abren para el servicio DNS. Ver 5.2.1
- *minimal-responses yes* → se analiza en 5.4.3
- *version ...* → se refiere a la ocultación de la versión. Ver 5.4.2
- *dnssec-enable ...* → sobre la configuración *DNSSEC*. Ver 10.1
- *allow-query ...* → no se permiten consultas sobre este servidor DNS por tratarse de un servidor maestro oculto.
- *allow-transfer ...* → no se permite puesto que no se trata de un servidor esclavo
- *notify-source ...* → indica la interfaz desde la que se va a notificar a los servidores esclavos indicando que hay una nueva actualización de la zona. En cuanto los servidores esclavos reciban esta notificación, iniciarán la transferencia de la zona correspondiente para actualizar la zona. De esta forma los servidores esclavos se actualizan con el maestro al instante.
- *notify-to-soa ...* → en las configuraciones en hidden master los *SOA (Start Of Authority)*, o servidores con autoridad de cara al mundo exterior serían los esclavos, no el maestro (de no ser así el maestro no estaría oculto). En caso de que se produzca un cambio en la serialización del *SOA* (número de serie creciente utilizado como versionado de las zonas) el maestro deberá anunciar a los *SOA* que en este caso son los esclavos.
- *<acl-slaves>* → Definimos esta *ACL* para identificar los servidores esclavos que recibirán las transferencias de zona. En este caso se indicará la IP de la interfaz de gestión de los servidores esclavos, no la interfaz de servicio donde atiende las consultas DNS.

- *<key-master-slave>* → es la clave que compartirán entre maestro y esclavos para transferir la zona. La transferencia de zona no se completa si no se autoriza la IP del esclavo y posee esta clave.
- *zone "midominio.es"...* → se indica que este servidor DNS posee una zona maestra para este dominio. Los servidores esclavos indicarán que son servidores esclavos para este mismo dominio.
- *allow-transfer ...* → esta configuración es poco común, pero de esta forma obligamos a que se cumplan dos condicionantes: que sólo se transfiera a los esclavos, y a aquellos que poseean la clave compartida.
  1. En primer lugar negamos el conjunto: *!{...}*
  2. Negamos el primer elemento: *!<acl-slaves>*
  3. Permitimos el segundo elemento: *key <key-master-slave>*
  4. En consecuencia el resultado es que se deben cumplir ambas condiciones 3.1

3.3 Instalación y configuración específica de un nodo esclavo

En este caso se trata de uno o varios servidores esclavos, complementarios del maestro, encargados de recibir las zonas modificadas del maestro, y responder las consultas DNS.

Fichero de configuración **/etc/named.conf**

```
options {
listen-on port 53 { <slave-service-IP>; };
listen-on port 7766 { <slave-management-IP>; };
directory       "/var/named";
dump-file       "/var/named/data/cache_dump.db";
statistics-file "/var/named/data/named_stats.txt";
memstatistics-file "/var/named/data/named_mem_stats.txt";
zone-statistics yes;
version         "not currently available";
allow-query     { any; };
minimal-responses yes;
recursion no;
notify no;
allow-transfer {"none";};
transfer-source <master-management-IP> port 7766;
rate-limit  {
                    window 5;
                    responses-per-second 20;
                    referrals-per-second 20;
                    nodata-per-second 20;
                    nxdomains-per-second 20;
                    errors-per-second 20;
                    all-per-second 30;
                    log-only no;
```

```
                    exempt-clients { <aclExcepciones>; };
                    ipv4-prefix-length 32;
            };

    dnssec-enable no;

    pid-file "/run/named/named.pid";
    session-keyfile "/run/named/session.key";
    };

    ...

    key "<key-master-slave>" {
            algorithm hmac-sha256;
            secret "secret-key-master-slave";
    };

    ...

    server <Master-IP1> {
            keys { <key-master-slave>; };
    };

    ...

    masters "<masterSource>" {
            <Master-IP1> port 7766;
    };

    ...

    include "/etc/named/named.zones.conf";
```

Fichero de configuración específico para las zonas DNS de cada dominio. **/etc/named/named.zones.conf**

```
zone "midominio.es" {type slave; file "slaves/midominio.es.dns";
     masters { "<acl-master>"; }; };
```

3.3.1 Análisis del fichero de configuración dns esclavo

Se revisa, a continuación, varias de las opciones fijadas para el servidor esclavo que difieran de la configuración del nodo maestro.

- *listen-on ...* → en este caso disponemos de dos directivas de este tipo, una para la gestión y otra para el servicio DNS 5.2.1
- *allow-query ...* → esta directiva está asignada a *any*, lo que significa que desde cualquier IP se podrán realizar consultas a este servidor DNS. Esto

debe ser así pues el tráfico de Internet no se puede limitar. Como configuración alternativa podríamos excluir de las consultas las IP's asociadas a rangos privados, aunque no se ha considerado necesario.

- *allow-transfer ...* → esta directiva está fijada a none, pues desde el nodo esclavo nunca deberá transferirse zona alguna por defecto. Esta política es importante desde un punto de vista de la seguridad, pues hay ataques que consisten en obtener todos los registros DNS de un dominio mediante la transferencia de toda la zona DNS y en base a los nombres DNS existentes y sus IP's fijar un objetivo y atacarlo.
- *transfer-source ...* → con esta propiedad les indicamos a los servidores esclavos qué IP y puerto de los disponibles utilizarán para conectarse al nodo maestro y realizar la transferencia de zona. En caso de no configurar esta directiva, podría intentar la transferencia de zona por la interfaz incorrecta. La IP asociada a esta interfaz también será la misma que se le permite la transferencia de zona en el nodo maestro con la directiva *allow-transfer*.
- *rate-limit ...* → directiva utilizada para limitar la tasa de consultas por segundo, ver 5.3.4
- *<key-master-slave>* → comparten la misma clave entre maestro y esclavo, para habilitar las transferencia de zona entre ambos.
- *server <MasterIP1>* → establecemos una *ACL* asociando la IP del servidor maestro con la clave del apartado anterior.
- *masters ...* → esta directiva se utilizapara indicar la IP y el puerto del nodo maestro al que se conectarán los esclavos para realizar las transferencias de zona a través de la interfaz/puerto indicados en la directiva *transfer-source*.

3.4 Aislamiento del tráfico

El servidor dispondrá de una única interfaz de red de gestión.

Por defecto, un servidor DNS tiene asociado el puerto *53/UDP* en la interfaz de servicio, pero, dado que el maestro es oculto, no debería tener ese puerto abierto para evitar peticiones. En este caso, indicamos que escuche únicamente en el puerto *7766* de la interfaz de gestión en la directiva (*listen-on ..*). Esta interfaz de gestión la emplearemos también para las transferencias de zona con los servidores esclavos; de esta forma, el tráfico de gestión y el tráfico de servicio de los nodos esclavos estarán completamente aislados a nivel de red como se indica en 5.2.

Desde el equipo de administración se realiza la conexión a la interfaz de gestión en el puerto 7766 para realizar tareas y lanzar comandos al servidor maestro. Sólo se permitirá la gestión desde las IP's indicadas en la *ACL rndc-remoteservers* mostrada en 4.1.2

3.4.1 Aislamiento del tráfico (nodo esclavo)

El servidor esclavo dispondrá de dos interfaces de red: una de gestión, por la que fluye el tráfico relativo a la gestión (tráfico de transferencia de zona y tráfico de gestión de *RNDC* en el puerto 7766) y otra de servicio, por el que fluye el tráfico relativo a las consultas DNS (*TCP y UDP*) a través del puerto 53.

El tráfico se aislará mediante dos interfaces 5.2, en segmentos de red separados.

4 Gestión remota del DNS

A continuación se muestra la manera de controlar:

- El servicio named
- Registros y zonas del DNS

Desde un equipo de administración situado a nivel de red en la capa de gestión y sin la necesidad de loguearse en el servidor DNS, prodrán ralizarse acciones tales como:

- recargar el DNS
- añadir registros
- alterar registros
- añadir zonas (a través de *SSH*)

La idea es que las tareas de administración principales sean realizadas directamente desde un equipo remoto habilitado a tal efecto a través de la interfaz de red de gestión, y no directamente desde el propio servidor DNS. Esto aportará seguridad y coherencia en la gestión de zonas DNS. Se guardará un registro de las órdenes enviadas desde el equipo remoto, y se irá versionando en un repositorio de código por cada cambio realizado 10.3

4.1 Cliente de gestión remota: RNDC

Para la administración remota del servicio DNS se emplea la utilidad de administración del demonio *named* a través de línea de comando, *rndc*, tanto desde el propio servidor como desde un el equipo de administración remota. Para evitar un uso no autorizado en el acceso al demonio named, *BIND* emplea un método de autenticación basado en una clave secreta compartida (*shared secret key*), garantizando privilegios a determinados equipos/usuarios. Se debe hacer referencia a esta clave tanto desde el punto de vista del servidor en el fichero */etc/named.conf (include "/etc/rndc.key")*, como desde el punto de vista del equipo cliente a través del fichero */etc/rndc.conf*.

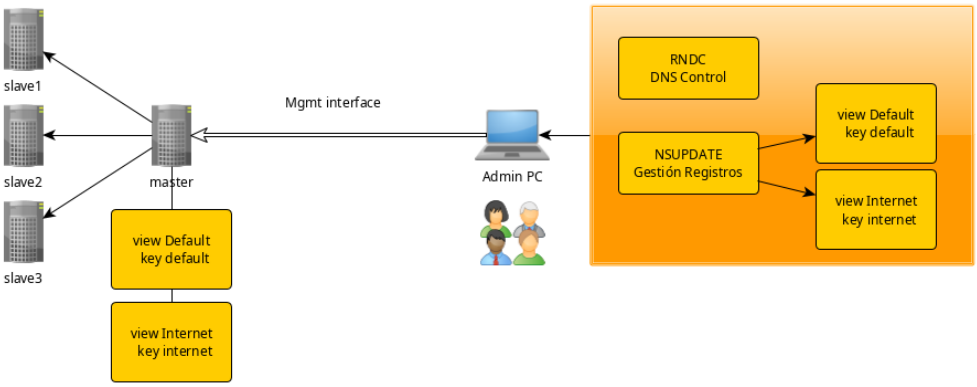

Figura 4.1: Esquema de gestión remota de zonas DNS

4.1.1 Instalación RNDC

El paquete *BIND* debe estar instalado en la estación , pero sería conveniente deshabilitar el servicio en equipos de administración.

```
~ # yum install bind
~ # systemctl mask --now named
```

4.1.2 Configuración RNDC

Generando clave compartida

El primer paso será generar las claves. Una vez generadas se pueden emplear tanto para el control del servicio (RNDC) como para gestión de registros de dominios (*NSUPDATE*). Para generar la clave secreta compartida se emplea el algoritmo HMAC-MD5, es el único algoritmo soportado. Es una buena práctica generar una clave de al menos 256-bits de longitud. La clave se generará mediante el siguiente comando:

```
~ # dnssec-keygen -a hmac-md5 -b <bit-length> -n HOST <key-file-name>
```

A título de ejemplo:

```
~ # dnssec-keygen -a hmac-md5 -b 256 -n HOST adminpc
```

La ejecución de este proceso puede tardar algún tiempo. Cuando finalice, generará dos ficheros:

- Kadminpc.+157+51708.key
- Kadminpc.+157+51708.private

El contenido de los ficheros generados es similar al siguiente:

```
~ #cat Kadminpc.+157+51708.key
    Kadminpc. IN KEY 512 3 157 hU9utBAdP6/dVKKfxOlv0bPOTnAd4A1qosMbs/dwVJI=

~ #Kadminpc.+157+51708.private
    Private-key-format: v1.3
    Algorithm: 157 (HMAC_MD5)
    Key: hU9utBAdP6/dVKKfxOlv0bPOTnAd4A1qosMbs/dwVJI=
    Bits: AAA=
    Created: 20230415211448
    Publish: 20230415211448
    Activate: 20230415211448
```

Lo esencial de estos ficheros es la clave secreta compartida, que comienza por *hU...*, y que será la que se emplee en cada caso.

Usaremos esta clave en dos supuestos:

- Claves compartidas para el control del demonio BIND desde la herramienta rndc: rndc.key
- Claves compartidas para la gestión de los registros de zona. Podemos gestionar los registros a través de la herramienta nsupdate u otra similar, que, previamente, deberá estar autorizada a través de esta clave compartida.

En primer lugar, se creará en los servidores DNS el fichero *rndc.key* donde se guardarán las claves, y se configurarán los permisos adecuados

```
~ # touch /etc/named/keys.d/rndc.key
~ # chown root.named /etc/named/keys.d/rndc.key
~ # chmod 640 /etc/named/keys.d/rndc.key
```

A continuación, se edita el fichero incluyendo las entradas de clave compartida necesarias para la gestión del demonio *BIND*. Como mínimo, se crearán dos:

- Una para la gestión desde el propio servidor DNS (*localhost*)
- Otra para la gestión remota. Las máquinas desde las que se emitan las órdenes al *BIND* deberán disponer de esta clave remota.

Se almacenarán cada una de las claves en el directorio */etc/named/keys.d/* generando los ficheros:

- Krndc-local.key.+157+51708.key
- Krndc-local.key.+157+51708.private
- Krndc-remote.key.+157+20845.key
- Krndc-remote.key.+157+20845.private

Se extraen las claves de sendos ficheros .key y se edita el fichero */etc/named/keys.d/rndc.key* introduciendo las claves compartidas en cada caso

```
key "rndc-local" {
    algorithm hmac-sha256;
    secret "hU9utBAdP6/dVKKfxOlv0bPOTnAd4A1qosMbs/dwVJI=";
};

key "rndc-remote" {
    algorithm hmac-sha256;
    secret "IBsAAFQ2aaRkKaOjAgMEWOG+dV3P7VjkE9JsvNskhBA=";
};
```

Configuración lado servidor

Posibilita la gestión tanto del demonio de BIND (named), como de las zonas y su transferencia. Para que rndc pueda conectarse al servicio named, se deberá crear una configuración específica de controles en el fichero de configuración del servidor BIND, /etc/named.conf:

```
include ‘‘/etc/named/keys.d/rndc.key’’;

acl "<rndc-remoteservers>" {
        <IPadminpc>;     // remote admin
};

controls {
    inet 127.0.0.1 port 953
        allow { localhost; } keys { ‘‘<rndc-local>’’; };
    inet <ip-management-interface> port 7766
        allow { ‘‘<rndc-remoteservers>’’; } keys { ‘‘<rndc-remote>’’; };
};
```

Estas declaraciones indican al demonio named que ha de escuchar en el puerto por defecto (TCP 953) en la dirección IP de loopback, permitiendo comandos

rndc desde el propio equipo a través de localhost, especificando el nombre de la clave secreta que está permitida (<rndc-local>). Además, escuchará comandos rndc en el puerto TCP 7766 de la interfaz de management del servidor desde equipos remotos que se encuentren en los rangos de direccionamiento permitidos en la claúsula <rndc-remoteservers>, autenticándose con la clave secreta definida en <rndc-remote>. Al principio de la sentencia, se hace un *include* del fichero donde están almacenadas las keys de forma segura. Para que esta configuración funcione al realizar una conexión desde un equipo remoto debidamente autorizado, se debe tener el puerto 7766 autorizado en las iptables, como se indicará más adelante en la sección de Seguridad.

Configuración lado equipo remoto

El equipo remoto será el equipo que se empleará para el control del BIND, evitando, de este modo, conectarnos directamente al servidor DNS. Al tener una clave única para los equipos remotos, todo equipo que posea esa clave, y siempre y cuando su IP esté autorizada explícitamente, podrá ejecutar comandos a través del cliente rndc. La configuración en el equipo remoto se realiza del siguiente modo: El fichero a configurar es */etc/rndc.conf*, con el siguiente contenido:

```
key "rndc-remote" {
    algorithm hmac-sha256;
    secret "<rndc-remote->";
};

options {
    default-key "<rndc-remote>";
    default-server <ip-management-interface>;
    default-port 7766;
};
```

4.1.3 Operación RNDC

Aquí se indicarán los comandos esenciales a la hora de operar con el servicio named a través de RNDC

Operación General

El comando ejecutado en local o remoto, utilizará por defecto el fichero de configuración /etc/rndc.conf. En caso de emplear un fichero de configuración específico en una ruta concreta, se deberá ejecutar el comando como se detalla a continuación:

```
$ ~ rndc -c /path/to/file/rndc.conf <commands>
```

Los comandos que es posible ejecutar son:

1. *reload* → recarga toda la configuración
2. *restart* → reinicia el servicio
3. *status* → muestra el estado del servicio
4. *stop* → salva las actualizaciones pendientes y para el servicio
5. *stats* → muestra estadísticas del servicio
6. *sync* → salva los cambios a disco y borra los ficheros temporales .jnl
7. *freeze* → no permite actualizaciones dinámicas en las zonas
8. *thaw* → permite actualizaciones dinámicas en las zonas
9. *flush* → vacía la caché del sevidor DNS

Operaciones de Zona

Existen, así mismo, una serie de comandos que se aplican por zonas

- *reload <zone>* → recargar la configuración de la zona
- *retransfer <zone>* → volver a retransmitir la zona independientemente de que haya variado el número de serie de la misma
- *freeze <zone>* → no permitir actualizaciones en la zona
- *thaw <zone>* → permitir actualizaciones en la zona
- *notify <zone>* → reenviar notificaciones para una zona concreta
- *refresh <zone>* → actualizar el estado de la zona desde un punto de vista de las actualizaciones

4.2 Actualizaciones dinámicas de las zonas: *DDNS*

DDNS se refiere a *Dynamic Domain Name System*, un protocolo creado con el objetivo de realizar actualizaciones dinámicas en el servidor DNS en caliente, es decir, sin ningún corte de servicio. El formato está definido en el *RFC 2136* [25].

DDNS permite añadir, modificar y eliminar registros en las diferentes zonas 6 DNS. Para realizar este tipo de acciones de escritura en las zonas, las modificaciones deberán realizarse sobre el nodo maestro. Éste una vez completada la modificación notificará a los esclavos de que la zona ha sido actualizada iniciando una transferencia de zona, disponiendo de los cambios en cuestión de segundos.

La política para permitir las actualizaciones se aplica tanto a nivel global en la sección de *options* a través de la directiva *allow-update*, o específicamente en una zona DNS mediante la política *update-policy*. Podría suceder que existen

zonas DNS en las que se dispone de autorización para realizar actualizaciones dinámicas y otras en las que no.

4.2.1 Configuración

```
options {
    ...
    allow-update { !{ !NSupdateExternal; any; }; key external-key-nsupdate; };
    ...
}

acl "NSupdateExternal" {
    <IP1Gestion>;
};

key "external-key-nsupdate" {
    algorithm hmac-sha256;
    secret "505bm3Q0nS8ROt.iyvE2C9/Ix57g7VYJkAsi15+QkY7=";
};
```

En este caso se emplea la directiva global *allow-update* dentro de la sección *options*. Se establece el mismo criterio empleado en 3.1 de tal forma que cumple que la IP que realiza la actualización esté en la ACL *NSupdateExternal* y posee la misma clave almacenada en *external-key-nsupdate*. De esta forma se autoriza al equipo de gestión con la IP *<IP1Gestion>* a actualizar el DNS a través de DDNS.

4.2.2 El fichero de *journal*

Todas las zonas DNS poseen un lugar propio en el que se almacena la información referente a esa zona (ver 6). En caso de actualizar una zona a través del protocolo *DDNS* no se escribe el cambio directamente en el fichero, sino que se guarda en uno temporal con extensión *.jnl* y con el mismo nombre que el de zona. Este archivo es de tipo binario y no puede ser leído directamente.

De forma periódica (cada 15 minutos, normalmente) el servidor DNS aplica los cambios sobre el fichero de zona, eliminando el temporal.

Este tipo de zonas no pueden ser editadas a mano ya que están configuradas para ser modificadas de forma dinámica. De todas formas existe un mecanismo que permite temporalmente que sean editadas a través del cliente *rndc* 4.1.3.

- En primer lugar se *congela* la zona mediante el comando *rndc freeze mydomain.com*. Esto consolida el fichero de con los cambios existentes en el archivo de *journal*.

- Se realizan los cambios en la zona.
- Modificar el número de serie del registro *SOA*. Ver B.1
- Se libera la zona para que se convierta de nuevo en dinámica *rndc thaw mydomain.com*

Existe otro modo de consolidar los cambios del *journal* en el fichero de zona sin tener que *congelarlo* a través del comando *rndc sync mydomain.com*. Si se añade el modificador *-clean* eliminaremos los archivos *.jnl*

4.2.3 Zona de transferencia incremental *IXFR*

La transferencia de zona no tiene por qué realizarse íntegramente en cada actualización, pues implicaría enviar el fichero completo. Existe una alternativa que es transferir únicamente la diferencia entre el existente y su estado final, lo que reduciría el tamaño de *bytes* a transferir. Esto se activa a través de la directiva *ixfr-from-differences yes*. La directiva está activada por defecto si el nodo es de tipo esclavo. Los pormenores del protocolo *IXFR* están recogido en el siguiente *RFC* [26].

4.2.4 Scripts para *DDNS*

Como se indicó en 4.2 existe un procedimiento a la hora de realizar tareas de actualización de las zonas DNS de forma dinámica. En este apartado se muestran una serie de scripts que sirven como envoltorio para facilitar esta tarea de edición. Dado que el procedimiento no es el mismo para todos los tipos de registros [27] se añadirá un script para los más comunes.

Por otro lado el protocolo *DDNS* no dispone de una directiva que nos permita crear dinámicamente las zonas DNS. Se implementará a través de un script específico que se conecte directamente al servidor maestro.

Actualización registro SOA

Cada zona DNS dispone de un registro denominado *SOA*. Asociado a éste existe un número de serie asignado que va aumentando a medida que la zona se modifica, de tal forma que este número siempre crece. En caso de que se realice un cambio en la zona, siempre se deberá incrementar este número.

El convenio más comúnmente utilizado a la hora de fijar este número consiste en escribir la fecha completa en formato *añoMesDiaXX*, siendo el código xx un número de serie de dos dígitos, que permitiría realizar hasta 99 modificaciones en el mismo día.

Se creará un script separado con una función que nos permita llamarla desde cualquier otro script: B.1

Actualización registros A/CNAME

Este script B.2 actualiza las entradas DNS de tipo A y CNAME. Uso y ejemplos:

```
        ./dns-actualizar.sh <zonaDNS> <registro-sinZona> <tipoRegistro> <IP> "Comentario
Opcional"
        ./dns-actualizar.sh midominio.es test A 127.0.0.2 "Alta entrada DNS nueva Web"
```

Borrado registros A/CNAME

Este script B.3 elimina las entradas DNS de tipo A y CNAME. Uso y ejemplos:

```
        ./dns-borra.sh <zonaDNS> <registro-sinZona> <tipoRegistro> "Comentario Opcional"
        ./dns-borrar.sh midominio.es test A "Borrado entrada DNS nueva Web"
```

Creación zona

Este script B.4 crea una nueva zona DNS en base a una plantilla. Uso y ejemplos:

```
        ./zona-crear.sh <zona(dominio)> <IPprincipal> <ficheroZona(opcional)>
        ./zona-crear.sh mydomain.com  127.0.0.2
```

Utiliza un fichero plantilla como el siguiente *domain-view-0.profile*

```
        # defaults profile for nameserver ns1.javieralvarezcalvo.com
        #
        TTL="3h"                      # Default TTL
        ATTL="3600"                   # Default TTL for each DNS rec
        EMAILID="root.javieralvarezcalvo.com." # hostmaster email
        REFRESH="3h"                  # Refresh After 3 hours
        RETRY="1h"                    # Retry Retry after 1 hour
        EXPIER="1w"                   # Expire after 1 week
        MAXNEGTIVE="1h"               # Minimum negative caching of 1 hour
        # name server names FQDN
        NAMESERVERS=("ns1.javieralvarezcalvo.com." "ns2.javieralvarezcalvo.com.")
        # name server IPs,
```

```
# leave it blank if you don't need them as follows
NAMESERVERSIP=("52.215.146.119" "54.75.48.125")
# mail server names
# leave it blank if you don't need them
#MAILSERVERS=("mail.javieralvarezcalvo.com.")
MAILSERVERS=()
################## add your own A recored here ##########################
# You can add additonal A recs using following function
function LoadCutomeARecords(){
echo >/dev/null # keep this line
# Uncomment or add A recoreds as per your requirments
# echo "ftp                    $ATTL   IN      A       202.54.2.2"
# echo "webmail                $ATTL   IN      A       202.54.2.5"
# echo "ipv6host                       $ATTL   IN      AAAA    2001:470:1f0e:c2::1"
```

4.2.5 Automatización

En el mercado diferentes herramientas de automatización que permiten interaccionar con el protocolo *DDNS*. Dos de las más conocidas son *Ansible* y *Terraform*, aunque no son las únicas. Ambas permiten gestionar los registros a través de *nsupdate* utilizando su propio lenguaje. En caso de tener implantada en la organización alguna de ellas, sería lógico implementar la actualización de registros con el código apropiado. El motivo principal es que permite integrar tareas de gestión del DNS en otros flujos o proyectos, automatizando así el proceso completo.

Cada uno de ellos posee sendos módulos como es el caso de *Ansible* [29] o de Terraform [30]

5 Securización

5.1 Niveles de securización

En lo que a un servicio de DNS concierne, existen varios niveles de securización, a destacar:

- Red
- Servicio
- Sistema Operativo

Los vectores de ataque podrían explotar vulnerabilidades en cualquiera de los elementos indicados, por lo que se deben adoptar medidas de cara a garantizar una protección adecuada.

5.2 Securización de la red

Como medida extra de seguridad a la hora de configurar las interfaces de red de los servidores DNS, el tráfico de gestión esté completamente separado del tráfico de consulta.

El mayor número de peticiones que se realizan a un servidor DNS expuesto a Internet son sobre el puerto 53 (TCP/UDP). Por defecto la transferencia de zona se lleva a cabo sobre este mismo puerto. La gestión de zona a través de *rndc* y *nsupdate* se suele realizar en puertos diferentes. La medida adoptada consiste en dedicar la interfaz de red de servicio de manera exclusiva a las consultas DNS (en el puerto 53). Tanto las actualizaciones de zona, como la gestión del servicio y registros DNS se llevarán a cabo a través de la interfaz de gestión. De este modo, el tráfico de "gestión" estará aislado del tráfico de "consulta", y la interfaz de "gestión" no será de acceso público bajo ninguna circunstancia. Se consigue, de este modo, establecer una fuerte barrera a nivel de red que reduce la exposición del servidor DNS. Esta configuración podría implicar que, por seguridad, el servidor maestro, al no atender peticiones DNS, pueda tener el

puerto 53 cerrado en la interfaz de servicio, dado que las transferencias de zona y las consultas de los DNS esclavos se podrían realizar por un puerto alternativo.

5.2.1 Aislamiento del tráfico

El servidor maestro dispondrá de una única interfaz de red de gestión, mientras que el servidor esclavo dispondrá de dos interfaces, una de gestión y otra de servicio.

Nodo maestro

Por defecto, un servidor DNS tiene asociado el puerto *53/UDP* en la interfaz de servicio, pero, dado que el maestro es oculto, no debería tener ese puerto abierto para evitar peticiones, ni tampoco disponer por lo tanto de la interfaz de servicio, como se vió en 3.2.1. En este caso, se indicará que escuche únicamente en el puerto *7766* de la interfaz de gestión con la directiva (*listen-on ..*). Esta interfaz de gestión se empleará, así mismo, para las transferencias de zona con los servidores esclavos; de esta forma, el tráfico de gestión y el de servicio de los nodos esclavos estarán completamente aislados a nivel de red.

Situados en el equipo de administración se conectará a la interfaz de gestión en el puerto 7766 para realizar tareas y lanzar comandos al servidor maestro. Sólo se permitirá la gestión desde las IP's indicadas en la *ACL rndc-remoteservers* mostrada en 4.1.2

Nodo esclavo

Tal y como se plasma en la configuración expuesta en 3.3.1 el nodo esclavo posee dos directivas *listen-on*:

- Puerto 53 asociada a la interfaz de servicio
- Puerto 7766 asociada a la interfaz de gestión

En este caso, el servicio de resolución de nombres se ofrecerá por la interfaz de servicio en el puerto *53/UDP* y *53/TCP* (lo estándar), mientras que la transferencia de zona con el maestro y los comandos de gestión que son enviados desde el equipo de administración lo harán a través de la interfaz de gestión, en el puerto sugerido *7766/TCP* (aunque podría ser cualquier otro acordado).

5.3 Securización del servicio

5.3.1 Securización del nodo maestro

La securización principal del nodo maestro pasa por configurarlo como oculto. Esto implica, en la práctica, que en la declaración de los servidores de zona (entradas de tipo *NS*) sólo estarán declarados los servidores DNS esclavos, nunca el maestro. De esta forma, el servidor maestro, único con autoridad sobre la zona, no estará visible y consecuentemente no se facilita el acceso al mismo. Como consecuencia el servidor maestro dejará de ser un punto más de consulta, pero este hándicap lo compensa con creces la seguridad que proporciona ocultarlo de cara a mantener la integridad del DNS, y con un nodo esclavo extra de ser necesario.

A modo de síntesis, destacar que las funciones principales del servidor DNS maestro serán:

- Modificación de zonas a través de herramientas autorizadas (Web, nsupdate)
- Transferencia de zona a los servidores esclavos

5.3.2 Securización del nodo esclavo

De entre los principales cometidos de los servidores esclavos, destacan:

- Ser autoritativos públicamente
- Atender las consultas DNS
- Recibir las transferencias de zona del servidor maestro

5.3.3 Securización del cliente de gestión

5.3.4 Securización frente a ataques de denegación de servicio

Límite de peticiones

Al objeto de combatir ataques por amplificación, es recomendable implementar en todos los servidores autoritativos la solución Response Rate Limiting (RRL); esta solución permitirá que el servidor responda mejor ante ataques masivos. En este caso se incluirá una tasa de respuestas por segundo máxima de

15 calculadas en una ventana de 5 segundos. A continuación, se muestra la configuración activa. El parámetro log-only no/yes indica si está en modo logging (yes) o en modo activo (no).

```
rate-limit {
           window 5;
           responses-per-second 5;
           referrals-per-second 5;
           nodata-per-second 5;
           nxdomains-per-second 5;
           errors-per-second 5;
           all-per-second 20;
           log-only no;
            exempt-clients { <aclExcepciones>; };
    };
```

A modo de explicación de la configuración seleccionada:

- *window 5* → tamaño de ventana: a parte de la tasa de peticiones por segundo, una IP origen no estará autorizada a más de 25 en total durante 5 segundos de ventana (5x5)
- *responses-per-second 5* → número máximo de peticiones por segundo permitidas para una IP origen dada. Sólo se contemplan como iguales las respuestas del mismo tipo (OK, REFUSED, NO-DATA...) y con la misma query (dominio dado) para la misma IP origen: no entraría en este límite consulta a diferentes dominios, o con respuestas distintas, por ejemplo.
- *referrals-per-second 5* → igual que el anterior pero para consultas de dominios delegados
- *nodata-per-second 5* → respuestas del tipo NODATA
- *nxdomains-per-second 5* → respuestas del tipo el registro no existe
- *errors-per-second 5* → respuestas con errores del tipo REFUSED, FORMERR, SERVFAIL
- *all-per-second 20* → cota superior para la IP origen: la misma IP no podrá superar este número en la ventana dada (5x20=100) independientemente de la consulta y su respuesta.
- *log-only no* → *rate-limit* activado
- *exempt-clients ...* → esta directiva nos permite crear excepciones a la tasa en el límite de peticiones. En la *ACL <aclExcepciones>* podríamos podríamos incluir todos aquellos servidores confiables que suelen realizar un número de peticiones elevado, como por ejemplo los DNS asociados a los principales *ISPs (Internet Service Providers)* como *Movistar, Vodafone, Orange o MasMovil* entre otros.

El bloqueo realizado es de rango /24, es decir, si la IP atacante es la 1.2.3.4, el rango bloqueado sería 1.2.3.0/24. Este rango es modificable a través de la opción

ipv4-prefix-length. Existe un parámetro extra de configuración que disminuye los límites establecidos por el rate-limit en función de la carga global del servidor: qps-scale. Esta opción no está aplicada, por ser de una fórmula más compleja, pero podría estudiarse su implantación. Se pueden excluir determinados rangos de IP's de los límites aplicados a través de la opción *exempt-clients*.

5.4 Securización del demonio BIND

5.4.1 Deshabilitar la recursión

Los servidores DNS externos autoritativos no deberán realizar consultas a otros servidores DNS, dado que las únicas zonas que van a servir a sus clientes son aquellas para las que están autorizados. En este caso, la mejor y más segura de las opciones es deshabilitar la recursividad en las peticiones. Para ello se editará el fichero de configuración */etc/named.conf* y se añadirá dentro de la sección *options* la siguiente línea:

```
recursion no;
```

En el supuesto de necesitar cierta recursión para un número reducido de usuarios es recomendable emplear la cláusula *allow-query-cache* para restringir el acceso a la caché, y, más concretamente, a los servicios recursivos de DNS. Esta configuración evitaría que, de forma inadvertida, el servidor se convirtiera en un OPEN DNS resolver (servidor DNS capaz de resolver de forma recursiva cualquier petición enviada desde cualquier origen de forma abierta, tal y como lo hacen los DNS's de Google, OpenDNS, etc).

Esto se aconseja, únicamente, para entornos de pruebas o internos, nunca en producción para servidores expuestos a Internet de tipo autoritativo (sí para los reenviadores), con el fin de evitar que los servidores DNS pudieran ser utilizados en ataques de denegación de servicio (*DDoS*), ataques de envenenamiento de caché, limitación en explotación de fallos por saturación o bugs del software,

El formato de la cláusula de restricción es la siguiente:

```
allow-query-cache { <address-match-list>;};
```

donde <*address-match-list*> está formada por uno o más elementos <*element*>:

```
address_match_list = <element> ; [ <element>; ... ]
```

donde cada uno de ellos puede tener la siguiente sintaxis:

```
element = [!] (<ip> [/<prefix>] | key <key-name> | "<acl_name>" | { <address_match_list> } )
```

Además de estos, existen 4 denominaciones para *<address-match-list>* predefinidos:

- *none* → ninguna dirección IP.
- *any* → cualquier dirección IP.
- *localhost* → tanto la IP de loopback 127.0.0.1 como cualquier IP asociada al propio servidor BIND.
- *localnets* → incluye la IP de loopback, así como cualquier dirección IP y máscaras de subred asociadas al servidor BIND.

5.4.2 Ocultación de versión

Por defecto, en una instalación de *BIND* se muestra la versión instalada, como puede aprecisarse a través de la ejecución del siguiente comando:

```
[tfg@master00 ~]$ dig -t txt -c chaos VERSION.BIND @172.16.127.129

                  ; <<>> DiG 9.11.36-RedHat-9.11.36-8.el8 <<>> -t txt -c chaos VERSION.BIND
@172.16.127.129
                  ;; global options: +cmd
                  ;; Got answer:
                  ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1600
                  ;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL:
1
                  ;; WARNING: recursion requested but not available

                  ;; OPT PSEUDOSECTION:
                  ; EDNS: version: 0, flags:; udp: 1232
                  ; COOKIE: 65ad62aa38782cb2d58573156473845eea133be429edf58f (good)
                  ;; QUESTION SECTION:
                  ;VERSION.BIND. CH TXT

                  ;; ANSWER SECTION:
                  VERSION.BIND. 0 CH TXT "9.11.36-RedHat-9.11.36-8.el8"

                  ;; AUTHORITY SECTION:
                  version.bind. 0 CH NS version.bind.

                  ;; Query time: 0 msec
                  ;; SERVER: 172.16.127.129#53(172.16.127.129)
                  ;; WHEN: dom may 28 12:42:06 EDT 2023
                  ;; MSG SIZE  rcvd: 136
```

Es conveniente no mostrar la versión de *BIND* en uso, debido a que facilita información extra que se puede aprovechar para explotación vulnerabilidades del paquete de software instalado. Para evitar que el servidor muestre la versión

instalada, se editará el fichero de configuración */etc/named.conf* y dentro de la sección *options* se añade la siguiente línea:

```
version         "not currently available";
```

Una vez aplicado el cambio, si repetimos el anterior comando, se obtiene el siguiente resultado:

```
[tfg@master00 ~]$ dig -t txt -c chaos VERSION.BIND @172.16.127.129

; <<>> DiG 9.11.36-RedHat-9.11.36-8.el8 <<>> -t txt -c chaos VERSION.BIND @172.16.127.129
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37739
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: e9ff010b1557eeecbe57c65464738590723ac75fda1edc52 (good)
;; QUESTION SECTION:
;VERSION.BIND. CH TXT

;; ANSWER SECTION:
VERSION.BIND. 0 CH TXT "not currently available"

;; AUTHORITY SECTION:
version.bind. 0 CH NS version.bind.

;; Query time: 0 msec
;; SERVER: 172.16.127.129#53(172.16.127.129)
;; WHEN: dom may 28 12:47:12 EDT 2023
;; MSG SIZE  rcvd: 131
```

5.4.3 Reducir la información en la respuesta

A la hora de contestar una query DNS, el servidor por defecto suele dar una respuesta como esta:

```
;; QUESTION SECTION:
;www.midominio.es.                IN      A
;; ANSWER SECTION:
www.midominio.es.          28800   IN      A       172.16.127.200
;; AUTHORITY SECTION:
midominio.es.              28800   IN      NS      ns1.midominio.es.
midominio.es.              28800   IN      NS      ns2.midominio.es.
;; ADDITIONAL SECTION:
ns1.midominio.es.          28800   IN      A       172.16.127.130
ns2.midominio.es.          28800   IN      A       172.16.127.131
;; Query time: 31 msec
;; SERVER: 172.16.127.129#53(172.16.127.129)
```

```
;; WHEN: dom may 28 12:57:18 EDT 2023
;; MSG SIZE  rcvd: 125
```

Esta respuesta contiene más información que la solicitada, pues está añadiendo información extra como los servidores de nombres NS y la resolución de los mismos. Si habilitamos esta opción, la respuesta será más breve, además de ser la estrictamente solicitada, sin información extra, haciendo que la carga del servidor se vea disminuída. La respuesta con esta opción habilitada sería:

```
;; QUESTION SECTION:
;www.midominio.es.                  IN      A

;; ANSWER SECTION:
www.midominio.es.           28800   IN      A       172.16.127.200

;; Query time: 31 msec
;; SERVER: 172.16.127.129#53(172.16.127.129)
;; WHEN: dom may 28 12:59:54 EDT 2023
;; MSG SIZE  rcvd: 57
```

Para habilitar esta opción se debe indicar el siguiente parámetro en la sección global del fichero de configuración */etc/named.conf*

```
minimal-responses yes;
```

5.4.4 Formato especial para nodos expuestos a Internet

Por defecto, el único servidor DNS que poseerá las zonas en un formato legible (*text*) será el Maestro, de tal forma que los esclavos no posean un formato legible (*raw*). Normalmente, se emplea en este formato, no sólo por aumentar la seguridad, sino por agilizar el proceso de carga de los servidores DNS esclavos. La conversión de la zona supone la ejecución del siguiente comando:

```
~ # named-compilezone -f raw -F text -o view-0.midominio.es.dns.txt midominio.es view-0.midominio.es.dns
```

Esto generará un fichero midominio.es.dns.txt con las entradas totalmente legibles (*text*)

Según se indica en [12] en versiones posteriores a la *9.10* de *BIND*, surgió este formato que era tan seguro como *raw* pero más rápido, gracias a que la zona no necesita ser preprocesada. En caso de utilizar este formato, se puede emplear el siguiente comando para convertir una zona a un texto legible:

```
~ # named-compilezone -f map -F text -o view-0.midominio.es.dns.txt midominio.es
view-0.midominio.es.dns
```

En el caso que nos ocupa, caso y siempre y cuando dispongamos de una versión posterior a la indicada, se empleará el parámetro *masterfile-format map;* en la definición de zona:

```
zone midominio.es {
    type secondary;
    file "slave/midominio.es";
    masterfile-format map;
    primaries {172.16.127.129;};
};
```

5.5 Securización del Sistema Operativo

En este apartado se incluirán las principales medidas de bastionado para el sistema operativo de los servidores que ofrecerán el servicio DNS, y su amplificación será especialmente crítica para aquellos nodos expuestos al exterior, concretamente para los nodos secundarios o esclavos.

5.5.1 Puertos abiertos

Según lo indicado en el apartado relativo a la securización de la red 5.2 dispondremos de dos interfaces de red, una de gestión y otra de servicio. Se limitará la apertura de puertos a la interfaz correspondiente no sólo a través de un cortafuegos, sino desde el propio servicio, de modo que se expongan, únicamente, los puertos en las interfaces que lo necesitan.

En atención a este criterio, se muestran, a continuación, los puertos expuestos asociados a la interfaz correspondiente:

- Nodos DNS
  - Interfaz de servicio - Acceso Internet
    - Puerto 53/UDP DNS
    - Puerto 53/TCP DNS
  - Interfaz de gestión - Acceso red local
    - Puerto 7766/TCP Transferencia de zona
    - Puerto 953/TCP Gestión por RDNC
    - Puerto 22/TCP Acceso SSH
- Equipo de gestión
  - Interfaz de gestión - Acceso red local
    - Puerto 22/TCP Acceso SSH

5.5.2 SELinux

Uno de los métodos más efectivos a la hora de definir los controles de acceso para las aplicaciones, procesos y archivos dentro de un sistema operativo Linux,

es a través de la herramienta *SELinux* [13]. Esta herramienta está instalada por defecto en las distribuciones heredadas de *RedHat*, por lo que no será preciso instalarla en *RockyLinux*. Por defecto está activada en modo *Enforced*.

Al objeto de evitar que los ficheros sean accedidos o modificados por procesos que no estén previamente autorizados, se dejará acttivada en en este modo. De esta forma únicamente el proceso asociado a *BIND* tendrá accesibilidad a los archivos de zona y de configuración del DNS.

Configuración

En caso de no estar activado en este modo, se ha de activar SELinux en modo *Enforced* tanto en el estado actual del sistema como ante un reinicio del mismo.

Para comprobar el estado, y activarlo en caso de ser necesario, se procede a ejecutar:

```
~ # getenforce
~ # setenforce Enforcing
```

A continuación se modificará el fichero de configuración de SELinux para que por defecto se active el modo Enforcing durante el arranque. Se edita el fichero de configuración */etc/sysconfig/selinux* y se modifica la opción SELINUX=permissive por SELINUX=enforcing

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of three two values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are
protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Contexto de los ficheros

El contexto de seguridad consiste en una base de datos en la que se permite o se deniega el acceso entre ficheros, procesos y aplicativos. Si se desea modificar el contexto de los ficheros se debe emplear la librería de gestión de *SELinux semanage*. Para poder disponer de ella, debe ser previamente instalada:

```
~ # yum install policycoreutils-python-utils -y
```

Dado que la creación de contextos específicos para los ficheros de DNS es algo complejo y laborioso, esta herramienta nos permite utilizar plantillas de DNS y copiar contextos entre ficheros. En el siguiente caso el fichero *named.conf* contiene el contexto correcto, mientras que el fichero de respaldo *named.conf.original* no. Para copiarlo de uno a otro:

```
~ # ls -Z /etc/named.conf.*
    -rw-r-----. root named  system_u:object_r:named_conf_t:s0 named.conf
    -rw-r-----. root root   unconfined_u:object_r:etc_t:s0   named.conf.original
~ # semanage fcontext -a -e /etc/named.conf /etc/named.conf.original
~ # restorecon -R -v /etc/named.conf.original
~ # ls -Z /etc/named.conf.*
    -rw-r-----. root named  system_u:object_r:named_conf_t:s0 named.conf
    -rw-r-----. root root   unconfined_u:object_r:named_conf_t:s0 named.conf.original
```

En caso de que se pretenda fijar un contexto concreto para un fichero o conjunto de ficheros dados, debe realizarse del siguiente modo:

```
~ # ls -Z /etc/named.conf.2023-05-17
    -rw-r-----. root root   unconfined_u:object_r:etc_t:s0   named.conf.2023-05-17
~ # chcon -R --type=named_conf_t /etc/named.conf.2023-05-17
~ # ls -Z /etc/named.conf.2023-05-17
    -rw-r-----. root root   unconfined_u:object_r:named_conf_t:s0 named.conf.2023-05-17
```

Los contextos existentes para los ficheros del servicio named en función de las carpetas en las que estén alojados, o en función de los binarios que son instanciados, son los siguientes. Contextos SELinux para *BIND - named*

- *named_cache_t* → contexto asociado a los ficheros almacenados en el directorio /var/cache.
- *named_checkconf_exec_t* → Contexto asociado a la ejecución del binario asociado al chequeo de la configuración: named_checkconf
- *named_conf_t* → contexto asociado a los ficheros almacenados en el directorio /etc, asociado a ficheros de configuración.
- *named_exec_t* → contexto asociado a la ejecución de los binario asociados a la ejecución del demonio named.
- *named_initrc_exec_t* → contexto asociado a la ejecución de los binario asociados a los scripts de inicio del demonio named.
- *named_keytab_t* → contexto asociado a los ficheros de autenticación keytab de kerberos.
- *named_log_t* → contexto asociado a los ficheros almacenados en el directorio /var/log, asociado a ficheros de sucesos del servicio.
- *named_tmp_t* → contexto asociado a los ficheros almacenados en el directorio /tmp, asociado a ficheros temporales del servicio.
- *named_var_run_t* → contexto asociado a los ficheros almacenados en el directorio /run, asociado a ficheros de control de ejecución del servicio.
- *named_zone_t* → contexto asociado a los datos de ficheros de zona.

- *named_initrc_exec_t* → contexto asociado a la ejecución de los binario asociados a los scripts de inicio del demonio named.
- *named_keytab_t* → contexto asociado a los ficheros de autenticación keytab de kerberos.
- *named_log_t* → contexto asociado a los ficheros almacenados en el directorio /var/log, asociado a ficheros de sucesos del servicio.
- *named_tmp_t* → contexto asociado a los ficheros almacenados en el directorio /tmp, asociado a ficheros temporales del servicio.
- *named_var_run_t* → contexto asociado a los ficheros almacenados en el directorio /run, asociado a ficheros de control de ejecución del servicio.
- *named_zone_t* → contexto asociado a los datos de ficheros de zona.

Puertos permitidos

Por defecto en *SELinux* para el DNS están autorizados los siguientes puertos:

```
# semanage port -l  | grep dns
dns_port_t                     tcp      53
dns_port_t                     udp      53
dnssec_port_t                  tcp      8955
```

Es preciso autorizar el puerto 7766 empleado para la comunicación y transferencia por las interfaces de gestión en todos y cada uno de los servidores DNS, maestro y esclavos:

```
# semanage port -a -t dns_port_t -p tcp 7766
```

Si se vuelve a comprobar los puertos permitidos para el DNS:

```
# semanage port -l | grep dns
dns_port_t                     tcp      7766, 53
dns_port_t                     udp      53
dnssec_port_t                  tcp      8955
```

Permiso escritura zona maestra

Por defecto, en *SELinux* para el DNS Maestro no está permitida la escritura de los ficheros de zona generados a raíz de una actualización dinámica del DNS (ficheros con extensión *.jnl*)

```
# getsebool named_write_master_zones
named_write_master_zones --> off
```

Será necesario autorizar la escritura de estos ficheros para permitir la actualización dinámica de la zona:

```
# setsebool -P named_write_master_zones true
```

Si se vuelve a comprobar los puertos permitidos para el DNS:

```
# getsebool named_write_master_zones
named_write_master_zones --> on
```

5.5.3 Cortafuegos

Procede definir los permisos de acceso al servidor a través del *firewall* instalado en el sistema operativo. En este caso, utilizaremos *iptables*, al tratarse de una herramienta disponible en la versión de *RockyLinux* elegida.

Configuración general

La configuración general del cortafuegos de un servidor con el servicio BIND es la siguiente:

```
\#
\# Rules for BIND Service: named and rndc
\#

\# Rules for BIND Service: Remote commands via RNDC
-A INPUT-MANAGEMENT -p tcp -m state --state NEW -m tcp --dport 953 -j ACCEPT

\# Rules for Named Service: DNS queries
-A INPUT-SERVICE -p udp -m state --state NEW -m udp --dport 53 -j ACCEPT

\# Specific Rules for master named server to allow zone transfers
\-A INPUT-SERVICE -p tcp -m state --state NEW -m tcp --dport 53 -j ACCEPT
```

Es preciso tener en cuenta que la *INPUT-MANAGEMENT* se refiere al tráfico de entrada en la interfaz de gestión, y que *INPUT-SERVICE* se refiere al tráfico entrante para la interfaz de servicio.

Configuración específica. Nodo maestro

Transferencia de Zona - DDNS En el caso particular de un servidor master de una o más zonas DNS, se ha de habilitar el tráfico udp/tcp relativo a la

transferencia de datos a través de la interfaz de gestión. Con esto, lo que conseguimos es separar completamente el tráfico de gestión (transferencia de zona y actualización dinámica), del de las consultas DNS. Las consultas no estarán permitidas en el master. Aplicaremos la siguiente regla de iptables en la cadena INPUT-MGMT:

```
# Specific Rules for master named server to allow zone transfers
-A INPUT-MANAGEMENT -p tcp -m state --state NEW -m tcp --dport 7766 -j ACCEPT
-A INPUT-MANAGEMENT -p udp -m state --state NEW -m udp --dport 7766 -j ACCEPT
```

Se podría limitar las IPs origen de los servidores slave que están autorizados a realizar solicitudes de transferencia de zonas mediante reglas del tipo:

```
# Specific Rules for master named server to allow zone transfers
-A INPUT-MANAGEMENT -p tcp -m state --state NEW -m tcp -s <address1>[,<address2>]
... --dport 7766 -j ACCEPT
-A INPUT-MANAGEMENT -p udp -m state --state NEW -m udp -s <address1>[,<address2>]
... --dport 7766 -j ACCEPT
```

o bien, si se trata de rangos de direcciones IP contiguas, empleando el módulo iprange:

```
# Specific Rules for master named server to allow zone transfers
-A INPUT-MANAGEMENT -p tcp -m state -m iprange --state NEW -m tcp --src-range
<address1>-<address2>]--dport 7766 -j ACCEPT
-A INPUT-MANAGEMENT -p udp -m state -m iprange --state NEW -m udp --src-range
<address1>-<address2>]--dport 7766 -j ACCEPT
```

Como se puede comprobar, no se permitiría la consulta en un puerto distinto al indicado, y estaría limitado a los orígenes autorizados en el fichero de configuración de BIND. El puerto 53 no estaría disponible en ningún caso.

Gestión servicio: RNDC También ha de habilitarse el acceso a la gestión del servicio named a través del puerto 953/tcp.

```
# Specific Rules for master named server to allow zone transfers
-A INPUT-MANAGEMENT -p tcp -m state --state NEW -m tcp --dport 953 -j ACCEPT
```

En caso de limitar el acceso a un determinado número de IP's, habría que especificarlo como en el apartado anterior. Tendrían que estar incluidas las IP's de todas las máquinas que tuvieran permitidas las actualizaciones dinámicas de DNS.

Configuración específica. Nodo esclavo

Transferencia de zona y consulta DNS En el caso particular de un servidor slave de una o más zonas DNS, se ha de habilitar el tráfico udp/tcp relativo a la notificación de actualización de la zona DNS (previo a la transferencia de zona) en la interfaz de Management, así como el tráfico udp/tcp relativo a la consulta DNS en la interfaz de Service. Con esto, lo que conseguimos es separar completamente el tráfico de gestión (transferencia de zona y actualización dinámica) del de las consultas DNS. Las consultas sí estarán permitidas en el slave. Se aplicará la siguiente regla de iptables en la cadena *INPUT-MGMT*:

```
# Specific Rules for master named server to allow zone transfers
-A INPUT-MANAGEMENT -p tcp -m state --state NEW -m tcp --dport 7766 -j ACCEPT
-A INPUT-MANAGEMENT -p udp -m state --state NEW -m udp --dport 7766 -j ACCEPT
```

En la cadena *INPUT-SERVICE*:

```
# Specific Rules for master named server to allow DNS queries
-A INPUT-SERVICE -p tcp -m state --state NEW -m tcp --dport 53 -j ACCEPT
-A INPUT-SERVICE -p udp -m state --state NEW -m udp --dport 53 -j ACCEPT
```

Gestión servicio: RNDC También ha de habilitarse el acceso a la gestión del servicio *named* a través del puerto 953/tcp.

```
# Specific Rules for master named server to allow control8.3.3.2 RNDC
-A INPUT-MANAGEMENT -p tcp -m state --state NEW -m tcp --dport 953 -j ACCEPT
```

6 Zonas

La zona DNS abarca todos los registros asociados a un dominio que se van a resolver (traducir nombre a IP o viceversa) bajo petición. Cabe destacar que no se debe confundir el servidor que ofrece el servicio de resolución de nombres, frente a la zona del dominio. La relación entre ambos es de varios a varios, es decir, un servidor DNS puede albergar varios dominios y, además, un dominio puede ser albergado por varios servidores DNS, aunque sólo uno puede ser autoritativo.

En la arquitectura vista en el capítulo 2, un dominio y su zona asociada podrán ser servidos por cualquiera de los nodos esclavos, pero, únicamente, el maestro será autoritativo para realizar cambios en la zona, y, a su vez, cada uno de estos nodos pueden servir múltiples dominios, ofreciendo resolución de nombres para todos ellos.

Existen, en esencia, dos tipos de zonas principalmente: las maestras y las esclavas. Los cambios se realizan en las primeras, y son transferidos a las zonas esclavas mediante transferencias de zonas. Estos no son los dos únicos tipos de zonas existentes, pero son las dos principales para un DNS autoritativo (más en [3] o en `https://www.zytrax.com/books/dns/ch7/zone.html#type`)

La actualización dinámica de las zonas se trata en el capítulo 4.2

6.1 Estándar

A continuación, se muestran las zonas estándar que se suelen cargar y que representan nombres y direcciones locales tanto para IPv4 como para IPv6 [7]. Está presente al objeto de que, en caso de que un cliente consulte una dirección local, exista respuesta y no se genere un error.

```
zone "localhost.localdomain" IN {
    type master;
    file "named.localhost";
    allow-update { none; };
```

```
};

zone "localhost" IN {
    type master;
    file "named.localhost";
    allow-update { none; };
};

zone "1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa" IN {
    type master;
    file "named.loopback";
    allow-update { none; };
};

zone "1.0.0.127.in-addr.arpa" IN {
    type master;
    file "named.loopback";
    allow-update { none; };
};

zone "0.in-addr.arpa" IN {
    type master;
    file "named.empty";
    allow-update { none; };
};
```

6.2 Respuesta condicional (*split DNS*)

Consiste en ofrecer una respuesta diferente en función de quién realiza la consulta. Este tipo de comportamiento es muy útil en determinados casos en los que nos interesa que la respuesta a una consulta no sea única y dependa del origen de quién la realiza, bien sea a nivel de red, o a nivel geográfico.

6.2.1 Vistas

Con el fin de generar esas distintas respuestas ante una misma consulta, se configuran las vistas. Una vista no es más que la definición de una zona, pero de forma condicional, en base a dos criterios fundamentalmente:

- IP origen
- Geolocalización de la IP (País de origen)

De esta forma, a través de las vistas, pueden definirse diferentes respuestas en función de la IP origen o en función de la geolocalización de la IP a través del módulo *GeoIP* que incorpora el paquete *BIND*.

En la figura 6.1 se muestra un ejemplo de vista típico en base a listas de acceso o *ACLs*. En primer lugar, se definen dos listas de acceso: la primera incorpora todos los rangos de IP privadas que se suelen ser empleadas en entornos locales; la segunda es la primera denegada, es decir, todas las IPs que no son locales o las IPs públicas de Internet.

Para este ejemplo, en la sección de vistas se distingue entre:

- *view intranet*: comprenderá a todos los clientes de ámbito local, y contiene un único dominio myzone.example, de tipo maestro y con una localización concreta.
- *view internet*: incluye todos los clientes de internet, con el mismo dominio, pero con una base de datos de registros totalmente diferente a la anterior

En síntesis, es posible realizar resoluciones de nombres diferentes en función de si el cliente pertenece a la red local o no. Si la IP origen de la consulta es local se podrá resolver un nombre a una IP local. En caso de que la petición venga desde Internet, deberá responder con una dirección pública. En cualquier caso, con las vistas se evita la necesidad de configurar servidores diferentes para el mismo dominio.

Esto no es más que un ejemplo de los posibles empleos que podemos darles a las vistas. En redes relativamente grandes no es extraño que pueda surgir esta necesidad por reglas de *NAT (Network Address Traslation)* o por diferentes enrutamientos. En el trabajo que nos ocupa no se emplearán vistas, al no disponer de un caso que justifique su uso.

```
##
# ACL'S
##
acl intranetIP {
        127.0.0.1;
        10.0.0.0/8;
        172.16.0.0/12;
        192.168.0.0/16;
};
acl internetIP   { !key intranetIP; };

##
# Views
##
view intranet {
    match-clients { intranetIP; };

    zone "myzone.example" {
        type master;
        file "intranet/db.myzone.example";
    };
};

view internet {
    match-clients { internetIP; };

    zone "myzone.example" {
        type master;
        file "internet/db.myzone.example";
    };
};
```

Figura 6.1: Ejemplo de vistas: intranet vs internet

7 Eventos y Monitorización

El servidor DNS *BIND* dispone de gestor de eventos que son generados a medida que se producen acciones en el servicio *named*. Estos eventos o *logs* obedecen a diferentes categorías, que engloban, a su vez, diferentes apartados de la gestión del DNS. Por defecto en *BIND* existe una configuración en un único fichero, pero también permite generar ficheros en función de la categoría del evento que se desea registrar.

7.1 Configuración de los eventos por defecto

La configuración que aparece por defecto recién instalado *BIND* es la siguiente:

```
logging {
    channel default_debug {
        file "data/named.run";
        severity dynamic;
    };
};
```

Esto almacenará en el directorio */var/named/data/named.run* todos los logs del demonio named. A continuación se describen los elementos que aparecen en la configuración:

- *channel ...* → Indica la fuente de logs, que puede llevar asociada una categoría, severidad, etc. El canal por defecto es *default_ debug* que incluye todos los eventos a excepción de las consultas DNS.
- *data ...* → Ruta relativa del fichero dentro del directorio */var/named* donde se guardarán todos los eventos.
- *severity ...* → Indica a partir de qué criticidad los eventos serán almacenados. *dynamic* significa modo depuración (*debug*)

7.2 Categorización de los registros

Para que que los logs queden correctamente categorizados y exista un fichero de log en función del tipo de información que se está generando, ha de modificarse el fichero de configuración de tal forma que cumpla los siguientes criterios:

- Los logs se separarán en ficheros situados en /var/log/named en función de las categorías de interés
- Los logs estarán rotados
- Se enviarán simultáneamente al demonio syslog de forma no permanente, para facilitar el filtrado de los mismos a través de la herramienta journalctl. En cada reinicio se perderán los logs asociados al syslog, y se conservarán los asociados a los ficheros de /var/log/named.

Dada la extensión de la configuración de eventos, se separarán en un fichero específico. Así, en el fichero */etc/named.conf* aparecerá:

```
logging {
include "/etc/named/named-logging.conf";
};
```

En el fichero */etc/named/named-logging.conf* se realizará la categorización de los eventos.

```
channel default_syslog {
        syslog;
        print-severity yes;
        print-category yes;
        severity dynamic;
};
channel default_channel {
        file "/var/log/named/default.log";
        print-time yes;
        print-category yes;
        print-severity yes;
        severity dynamic;
};
channel general_channel {
        file "/var/log/named/general.log";
        print-time yes;
        print-category yes;
        print-severity yes;
        severity dynamic;
};
channel xfer_channel {
        file "/var/log/named/xfer.log";
        print-time yes;
        print-severity yes;
```

```
        severity dynamic;
};
channel notify_channel {
        file "/var/log/named/notify.log";
        print-time yes;
        print-severity yes;
        severity dynamic;
};
channel network_channel {
        file "/var/log/named/network.log";
        print-time yes;
        print-severity yes;
        severity dynamic;
};
channel queries_channel {
        file "/var/log/named/queries.log";
        print-time yes;
        print-severity yes;
        severity dynamic;
};
channel query-errors_channel {
        file "/var/log/named/query-errors.log";
        print-time yes;
        print-category yes;
        print-severity yes;
        severity dynamic;
};
channel lame-servers_channel {
        file "/var/log/named/lame-servers.log";
        print-time yes;
        print-category yes;
        print-severity yes;
        severity dynamic;
};
channel rate-limit_channel {
        file "/var/log/named/rate-limit.log";
        print-time yes;
        print-category yes;
        print-severity yes;
        severity dynamic;
};
channel dynamic-updates_channel {
        file "/var/log/named/ddns.log";
        print-time yes;
        print-category yes;
        print-severity yes;
        severity dynamic;
};
category default { default_channel; default_syslog; };
category general { general_channel; default_syslog; };
category xfer-out { xfer_channel; default_syslog; };
category xfer-in { xfer_channel; default_syslog; };
category notify { notify_channel; default_syslog; };
category network { network_channel; default_syslog; };
#category queries { queries_channel; default_syslog; };
#category query-errors { query-errors_channel; default_syslog; };
category lame-servers { lame-servers_channel; default_syslog; };
category rate-limit { rate-limit_channel; default_syslog; };
```

```
category update { dynamic-updates_channel; default_syslog; };
category update-security { dynamic-updates_channel; default_syslog; };
```

En la parte final de esta configuración se ve cómo se muestran varias entradas de tipo *category*. Aquí es donde se realiza la asociación entre la categoría y el canal. Es posible publicar varias categorías en un mismo canal, lo que aporta cierta flexibilidad a la hora de generar eventos para el equipo de seguridad, por ejemplo.

La lista completa de categorías puede verse en [22]. Se aprecia que en la configuración se muestran un par de las categorías comentadas, concretamente las categorías *queries* y *query-errors*. Esto es debido a que los eventos relativos tanto a las consultas como a los errores derivados de cada consulta son tan elevados que suelen provocar (en función de la carga del servidor) una bajada de rendimiento en el servicio DNS. Un servidor DNS expuesto a Internet puede recibir una cantidad ingente de consultas por segundo, algunas exitosas y otras que generan errores diversos. No tiene sentido almacenar esta cantidad de logs a menos que, de modo puntual, sea necesaria una auditoría de seguridad, por ejemplo. Es posible visualizar estadísticas en tiempo real, como se indica en 7.4

Se procede al análisis de algunos de los parámetros de la configuración avanzada de registros.

- *print-time yes* → por cada registro indica la hora/día del mismo
- *print-severity yes* → por cada registro indica su severidad
- Categorías
  - *general* → captura todas las categorías de forma general. Útil cuando no se dispone de todos los eventos categorizados.
  - *xfer-out* → envíos de transferencia de zona
  - *xfer-in* → recepciones de transferencia de zona
  - *notify* → notificaciones enviadas entre servidor maestro y esclavo
  - *network* → operaciones relacionadas con la red
  - *lame-servers* → fallos de configuración en otros servidores remotos
  - *rate-limit* → eventos relativos a la superación de la tasa de consultas fijada. Sería válido para encontrar proveedores de servicio cuyos DNS realizasen una tasa elevada de consultas.
  - *update* → actualizaciones dinámicas de zona a través del protocolo *DDNS* para la actualización de los registros.
  - *udpate-security* → autorizaciones y denegaciones de actualizaciones dinámicas. Detectaría equipos que realizasen actualizaciones de zona sin permisos.

Para que esta configuración sea efectiva es necesario crear las carpetas y permisos adecuados. Se realizará a través de los siguientes comandos:

```
        # Creamos el directorio donde se almacenarán los logs
        ~ # mkdir /var/log/named
        # Establecemos los permisos adecuados
        ~ # chown -R named:named /var/log/named
        # Establecemos el contexto SELinux adecuado (named_log_t)
        ~ # chcon --type=named_log_t /var/log/named
        # Por último sólo tenemos que recargar el demonio named para que se escriban
los logs en la nueva ubicación
        ~ # systemctl reload named.service
```

Con esta configuración no se escriben los logs en el fichero /var/named/data/named.run (configuración por defecto del paquete *BIND* en *RedHat*), pero quedarán categorizados en varios archivos dentro de */var/log/named*, y a través del comando: *journalctl -u named*

7.3 Rotado de logs

La cantidad de eventos generados por un servidor DNS puede ser enorme (incluso excluyendo las consultas). El tamaño de los ficheros que almacenan los eventos no debieran superar un tamaño determinado, y será necesario *rotarlos*: se trunca el fichero, se cambia de nombre la parte con los datos más antiguos, se añade la fecha o simplemente un número y se comprime. Será necesario fijar esta configuración para el servicio *logrotate*.

Se procede a editar el fichero de configuración */etc/logrotate.d/named* y cambiar la primera línea, en la que se especifica el fichero de log por defecto, por la nueva carpeta de logs. El resultado será:

```
/var/log/named/*.log {
    missingok
    notifempty
    sharedscripts
    delaycompress
    size 50M
    compress
    rotate 10
    su named named
    create 0644 named named
    postrotate
        /usr/bin/systemctl reload named.service > /dev/null 2>&1 || true
        /usr/bin/systemctl reload named-chroot.service > /dev/null 2>&1 || true
        /usr/bin/systemctl reload named-sdb.service > /dev/null 2>&1 || true
        /usr/bin/systemctl reload named-sdb-chroot.service > /dev/null 2>&1 || true
        /usr/bin/systemctl reload named-pkcs11.service > /dev/null 2>&1 || true
    endscript
}
```

El detalle de configuración ha sido extraído de [28], pero no se detallan los parámetros por escapar del objetivo de este proyecto.

```
...
[View: _bind]
++ Name Server Statistics ++
1052270153 IPv4 requests received
948908766 requests with EDNS(0) received
        141 requests with unsupported EDNS version received
        43 requests with TSIG received
        43 requests with invalid signature
    765856 TCP requests received
666260421 auth queries rejected
    2232485 recursive queries rejected
        2699 transfer requests rejected
    91368 update requests rejected
1027710066 responses sent
    94123 truncated responses sent
925964905 responses with EDNS(0) sent
236944249 queries resulted in successful answer
378189715 queries resulted in authoritative answer
        5 queries resulted in non authoritative answer
        5 queries resulted in referral answer
    78049254 queries resulted in nxrrset
        60 queries resulted in FORMERR
    63196212 queries resulted in NXDOMAIN
    24560087 queries dropped
668201177 other query failures
    24560087 responses dropped for rate limits
    94095 responses truncated for rate limits
...
```

Figura 7.1: Porción de estadísticas recopiladas en *named.stats*

7.4 Monitorización

Las directivas *statistics-file, memstatistics-file y zone-statistics* que se muestran en la configuración de *BIND* (ver 3.2.1) posibilita la extracción de información acerca del comportamiento de BIND. El comando *rndc stats* vuelca las estadísticas en el fichero *named.stats*. Este fichero proporciona la información más completa acerca del estado actual del servidor, y si se habilita la directiva de las estadísticas de zona, facilita un desglose por zona. En la 7.1 se muestran estadísticas globales de un servidor DNS. Existen programas que permiten obtener datos en tiempo real, como *dnstop* (ver [23]). En la figura 7.2 aparece una estadística en tiempo real en la que se muestran varias IP's de Internet, y la tasa de consultas que realiza cada una.

Dado que el objetivo de este trabajo no es monitorizar un servidor DNS, no se implementará una herramienta de monitorización para el servidor DNS. Posibles opciones a día de hoy son *Nagios* (y sus derivados), *LibreNMS*, *Zabbix* o *Prometheus*. Pueden verse otras consideraciones al respecto en [24].

```
Queries: 25 new, 3120 total

Sources            Count      %   cum%
-------------- --------- ------ ------
54.154.201.73        102    3.3    3.3
69.171.249.129        64    2.1    5.3
69.171.231.130        61    2.0    7.3
3.228.171.235         53    1.7    9.0
10.199.71.9           49    1.6   10.5
10.199.71.8           49    1.6   12.1
10.199.71.7           49    1.6   13.7
10.199.71.6           48    1.5   15.2
193.232.160.51        33    1.1   16.3
69.171.249.172        28    0.9   17.2
3.238.178.158         27    0.9   18.0
3.209.83.109          25    0.8   18.8
3.228.172.89          25    0.8   19.6
193.232.160.49        23    0.7   20.4
3.209.84.250          23    0.7   21.1
44.192.176.35         23    0.7   21.9
93.158.164.103        21    0.7   22.5
3.238.129.119         19    0.6   23.1
3.228.181.100         19    0.6   23.8
187.86.214.16         19    0.6   24.4
193.232.160.48        18    0.6   24.9
44.197.205.112        18    0.6   25.5
3.228.181.69          17    0.5   26.1
3.239.155.19          17    0.5   26.6
```

Figura 7.2: Consultas por segundo en tiempo real con *dnstop* [23]

8 Alta disponibilidad y balanceo

8.1 Alta disponibilidad del servicio

Dada la naturaleza crítica del servicio DNS, este deberá estar siempre disponible, con independencia de las indisponibilidades que puedan surgir en los centros de procesamiento de datos.

Las *zonas de disponibilidad* pueden ser definidas como islas que no se ven afectadas por los errores ocasionados en otras zonas del mismo centro de datos. Este aislamiento es a todos los niveles: red eléctrica, red de datos, almacenamiento y capacidad de cómputo. En el supuesto de contar con diferentes regiones geográficas, podría hablarse de *regiones de disponibilidad*.

Con el fin de proporcionar un servicio de DNS ininterrumpido, se aplicará la siguiente política de ubicación:

- Se dispondrán dos nodos esclavos en dos regiones diferentes.
- El resto de nodos dentro de cada región estarán en diferentes zonas.

Con esta disposición, el servicio de resolución de nombres estará siempre asegurado, bien en modo de sólo lectura (sólo consultas), o en modo de lectura/escritura, en función de la disponibilidad del nodo maestro.

Se selecciona esta política dada la necesidad de superar cualquier contingencia, bien causada por una incidencia en el centro de datos, que se superaría con las diferentes zonas de disponibilidad, bien por contingencias de origen natural o geopolítico, que se superarían con la disposición de los nodos en diferentes regiones.

8.2 Importancia del TTL (*Time To Live*)

8.2.1 Empleo del TTL

El *TTL (Time To Live)* es el período temporal que se establece para la resolución de un registro presente en el servidor autoritativo del dominio asociado, y que tiene como objetivo indicar el tiempo de validez de esa resolución. Esto permite que el cliente pueda almacenar en su caché (memoria temporal) la respuesta, de tal forma que si se realiza una nueva consulta del mismo registro en un tiempo inferior al TTL, la respuesta estará cacheada localmente y por lo tanto la respuesta será inmediata. Una vez superado el período de TTL y ante una nueva consulta, la caché quedaría invalidada y habría que realizar nuevamente la consulta al DNS autoritativo.

La caché puede almacenarse no sólo en los equipos clientes que realizan las consultas, sino también en los servidores DNS recursivos que utilizan estos clientes para realizar la consulta. Así, si dos clientes diferentes realizan la misma consulta DNS en un tiempo inferior al TTL, la respuesta al segundo cliente se realizará en base a la caché.

El valor del TTL se puede fijar por cada registro, o, de manera global, a través de la directiva *$TTL*, véase [15]. La política a seguir suele ser fijada en base al siguiente criterio:

- Se fija un TTL reducido (5 minutos) para aquellos registros que cambien de manera periódica
- Los tiempos de TTL superiores a 5 horas suelen establecerse en los registros con pocas o nulas modificaciones

Los tiempos de TTL elevados permiten cachear la resolución durante más tiempo, lo que posibilita la obtención de una respuesta en base a una caché válida a pesar de la eventual falta de respuesta de un servidor DNS.

8.2.2 Asignación TTL por tipo de registro

No todos los tipos de registro suelen poseer un mismo valor de TTL. El registro de tipo *NS* se refiere a la IP del servidor de nombres autorizado para resolver peticiones sobre un dominio dado. No es usual cambiar el servidor de nombres, y, en todo caso, si existiese intencionalidad de un cambio a corto plazo, se reduciría el TTL del registro. En síntesis, el registro de tipo *NS* suele tener asociado un TTL más elevado que los registros de tipo *A* o *CNAME* por ejemplo.

Se indican en 8.1 una serie de valores de TTL razonables en base al tipo de registro, así como a la asiduidad con la que se suele cambiar el mismo [16]

Tipo de registro	TTL conservador	TTL recomendado	TTL agresivo
A / AAAA	1 hour	5 minutes	60 seconds
NS	>2 days	2 days	12 hours
MX	1 day	4 hours	1 hour
TXT (for SPF)	1 day	1 hour	5 minutes

Cuadro 8.1: Valores de TTL aconsejados según el tipo de registro DNS

8.3 Balanceo de carga

En la arquitectura descrita en 2, se indica que se dispondrá de varios servidores esclavo, no sólo uno. Dado que el servidor maestro está oculto, los únicos servidores que ofrecen servicio de consulta , serían los servidores esclavos. El hecho de que exista más de uno se justifica, en primer lugar, al objeto de permitir la alta disponibilidad 8.1, y, en segundo lugar, con la finalidad de ofrecer la posibilidad de balancear la carga entre los diferentes servidores esclavos existentes.

Existen dos metodologías principales a la hora de realizar esta tarea:

- Balanceo por *Round Robin*
- Balanceo inteligente

8.3.1 *Round Robin*

Al establecer los registros NS para el dominio han de definirse tantas entradas como servidores esclavos ofrezcan resolución del mismo. Si se efectúa una consulta a los servidores *DNS* de *Google* sobre los servidores *NS* del dominio *google.com* se obtendrá:

```
# drill google.com -t NS @8.8.8.8
;; ->>HEADER<<- opcode: QUERY, rcode: NOERROR, id: 31961
;; flags: qr rd ra ; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 0
;; QUESTION SECTION:
;; google.com. IN NS

;; ANSWER SECTION:
google.com. 20208 IN NS ns3.google.com.
google.com. 20208 IN NS ns4.google.com.
google.com. 20208 IN NS ns1.google.com.
google.com. 20208 IN NS ns2.google.com.
```

```
;; AUTHORITY SECTION:

;; ADDITIONAL SECTION:

;; Query time: 30 msec
;; SERVER: 8.8.8.8
;; WHEN: Sun Jun 11 13:59:02 2023
;; MSG SIZE  rcvd: 100
```

Como se puede apreciar, se muestran 4 registros de tipo *NS*, que se corresponden con los servidores que responden a las consultas del dominio. Dado que cualquiera de ellos será capaz de responder las consultas, el orden se establece en cada consulta de forma aleatoria, provocando que todos los servidores *NS* reciban el mismo número de consultas.

Este método de balanceo es aleatorio, y, ante los mismos condicionantes, todos los servidores esclavos responderán el mismo número de consultas.

8.3.2 Inteligente

Ante una situación como la del punto anterior, puede suceder que, aunque todos los servidores esclavos reciban el mismo número de peticiones, no todos respondan de igual forma, pues algunos servidores pudieran tener un tiempo de latencia mayor; estos servidores debieran recibir menos peticiones. En este caso, se pueden establecer valores ponderados a cada uno de los servidores, fijando una preferencia mayor en aquellos servidores de menor latencia, y viceversa.

En este tipo de casos, la inteligencia corre a cargo del denominado **balanceador de carga**. Éste consiste en un dispositivo hardware o software que presenta una *VIP (Virtual IP)*, una IP única asociada al dispositivo. Todas las peticiones dirigidas a esa *VIP* serán balanceadas entre todos los nodos esclavos disponibles.

La lógica o *inteligencia* del balanceo se implementa en el propio balanceador, el cual puede disponer de las siguientes funcionalidades:

- Monitorización del servicio DNS en cada uno de los nodos. En caso de que uno deje de responder, dejará de enviarle peticiones.
- Establecimiento de pesos ponderados para cada uno de los nodos de forma manual
- Capacidad de implementación de código que permita redirigir el tráfico en base a la latencia de la respuesta
- Sesiones persistentes. Esta capacidad no es útil para consultas simples, pero si por ejemplo para transferencias de zona.
- Conservación de la IP original. El servidor DNS verá la IP original de la consulta, en lugar de la del balanceador.

Este tipo de balanceadores son muy útiles a la hora de distribuir el tráfico de forma óptima, pero presentan un punto de fallo en sí mismos: si se dispone de un único balanceador y falla, el servicio dejará de responder. Lo ideal, en este caso, es distribuir tanto los nodos esclavos como dos o más balanceadores en diferentes zonas o regiones de disponbilidad.

9 Pruebas de rendimiento

Con la finalidad de comprobar el buen funcionamiento así como la capacidad del servicio DNS implementado se han de realizar una pruebas de rendimiento que permitan validar la solución. Estas pruebas se realizan desde un equipo *local* a través de la herramienta *dnsperf*, ver [14].

9.1 Preparación de la prueba

El aplicativo *dnsperf* necesita un fichero de registros DNS que utilizará a modo de base para sus consultas. Se muestra a continuación un formato válido:

```
www.mydomain.com A
cgi.mydomain.com A
control.mydomain.com A
list.mydomain.com A
mail.mydomain.com A
smtp.mydomain.com A
stats.mydomain.com A
ftp.mydomain.com CNAME
webmail.mydomain.com A
```

9.2 Ejecución de las pruebas

Comando ejecutado para la realización de las pruebas

```
dnsperf -s <slaveIP> -d domainTest.txt -f inet
```

El resultado con la configuración completa es el siguiente:

```
Statistics:

Queries sent:         100
Queries completed:    25 (25.00%)
Queries lost:         75 (75.00%)

Response codes:       NOERROR 25 (100.00%)
Average packet size:  request 51, response 63
Run time (s):         0.008107
Queries per second:   3083.754780

Average Latency (s):  0.000390 (min 0.000286, max 0.000695)
Latency StdDev (s):   0.000105
```

Vemos que hay muchas consultas que se pierden (el 75 %). Comprobamos que la configuración de *rate-limit* funciona correctamente. Repetimos la prueba esta vez sin rate-limit.

```
Statistics:

Queries sent:         100
Queries completed:    100 (100.00%)
Queries lost:         0 (0.00%)

Response codes:       NOERROR 100 (100.00%)
Average packet size:  request 51, response 67
Run time (s):         0.009197
Queries per second:   10873.110797

Average Latency (s):  0.000893 (min 0.000481, max 0.001383)
Latency StdDev (s):   0.000217
```

La salida muestra una tasa de casi 11000 peticiones por segundo sin pérdida de respuesta, lo que es un resultado excepcional para el hardware existente con requisitos mínimos y contra un sólo nodo esclavo.

En la figura 9.1 se muestra el rendimiento alcanzado con una tasa máxima de casi 32 mil peticiones por segundo, en la que sólo un 8,19 % ha quedado sin responder. En las gráficas se visualiza una latencia mínima en la respuesta hasta cerca del final que aumenta drásticamente. Esto podría obedecer a la herramienta empleada, que está en fase beta. El comando utilizado es el siguiente:

```
resperf-report -s <ipEsclavo> -d domainTest.txt -f inet -R -F 1000
```

En conclusión el rendimiento del servicio DNS implementado obedece a la calidad y rendimiento deseados, y el *rate-limit* funciona de manera adecuada.

Resperf output

```
DNS Resolution Performance Testing Tool
Version 2.12.0

[Status] Command line: resperf -P 20230618-1937.gnuplot -s 172.31.38.140 -d domainTestv2.txt -f inet -R -F 1000
[Status] Sending
[Status] Fell behind by 1000 queries, ending test at 36597 qps
[Status] Waiting for more responses
[Status] Testing complete

Statistics:

  Queries sent:         400792
  Queries completed:    388586
  Queries lost:         12206
  Response codes:       NOERROR 388586 (100.00%)
  Reconnection(s):      0
  Run time (s):         66.957499
  Maximum throughput:   32080.000000 qps
  Lost at that point:   8.19%
```

Plots

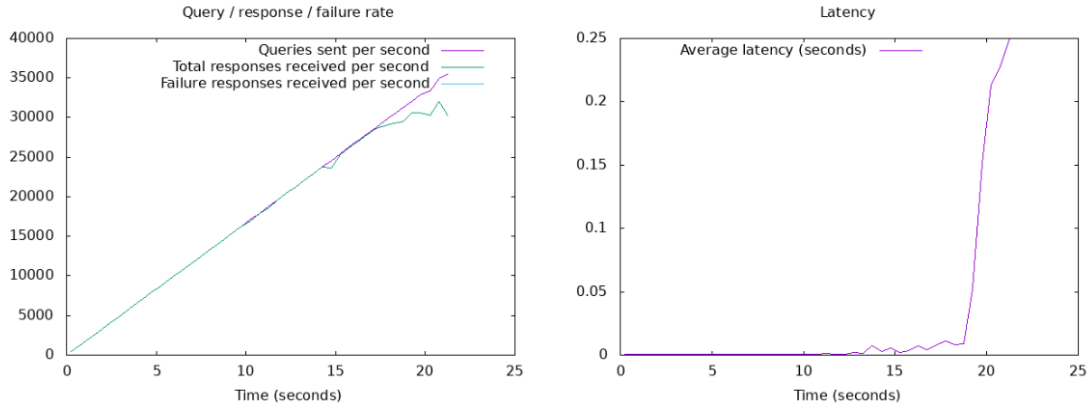

Figura 9.1: Resultado test de rendimiento DNS

10 Mejoras

A pesar de haber indicado los pasos para configurar un DNS de alto rendimiento válido como servidor raíz, existen ciertas mejoras que se podrían implementar en futuras revisiones, a medida que las versiones estables del software *BIND* dispongan de éstas.

Se abordan, a continuación, las de mayor relevancia para el caso que nos ocupa.

10.1 *DNSSEC*

El protocolo DNS es relativamente antiguo (1987), y, aunque ha ido cambiando a lo largo de los años, su funcionamiento básico sigue siendo en esencia el mismo. Originalmente no disponía de medidas de seguridad en lo que a la autenticidad de los datos se refiere, pero, con versiones posteriores, se han ido añadiendo modificaciones en este sentido. La necesidad de evitar la falsificación de registros DNS a través de diversas tácticas (*DNS Cache poisoning*, por ejemplo) ha generado la necesidad de autenticar las respuestas de las peticiones DNS, esto es, de que las respuestas puedan verificarse al venir acomapañadas de una firma digital que ha sido validada en cadena: el servidor raíz, la extensión de dominio (*.edu*), el dominio principal *uoc.edu* hasta llegar a los subdominios y registros.

A pesar de que *DNSSEC* presenta una clara ventaja, al aportar una mayor seguridad en el uso del protocolo DNS, su implantación ha sido, hasta el día de hoy, relativamente escasa. En la figura 10.1 ([19]) podemos ver la tasa de validaciones realizadas por los usuarios entre los años 2013 a 2023. Como puede observarse, su uso, a día de hoy, no ha alcanzado ni al 50 % de los usuarios. Existen varios motivos que explican este comportamiento [19]:

- La firma de las zonas con muchos registros y con una elevada tasa de modificaciones es una tarea compleja, al necesitar firmar el archivo de zona constantemente. Esto se suele solucionar con un frontal que firme las peticiones en lugar de hacerlo el propio DNS autorizado.

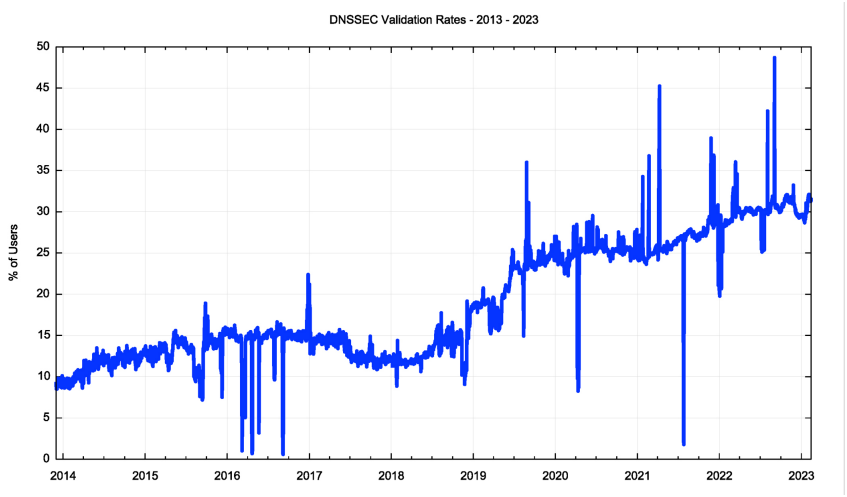

Figura 10.1: Tasa de validaciones de *DNSSEC* entre 2013 y 2023

- En el caso de utilizar un frontal para la firma de peticiones y tener delegados uno o varios servidores esclavos en varias organizaciones, cada organización deberá poseer su par de claves para firmar la zona. Esto implica que debe haber varias llaves de encriptación por cada zona, incrementando el tamaño de respuesta y reduciendo el rendimiento del DNS, llegando, incluso, a fragmentarse los paquets debido al tamaño de *MTU (Maximum Transmission Unit)*.
- Las validación de las peticiones conlleva mayor tiempo en la respuesta a una petición, pues es necesario comprobar la validez de esta respuesta como se muestra en 10.2, duplicando las consultas realizadas sin emplear *DNSSEC*.
- Normalmente, las comprobaciones relativas *DNSSEC* en cada petición DNS no las realizaría el propio equipo cliente, sino el servidor DNS recursivo que tendrían configurado (el del propio proveedor o uno de los servidores DNS públicos existentes). En caso de fallo en la comprobación, no existe un mensaje específico para indicarle al cliente que falló la verificación, enviándole como alternativa un mensaje de tipo *SERVFAIL*. Esto podría provocar que el cliente relanzara la petición ante un posible fallo del servidor, lo cual no tendría sentido.

Todos estos puntos recogidos en [19] muestran la dificultad a la hora de implantar el mecanismo de verificación de *DNSSEC*, dando lugar a que su adopción no sea masiva, principalmente, porque, a priori, no parece adecuarse bien al funcionamiento de un servicio DNS.

Realizado este análisis puede concluírse que el servidor a emplear en este proyecto no implementará *DNSSEC*, al tratarse de un servidor DNS de alto rendimiento, y no estar globalmente implantado.

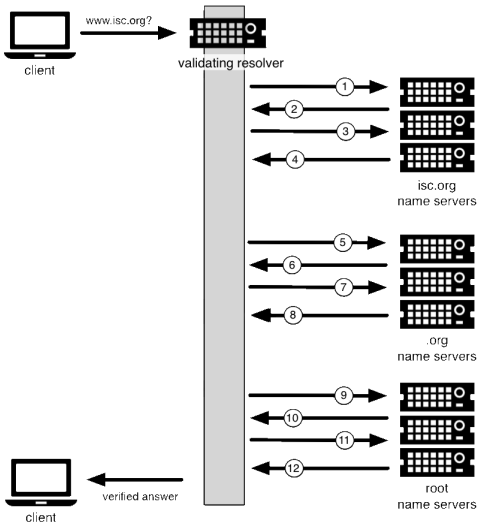

Figura 10.2: Proceso para validar una petición DNS con *DNSSEC*

10.2 *DoH (DNS over HTTPS)* y *DoT (DNS over TLS)*

Con el objeto de anonimizar las consultas realizadas al servidor de nombres, cada vez es más común encriptar la comunicación DNS a través de *HTTPS (Hypertext Transfer Protocol, Secure)* o *TLS (Transport Layer Security)*. Esta práctica no es usual en los servidores autoritativos, sino, más bien, en los servidores recursivos: servidores intermedios que no son autorizados para un determinado dominio sino que se encargan de recibir las consultas, redirigirlas al servidor DNS autorizado, responder al cliente y almacenar la respuesta en la caché.

Esta práctica es común en servidores *abiertos* como *Google DNS, OpenDNS o Quad9* entre otros, que suelen ofrecer este método de consulta como alternativa a los puertos *53/UDP* o *53/TCP* clásicos.

BIND implementó la funcionalidad en 2021 [17] a partir de la versión 9.17.10. Debido a la escasa relevancia en el caso que nos ocupa, no se realizará esta configuración.

10.3 Copia de seguridad y versionado

Se revisan, a continuación, los elementos a conservar de cara a disponer de una copia de seguridad del servicio DNS.

Dada la posibilidad de que se pueden realizar copias de seguridad a varios niveles (ficheros, sistemas de ficheros, máquina virtual o contenedores), se pondrá el foco en aquellos ficheros y carpetas que han de ser conservadas de cada uno de los nodos, de cara a disponer de una copia ante la pérdida de los ficheros de configuración o las zonas DNS.

Se indican a continuación las carpetas y ficheros de los que ha de realizarse copia se seguridad:

- */var/named* - carpeta con los registros de las diferentes zonas de *Bind* (maestras y esclavas)
- */etc/named* - carpeta con los ficheros de configuración de *Bind*
- */etc/named.conf* - fichero principal de configuración de *Bind*
- */etc/rndc.conf* - fichero de configuración de la herramienta de control de *Bind*

No sólo sería interesante mantener una copia de los ficheros en texto plano indicados, sino también sería interesante realizar un seguimiento de los cambios realizados en la configuración, así como los realizados en las propias zonas. En caso de disponer de un repositorio como Git [18], sería posible automatizar los cambios para que, cada vez que se realice una modificación en los ficheros de interés, se ejecute un *commit* indicando el tipo de cambio realizado. Esto permitirá disponer de una **trazabilidad** de todos los cambios realizados a través de un respositorio de código. Esta disposición, permitirá realizar un seguimiento de los cambios en las zonas DNS, así como en la configuración, permitiendo contrastar fechas especialmente relevantes con cambios realizados en los ficheros.

En el apéndice A.1 se muestra un script realizado en *bash* para realizar la copia de ficheros y carpetas a un directorio dado para, después, de forma manual los comandos necesarios para realizar un *commit* aportando información justificativa del cambio.

Se completa esta parte en el apéndice A.2, en el que se crea una función de copia de seguridad y versionado, que se podrá utilizar en otros scripts de forma transparente. La ejecución de éste y los restantes scripts que utilicen esta función, será realizada desde el equipo de administración.

10.4 Contenedores

En la actualidad es muy frecuente la utilización de los contenedores como alternativa a la instalación y configuración de un Sistema Operativo completo. Los contenedores aportan una gran versatilidad e independencia de la plataforma de despliegue.

Es totalmente plausible el despliegue de la solución aquí mencionada a través de simples contenedores (*Docker*) o alternativamente con una solución orquestada (*kubernetes*), pero ninguna de ellas posee una ventaja clara, pues tanto la rigidez del protocolo *DNS* como la robusted necesarias se obtienen directamente a través del sistema operativo.

11 Conclusiones

El servicio DNS ha formado parte de Internet desde los albores de su nacimiento. Es tal su su importancia y criticidad, que se hace necesario un amplio conocimento del mismo para obtener el mayor beneficio de su uso.

Se han empleado metodologías que ponen el foco en la configuración y su versionado así como en la actualización dinámica de zonas y registros. Este conocimiento pretende abordar casos de uso poco conocidos y servir como guía para encontrar las funcionalidades que mejor se adapten.

El uso adecuado del servicio DNS potencia sin duda la capacidad de gestión y la independencia frente a terceros para obtener valor de los registros que engloban todos los servicios de una organización. Permite, asimismo, integrarlo en los procesos de automatización internos adoptando la filosofía de infraestructura como código (*IaC*), pero sin perder un ápice de la solidez y disponibilidad necesarios.[ooo]

Bibliografía

[1] RootServers.org. (2023). *Root Server Technical Operations Association.* https://aws.amazon.com/es/

[2] Wikipedia. (2023). *Distributed denial-of-service attacks on root nameservers.* https://en.wikipedia.org/wiki/Distributed_denial-of-service_attacks_on_root_nameservers

[3] Aitchison, [Ron]. (2005). *Pro DNS and BIND.* Apress

[4] Huston, [Geoff] (2023, 8 de Febrero). The Root Zone of the DNS Revisited [entrada de blog]. *blabs | Postings from labs.apnic.net* https://labs.apnic.net/index.php/2023/02/08/the-root-of-the-dns-revisited/

[5] Thomas, [Matthew]. (2020). Chromium's impact on root DNS traffic. *APNIC Blog, the Regional Internet Registry administering IP addresses for the Asia Pacific* https://blog.apnic.net/2020/08/21/chromiums-impact-on-root-dns-traffic/

[6] Chandramouli, [Ramaswamy] y Rose, [Scott] (2013) Secure Domain Name System (DNS) Deployment Guide, *NIST Special Publication 800-81-2* https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-81-2.pdf

[7] IETF - Internet Engineering Task Force (2011). RFC 6303:*Locally Served DNS Zones.* IETF https://datatracker.ietf.org/doc/html/rfc6303

[8] Morvan, [Antoine]. (2023). Logical Volume Manager in RockyLinux. *RockyLinux Documentation* https://docs.rockylinux.org/es/books/admin_guide/07-file-systems/#logical-volume-manager-lvm

[9] Amazon. (2023). *Amazon Web Services.* https://aws.amazon.com/es/

[10] Amazon. (2023). *Amazon EC2 Instance Types.* https://aws.amazon.com/ec2/instance-types/

[11] RockyLinux. (2023). *Download RockyLinux.* https://rockylinux.org/download/

[12] ISC - Internet System Consortium. (2021). *Using the 'map' zone file format in BIND*. https://kb.isc.org/docs/aa-01120

[13] RedHat. (2019). *¿Qué es SELinux?*. https://www.redhat.com/es/topics/linux/what-is-selinux

[14] DNS-OARC. (2019). *dnsperf*. https://www.dns-oarc.net/tools/dnsperf

[15] Zytrax. (2022). *Some thoughts on TTL Values*. https://www.zytrax.com/books/dns/info/ttl.html

[16] nslookup.io. (2023). *What is a good TTL for DNS?*. https://www.nslookup.io/learning/what-is-a-good-ttl-for-dns/

[17] Boldariev, [Artem]. (2021). DNS over HTTPS (DoH) in BIND 9. *Internet Systems Consortium* https://www.isc.org/blogs/bind-implements-doh-2021/

[18] Git. (2023). *git –distributed-even-if-your-workflow-isnt*. https://git-scm.com

[19] Huston, [Geoff]. (2023). [Opinion] To DNSSEC or not? *APNIC Blog, the Regional Internet Registry administering IP addresses for the Asia Pacific* https://blog.apnic.net/2023/02/20/opinion-to-dnssec-or-not/

[20] BIND9. (2023). *DNSSEC Guide*. https://bind9.readthedocs.io/en/v9.18.14/dnssec-guide.html

[21] ISC Kb. (2023). *Using Access Control Lists (ACLs) with both addresses and keys*. https://kb.isc.org/docs/aa-00723

[22] Readthedocs. (2023). *BIND 9 Administrator Reference Manual*. https://bind9.readthedocs.io/en/v9.18.14/reference.html#namedconf-statement-category

[23] The Measurement Factory. (2020). *DNSTOP: STAY ON TOP OF YOUR DNS TRAFFIC*. http://dns.measurement-factory.com/tools/dnstop/index.html

[24] ISC Kb. (2022). *Monitoring Recommendations for BIND 9*. https://kb.isc.org/docs/monitoring-recommendations-for-bind-9

[25] IETF - Internet Engineering Task Force (1997). RFC 2136:*Dynamic Updates in the Domain Name System (DNS UPDATE)*. IETF https://datatracker.ietf.org/doc/html/rfc2136.html

[26] IETF - Internet Engineering Task Force (1996). RFC 1995:*Incremental Zone Transfer in DNS*. IETF https://datatracker.ietf.org/doc/html/rfc1995.html

[27] WIKIPEDIA. (2023). *List of DNS record types*. https://en.wikipedia.org/wiki/List_of_DNS_record_types

[28] DIE.NET. (-). *logrotate(8) - Linux man page*. https://linux.die.net/man/8/logrotate

[29] ANSIBLE DOCUMENTATION. (2023). *community.general.nsupdate module – Manage DNS records*. https://docs.ansible.com/ansible/latest/collections/community/general/nsupdate_module.html

[30] TERRAFORM DOCUMENTATION. (2023). *DNS Provider*. https://registry.terraform.io/providers/hashicorp/dns/latest/docs

A *Scripts* de backup - versionado

A.1 Script versionado manual

```
        #!/bin/bash
        # Nombre script: backupVersionadoManual.sh
        # Francisco Javier Alvarez Calvo
        # Copia los ficheros modificados al repositorio ~/bind
        # Se ejecuta de forma manual
        echo ""
        rsync -avuqX --delete --exclude '*.jnl' --exclude 'data' /var/named/ ./bind/var/named/
&& rsync -avuqX --delete --exclude keys.d /etc/named/ ./bind/etc/named/ && rsync -avuqX
--delete /etc/named.conf ./bind/etc/named.conf && rsync -avuqX /etc/rndc.conf ./bind/etc/rndc.conf
        cd ./bind ; git status
        echo ""
        echo "Ejecutar los siguientes comandos para completar el commit: cd bind; git
commit -m 'Motivo del cambio realizado' -a"
        echo ""
```

A.2 Función versionado automatizada

Creamos una función que será llamada desde otro script.

```
        #!/bin/bash
        # Nombre script: libGIT.sh
        # Francisco Javier Alvarez Calvo
        # Este script se encarga de actualizar el repositorio GIT del HIDDEN MASTER
DNS. Se llama desde otros scripts
        # Importante indicar en la variable $SERVER la IP del servidor hidden master.
        function suberepo() {
            # Variables
            FILE=$1
            FILE2=$(echo $FILE | cut -d/ -f2)
            REMOTEPATH=/tmp/$FILE2
            USUARIO="admin"
            REPOPATH="/home/$USUARIO/bind"
            SERVER="<IPHiddenMaster>" <-- Deberemos fijar la IP del servidor hidden
master!!
```

```
            RNDCCONF=".rndc.conf"
            #Principal
            rndc -c $RNDCCONF sync
            ssh -X -o LogLevel=error $USUARIO@$SERVER "rsync -avuqX --delete --exclude
'*.jnl' --exclude 'data' /var/named/ $REPOPATH/var/named/ && rsync -avuqX --delete
--exclude keys.d /etc/named/ $REPOPATH/etc/named/ && rsync -avuqX --delete /etc/named.conf
$REPOPATH/etc/named.conf && rsync -avuqX /etc/rndc.conf $REPOPATH/etc/rndc.conf"
            scp -q -o LogLevel=error $FILE $USUARIO@$SERVER:$REMOTEPATH
            ssh -X -o LogLevel=error $USUARIO@$SERVER "echo \"\" >> $REMOTEPATH ; journalctl
--no-pager -u named --since \"5 seconds ago\" >> $REMOTEPATH ; cd $REPOPATH ; git add
-A * ; git commit -q -a -F $REMOTEPATH ; rm -f $REMOTEPATH ; git push -q -u origin
master"
            return 0
        }
```

Este script estaría situado en el equipo de administración, y se conectaría a un repositorio situado en el servidor DNS *Hidden Master*. El comando *rndc ... sync* se utiliza para consolidar la información de las zonas dinámicas DNS sobre su fichero correspondiente, evitando así copiar los ficheros temporales *.jnl*. Ver 4.2.2

En cada subida al repositorio se incorpora el log generado al realizar el cambio en el servidor DNS: *journalctl ... -u named ...*, se realiza un *commit*, y se completa un *push* sobre el servidor remoto de Git. De esta forma cada vez que se ejecute un script provocando un cambio de configuración o de zona en el servidor DNS, será automáticamente plasmado en un repositorio, con los logs de la ejecución de ese cambio.

Si queremos emplear esta función desde cualquier otro script se procederá de la siguiente manera:

```
        #!/bin/bash
        ...
        source libGIT.sh #Cargamos el script con la función
        ...
        suberepo $FICHERO #Subimos al repositorio el nombre del fichero en la variable
$FICHERO
        ...
```

B Scripts *DDNS*

B.1 Script para actualizar número serie de registro *SOA*

```
        #!/bin/bash
        # No usar directamente!
        # title           : libSOA.sh
        # Francisco Javier Álvarez Calvo
        # Este script se encarga de actualizar el Serial del DNS según el formato:
YYYYMMDDNN. Se llama desde otros scripts
        # Modificaciones en la seccion #Variables
        function goodSOAserial() {
                zone=$1
                serial=$2
                declare -i serialdate todaydate
                serialdate=$(echo $serial|cut -c1-8)
                todaydate=$(date +%Y%m%d)
                if [ "$serialdate" -lt "$todaydate" ]; then
                        nSOAserial=$(($todaydate*100+1))
                        cSOAserial="yes"
                        return 0
                fi
                nSOAserial=$(($serial+1))
                return 0
        }
        function updateSOAserial() {
                zone=$1
                oldSerial=$2
                SOATTL=$(dig -p $PORT -t SOA $zone @$MASTERDNS|grep -A1 ";; ANSWER
SECTION"|grep -v \; | awk '{print $2}')
                SOARR=$(dig +short -p $PORT -t SOA $zone @$MASTERDNS)
                SOARR=${SOARR/$oldSerial/$nSOAserial}
                echo "update add $zone $SOATTL SOA $SOARR" >> $FICHERO
```

B.2 Creación/Modificación de registros A/CNAME

```
        #!/bin/bash
```

```
        #title           : dns-actualizar.sh
        #description     : Este script actualiza las entradas DNS de tipo A, CNAME...
(pero no MX, TXT o SOA, para existen scripts especificos)
        #author          : Francisco Javier Alvarez Calvo
        #version         : 1.4
        #notes           : Modificaciones en la seccion #Variables
        #notes2          : Este script actualiza las entradas DNS del servidor (HIDDEN)
MASTER a través de nsupdate, y guarda logs del cambio
        #                  en un repo GIT dentro del mismo asi como en un fichero local
        # Ayuda
        DESCRIPTION="Este script actualiza las entradas DNS de tipo A, CNAME... (pero
no MX, TXT o SOA, para existen scripts especificos)"
        USAGE="Uso: $0 <zonaDNS> <registro-sinZona> <tipoRegistro> <IP> [\"Comentario
Opcional\"]"
        EXAMPLE="Ejemplo registro A: $0 mapu.com.es test A 127.0.0.2 \"Ticket #23456
- Alta entrada DNS nueva Web\""
        EXAMPLE2="Ejemplo registro CNAME: $0 mapu.com.es test66 CNAME test.mapu.com.es."
        EXAMPLE3="Ejemplo registro PTR: $0 67.91.85.in-addr.arpa 254 PTR test.mapu.com.es."
        EXAMPLE4="Ejemplo registro @: $0 mapu.com.es @ A 127.0.0.2"
        if [ $# -lt 4 ] ; then
            echo ""
            echo $DESCRIPTION
            echo ""
            echo $USAGE
            echo $EXAMPLE
            echo $EXAMPLE2
            echo $EXAMPLE3
            echo $EXAMPLE4
            echo ""
            exit 1;
        fi
        #Variables
        FICHERO=".history/actualizacion-$(whoami)-$(date +%F-%H-%M).tmp"
        KEYPATH=".Kview-external-00-nsupdate.+163+62985.key"
        MASTERDNS="172.31.100.177"
        PORT="7766"
        DEFTTL="3600"
        #Functions
        source ./.libSOA.sh
        source ./.libGIT.sh
        # Principal
        # Comprobamos que el dominio existe
        if [ ! "$(dig -p $PORT -t ANY $1 @$MASTERDNS|grep -A1 ";; ANSWER")" ]; then
                echo ""
                echo "El dominio $1 no está dado de alta en estos servidores DNS"
                echo "Pruebe a darlo de alta con el comando dns-zona-crear.sh"
                echo ""
                exit 1
        fi
        oSOAserial=$(dig +short -p $PORT -t SOA $1 @$MASTERDNS|cut -d " " -f 3)
        nSOAserial=""
        cSOAserial="no"
        goodSOAserial $1 $oSOAserial
        echo ""
        echo "El registro a modificar: $2.$1 resuelve en la actualidad de la siguiente
manera:"
        echo "----------------------------------------"
        echo "SERIAL:$oSOAserial"
```

```
        if [ "$2" == "@" ]; then
            oldRR="@  IN  $3  $(dig +short -p $PORT -t $3 $1 @$MASTERDNS)"
        elif [ "$(dig -p $PORT -t ANY $2.$1 @$MASTERDNS|grep -A1 ";; ANSWER"|grep -v
"ANSWER"|grep -v MX)" ]; then
                oldRR="$(dig -p $PORT -t ANY $2.$1 @$MASTERDNS|grep -A1 ";; ANSWER"|grep
-v "ANSWER"|grep -v MX)"
        else
                oldRR="$2.$1  IN  $3  $(dig +short -p $PORT -t $3 $2.$1 @$MASTERDNS)"
        fi
        echo "$oldRR"
        echo "----------------------------------------"
        echo ""
        echo "¿Desea modificar el registro de la siguiente manera? (y para confirmar)"
        echo "----------------------------------------"
        echo "SERIAL:$nSOAserial"
        if [ "$2" == "@" ]; then
            newRR="@  IN  $3  $4"
        else
            newRR="$2.$1  IN  $3  $4"
        fi
        echo "$newRR"
        echo "----------------------------------------"
        echo ""
        read resposta
        if [ "$resposta" == "y" ]; then
                echo ""
                ##
                # Info de interes
                ##
                echo ";$(date +%F-%H-%M)" > $FICHERO
                # Comentario Usuario sobre modificacion: ticket, motivo, etc
                if [ $# == 5 ] ; then
                        echo ";Comentario $(whoami):" >> $FICHERO
                        echo ";$5" >> $FICHERO
                        echo ";" >> $FICHERO
                fi
                echo ";Modificada entrada por el usuario $(whoami) desde el equipo
$(hostname) al que se ha conectado con la IP $(echo $SSH_CLIENT | cut -f1 -d' ')" >>
$FICHERO
                echo ";" >> $FICHERO
                echo ";SOA SERIAL: $nSOAserial" >> $FICHERO
                echo ";$oldRR ----> $newRR" >> $FICHERO
                echo ";" >> $FICHERO
                echo ";Comando enviado:" >> $FICHERO
                echo ";" >> $FICHERO
                #
                echo "server $MASTERDNS $PORT" >> $FICHERO
                echo "zone $1" >> $FICHERO
            if [ "$2" == "@" ]; then
                echo "update delete $1 $DEFTTL $3" >> $FICHERO
                echo "update add $1 $DEFTTL $3 $4" >> $FICHERO
            else
                    echo "update delete $2.$1 $DEFTTL $3" >> $FICHERO
                    echo "update add $2.$1 $DEFTTL $3 $4" >> $FICHERO
            fi
                if [ $cSOAserial == "yes" ]; then
                        updateSOAserial $1 $oSOAserial
                fi
```

```
                echo "send" >> $FICHERO
                echo "answer" >> $FICHERO
                echo ";" >> $FICHERO
                nsupdate -v -k $KEYPATH $FICHERO | sed -e '/status/!d' -e 's/\(.*\)status:
\(.*\), \(.*\)/Resultado: \2 /' -e 's/ \(.*\)/\x1b[7m&\x1b[0m/' | tee -a $FICHERO
                suberepo $FICHERO
                echo ""
                exit 0
        fi
        echo "No se realiza ningun cambio"|sed -e 's/.*/\x1b[7m&\x1b[0m/'
        echo ""
        exit 1
```

B.3 Eliminación de registros A/CNAME

```
        #!/bin/bash
        #title           : dns-borrar.sh
        #description     : Este script borra entradas DNS de tipo A, PTR, TXT... (no
borra la de tipo MX, para ello dns-borrar-MX.sh)
        #author          : Francisco Javier Alvarez Calvo
        #version         : 1.3
        #notes           : Modificaciones en la seccion #Variables
        #notes2          : Este script actualiza las entradas DNS del servidor (HIDDEN)
MASTER a través de nsupdate, y guarda logs del cambio
        #                  en un repo GIT dentro del mismo asi como en un fichero local
        # Ayuda
        DESCRIPTION="Este script borra entradas DNS de tipo A, PTR, TXT... (no borra
la de tipo MX, para ello dns-borrar-MX.sh)"
        USAGE="Uso: $0 <zonaDNS> <registro-sinZona> <tipoRegistro> [\"Comentario Opcional\"]"
        EXAMPLE="Ejemplo A: $0 mydomain.com test A \"Ticket #23456 - Borrado entrada
DNS\""
        EXAMPLE2="Ejemplo PTR: $0 95.91.85.in-addr.arpa 255 PTR \"Ticket #23456 - Borrado
entrada DNS\""
        if [ $# -lt 3 ] ; then
            echo ""
            echo $DESCRIPTION
            echo ""
            echo $USAGE
            echo $EXAMPLE
            echo $EXAMPLE2
            echo ""
            exit 1;
        fi
        #Variables
        FICHERO=".history/borrado-$(whoami)-$(date +%F-%H-%M).tmp"
        KEYPATH=".Kview-external-00-nsupdate.+163+62985.key"
        MASTERDNS="172.31.100.177"
        PORT="7766"
        DEFTTL="3600"
        #Functions
        source ./.libSOA.sh
        source ./.libGIT.sh
        # Principal
```

```
        if [ $2 == "@" ] ; then
        registro=$1
        else
        registro=$2.$1
        fi
        oSOAserial=$(dig +short -p $PORT -t SOA $1 @$MASTERDNS|cut -d " " -f 3)
        nSOAserial=""
        cSOAserial="no"
        goodSOAserial $1 $oSOAserial

        echo ""
        echo "El registro a modificar: $registro resuelve en la actualidad de la siguiente
manera:"
        echo "----------------------------------------"
        echo "SERIAL:$oSOAserial"
        oldRR="$registro  IN  $3  $(dig +short -p $PORT -t $3 $registro @$MASTERDNS)"
        echo "$oldRR"
        echo "----------------------------------------"
        echo ""
        echo "¿Desea borrar el registro? (y para confirmar)"
        echo "----------------------------------------"
        echo "SERIAL:$nSOAserial"
        newRR="$registro  IN  $3"
        echo "$newRR"
        echo "----------------------------------------"
        echo ""
        read resposta
        if [ "$resposta" == "y" ]; then
                echo ""
                ##
                # Info de interes
                ##
                echo ";$(date +%F-%H-%M)" > $FICHERO
                # Comentario Usuario sobre modifacion: ticket, motivo, etc
                if [ $# == 4 ] ; then
                        echo ";Comentario $(whoami):" >> $FICHERO
                        echo ";$4" >> $FICHERO
                        echo ";" >> $FICHERO
                fi
                echo ";Borrada entrada por el usuario $(whoami) desde el equipo $(hostname)
al que se ha conectado con la IP $(echo $SSH_CLIENT | cut -f1 -d' ')" >> $FICHERO
                echo ";" >> $FICHERO
                echo ";SOA SERIAL: $nSOAserial" >> $FICHERO
                echo ";$oldRR ----> $newRR" >> $FICHERO
                echo ";" >> $FICHERO
                echo ";Comando enviado:" >> $FICHERO
                echo ";" >> $FICHERO
                #
                echo "server $MASTERDNS $PORT" >> $FICHERO
                echo "zone $1" >> $FICHERO
                echo "update delete $registro $DEFTTL $3" >> $FICHERO
                if [ $cSOAserial == "yes" ]; then
                        updateSOAserial $1 $oSOAserial
                fi
                echo "send" >> $FICHERO
                echo "answer" >> $FICHERO
                echo ";" >> $FICHERO
                nsupdate -v -k $KEYPATH $FICHERO | sed -e '/status/!d' -e 's/\(.*\)status:
```

```
\(.*\), \(.*\)/Resultado: \2 /' -e 's/ \(.*\)/\x1b[7m&\x1b[0m/' | tee -a $FICHERO
                suberepo $FICHERO
                echo ""
                exit 0
        fi
        echo "No se realiza ningun cambio"|sed -e 's/.*/\x1b[7m&\x1b[0m/'
        echo ""
        exit 1
```

B.4 Creación de nueva zona

```
        #!/bin/bash
        #title           : dns-zona-crear.sh
        #description     : Este script permite anhadir una nueva zona para un dominio
dado
        #author          : Francisco Javier Alvarez Calvo
        #date            : 20170420
        #notes           : Modificaciones en la seccion #Variables
        #notes2          : Este script zonas DNS del servidor (HIDDEN) MASTER y sus
esclavos
        # Ayuda
        DESCRIPTION="Este script permite anadir una nueva zona para un dominio dado.
El fichero de zona es opcional"
        USAGE="Uso: $0 <zona(dominio)> <IPprincipal> [<ficheroZona(opcional)>]"
        #EXAMPLE="Ejemplo: $0 mydomain.com  127.0.0.2 \"Redmine #23456 - Creación nova
zona DNS\""
        EXAMPLE1="Ejemplo: $0 mydomain.com  127.0.0.2"
        EXAMPLE2="Ejemplo: $0 mydomain.com  127.0.0.2 mydomain.com.dns"
        if [ $# -lt 2 ] ; then
            echo ""
            echo $DESCRIPTION
            echo ""
            echo $USAGE
            echo $EXAMPLE1
            echo $EXAMPLE2
            echo ""
            exit 1
        fi
        #Variables
        ZONA=$1
        ZONAIP=$2
        #COMENTARIO=$3
        VISTA="view-ext-00"
        PROFILE=".xunta-view-0.profile"
        FICHERO=".history/zonaNueva-$(whoami)-$(date +%F-%H-%M).tmp"
        LOGGING=".history/zonaNueva-$(whoami)-$(date +%F-%H-%M).log"
        RNDCPATH="/usr/sbin/rndc"
        RNDCCONFIG=".rndc.conf"
        #KEYPATH="Kview-0-nsupdate-key.+163+56653.key"
        MASTERDNS="172.31.100.177"
        PORT="7766"
        SLAVE1DNS="172.31.100.28"
        SLAVE2DNS="172.31.100.157"
```

```
        FICHEROCUSTOM="NO"
        #DEFTTL="3600"
        #Functions
        #source ./libSOA.sh
        source .libGIT.sh
        source .libMkzone.sh
        source .libMaster.sh
        # Principal
        if [ $(dig +short -p $PORT $ZONA @$MASTERDNS) ] || [ $(dig +short -p $PORT
$ZONA @$SLAVE1DNS) ] || [ $(dig +short -p $PORT $ZONA @$SLAVE2DNS) ]; then
            echo ""
            echo "Esta zona ya existe en los servidores DNS"
            echo "No se realiza ningun cambio"|sed -e 's/.*/\x1b[7m&\x1b[0m/'
            echo ""
            exit 1
        fi
        if [ $# -eq 3 ] ; then
            FICHERO=$3
            FICHEROCUSTOM="YES"
            grep -i $ZONA $FICHERO
            if [ $? -ne 0 ]; then
                echo ""
                echo "La zona especificada no se corresponde con la del fichero"
                echo ""
                exit 1
            fi
        else
            mkzone $ZONA $ZONAIP $PROFILE > $FICHERO
        fi
        echo ""
        echo "Se va a añadir al DNS la siguiente zona para el dominio $ZONA"
        echo "----------------------------------------"
        cat $FICHERO
        echo "----------------------------------------"
        echo ""
        echo "¿Desea testear la configuración? (y para confirmar)"
        echo ""
        read resposta
        if [ "$resposta" == "y" ]; then
            zoneCheck $FICHERO $ZONA
            if [ $? != 0 ]; then
                echo ""
                echo "La zona no es correcta"
                echo ""
                exit 1
            fi
            echo "¿Desea publicar la zona en la plataforma DNS?"
            read resposta2
            if [ "$resposta2" == "y" ]; then
                zoneAdd $FICHERO $ZONA $VISTA $MASTERDNS $SLAVE1DNS $SLAVE2DNS
                $RNDCPATH -s $MASTERDNS -c $RNDCCONFIG reconfig
                $RNDCPATH -s $SLAVE1DNS -c $RNDCCONFIG reconfig
                $RNDCPATH -s $SLAVE2DNS -c $RNDCCONFIG reconfig
                echo "; -- NUEVA ZONA $ZONA CREADA -- " > $LOGGING
                echo ";Creacion de zona por el usuario $(whoami) desde el equipo $(hostname)
al que se ha conectado con la IP $(echo $SSH_CLIENT | cut -f1 -d' ')" >> $LOGGING
                echo ";" >> $LOGGING
                echo "Se han aplicado los cambios con exito"|sed -e 's/.*/\x1b[7m&\x1b[0m/'
```

```
| tee -a $LOGGING
                echo ";" >> $LOGGING
                cat $FICHERO >> $LOGGING
                if [ "$FICHEROCUSTOM" == "NO" ]; then
                    rm $FICHERO
                fi
                suberepo $LOGGING
                echo ""
                exit 0
            fi
            zoneDel $FICHERO $ZONA $VISTA $MASTERDNS $SLAVE1DNS $SLAVE2DNS
            exit 1
        fi

        echo "No se realiza ningun cambio"|sed -e 's/.*/\x1b[7m&\x1b[0m/'
        echo ""
        exit 1
```

La dependencia *.libMaster.sh* y *.libMkzone.sh* son demasiado grandes en tamaño para el proyecto. Se deja alojado en el servidor de pruebas. Ver B.5

B.5 Entorno de pruebas

Se ha creado un entorno de pruebas en *AWS* donde se han configurado los siguientes nodos:

- Un servidor DNS maestro
- Dos servidores DNS esclavos, cada uno con su IP pública
- Un equipo de gestión al que nos podremos conectar

Se incluye en el entregable la clave privada de acceso al equipo de gestión a través del protocolo *SSH*. El comando completo para acceder será:

```
ssh -i clavePrivadaEquipoGestion.pem rocky@54.195.225.40
# O si lo preferimos a través de su registro DNS
ssh -i clavePrivadaEquipoGestion.pem rocky@gestiondns.javieralvarezcalvo.com
```

El usuario por defecto es *rocky*, y con el comando *sudo -s* cambia al usuario root.

En el fichero */etc/hosts* se han identificado alias para las IP's privadas de gestión y servicio de los diferentes nodos

```
172.31.100.177 master00-gestion master00 master
```

```
172.31.100.28 slave01-gestion slave01 ns1
172.31.100.157 slave02-gestion slave02 ns2
172.31.41.164 master00-service
172.31.38.140 slave01-service
172.31.46.142 slave02-service
```

Se ha adquirido el dominio *javieralvarezcalvo.com* y registrado las entradas:

- *ns1.javieralvarezcalvo.com*
- *ns2.javieralvarezcalvo.com*

Se han cambiado los servidores de nombres para los dos siguientes dominios, que podrán ser utilizados en este laboratorio.

- *javieralvarezcalvo.com*
- *mapu.com.es*

Estos dominios ya están configurados y funcionales en el entorno de pruebas y pueden consultarse sus registros desde cualquier equipo con salida a Internet:

```
$ dig -t NS javieralvarezcalvo.com
$ dig -t NS mapu.com.es
```

Se dejan configurados en la *home* del usuario *rocky* los comandos necesarios para acceder a los diferentes equipos, así como los diferentes scripts.