1 Servidor DNS bind

O servidor DNS máis característico en todas as versións de Linux/Unix é *BIND*[1]. A libraría de resolución incluida na distribución de *bind* provee unha API estandar para traducir entre nomes e enderezos IP, e está enlazada con todas as aplicacións que requiren o servizo de resolución de nomes.

BIND version 9 é a derradeira reescritura da gran maioría de características de 'bind. *As máis importantes características son: Seguridade DNS (DNSSEC, TSIG), IPv6, melloras do protocolo DNS (IXFR, DDNS, DNS Notify, EDNS0), Vistas, Soporte multiprocesadores, e unha arquitectura portable.*

1.1 Sumario

- 1 Instalación en Debian/Ubuntu
- 2 Configuración básica
  - 2.1 Servidor DNS Caché de reenvío
  - 2.2 Zonas mestras
    - 2.2.1 Zona de resolución directa ou de reenvío
    - 2.2.2 Zona de resolución inversa
  - 2.3 Rexistros de recursos
  - 2.4 Zonas secundarias
    - 2.4.1 Notificacións aos servidores subordinados
  - 2.5 Subdominios e delegación de zonas
    - 2.5.1 Crear un subdominio virtual
    - 2.5.2 Delegación de subdominios
  - 2.6 Logging
- 3 Configuración avanzada
  - 3.1 acl
  - 3.2 Chaves TSIG
  - 3.3 Transferencias de zonas seguras empregando chaves TSIG
  - 3.4 Control remoto
    - 3.4.1 Configuración do equipo remoto
    - 3.4.2 A utilidade rndc
  - 3.5 DNSSEC
    - 3.5.1 Configuración de Bind para soportar DNSSEC
    - 3.5.2 Comprobación de que DNSSEC está activado
    - 3.5.3 Creación das chaves de asinado
    - 3.5.4 Asinando o ficheiro de zona
    - 3.5.5 Informando da KSK no rexistrador
    - 3.5.6 Validar a implementación de DNSSEC
- 4 Actualizacións dinámicas
  - 4.1 Configuración do servidor para poder recibir actualizacións dinámicas
- 5 Referencias externas

1.2 Instalación en Debian/Ubuntu

O único necesario é escribir nunha liña de comandos:

```
sudo apt-get install bind9
```

Un paquete moi útil para probar e solucinar problemas relacionados co servidor DNS é o paquete *dnsutils*. Con frecuencia estas ferramentas ou veñen instaladas de serie, ou instálanse canda o servidor. Para asegurarse escribimos:

```
sudo apt-get install dnsutils
```

Sempre que fagamos cambios, hai que lembrarse de reiniciar o servidor DNS.

```
sudo service bind9 restart
```

Tamén podemos botar unha ollada ao log do sistema, por se algo vai mal. É útil ter aberto noutro terminal o ficheiro de log do sistema:

```
tail -f /var/log/syslog
```

1.3 Configuración básica

Hai varias maneiras de configurar BIND9. As máis comúns son como servidor caché de reenvío, primario o secundario.

- Cando se configura como servidor DNS caché de reenvio, BIND9 buscará respostas a consultas de nomes e recordaraas para cando sexa preguntado de novo por ese mesmo nome.
- Como servidor primario BIND9 le os datos do ficheiro de zona do que é a autoridade competente.
- Como servidor secundario doutro, obten os datos da zona do servidor que é autoritativo da zona en cuestión.

Tanto servidores primarios coma secundarios, poden ser configurados tamén a maiores coma servidores cache de reenvío. E tamén pode ser configurado coma primario para unha zona, e secundario para outra.

Os ficheiros de configuración do servidor DNS están almacenados no directorio */etc/bind*. O ficheiro principal de configuración é `/etc/bind/named.conf`.

Contido do ficheiro */etc/bind/named.conf*:

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

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

As liñas coa clausula `include` especifican o nome dos ficheiros que conteñen as opcións, e cuxo contido é "copiado" no ficheiro *named.conf* como resultado do procesado do ficheiro *named.conf*.

No ficheiro **`/etc/bind/named.conf.options`** especificanse diversas opcións de configuración. Unha delas denominada `directory` especifica onde están gardados os ficheiros de configuración das zonas. Todas as rutas relativas na configuración de BIND serán relativas a ese cartafol. O cartafol por defecto é */var/cache/bind*

Contido do ficheiro */etc/bind/named.conf.options*:

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

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

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

        // forwarders {
        //      0.0.0.0;
        // };

        //========================================================================
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/bind-keys
        //========================================================================
        dnssec-validation auto;

        auth-nxdomain no;    # conform to RFC1035
        listen-on-v6 { any; };
};
```

O ficheiro */etc/bind/db.root* describe os servidores de nomes raíz (suxestións de raíz) do mundo. Estes servidores cambian ao longo do tempo, polo que o ficheiro */etc/bind/db.root* debe ser mantido. Isto faise habitualente cando se actualiza o paquete *bind9*. Nese ficheiro almacenase a zona mestra

raíz(.).

É posible configurar o mesmo servidor para ser servidor caché de reenvío, servidor mestre e servidor secundario. Un servidor pode ser Start of Authority (SOA) para unha zona, mentras funciona coma servidor secundario para outra zona diferente, todo mentres fai de servidor caché DNS para os equipos da LAN.

Sempre que fagamos un cambio na configuración é desexable comprobar se a nova configuración está correcta. O comando `named-checkconf` pódenos axudar a detectar posibles erros:

```
root@ns:~# named-checkconf -z
zone localhost/IN: loaded serial 2
zone 127.in-addr.arpa/IN: loaded serial 1
zone 0.in-addr.arpa/IN: loaded serial 1
zone 255.in-addr.arpa/IN: loaded serial 1
```

1.3.1 Servidor DNS Caché de reenvío

Unha vez instalado o servidor DNS BIND9, xa está listo para funcionar coma servidor caché DNS e resolver calquera nome de Internet. Pódese comprobar con calquera resolvedor como *dig*, *nslookup*, ou *host*.

É probable que nos aparezan certos no ficheiro de log `/var/log/syslog`, como poden ser os seguintes:

```
named[1719]: error (network unreachable) resolving './DNSKEY/IN': 2001:500:2f::f#53
named[1719]: error (network unreachable) resolving './DNSKEY/IN': 2001:500:3::42#53
named[1719]: error (network unreachable) resolving './DNSKEY/IN': 2001:503:ba3e::2:30#53
named[1719]: error (network unreachable) resolving 'google.es/DS/IN': 2001:67c:21cc:2000::64:41#53
```

Estes erros están relaccionados con IPv6, e solucionase facendo que o DNS so traballe con IPv4. No ficheiro `/etc/default/bind9`, no cal se establecen as opcións por defecto para o arranque do proceso bind9, debemos engadir a opción `-4`

```
# startup options for the server
OPTIONS="-u bind -4"
```

Bug na versión 8 de Debian

Na versión 8 de Debian, existe un bug, que fai que se obvie a configuración do ficheiro */etc/default/bind9*. Habería que engadir a opción "-4" no ficheiro */lib/systemd/system/bind9.service*

```
ExecStart=/usr/sbin/named -f -4 -u bind
```

Feito este cambio volvemos a reiniciar o servidor, e comprobaremos que non volven aparecer máis erros no ficheiro de log.

Tamén é posible facer funcionar o servidor DNS coma servidor caché empregando reenviadores.

O único necesario é introducir no ficheiro `/etc/bind/named.conf.options` esta información, coma por exemplo:

```
forwarders {
                1.2.3.4;
                5.6.7.8;
          };
```

Reemplazar 1.2.3.4 e 5.6.7.8 cos enderezos IP dos servidores DNS correctos.

Tamén debemos engadir a opción para forzar que empregue reenviadores. Do contrario, intentará sempre primeiro unha consulta iterativa a través dos servidores raíz, no canto dunha recursiva

```
 forward only;
```

A continuación é necesario reiniciar o servidor DNS para habilitar a nova configuración. Introducimos na liña de comandos:

```
sudo service bind9 restart
```

1.3.2 Zonas mestras

Nesta sección configuraremos BIND9 como servidor primario dunha zona mestra chamada "*exemplo.com*". Simplemente necesitarás substituír este nome co teu FQDN comleto.

1.3.2.1 Zona de resolución directa ou de reenvío

Para engadir unha zona DNS de reenvío ao BIND, debemos tornalo en servidor mestre primario, e o primeiro paso é editar o ficheiro `/named.conf.local`:

```
zone "exemplo.com" {
type master;
        file "db.exemplo.com";
};
```

Agora empregaremos un modelo de zona para crear o arquivo de zona `/etc/bind/db.exemplo.com`:

```
sudo cp /etc/bind/db.local /var/cache/bind/db.exemplo.com
```

Editamos o arquivo de zona `/var/cache/bind/db.exemplo.com` cambiando localhost. polo FQDN do noso servidor, deixando o "." adicional ao final. Tamén cambiamos 127.0.0.1 polo enderezo IP do servidor de nomes DNS e root.localhost por un email válido, pero cun carácter "." no canto do típico símbolo "@", deixando tamén o "." ao final. Anque non é necesario, tamén cambiamos o comentario para indicar a que zona pertence o ficheiro.

Os demáis parámetros a configurar no SOA son os seguintes:

- *Serial*: Versión actual do arquivo de zona. Se temos un servidor secundario que se actualiza contra este, comprobará se o seu Serial é menor ou igual. Se é menor fará unha transferencia de zona para actualizarse.
- *Refresh*: Indicaralle ao servidor secundario con que frecuencia se comproba se hai novos cambios no principal.
- *Retry*: Se o intento anterior de actualización fallou indica canto espera para volver a tentalo de novo.
- *Expire*: No caso de que non se puidese facer ningunha actualización indica cando deixa de ser válido o arquivo de zona do servidor secundario.
- *Negative Cache TTL*: Tempo de vida de cada rexistro de recurso nas cachés de clientes e servidores.

Creamos un rexistro de recursos tipo A, para o dominio base *exemplo.com*. Adicionalmente, tamén creamos outro rexistro tipo A record para ns.exemplo.com, que corresponde ao enderezo IP do servidor DNS:

```
;
; BIND data file for exemplo.com
;
$TTL    604800
@       IN      SOA     ns.exemplo.com. root.exemplo.com. (    ; O rexistro soa debe ter o mesmo nome que o NS
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
@       IN      NS      ns.exemplo.com.                        ; O rexistro NS debe ter o mesmo nome que o soa e apuntar a un rexist
ns      IN      A       192.168.1.10
@       IN      A       192.168.1.10
```

Deberáse incrementar o número de serie especificado con **Serial** cada vez que fagamos cambios no arquivo de zona. Se se fan múltiples cambios antes de reiniciar o servidor BIND, so é necesario incrementalo unha única vez.

Agora xa podemos engadir rexistros de recursos DNS no fondo do arquivo de zona.

É comun que moitos administradores engadan como *Serial* ao arquivo de zona un número tal como 2012010100 que ven sendo a data no seguinte formato *yyyymmddss* sendo ss o *número de serie*. Así sabemos cando foi a última modificación.

Antes de reiniciar os cambios, podemos comprobar se o ficheiro de zona é sintácticamente correcto, co comando

```
named-checkzone
```

Unha vez feitos os cambios ao arquivo de zona é necesario reiniciar BIND9 para que os cambios sexan aplicados:

```
sudo service bind9 restart
```

1.3.2.2 Zona de resolución inversa

A zona inversa permítenos resolver enderezos IP en nomes:

Editamos `/etc/bind/named.conf.local` e engadimos a definición da zona:

```
zone "0.168.192.in-addr.arpa" {
        type master;
        file "/var/cache/bind/db.192";
};
```

 Débese reemplazar 0.168.192 cos octetos da rede correspondente que se estea empregando. Do mesmo xeito que o ficheiro /var/cache/bind/db.192.168.0, que debe encaixar co primeiro octeto do enderezo de rede.

Agora empregaremos un modelo de zona para crear o arquivo de zona `/etc/bind/db.192`:

```
sudo cp /etc/bind/db.127 /var/cache/bind/db.192
```

Editamos o arquivo de zona `/var/cache/bind/db.192.168.0` cambiando localhost. polo FQDN do noso servidor, deixando o "." adicional ao final. Tamén cambiamos 127.0.0.1 polo enderezo IP do servidor de nomes DNS e root.localhost por un email válido, pero cun carácter "." no canto do típico símbolo "@", deixando tamén o "." ao final. Anque non é necesario, tamén cambiamos o comentario para indicar a que zona pertence o ficheiro.

Creamos un rexistro de recursos tipo A, para o dominio base *exemplo.com*. Adicionalmente, tamén creamos outro rexistro tipo A record para ns.exemplo.com, que corresponde ao enderezo IP do servidor DNS:

```
;
; BIND reverse data file for local 192.168.0.XXX net
;
$TTL    604800
@       IN      SOA     ns.exemplo.com. root.exemplo.com. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns.exemplo.com.
10      IN      PTR     ns.exemplo.com.
```

 Deberáse incrementar o número de serie especificado con **Serial** cada vez que fagamos cambios no arquivo de zona. Se se fan múltiples cambios antes de reiniciar o servidor BIND, so é necesario incrementalo unha única vez.

Agora xa podemos engadir rexistros de recursos DNS no fondo do arquivo de zona.

Unha vez feitos os cambios ao arquivo de zona é necesario reiniciar BIND9 para que os cambios sexan aplicados:

```
sudo service bind9 restart
```

1.3.3 Rexistros de recursos

- Os rexistros tipo **A** enlazan nomes con enderezos IPv4, e os **AAAA** para enderezos IPv6

```
www      IN    A      192.168.0.12
www      IN    AAAA      2001:0db8::12
```

- Os rexistros tipo **CNAME** empreganse para crear un "alias" doutro nome definido por outro rexistro tipo CNAME ou tipo A. Taén traballan con rexistros tipo AAAA

```
web     IN    CNAME  www
```

- Os rexistros **PTR** nas zonas de resolución inversa asocian un enderezo IP con un nome, ao contrario dos rexistros tipo A ou AAA. So se poden atopar nas zonas de resolución inversa.

```
10      IN      PTR     ns.example.com.
```

- Os rexistros **MX** permitennos definir cales son os intercambiadores de correo, é dicir, a onde os servidores de correo van enviar os correos electrónicos. Debe apuntar obrigatoriamente a un rexistro tipo A ou AAAA. Estes rexistros a maiores incluen un número que indica a prioridade dese intercambiador de correo. Os rexistros MX so se poden atopar nas zonas de resolución directa.

```
@       IN    MX  1   mail.exemplo.com.
mail    IN    A       192.168.0.13
```

- Os rexistros **NS** empreganse para definir que servidores son os que teñen copias da zona. Debe obrigatoriamente apuntar a un rexistro tipo A ou AAAA, nunca a un CNAME. Aquí un deles será o primario e o rexistro SOA da zona apuntará cara el.

```
@        IN    NS     ns.exemplo.com.
@        IN    NS     ns2.exemplo.com.
ns      IN    A      192.168.0.10
ns2     IN    A      192.168.0.11
```

- Os rexistros **TXT** permiten introducir calquera información arbitraria, como pode ser unha frase, un comentario, ou calquera outro texto

```
joe        IN      TXT    "somewhere over the rainbow"
```

 É moi típico introducir o símbolo @ no lugar do nome dun recurso ou do nome da zona. Este @ substituirase polo valor introducido na directiva **`$ORIGIN`**. Esta directiva pode estar definida no ficheiro da zona (sóese poñer ao inicio do ficheiro antes do rexistro SOA, ou pode omitirse. Nese caso tomará o nome da zona.

```
$ORIGIN exemplo.com.
@     IN  SOA  ns1.exemplo.com. hostmaster.exemplo.com. (
...
```

Tamén é habitual (e aparece frecuentemente en moitos manuais) en vez de introducir o nome do rexistro de recursos ou o simbolo @, **deixar ese parámetro en branco** e introducir todos os demáis. Esta práctica pode ser un pouco confusa e non é moi aconsellable. No caso de que se deixe en branco, tomará como valor o definido na clausula **$ORIGIN** se está definida, ou último nome introducido (o último antes de definir o rexistro no que se deixa o nome en branco).

Este arquivo de zona é perfectamente válido:

```
$TTL    604800
@       IN      SOA     ns.exemplo.com. root.exemplo.com. (
                        3
                        604800
                        86400
                        2419200
                        604800 )
        IN      NS      ns.exemplo.com.
        IN      MX      10      mail.exemplo.com.
        IN      A       192.168.0.10
ns      IN      A       192.168.0.10
mail    IN      A       192.168.0.10
web     IN      A       192.168.0.14
        IN      A       192.168.0.15
```

Pódese atopar máis información deste tipo de substitucións na seguinte páxina

1.3.4 Zonas secundarias

Unha vez que o servidor mestre primario foi configurado, é aconsellable ter un servidor mestre secundario, para manter a dispoñibilidade (por se cae o primario), balanceo de carga, ...

Primeiro no servidor mestre primario, débense permitir as transferencias de zona. (Máis ben habería que restrinxilas, xa que por defecto está permitido transferir a zona a calquera servidor que o solicite, o cal pode ser inseguro). Engadimos a cláusula **`allow-transfer`** as zonas que o desexemos. Nos

faremolo coas zonas directas e inversas:

```
zone "exemplo.com" {
        type master;
        file "/etc/bind/db.example.com";
        allow-transfer { 192.168.0.11; };
};

zone "0.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/db.192";
        allow-transfer { 192.168.0.11; };
};
```

 . Reemplazar 192.168.0.11 polo enderezo do servidor que vai ser mestre secundario. Tamén podemos poñer un enderezo de rede, ou unha acl.

Reiniciamos o bind9 no servidor mestre primario:

```
sudo service bind9 restart
```

A continuación no equipo que vai ser mestre secundario, instalamos o paquete bind9 ao igual que no servidor mestre primario. Logo, editamos o ficheiro `/etc/bind/named.conf.local` e engadimos as seguintes declaracións para as zonas de resolución inversa e directa.

```
zone "exemplo.com" {
type slave;
        file "db.exemplo.com";
        masters { 192.168.0.10; };
};

zone "0.168.192.in-addr.arpa" {
type slave;
        file "db.192";
        masters { 192.168.0.10; };
};
```

 . Reemplazamos 192.168.0.10 co enderezo IP do servidor DNS mestre primario.

Reiniciamos o bind9 no servidor mestre primario:

```
sudo service bind9 restart
```

Podemos observar no ficheiro de log (/var/log/syslog) do servidor mestre secundario que se fixo correctamente a transferencia de zona.

```
Oct  9 14:34:24 ns2 named[1851]: zone exemplo.com/IN: Transfer started.
Oct  9 14:34:24 ns2 named[1851]: transfer of 'exemplo.com/IN' from 192.168.0.10#53: connected using 192.168.0.11#41341
Oct  9 14:34:24 ns2 named[1851]: zone exemplo.com/IN: transferred serial 3
Oct  9 14:34:24 ns2 named[1851]: transfer of 'exemplo.com/IN' from 192.168.0.10#53: Transfer completed: 1 messages, 10 records, 250 
Oct  9 14:34:25 ns2 named[1851]: zone 0.168.192.in-addr.arpa/IN: Transfer started.
Oct  9 14:34:25 ns2 named[1851]: transfer of '0.168.192.in-addr.arpa/IN' from 192.168.0.10#53: connected using 192.168.0.11#44011
Oct  9 14:34:25 ns2 named[1851]: zone 0.168.192.in-addr.arpa/IN: transferred serial 2
Oct  9 14:34:25 ns2 named[1851]: transfer of '0.168.192.in-addr.arpa/IN' from 192.168.0.10#53: Transfer completed: 1 messages, 4 rec
```

No log do servidor DNS mestre primario, tamén queda constancia da transferencia feita:

```
Oct  9 14:34:31 ns named[4965]: client 192.168.0.11#41341 (exemplo.com): transfer of 'exemplo.com/IN': AXFR started
Oct  9 14:34:31 ns named[4965]: client 192.168.0.11#41341 (exemplo.com): transfer of 'exemplo.com/IN': AXFR ended
Oct  9 14:34:32 ns named[4965]: client 192.168.0.11#44011 (0.168.192.in-addr.arpa): transfer of '0.168.192.in-addr.arpa/IN': AXFR st
Oct  9 14:34:32 ns named[4965]: client 192.168.0.11#44011 (0.168.192.in-addr.arpa): transfer of '0.168.192.in-addr.arpa/IN': AXFR en
```

Unha vez que se fixo a transferencia, os ficheiros das zonas subordinadas, non se almacenan no directorio `/etc/bind` senón que se gardan en `/var/cache/bind` e tampouco se poderá ver o contido xa que están en formato **raw** en vez de formato texto como ocorre no servidor mestre primario.

Se quixeramos ver o contido, hai utilidades para transformar o ficheiro de zona, de modo raw a modo texto

```
named-compilezone -f raw -F text -o <arquivo da zona novo a gardar> <nome da zona> <arquivo da zona orixinal>
```

so cada vez que se transfira a zona, habería que volver a convertir o arquivo de modo raw a modo texto

E a outra posibilidade e forzar a que a transferencia se faga sempre en modo texto. So temos que indicar na configuración da zona no servidor secundario (/etc/bind/named.conf.local) a seguinte opción:

```
masterfile-format text
```

Esta opción si é permanente e perdurable no tempo.

Destacar, que todo isto non é necesario, xa que os clientes son capaces de resolver os nomes, que é o realmente importante.

1.3.4.1 Notificacións aos servidores subordinados

Sempre e cando fagamos un cambio na zona mestra, ese cambio débese propagar aos servidores secundarios. Hai que lembrarse sempre de **aumentar o Serial** cando fagamos cambios antes de reiniciar o servidor mestre primario.

Tamén no servidor mestre primario, se debe notificar aos servidores mestre subordinados. Por defecto, so se lles notifica aos servidores que teñen un rexistro NS definido. Se queremos notificar cambios a servidores que non teñen o seu rexistro NS definido, debemos incluír unha clausula **`also-notify`** na declaración da zona. Exemplo:

```
zone "exemplo.com" {
        type master;
        file "/etc/bind/db.example.com";
        allow-transfer { 192.168.0.11; };
        also-notify { 192.168.0.12; };
};

zone "0.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/db.192";
        allow-transfer { 192.168.0.11; };
        also-notify { 192.168.0.12; };
};
```

Tamén hai que ter a opción de notificar os cambios activada. Esa opción pode configurarse no ficheiro `/etc/named.conf.options`

```
notify yes;
```

ou dentro da definicion das zonas no ficheiro `/etc/bind/named.conf.local`

```
zone "exemplo.com" {
        type master;
        notify yes;
        file "/etc/bind/db.example.com";
        allow-transfer { 192.168.0.11; };
        also-notify { 192.168.0.12; };
};

zone "0.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/db.192";
        allow-transfer { 192.168.0.11; };
        also-notify { 192.168.0.12; };
};
```

Por defecto, se non aparece, a cláusula **notify** está sempre activa.

1.3.5 Subdominios e delegación de zonas

Para conseguir configurar subdominios temos dúas alternativas:

- Crear un subdominio virtual, neste caso é un so servidor DNS o que vai ter autoridade sobre o dominio e sobre o subdominio.
- Delegar o subdomio, é dicir o servidor DNS autorizado para o dominio vai delegar a xestión e autorización do subdominio a outro servidor DNS.

1.3.5.1 Crear un subdominio virtual

Neste caso supoñemos que temos configurado un servidor DNS onde configuramos a zona exemplo.com no fichero /etc/bind/db.exemplo.com. A configuración do subdominio virtual está ao final do ficheiro:

```
$TTL    86400
@       IN      SOA     ns1 mail (
                              4
                         604800
                          86400
                        2419200
                          86400 )
@               NS      ns1
ns1     IN      A       192.168.0.2
www     IN      A       192.168.0.1

$ORIGIN gl.exemplo.com.
www     IN      A      10.10.0.3
```

Despois de reiniciar o servidor podemos facer unha consulta coa utilidade dig, da seguinte maneira:

```
dig @localhost www.gl.exemplo.com
```

1.3.5.2 Delegación de subdominios

Nesta ocasión partimos dun servidor DNS con autoridade sobre o dominio exemplo.com (ns1.exemplo.com), que vai delegar a xestión do subdominio es.exemplo.com a otro servidor DNS (`nssub.es.exemplo.com`). Vexamos a configuración do arquivo de zona no servidor principal:

```
$TTL    86400
@       IN      SOA     ns1 mail (
                              4
                         604800
                          86400
                        2419200
                          86400 )

                NS      ns1
ns1     IN      A       192.168.0.2
www     IN      A       192.168.0.1
```

Podemos observar que a resolución faise correctamente, e como sinalamos anteriormente o servidor con autoridade (rexistro NS) é o servidor ns1.exemplo.com que, en realidad, é o servidor con autoridade do dominio exemplo.com

No dominio principal:

A zona está definida no fichero /etc/bind/db.exemplo.com, onde teremos que indicar cal é o servidor DNS con autoridade para o subdominio, é dicir indicaremos o servidor DNS ao que vamos a delegar a xestión do subdominio gl.exemplo.com, que no noso caso será nssub.es.exemplo.com.

```
$TTL    86400
@       IN      SOA     ns1 mail (  ;Dominio principal
                              4
                         604800
                          86400
                        2419200
                          86400
                        )
                NS      ns1
ns1     IN      A       192.168.0.2
www     IN      A       192.168.0.1

$ORIGIN es.exemplo.com.
@       IN      NS      nssub
nssub   IN       A      192.168.0.6 ; glue record
```

Como podemos observar o servidor DNS con autoridade sobre a zona gl.exemplo.com, será nssub.gl.exemplo.com que se atopa no enderezo 192.168.0.6.

No **servidor DNS do subdominio** (es.exemplo.com):

Configuramos o segundo servidor DNS (nssub.es.exemplo.com), ao que vamos delegar la xestión do dominio gl.exemplo.com. O primeiro que temos que facer é definir a zona que corresponde co subdominio no fichero /etc/bind/named.conf.local:

```
zone "es.exemplo.com" {
  type master;
  file "db.es.exemplo.com";
};
```

e no fichero /var/cache/bind/db.es.exemplo.com, quedaría:

```
$TTL    86400
@       IN      SOA     nssub mail ( ;Subdominio delegado
                              4
                         604800
                          86400
                        2419200
                          86400 )

                  NS      nssub
nssub     IN      A       192.168.0.6
www       IN      A       192.168.0.3
```

Dominios ficticios

No caso de empregar dominios ficticios, como por exemplo *.lan*, deberemos ter desactivados os reenviadores no servidor principal, para que faga sempre unha búsqueda iterativa. De non facer isto. calquera consulta dun rexistro do subdominio contra o servidor principal (o que delega) sería remitida ao seu reenviador (xa que non é autoritativo nesa zona) e como os dominios ficticios, non poden ser alcanzados desde o raíz "." pois non se resolve a consulta. Unha solución menos drástica sería deshabilitar os reenviadores para unha zona determinada, coma no seguinte exemplo

```
zone "exemplo.com" {
type master;
        file "db.exemplo.com";
        forwarders {};
};
```

1.3.6 Logging

BIND9 ten unha cantidade de opcións de log enorme. Hai dúas opcións principais de configuración: A opción *channel* indica a lugar se escriben os logs, e a opción *category* indica que se escribe no ficheiro de log.

Se non se configura nada por defecto, tómase o seguinte:

```
logging {
     category default { default_syslog; default_debug; };
     category unmatched { null; };
};
```

No seguinte exemplo, gardamos todas as consultas no ficheiro `/var/log/named/query.log`:

```
# A engadir en /etc/bind/named.conf.options
logging {
    channel query.log {
        file "/var/log/named/query.log";
        // Set the severity to dynamic to see all the debug messages.
        severity dynamic;
    };

    category queries { query.log; };
};
```

Se o queremos sacar polo log do sistema

```
# A engadir en /etc/bind/named.conf.options
logging {
    channel query.log {
        syslog;
        // Set the severity to dynamic to see all the debug messages.
        severity dynamic;
    };

    category queries { query.log; };
};
```

Poden atoparse máis opcións de configuración neste enlace

Posto que o demonio named estase executando como usuario *bind* hai que cambiar o usuario propietario do ficheiro `/var/log/named/query.log` :

```
mkdir -p /var/log/named
touch /var/log/named/query.log
chown bind /var/log/named -R
```

Tamén debemos comprobar que o *AppArmor* de Ubuntu é capaz de escribir nese directorio e/ou ficheiro de log. Chequeamos que no ficheiro `/etc/apparmor.d/usr.sbin.named` hai algo similar a

```
 /var/log/named/** rw,
 /var/log/named/ rw,
```

Reiniciamos o servizo bind9

```
sudo /etc/init.d/bind9 restart
```

1.4 Configuración avanzada

Nesta sección descríbese diversa funcionalidade que controla o comportamento do ficheiro "`named.conf`". BIND9 soporta unha lista ben longa de opcións soportadas. Aquí trataranse as principais.

A orde de introdución das distintas opcións sería a seguinte:

```
// acl clause if required
// defining first avoids forward name references
acl "name" {...};
logging {...};
// usually requires at least a file statement
// unless you are using the system log
options {...};
// other clauses/statements (as required)
// zones clauses including 'required' zones
zone {...};
....
zone {...};
```

1.4.1 acl

A clausula **acl**[2]

```
permite afinar o control sobre que usuarios ou equipos poden realizar operacións co servidor de nomes. A súa sintaxe é a seguinte:

acl acl-name {
    address_match_list
};
```

Exemplo:

```
 acl "moreips" {
  10.0.0.1;
  192.168.23.128/25; // 128 IPs
};
```

Están predefinidas as seguintes acl's:

- **"none"**: ningún host
- **"any"**: calquera host
- **"localhost"**: encaixa con todos os enderezos do host, incluido o 127.0.0.1
- **"localnets"**: encaixa con todas as redes e subredes as que está conectado directamente o host

1.4.2 Chaves TSIG

As chaves TSIG, son un mecanismo de cifrado simétrico, que necesita unha chave segreda, pero compartida entre multiples host. As chaves TSIG, poden empregarse para encriptar as comunicacións entre servidores DNS, entre servidores DNS e servidores DHCP, entre servidores DNS e clientes que actualizan os servidores, ...

Problemas de sincronización do reloxo dos equipos

Se os equipos teñen a súa hora moi desfasada entre sí, é probable que a transferencia de zonas con chaves TSIG (ou calquera outro servizo que utilice encriptación en xeral) non funcione correctamente. Para sincronizar a hora, instalamos o paquete *ntpdate* en todos os equipos, e despois sincronizamos coa seguinte instrucción:

```
ntpdate -u hora.roa.es
```

Todas as chaves, deben gardarse nun ficheiro que se debe incluír desde o `/etc/bind/named.conf`. O formato de configuración da chave é o seguinte:

```
key "<nome da chave>"{
  algorithm <algoritmo de xeneración da chave>;
  secret "<chave xerada co algoritmo>";
};
```

Exemplo:

```
key "TRANSFER" {
          algorithm hmac-md5;
          secret "0jnu3SdsMvzzlmTDPYRceA==";
};
```

As chaves, poden xerarse coa utilidade *`tsig-keygen`*

```
 tsig-keygen nomechave
```

que producirá unha saída por pantalla indicando, entre outras a configuración da chave, que debemos copiar e pegar no ficheiro que conteña as chaves TSIG.

Por último so nos basta incluír o ficheiro de chaves desde /etc/bind/named.conf

1.4.3 Transferencias de zonas seguras empregando chaves TSIG

Unha vez configurada a chave en **ambos servidores** primario e secundario, indicamos na configuración da zona no servidor mestre primario

```
zone "exemplo.com" {
        type master;
     "/etc/bind/db.example.com";
        allow-transfer { key "nomechave"; };
};
```

No servidor secundario, deberemos indicar que chaves emprega o servidor primario. Para elo, incluímos no ficheiro de configuración

```
server 192.168.0.10 { //Indicar o enderezo IP do servidor primario
 keys { nomechave; };
};
```

1.4.4 Control remoto

O servidor BIND pode ser controlado tanto de forma remota coma desde o equipo local coa utilidade **`rndc`**. A utilidade *rndc* instálase co paquete *bind9utils* Por defecto, todo servidor BIND permite ser controlado desde o propio equipo coa utilidade rndc desde o enderezo 127.0.0.1 e o porto 953.

Esta utilidade empega unha chave md5. Cando se instala o paquete bind9, esta chave está incluida no ficheiro `/etc/bind/rndc.key` e non fai falta incluir ese ficheiro na configuración de BIND

Se queremos configurar outro a maiores, introducimos unha sección `controls`[3] no ficheiro *named.conf.options*

```
controls {
   inet inet_spec [inet_spec]  ;
};
```

A clausula inet define o método de acceso a utilidade *rndc*. O parámetro `inet_spec` pode ter o seguinte formato:

```
inet_spec = ( ip_addr | * ) [ port ip_port ] allow {  address_match_list  } [keys {  key_list  }];
```

ip_addr define o enderezo IP do servidor DNS no que atenderá conexións da utilidade *rndc*. O comodin * engloba a todos os enderezos do servidor DNS incluido o enderezo do interface *loopback*. O parámetro *ip_port* permite escoller o porto no que atende peticións. Por defecto ese porto é o 953. *address_match_list* restrinxe a que equipos se lles permite establecer conexións ao servidor DNS coa utilidade *rndc* e pode ser tanto unha *acl* coma un enderezo IP coma un par enderezo IP/máscara de rede. Por último *key_list* define a lista de chaves que deben posuír os equipos remotos para poder interactuar coa utilidade *rndc* contra o servidor DNS. Se non se emprega ningunha, usarase a chave gardada no ficheiro /etc/bind/rndc.key

Un exemplo de todo isto podería ser:

```
// named.conf.options fragment
acl "rndc-users" {
     10.0.15.0/24;
     !10.0.16.1/24; // negated
     2001:db8:0:27::/64; // any address in subnet
 };
....
key "rndc-remote" {
     algorithm hmac-md5;
     secret "OmItW1lOyLVUEuvv+Fme+Q==";
};
controls {
     // local host - default key
     inet 127.0.0.1 allow {localhost;};
     inet * port 7766 allow {"rndc-users";} keys {"rndc-remote";};
};
```

Desta maneira, ademais de permitirse o acceso desde localhost ao porto 953 e a chave incluída no ficheiro /etc/bind/rndc.key, tamén se permite a través de todos os enderezos IP do servidor DNS (icluído localhost) o acceso desde os equipos cuxo enderezo IP encaixe na *acl* **rndc-users** empregando a chave rndc-remote.

1.4.4.1 Configuración do equipo remoto

Os equipos remotos, deberán ter instalado o paquete **bind9utils**, e ter configurado no ficheiro */etc/bind/rndc.conf* a mesma chave que no servidor DNS. Seguindo o exemplo anterior, debería haber un tal que así:

```
key "rndc-remote" {
     algorithm hmac-md5;
     secret "OmItW1lOyLVUEuvv+Fme+Q==";
};

options {
        default-key "rndc-remote";
        default-server 10.0.15.1; #IP do servidor
        default-port 7766;
};
```

Todo o definido dentro de options, pode omitirse e introducirse mediante opcións do comando *rndc*

1.4.4.2 A utilidade rndc

A utilidade **rndc** permítenos controlar o servidor de forma remota pero tamén desde o propio equipo. Se se fai desde un equipo remoto é recomendable configurar o ficheiro /etc/bind/rndc.conf como se indicou anteriormente. Esta utilidade permítenos moitas máis opcións que o simple reinicio do servizo bind9.

```
rndc <options> <command> <command-options>
```

As opcións máis empregadas son as seguintes:

- **halt**: detén o servidor sen engadir as actualizacións dinámicas no ficheiro de zona. Estas actualización estarán dispoñibles cando se reinicie o servidor.
- **stop**: detén o servidor engadindo as actualizacións dinámicas no ficheiro de zona.
- **status**: Amosa o estado do servidor.
- **flush**: Borra a caché do servidor DNS.
- **reload**: Recarga todas as zonas mantendo o contido da caché. Se engadimos o nome da zona a continuación so recarga a zona indicada.d
- **refresh**: Recarga a base de datos do servidor DNS.
- **freeze**: Suspende as actualizacións dinámicas en todas as zonas ou na zona indicada.
- **thaw**: Habilita de novo as actualizacións dinámicas en todas as zonas conxeladas ou na zona indicada.
- **sync**: Sincroniza os cambios das actualizacións dinámicas almacenadas nos ficheiros .jnl, e incorporaas ao ficheiro de zona. Se se emprega a opción *-clean* bórrase o ficheiro .jnl. Se non se indica o nome de ninguna zona, faise a sincronización de todas elas.
- **reconfig**: Recarga os ficheiros de configuración, engadindo as novas zonas que non existían antes. Non recarga os ficheiros de zonas xa existentes.
- **addzone**: Engade unha zona sen reiniciar o servidor. Require que a opción **allow-new-zones** teña o valor **yes** no ficheiro */etc/bind/named.conf.options* . A cadea de texto que se introduce neste comando sería a mesma que se introduciria na definición da zona no ficheiro '/etc/bind/named.conf.local*. O ficheiro de zona debe existir previamente. A configuración gárdase nun ficheiro* hash.nzf *sendo hash unha cadea HASH criptográfica, no cartafol /var/cache/bind. As zonas engadidas así son persistentes ao reinicio do servizo bind.*

  Exemplo:

  ```
  rndc addzone example.com '{ type master; file "example.com.db";};'
  ```

   nótese o uso das comiñas simples e os puntos e comas.
- **delzone**: Permite borrar zonas sen reiniciar o servidor. So se poden borrar as zonas engadidas con *addzone*

Para máis información consultar a páxina de manual de rndc[4]

1.4.5 DNSSEC

Para ofrecer DNSSEC nas respostas do noso dominio, garantindo aos usuarios/aplicacións a validez destas, necesitamos ter cceso ao panel de control do rexistrador do dominio. Nel será onde informaremos do rexistro DS (Delegation Signer) que indicará cal é a nosa chave (KSK) na zona pai do noso dominio (normalmente .com, .net, .org, etc,...).

Antes de nada, deberemos asegurarnos que o noso BIND está configurado para soportar e validar DNSSEC.

Despois crearemos as chaves KSK e ZSK. Compostas, respectivamente, dunha chave privada que usaremos para asinar a chave ZSK e a zona, e dunha chave pública que irá informada o ficheiro de zona do noso dominio.

Posteriormente asinaremos a zona do noso dominio coas chaves creadas e publicaremos, mediante o panel de control do noso rexistrador, o rexistro DS correspondente á nosa KSK. Ésto permitirá a outros servidores DNS validar a información provinte dos nosos servidores DNS.

1.4.5.1 Configuración de Bind para soportar DNSSEC

Tres son as opcións principais que controlan el soporte DNSSEC en Bind. Haberá que introducilas no ficheiro */etc/bind/named.conf.options*:

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

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

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

        // forwarders {
        //      0.0.0.0;
        // };

        auth-nxdomain no;    # conform to RFC1035
        listen-on-v6 { any; };
        dnssec-enable yes;
        dnssec-validation yes;
        dnssec-lookaside auto;
        //bindkeys-file "/etc/bind/bind.keys";
};
```

Tamén deberemos incluír as chaves da zona raíz "." para que se poida comezar a cadea de confianza por algún lugar. Podemos incluír no ficheiro "/etc/bind/named.conf" o ficheiro "/etc/bind/bind.keys" ou o que coneña as chaves da zona raíz.

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

Por último reiniciamos o servidor.

1.4.5.2 Comprobación de que DNSSEC está activado

Desde o propio servidor, mediante a utilidade *dig* coa opción *+dnssec* podemos comprobar se o noso servidor ten activo DNSSEC ou non:

```
$dig org. SOA +dnssec

; <<>> DiG 9.7.3 <<>> org. SOA +dnssec
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 31736
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 7, ADDITIONAL: 1

...[cut]...

;; Query time: 597 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu May 12 21:44:43 2011
;; MSG SIZE  rcvd: 536
```

Se o flag **ad** está activo, DNSSEC está configurado.

Tamén podemos preguntar se podemos resolver o nome *www.dnssec-failed.org* que ten deliberadamente unha configuración DNSSEC inválida.

```
# Se o noso DNS non usa (correctamente) DNSSEC:
$ host www.dnssec-failed.org
www.dnssec-failed.org has address 69.252.193.191
www.dnssec-failed.org has address 68.87.109.242

# Se o noso DNS usa correctamente DNSSEC:
$ host www.dnssec-failed.org
Host www.dnssec-failed.org not found: 2(SERVFAIL)
```

1.4.5.3 Creación das chaves de asinado

crearemos dos chaves. La primera, KSK (Key Signing Key), será usada para asinar la ZSK (Zone Signing Key), permitiendo cambiar esta última de forma periódica de forma sencilla y sin realizar cambios en el registrador del dominio:

```
dnssec-keygen -K /etc/bind/keys/ -a RSASHA256 -b 4096 -n ZONE -3 -f KSK exemplo.com
```

Este comando creará un par de chaves no directorio (-K) /etc/bind/keys. O nome dos ficheiros será algo como Kexemplo.com.+008+51772.private ou .key. Onde o ficheiro .private contén a chave privada e o .key a pública. O "+008" indica o algoritmo elixido (-a RSASHA256) e o "+51772" o identificador da chave (usarémolo logo no rexistrador). Con "-b 4096" eliximos o tamaño da chave a xerar, 4096 é unha boa opción para a KSK. "-n ZONE" indica que a chave será usada para traballar con zonas DNSSEC, "-3" que usaremos NSEC3 e por último "-f KSK" indica o tipo de chave.

Se vemos que tarda moito, poderíamos instalar o paquete **haveged** para que xenere números aleatorios de forma máis rápida.

Si nos fixamosno ficheiro *.key*, veremos unha entrada de zona DNS (DNSKEY neste caso) coa seguinte información:

```
exemplo.com. IN DNSKEY 257 3 8 AwEA......
```

- **257** indica que se trata de unha KSK
- **3** é o protocolo, e sempre ten ese valor RFC4034
- **8** é algoritmo elixido. Ver lista na wikipedia. Neste caso RSA/SHA-256
- **AwEA.....** e o resto será a chave pública

A única diferenza co .key da ZSK é que o primeiro campo terá un valor de 256 (en vez de 257)

Despois crearemos a ZSK, cun comando practicamente igual ao anterior, neste caso cun tamaño de chave algo menor (2048 bits) e sen "marcar" como KSK:

```
dnssec-keygen -K /etc/bind/keys/ -a RSASHA256 -b 2048 -n ZONE exemplo.com
```

Unha vez máis obteremos dous ficheiros en */etc/bind/keys* (un .private e un .key) do tipo Kexemplo.com.+008+50946. Onde, 008 é o algoritmo e 50946 o identificador/etiqueta da chave.

1.4.5.4 Asinando o ficheiro de zona

Agora xa podemos asinar a nosa zona DNS. No meu caso o ficheiro chámase exemplo.com e atópase en /etc/bind. Ese ficheiro é o que seguirei modificando cando queira facer cambios na miña zona, só que despois dos cambios haberá que volver asinar e recargar a zona.

```
dnssec-signzone -d /var/cache/bind/ -K /etc/bind/keys/ -N "increment" -S \
   -3 $( dd if=/dev/random bs=16 count=1 2>/dev/null | hexdump -e \"%08x\" ) \
   -o exemplo.com /var/cache/bind/exemplo.com
```

O resultado deste comando será un ficheiro /var/cache/bind/exemplo.com.signed. Vexamos o por que das opcións empregadas:

- **-d** directorio de traballo, onde deixar a zona asinada
- **-K** directorio coas chaves xeradas anteriormente
- **-N "increment", esta opción indica que debe aumentarse o número de serie da zona ademais de asinala. É dicir, o ficheiro .signed terá un número de serie (no SOA) maior ao do ficheiro orixinal. Non é necesario poñela, xa que o aumentamos na zona orixinal.**
- **-S** opción "Smart", é dicir, que busca as chaves (no directorio indicado), sabe (polo 257/256 que vimos antes) cal é a KSK e a ZSK, úsaas correctamente (asinando os RR tipo DNSKEY coa KSK e o resto coa ZSK), inclúeas no novo ficheiro xerado, limpa e abrillanta. No caso de que as chaves se xeren con períodos de validez, encargaríase de xestionar a rotación por caducidade das mesmas. Unha marabilla!
- **-3 SALT**. Xera os rexistros NSEC3. Estes son necesarios para que non se poida enumerar por completo a zona. Como están baseados nun hash, a "salt" varía o valor do hash para que non poida inferirse o seu valor orixinal, por exemplo con dicionarios. Neste comando uso /dev/random e hexdump para xerar unha "salt" completamente aleatoria. Se non queremos usar "salt", podemos especificar un guión (-3 -) ou mesmo pór unha fixa (en hexadecimal, 32 carácteres, ...)
- **-o exemplo.com**. Indica o ORIGIN da zona, é dicir o dominio (para que non o teña que deducir do nome do ficheiro de zona, aínda que neste exemplo sexa obvio). Así que se trata dunha opción importante.
- **/var/cache/bind/exemplo.com**. O ficheiro orixinal da zona. O destino terá o mesmo nome, coa extensión .signed.

É importante notar que a sinatura dos rexistros ten caducidade. Por defecto 30 días desde a súa creación. Este dato é vital, xa que pasados 30 días o noso dominio deixará de validar correctamente e os DNS que teñan soporte para DNSSEC (hoxe case todos) deixarán de aceptar como válidas nosas respostas DNS, é dicir, desapareceremos de Internet LOL!. Se queremos modificar o período de validez da asina, podemos usar a opción -e. Por exemplo, o valor por defecto é: -e now+30D (30 Días desde agora). BAsta, por agora, dicir que basta executar o mesmo comando de sinatura de zona outra vez para xerar as asinas (coa súa validez de 30 días) unha vez máis.

Unha vez temos a zona asinada, o único que nos queda no servidor é servir a zona asinada no canto da orixinal. É dicir, na configuración do DNS cambiar:

```
zone "exemplo.com" {
 type master;
 file "/var/cache/bind/exemplo.com";
 .....
};
```

por

```
zone "exemplo.com" {
 type master;
 file "/var/cache/bind/exemplo.com.signed";
 .....
};
```

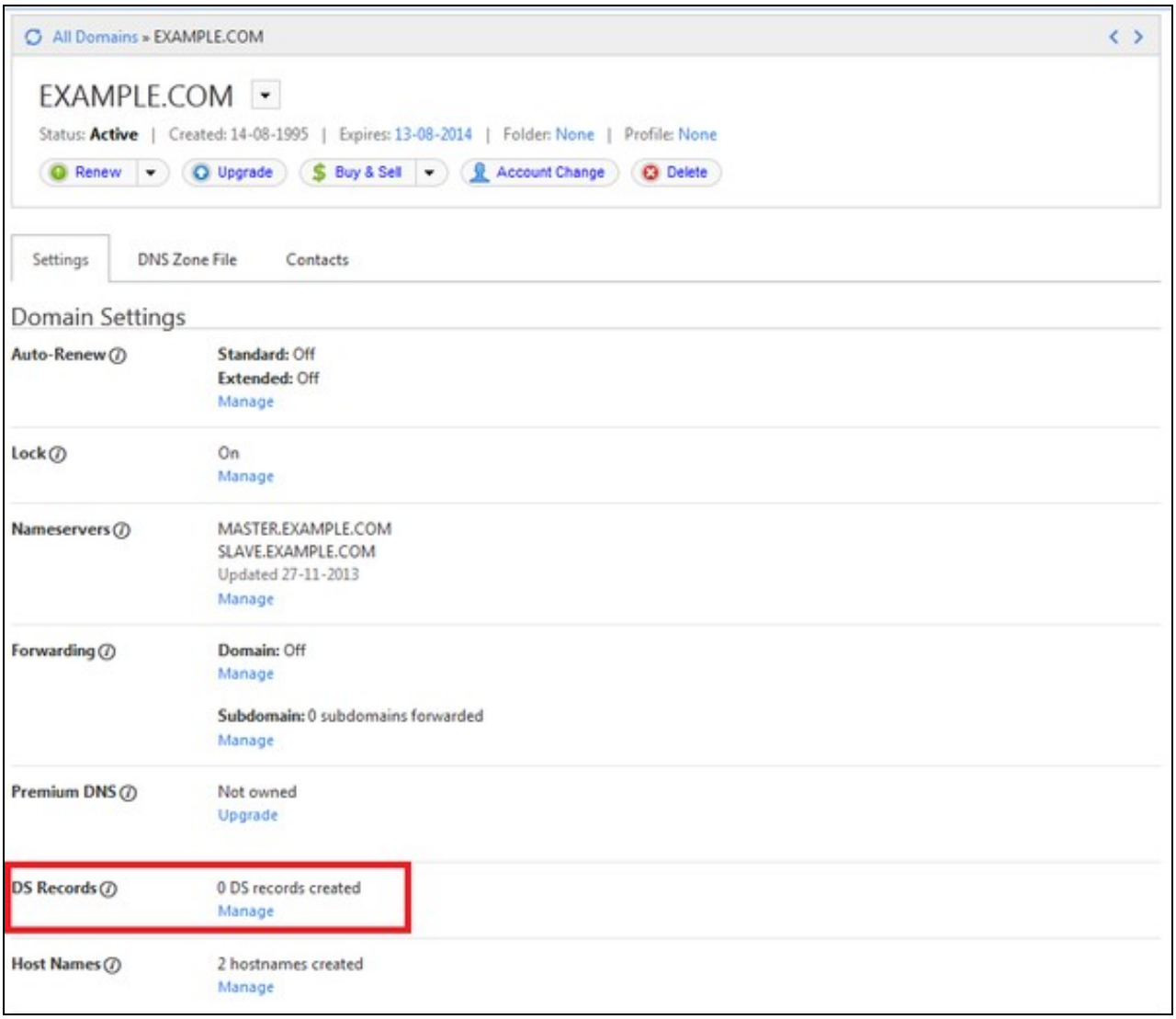
E recargala con:

```
rndc reload exemplo.com
```

1.4.5.5 Informando da KSK no rexistrador

Unha vez o noso servidor DNS está a servir a zona asinada, quédanos o último paso. Engadir a "ligazón" entre a zona pai (.com neste exemplo) e a nosa. É dicir, informar (no rexistrador) de cal é nosa KSK, para que as respostas dos nosos DNS poidan ser validadas correctamente.

Para dar de alta no rexistrador esas chaves, iso iremos ao panel de control do noso dominio e encheremos os datos do rexistro DS (Delegation Signer). Ese formulario correspondente ten esta pinta:



Pero, de onde sacamos toda esta información?. No directorio de saída, no noso caso */var/cache/bind* hai un ficheiro que comeza con nome **dsset-<nome da zona>**. No noso caso temos isto:

```
example.com.          51772 IN DS 1 F6E0738F65E7F9A8231C7BE5052ADE76FBEE3A35
example.com.          51772 IN DS 2 8040CC07D45693111A17F81DE381B26DDC2B81FFAC9324582153682C 54EB3E3C
```

Onde **51772** é o identificador/etiqueta da chave (campo keytag), o **8** o o algoritmo (alg) correspondente á chave (lembrade que neste exemplo era RSA/XA-256) e o **1** e **2** as funcións hash usadas (digest type) para crear cada resumo da chave (digest), que é o chorro hexadecimal do final. É dicir, para unha mesma chave (KSK neste caso) xéranse dous rexistros DS, con diferentes funcións de resumo (1 é XA-1 e 2 é XA-256). No rexistrador podemos informar só unha, aínda que o ideal é ter as dúas. Se por algún motivo o ficheiro dsset-<nome da zona> non se creou, podemos facelo partindo do compoñente público da clave KSK co comando dnssec-dsfromkey:

```
ddnssec-dsfromkey Kexemplo.com.+008+51772.key
#Este ficheiro debería estar en /etc/bind/keys/
```

Á hora de introducir os dous rexistros no panel de control do rexistrador, hai que eliminar o espazo no segundo rexistro.

1.4.5.6 Validar a implementación de DNSSEC

Se seguimos as indicacións anteriores, agora teremos o noso dominio baixo DNSSEC. Neste momento é vital asegurarse de que todo está correcto, xa que se o noso dominio "anuncia" (mediante a publicación dun rexistro DS no rexistrador) que soporta DNSSEC, pero a implementación é errónea (chaves non corresponden ao DS, sinaturas caducadas, ...) o noso dominio deixará de ser visible para aqueles que usen DNSSEC (que cada día serán máis). É dicir, adeus ao correo, a web, etc, ...

Para comprobar que todo está OK podemos usar servizos web como:

- http://dnssec-analyzer.verisignlabs.com/ Que lista todos os pasos para validar o soporte DNSSEC do teu dominio.

- http://dnsviz.net/ Que visualiza a relación (sinaturas) entre as claves desde o dominio raíz até o teu. Ademais de comprobar que está correcto, axuda a entender a relación entre os diferentes (sub-)dominios e as súas claves.

Se somos máis de consola, os seguintes comandos permitirannos comprobar que o noso dominio valida correctamente:

Primero gardamos as chaves (KSK y ZSK) da zona raíz (.)

```
dig . DNSKEY | grep -Ev '^($|;)' > root.keys
```

Logo facemos a validación da cadea completa (desde a raíz ata o noso dominio)

```
dig +sigchase +trusted-key=./root.keys inittab.net. SOA
```

1.5 Actualizacións dinámicas

Desde calquera cliente, podemos facer actualizacións dinámicas contra o servidor BIND co comando *nsupdate*[5]

Exemplo:

```
nsupdate actualiza.txt
```

Sendo o contido de *actualiza.txt*

```
server ns.exemplo.lan
zone exemplo.lan
update delete cliente1.exemplo.lan
update add cliente1.exemplo.lan 86400 A 192.168.0.124
show
send
```

Para permitir isto, hai que configurar as zonas no ficheiro */etc/bind/named.conf.local* para que permitan recibir actualizacións

1.5.1 Configuración do servidor para poder recibir actualizacións dinámicas

As zonas que lle permitamos recibir actualizacións dinámicas, tanto por DHCP coma por medio do comando *nsupdate* debemos indicarllo na definición da zona. Por exemplo:

```
// fragmento de named.conf.local fragment
// a clausula key so se mostra como exemplo. Pode ser definida ou incluida noutro ficheiro.
key "update-key" {
        algorithm hmac-md5;
        "HnYDB82NMCr4NBjv+YTsyw==";
};
....
zone "example.net" {
    type master;
    allow-update {none;}; // Por defecto, non se permiten actualizacións dinámicas
    ....
};
....
zone "example.com" {
....type master;
    allow-update {192.168.0.23;}; // DDNS so desde este host
    ....
};
zone "example.org" {
    type master;
    allow-update {key "update-key";};  //So aos que equipos que posúan esta chave.
    ....
};
```

Se empregamos chaves, é necesario xeralas desde o equipo que se emprega o comando *nsupdate*.

```
nsupdate -y update-key:HnYDB82NMCr4NBjv+YTsyw== actualiza.txt
```

que ven sendo inseguro xa que se escriben as chaves na liña de comandos. É máis seguro gardar o contido da chave

```
key "update-key" {
        algorithm hmac-md5;
        "sec2NMCr4NBjv+YTsyw==";
};
```

nun ficheiro de texto con premisos de lectura limitados e executar o comando

```
nsupdate -k chave.key actualiza.txt
```

Ubicación dos ficheiros de zona

A actualización dos ficheiros de zona, non se fai directamente contra o propio ficheiro de zona, senón contra outro co mesmo nome, pero rematado en ".jnl" chamado *journal file*. Periodicamente, ou cando se reinicia o servidor vólcanse os cambios contra o ficheiro de zona. Hai que asegurarse de que os ficheiros son escribibles no directorio onde están ubicados. O mellor é gardalos en */var/cache/bind*

1.6 Referencias externas

1. ? The most widely used Name Server Software
2. ? DNS BIND acl clause
3. ? DNS BIND controls clause
4. ? Ubuntu Manpage: rndc - name server control utility
5. ? Ubuntu Manpage: nsupdate - Dynamic DNS update utility

- Bruno Vila Vilariño (out 2014)