BIND, die Neunte

Wem der große Wurf gelungen

Einer der wichtigsten Dienste im Internet ist der Domain Name Service. Ohne die automatische Namensauflösung müssten wir für jede Rechner-zu-Rechner-Kommunikation numerische Adressen verwenden. Thomas Bader

Ursprünglich geht der Domain Name Service (DNS) auf eine Entwicklung von Jon Postel aus dem Jahr 1979 zurück – unter der Bezeichnung IEN-116 veröffentlichte er eine Publikation mit dem Namen „Internet Name Server“. 1982 überholte sie der RFC 830 „A Distributed System for Internet Name Service“ [13]. Der folgende Artikel geht näher auf die aktuelle Version 9 von BIND (Berkeley Internet Name Domain) und dessen neue Möglichkeiten ein. Dazu gehören zum Beispiel mehr Stabilität und Sicherheit. Der Artikel zeigt auch, wie man sich einen Nameserver von Grund auf installieren kann.

BIND [1] (oder auch ISC BIND genannt), eine Implementation des Domain Name System (DNS), wurde vom ISC (Internet Software Consortium) [2] geschrieben. Er läuft auf nahezu jedem Unix-System. An Version 9 ist unter anderem neu:

- DNSSEC (signierte Zonen)
- TSIG (signierte DNS Requests)
- Unterstützung von IPv6
- Multiple Views von Zonen
- Unterstützung von Multiprozessor-Systemen

Für die meisten Anwender werden die beiden ersten Punkte ausschlaggebend sein, auf BIND 9 zu wechseln. Ferner soll BIND 9 auch stabiler und weniger anfällig gegen Exploits sein als seine Vorgängerversionen.

Listing 1: Minimale »named.conf«

```
01 // /etc/named.conf
02
03 options {
04                 // Wo BIND seine Datenfiles speichert
05                 directory "/etc/bind";
06                 // In dieser Datei wird die PID
   hinterlegt
07                 pid-file "/var/run/named.pid";
08         };
09
10 //  Ist ist die root-Zone. Hier stehen die root-
   Nameserver drin.
11 zone "." {
12                 type hint;
13                 file "named.root";
14         };
```

Basis-Installation

Die aktuelle Version lädt man sich am besten von der BIND-Homepage [1] herunter, entpackt und übersetzt sie:

```
tar xvfz bind-9.1.3.tar.gz
cd bind-9.1.3.tar.gz
```

BIND benutzt GNU Automake. Das heißt, dass wir ihn so kompilieren können, wie von anderer Software her gewohnt:

```
./configure
make
make install
```

Das war’s im Prinzip schon. BIND wird auf diese Weise nach »/usr/local« installiert. Ich gehe im weiteren Verlauf davon aus, dass der Leser BIND auch so konfiguriert hat.

Für einige Leute kann »--sysconfdir« beim »./configure« noch interessant sein. Damit kann man den Ort, an dem BIND seine Konfiguration ablegt, bestimmen. Ich wähle dafür meist »/etc/bind«, wobei wir für die erste Installation diesen Schalter mal weglassen.

Konfiguration

Nachdem BIND 9 frisch kompiliert ist, muss er noch eine Grundkonfiguration erhalten. Zuerst erstellen wir das Verzeichnis »/etc/bind« und darin die Datei »named.root« [3]. In dieser Datei sind die so genannten Root-Nameserver eingetragen. Das sind jene Nameserver, die offiziell die Aufgabe haben, die Delegationen der Second-Level-Domains herauszugeben.

Das heißt: Der soeben installierte BIND, kann diese Root-Server kontaktieren und anfragen, welche Nameserver zum Beispiel für die Toplevel-Domain ».de« zuständig sind, die wiederum wissen, welcher Nameserver beispielsweise für die Second-Level-Domain »linux.de« zuständig ist. Die Minimalversion der »named.conf« zeigt Listing 1.

Von jetzt an ist unser BIND im Prinzip schon lauffähig. Wir müssen ihn lediglich noch in den Startup-Prozess des Betriebssystems einbinden. In der Regel geht das, indem man ein Skript in »/etc/init.d« erstellt und BIND dann entsprechend in den Runlevel-Verzeichnissen (»/etc/rcX.d«, je nach Distribution) verlinkt. Ein Beispielskript findet sich in Listing 2.

Um das Skript in den Bootprozess einzubinden, muss man es in die entsprechenden Runlevel-Verzeichnis verlinken. Ich habe folgende Symlinks gesetzt:

```
/etc/rc0.d/K19bind
/etc/rc1.d/K19bind
/etc/rc2.d/S19bind
/etc/rc3.d/S19bind
/etc/rc4.d/S19bind
/etc/rc5.d/S19bind
/etc/rc6.d/K19bind
```

Diese zeigen jeweils auf das Skript in »/etc/init.d«

Erster Start und Test

Nun können wir den BIND mit »/etc/init.d/bind start« starten. Im Syslog lassen sich die einzelnen Startmeldungen verfolgen (Listing 3): Welche Version von BIND gestartet wurde, wie viele CPUs zur Verfügung stehen (BIND ist ab 9.* multithreaded), welche Konfiguration er gelesen hat und an welche Interfaces er sich bindet.

Zum Test, ob der BIND auf Anfragen richtig antwortet, ist das Tool »nslookup« sehr nützlich. Dazu startet man es auf der Shell und tippt als erstes Kommando »server 127.0.0.1« ein, das den lokalen Nameserver für alle folgenden Anfragen auswählt. Nach der Eingabe von »www.linux.de« sollte die IP-Adresse zurückkommen.

Eigene Zonen

Damit ist ein Nameserver entstanden, der zwar alle Anfragen nach fremden Rechnernamen beantwortet, aber wir wollen auch unsere eigene Domain darauf laufen lassen. Dazu müssen wir nun in der »named.conf« entsprechende Zonen definieren (Beispiel in Listing 4). BIND unterscheidet zwei Arten von Zonen: Primäre und sekundäre. Eine primäre Zone ist das Original einer Zone. Alle Änderungen werden in ihr gemacht. Die sekundäre Zone dagegen ist eine Kopie davon. Die Idee dahinter ist, dass man für jede Domain mehr als einen Nameserver angibt, um eine möglichst große Ausfallsicherheit zu haben.

Das erste Beispiel ist für eine primäre Zone, das zweite für eine sekundäre Zone. Das Schlüsselwort »zone« leitet eine Zonendefinition ein. Die Syntax ist sehr stark an C angelehnt. Jede Anweisung muss wie dort üblich mit einem Semikolon abgeschlossen werden. Die geschweiften Klammern zeigen einen Block an, in dem wiederum Schlüsselworte stehen können.

Anzuzeigen, ob die Zone eine primäre oder sekundäre ist, ist Aufgabe des Schlüsselworts »type«. Es kann die Parameter »master« oder »slave« haben. Ein Spezialfall ist »hint«, das wir bei der Root-Zone (».«) kennengelernt haben. »hint« zeigt an, dass sich in dieser Zone Verweise auf die Root-Nameserver verbergen. Falls es sich dabei um eine Slave-Zone handelt, muss man zusätzlich einen »masters«-Block angeben, damit BIND weiß, von welchem DNS-Server er die Zonen-Informationen (zone transfers) holen soll.

Jede Zone wird auch in einer Datei abgelegt. Dazu dient das Schlüsselwort »file«. Man kann sowohl einen absoluten als auch einen relativen Pfad angeben. Relative Pfade werden dann vom »directory«, das im »options«-Block gesetzt wurde, aus gesehen.

Ich mache es in der Regel so, dass ich Master-Zonen in einem Unterverzeichnis namens »primary/« und Slave-Zonen in einem Unterverzeichnis namens »secondary/« ablege.

Nachdem eine Zone auf diese Weise eingebunden ist, müssen wir bei den pri-

Listing 2: Init-Skript für BIND

```
01 #!/bin/sh
02
03 # /etc/init.d/bind
04
05 PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin
06
07 test -x /usr/local/sbin/named || exit 0
08
09 case "$1" in
10     start)
11         echo -n "Starting domain name service: named"
12         /usr/local/sbin/named
13         echo "."
14     ;;
15
16     stop)
17         echo -n "Stopping domain name service: named"
18         kill -TERM `cat /var/run/named.pid"
19         echo "."
20     ;;
21
22     restart)
23         $0 stop
24         sleep 5
25         $0 start
26     ;;
27
28     reload)
29         kill -HUP `cat /var/run/named.pid"
30     ;;
31
32     *)
33         echo "Usage: $0 {start|stop|reload|restart}" >&2
34         exit 1
35     ;;
36 esac
37
38 exit 0
```

mären Zonen noch ein Zonen-File erstellen. Bei Slave-Zonen wird das Zonen-File anhand der Informationen vom Master-Server erstellt; mehr dazu im folgenden Abschnitt.
Um die Änderungen gültig zu machen, müssen wir den BIND zum Einlesen der Konfigurationsdatei auffordern, und zwar mit »killall -HUP named«.

Tabelle 1: Resource-Record-Typen

Record	Bedeutung
A	IP-Adresse für eine Ressource
CNAME	Auf welchen DNS-Namen die Ressource zeigt (Alias)
MX	Mailempfänger für die Ressource
NS	Eine Subdomain an einen anderen DNS delegieren
PTR	Reverse-Lookup-Record

Format der Zonen-Datei

Das Format eines Zonen-Files ist streng vorgegeben. Zunächst hat man einen SOA-Record (Start of Authority) . In dem steht zum einen, wer der Master-Nameserver für die Zone ist, wer für die Zone verantwortlich ist (der so genannte Hostmaster), zudem sind die Details zum Timing der Zone festgehalten. Darauf folgen die verschiedenen Resource-Records (RR) (Tabelle 1).
Eine längere Liste der möglichen Resource-Records findet sich unter [4]. Zur Erklärung des Aufbaus einer Zonen-Datei dient Listing 5.
In der zweiten Zeile beginnt der SOA-Record mit dem Master-Server (»ns.linux-magazin.de«). Darauf folgt die Mailadresse des Domain-Verantwortlichen. In diesem Fall wäre das (»hostmaster@linux-magazin.de«; das Et-Zeichen @ wird durch einen Punkt ersetzt).
Das zweite Argument ist gleich ein ganzer Block mit einigen Angaben zum Timing der Zone. Die Serial-Nummer der Zone ist bei jeder Änderung an der Zone zu erhöhen, damit die Secondary-Server die Änderung(en) mitbekommen. RFC 1982 [6] bietet einige Anregungen und Vorschriften, wie man die Serial-Nummer berechnen kann. Es hat sich allgemein eingebürgert, die Serial-Nummer als YYYYMMTTXX zu schreiben, wobei XX eine fortlaufende Nummer ist.
Gegenüber früheren Versionen verlangt BIND 9 zusätzlich, mit dem Schlüsselwort »$TTL« eine Lebenszeit (Time-To-Live) zu spezifizieren.
Da es den Artikel sprengen würde, hier genauer auf das Format eines Zonen-Files einzugehen, mag der Verweis auf Kapitel 5 der DNS-HOWTO [8] genügen.

Daemon-Steuerung

Von BIND 8 ist das Tool »ndc« zur Steuerung des Nameserver-Daemon bekannt. BIND 9 ersetzt es durch »rndc«, mit dem man einen BIND-Daemon nun auch über das Netz (oder besser gesagt über TCP) steuern kann. Dieses Tool ist viel bequemer als das Schicken von Signalen mit »kill« und funktioniert auch über das Netzwerk. Der DNS-Server stellt mit Hilfe digitaler Signaturen sicher, dass das »rndc«-Kommando wirklich von einem vertrauten Host kommt. Dazu benötigt BIND allerdings eine weitere Konfigurationsdatei »/etc/rndc.conf«, in der der Administrator seinen Signatur-Schlüssel hinterlegt. Ein Beispiel ist in Listing 6 zu sehen.
Mit dem Schlüsselwort »key« wird ein Schlüssel eingeleitet. Innerhalb des Blocks folgt nun die Angabe das Algorithmus sowie des Schlüssels selbst. »hmac-md5« ist der einzige bisher unterstütze Algorithmus. Im Prinzip ist es nur ein Base64-kodierter String, weshalb wir Perl zu seiner Generierung benutzen können:

```
perl -MMIME::Base64 -e \
 'print encode_base64("Geheimer String");'
```

Am besten wählt man einen etwas längeren String. Auch in der »named.conf« muss entsprechender Eintrag stehen (siehe Listing 8).
Mit »inet 127.0.0.1« lassen wir einen Control-Channel vom BIND auf Loopback-Adresse 127.0.0.1 und Port 953 laufen. Alle Verbindung von Localhost mit dem entsprechenden Key werden akzeptiert. In Tabelle 2 finden wir alle Argumente zu »rndc«.

BIND 9 im Sandkasten

BIND war schon immer etwas heikel, was das Thema Security angeht. In der Vergangenheit gab es viele Remote-Exploits. Obwohl BIND 9 sicherer sein soll als seine Vorgänger, lohnt es sich, dem DNS-Server eine »chroot«-Umgebung aufzubauen.
Das ist gar nicht so schwer. Zuerst müssen wir einen User erstellen, unter dem BIND laufen wird. Es wäre zwar mög-

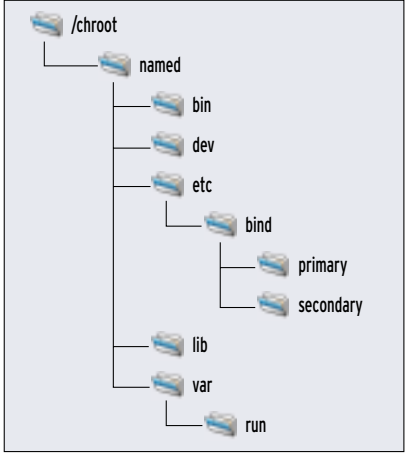
Listing 3: Syslog-Startmeldungen des BIND

```
01 Aug 30 20:54:15 frodo named[20267]: starting BIND 9.1.3
02 Aug 30 20:54:15 frodo named[20267]: using 1 CPU
03 Aug 30 20:54:15 frodo named[20269]: loading configuration from '/etc/named.conf'
04 Aug 30 20:54:15 frodo named[20269]: the default for the 'auth-nxdomain' option is now 'no'
05 Aug 30 20:54:15 frodo named[20269]: no IPv6 interfaces found
06 Aug 30 20:54:15 frodo named[20269]: listening on IPv4 interface lo, 127.0.0.1#53
07 Aug 30 20:54:15 frodo named[20269]: listening on IPv4 interface eth0, 192.168.1.1#53
08 Aug 30 20:54:15 frodo named[20269]: running
```

Listing 4: Beispiel einer Zonendefinition

```
01 Für eine primäre Zone:
02
03         zone "linux-magazin.de" {
04                 type master;
05                 file "primary/linux-
   magazin.de.zone";
06             };
07
08 Für eine sekundäre Zone:
09
10         zone "linux-magazin.de" {
11                 type slave;
12                 file "secondary/linux-
   magazin.de.sec";
13                 masters {
14                         a.b.c.d;
15                         };
16             };
```

Struktur im Jail



lich, einen existierenden User wie Nobody zu benutzen. Das ist aber nicht empfehlenswert, da ein Einbrecher so alle anderen Prozesse, die als »nobody« laufen, steuern könnte. Aus diesem Grund müssen einige Dateien in »/etc« geändert werden, um den Benutzer hinzuzufügen. Zudem ist sicherzustellen, dass die UIDs und GIDs noch nicht vergeben sind.

Dazu hängen wir an »etc/passwd«

```
named:x:200:200:Nameserver:/chroot/named:/ ➥
bin/false
```

an und an »/etc/shadow«:

```
named:NP:::::::
```

Und an »/etc/group« ist Folgendes anzuhängen:

```
named:x:200:
```

Wir haben an den Benutzer »named« nun »/chroot/named« als Homedirectory vergeben. Wir werden auch noch den »chroot«-Sandkasten darin aufbauen. Natürlich kann man diesen Pfad auch noch ändern. Für ganz paranoide Installationen kann man zusätzlich ein separates Volume für den Jail nehmen, um Denial-of-Service-Angriffe wie das Vollmüllen des Dateisystems im Zaum zu halten.

In »/chroot/named« müssen wir eine Struktur aufbauen, wie sie im Kasten „Struktur im Jail“ ersichtlich ist. Nun können wir die Umgebung füllen. Da wir ja bereits eine funktionierende Installation haben, können wir die vorhanden Files verschieben oder – um sicherzugehen – kopieren):

```
cp -p /etc/named.conf /chroot/named/etc/
cp -a /etc/bind /chroot/named/etc/
```

Der Daemon muss einige Files auch schreiben können. Deshalb setzen wir einige Rechte passend:

```
chown -R named:named ➥
  /chroot/named/etc/bind/ ➥
  /chroot/named/var/run/
```

Die erste Rechtevergabe ist erforderlich, damit BIND sein Dumpfile und die Slave-Zonen updaten kann, und die Letztere brauchen wir, damit »named« sein PID-File und den Socket für »ndc« anlegen kann.

Nachdem die Konfiguration kopiert ist, ist noch eine Reihe von Systemdateien wie Bibliotheken oder Devices-Files nötig (siehe Listing 7).

Beim Aufruf von »ldconfig« sollten die beiden Libraries gefunden werden. Eine Konfigurationsdatei brauchen wir in diesem Fall nicht, da »ldconfig« von sich aus in »/lib« schaut. Unerlässlich ist aber »/dev/null«:

```
mknod /chroot/named/dev/null c 1 3
```

Tabelle 2: Argumente für »rndc«

Argument	Zweck
stop	Alle anstehenden Aufgaben beenden und den Daemon stoppen
reload	Konfigurationsdatei und Zonen neu laden
reload <zone>	<zone> neu laden
refresh <zone>	<zone> sofort neu laden
stats	Server-Statistiken in ein File schreiben
querylog	Logging von Querys an-/abstellen
dumpdb	Cache in das Dump-File (»named_dump.db«) schreiben
halt	Server sofort stoppen, ohne anstehende Aufgaben zu beenden
* status	»ps(1)«-Status des Daemon anzeigen
* trace	Debugging-Level um 1 erhöhen
* notrace	Debugging-Level auf 0 setzen
* restart	Server neu starten
(Die mit * markierten Optionen sind in 9.1.3 noch nicht implementiert)	

Listing 5: Ein Zonen-File

```
01 $TTL 86400
02 @    IN SOA ns.linux-magazin.de.
   hostmaster.linux-magazin.de. (
03                 200108300 ; Serial number
04                 10800     ; Refresh
05                 3600      ; Retry
06                 604800    ; Expire after one week
07                 86400 )   ; Minimum TTL of one day
08      IN NS    ns
09      IN NS    ns.provider.de.
10
11 ; Add MX records
12 @  IN   MX     5   mail
13
14 ; Define the hosts
15
16 mail            IN      A       192.168.0.1
17 www             IN      A       192.168.0.2
18 ns              IN      A       192.168.0.1
```

Listing 6: »rndc.conf«

```
01 key rndc_key {
02    algorithm "hmac-md5";
03    secret "c3Ryb25nIGVub3VnaCBmb3IgYSBtYW4gYnV0IG1hZGU ➥
   gZm9yIGEgd29tYW4K";
04 };
05 options {
06    default-server localhost;
07    default-key    rndc_key;
08 };
```

Einige Konfigurationsdateien dürfen bei diesem Vorgehen natürlich auch nicht fehlen:

```
cp /etc/localtime /etc/nsswitch.conf \
   /etc/resolv.conf /chroot/named/etc
```

In den Dateien »/chroot/named/etc/passwd«, »/chroot/named/etc/shadow« und in der »/chroot/named/etc/group« müssen wir noch die Informationen für den Benutzer und die Gruppe »named« hinterlegen.

Recht heikel bei »chroot«-Umgebungen ist das Logging. Die einfachste Methode ist es, den »syslogd« mit der Option

```
-a /chroot/named/dev/log
```

zu starten. Falls das aus irgendeinem Grund aber nicht gehen sollte (beispielsweise weil man einen zu alten »syslogd« hat), lässt sich auch »holelogd« einsetzen, der eine Art Proxy für den »syslogd« darstellt.

Zuletzt müssen wir noch die BIND-Binaries hineinkopieren. Wenn wir den BIND wie am Anfang des Artikels kompiliert haben, dann reicht ein:

```
cp /usr/local/sbin/named /chroot/named/bin
chmod 000 /usr/local/sbin/named
```

Nun müssen wir nur noch das Init-Skript (das aus Listing 2) anpassen, damit »named« mit den Optionen

```
-u named -t /chroot/named
```

startet. Wir sollten auch nicht vergessen, die PID aus »/chroot/named« zu lesen. Danach können wir den »named« aus dem Init-Skript heraus starten.

Listing 7: Bau des Jails

```
01 cd /chroot/named/lib
02 cp -p /lib/libc-2.*.so .
03 ln -s libc-2.*.so libc.so.6
04 cp -p /lib/ld-2.*.so .
05 ln -s ld-2.*.so ld-linux.so.2
06 cp /usr/lib/libcrypto.so.* .
07 cp /lib/libnsl-2.*.so .
08 ln -s libnsl-2.*.so libnsl.so.1
09 cp /lib/libpthread-*.so .
10 ln -s libpthread-*.so libpthread.so.0
11 cp /lib/libdl-2.*.so .
12 ln -s libdl-2.*.so libdl.so.2
13
14 cp /sbin/ldconfig /chroot/named/bin
15 chroot /chroot/named /bin/ldconfig -v
```

Listing 8: »named.conf« für »rndc«

```
01 key rndc_key {
02    algorithm "hmac-md5";
03    secret
   "c3Ryb25nIGVub3VnaCBmb3IgYSBtYW4gYnV0IG1hZGUgZm9y➥
   IGEgd29tYW4K";
04 };
05
06 controls {
07    inet 127.0.0.1 allow { localhost; } keys
   { rndc_key; };
08 };
```

Neuigkeiten vom Master

Die Notify-Funktion (Benachrichtigen), beschrieben in RFC 1996 [9], nutzt der Master-Server, um jedem Slave eine Benachrichtigung zu schicken, wenn an einer Zone etwas verändert wurde. So verbreiten sich DNS-Änderungen schneller auf alle Slave-Server.

Grundsätzlich sendet BIND allen Secondary-Nameservern, die im Zonenfile deklariert sind, ein Notify. Falls eine Zone auch Stealth-Server hat, kann man mit der Option »also-notify« auch unbekannten DNS ein Notify schicken. Man spezifiziert diese Option in der Definition der Zone oder in den globalen Optionen (Listing 9).

Listing 9: »also-notify«

Bei den globalen Optionen:

```
01 options {
02         // andere Optionen
03         also-notify {
04                 a.b.c.d;
05         };
06         // andere Optionen
07 };
```

Bei einer Zone:

```
08 zone "linux-magazin.de" {
09         // andere Optionen
10         also-notify {
11                 a.b.c.d;
12         };
13         // andere Optionen
14 };
```

Zugriffsteuerung mit ACLs

Mit Hilfe von ACLs (Access Control Lists) kann man den Zugriff auf den Nameserver einschränken. Man unterscheidet zwischen folgenden Berechtigungen: Notify, Query (Abfragen), Transfer (komplette Zone transferieren) und Update (dynamische Änderungen an Zonen vornehmen).

Bevor man eine ACL nutzen kann, ist eine Address-Match-List unter einem symbolischen Namen zu erstellen. Dazu gibt es das »acl«-Schlüsselwort. Im Block kann man entweder IP-Adressen oder Subnetze angeben. Zusätzlich gibt es noch die in Tabelle 3 aufgeführten Definitionen. Hier zwei Beispiele dafür:

```
acl name1 {
  192.168.1.0/24; 192.168.9.10;
};
acl name2 { localnets; };
```

Die auf diese Weise definierten symbolischen Namen »name1« und »name2« können wir für die ACLs »allow-notify«, »allow-transfer« und »allow-update« benutzen (siehe Tabelle 4). Die ACLs kann man entweder in den »options« oder für

Tabelle 3: Spezialfälle bei Adresslisten

Definition	Bedeutung
any	Matcht für alle Hosts
none	Matcht für keinen Host
localhost	Matcht für alle lokalen Interfaces
localnets	Matcht für alle Subnetze, in denen das System ein Interface hat

Tabelle 4: ACLs und ihre Bedeutungen

Schlüsselwort	Bedeutung
allow-notify	Von welchen Hosts ein Notify angenommen wird
allow-query	Welche Hosts ein Abfrage von unserem DNS-Server machen dürfen
allow-transfer	Welche Hosts einen kompletten Zonen-Transfer machen dürfen
allow-update	Welche Hosts dynamische Updates vornehmen dürfen

jede Zone einzeln festlegen (oder an beiden Orten, wobei die Zonen-ACLs jeweils die globalen ACLs überschreiben). Ein Beispiel für ACLs ist im nächsten Abschnitt zu finden, in dem es um Split DNS geht.

Split DNS - die alte Art

Das Aufsetzen mehrerer Ansichten (Views) einer Zone gegenüber internen oder externen Clients wird als „Split DNS Setup“ bezeichnet. Es gibt viele Einsatzzwecke, für die das nützlich ist. Der häufigste ist wahrscheinlich, wenn man Hosts, die in einem maskierten Firmen-LAN stehen, nicht von der ganzen Öffentlichkeit auflösen lassen will.

Nehmen wir als Beispiel die rein fiktive Firma Test AG (Test.ch), die ein Firmen-LAN mit reservierten (RFC 1918) IP-Adressen sowie eine DMZ (Demilitarized Zone) eingerichtet hat. Die Firma will nun eine interne Ansicht der Zone, die nur vom LAN aus sichtbar ist, und eine Ansicht der Zone, die für das Internet bestimmt ist. Das heißt also, dass die Firma zwei DNS-Server aufsetzen muss. Der eine steht im Firmen-LAN, der andere in der DMZ (kann aber das Firmen-LAN trotzdem erreichen).

Der interne DNS-Server muss nun so konfiguriert werden, dass er alle Anfragen – außer für Test.ch – an den DNS-Server in der DMZ weiterleitet. Für die eigene Zone hält der interne DNS-Server alle Informationen bereit. Um ganz sicher zu sein, dass die Informationen von Test.ch von außen nicht abgefragt werden können, müssen wir noch eine ACL setzen. Auf dem internen DNS-Server sieht das Setup also wie in Listing 10 gezeigt aus. An demselben Listing kann man auch gleich erkennen, wie ACLs und Forwarding-Only-Nameserver funktionieren.

Der Server in der DMZ wird also so konfiguriert, dass auch er Master für Test.ch ist. Allerdings kann diese Version der Zone von jedem Host abgefragt werden. Ein Beispiel dazu ist in Listing 11 nachzulesen.

Bei dem NIC (Network Information Center) für die Top-Level-Domain .ch müssen nun der Server aus der DMZ sowie zusätzlich beliebig viele Slave-Server (die einen Zonen-Transfer vom Server in der DMZ ausführen) als autoritativ angegeben werden. Wie man Zonentransfers für die Slaves erlauben kann, haben wir einen Abschnitt vorher gesehen.

Split DNS - die neue Art

Wie im vorherigen Abschnitt beschrieben kann man ein Split DNS Setup natürlich auch realisieren. Mit BIND 9 gibt es allerdings eine weitere, elegante Methode, ein Split-DNS-Setup auszuführen, die zudem mit einem Server auskommt. Dazu gibt es das »view«-Schlüsselwort. In dessen Block kann man die Zonen-Definition platzieren.

Das Ganze passiert ebenfalls wieder in der »named.conf« (Listing 12). »match-clients« kann dabei gleich wie ein »acl«-Statement behandelt werden. Natürlich kann man im »zone«-Block innerhalb einer View dieselben Optionen wie außerhalb einer View benutzen.

Sichere Signaturen: TSIG

Transaction Signatures (TSIG) sichern die Server-zu-Server-Kommunikation. Das umfasst Zonen-Transfers, Notifies sowie rekursive Queries. Neuere Versionen von BIND 8 enthalten zwar auch bereits eine limitierte Anzahl der TSIG-Features. Am häufigsten wird man TSIG jedoch für dynamische Updates von Zonen benutzen.

Zuerst muss man einen Schlüssel für die Hosts erzeugen. Beide Hosts, die an der Kommunikation beteiligt sind, benutzen denselben Key. Dieser Key muss auch gleich heißen. In diesem Beispiel gehen wir davon aus, dass »frodo« mit »zeus« (oder umgekehrt) kommunizieren will. Der Einfachheit halber sollte man in diesem Fall den Key »frodo-zeus« nennen.

```
dnssec-keygen -a hmac-md5 -b 128 ➥
  -n HOST frodo-zeus.
```

Das erzeugt einen 128-Bit-HMAC-MD5-Schlüssel. Längere Keys sind generell besser, kürzere Schlüssel sind jedoch besser zu lesen. Außerdem sollte man beachten, dass man den Key nicht länger als 512 Bit machen kann – längere Keys werden mit MD5 digested (zusammengefasst), um einen 128-Bit-Key zu erhalten.

Der Schlüssel ist nun im File »Kfrodo-zeus.+157+32808.private« (der Dateiname kann variieren). Nichts benutzt diese Datei direkt, aber man kann den Base64-encodierten String (der in der Datei hinter »Key:« steht) extrahieren und als so genanntes Shared Secret benutzen. Ein Key sieht etwa so aus:

Listing 10: Setup vom internen Server

```
01 // Die Range hier anpassen
02 acl internals { 192.168.1.0/24; };
03 acl externals { range.der.dmz; };
04
05 options {
06        // alle anderen Optionen
07        forward only;
08        forwarders {
09                dns.aus.der.dmz;
10        };
11        allow-transfer { none; };
12        allow-query { internals; externals; };
13         allow-recursion { internals; };
14 };
15
16 zone "test.ch" {
17        type master;
18        file "primary/test.ch.zone";
19        forwarders { };
20
21        allow-query { internals; };
22        allow-transfer { internals; };
23
24 };
```

Listing 11: Setup vom DMZ-Server

```
01 // Den Range hier anpassen
02
03 acl internals { 192.168.1.0/24; };
04 acl externals { range.der.dmz; };
05
06 options {
07    // alle anderen Optionen
08    allow-transfer { none; };
09    allow-query { internals; externals; };
10    allow-recursion { internals; externals; };
11 };
12
13 zone "test.ch" {
14    type master;
15    file "primary/test.ch.zone";
16    allow-query { any; };
17    allow-transfer { internals; externals; };
18 };
```

```
nXtCC4k7bkmflqOR7rJxpA==
```

Alternativ zu »dnssec-keygen« kann man auch den im Kapitel zu »rndc« erwähnten Perl-Aufruf benutzen, das Resultat bleibt gleich.

Nun ist es wichtig, den Key über eine sichere Methode (»sftp«, »scp«, Telefon) zu beiden DNS-Servern zu bringen. Wenn das geschehen ist, kann man die »named.conf« entsprechend anpassen (Listing 13).

Zuerst machen wir mit dem »key«-Statement den Schlüssel bekannt. Anschließend können wir mit der »server«-Direktive festlegen, welcher Key wann zu benutzen ist. So kann man auch mit verschiedenen Keys arbeiten, beispielsweise mit einem Schlüssel pro Host. Man muss dabei selbstverständlich auf beiden Hosts jeweils die IP-Adresse des jeweils anderen Hosts nehmen.

Wenn »frodo« zu »zeus« verbindet (oder umgekehrt), werden alle Anfragen automatisch mit dem Schlüssel signiert (ein Resultat des »server«-Statements).

Nun kann man in den ACL-Definitionen neben IP-Adressen auch Schlüssel angeben:

```
allow-update { key frodo-zeus. ;};
```

Damit wären alle dynamischen Updates, die mit dem Key »frodo-zeus.« signiert sind, automatisch als gültig angenommen. Das ist viel sicherer und vor allem auch einfacher, als einfach IP-Adressen zu definieren. Auf die gleiche Weise kann man natürlich auch Zonen-Transfer oder ein Notify auf gültige Signaturen beschränken.

Listing 12: Split DNS mit der neuen Methode

```
01 view "internal" {
02         // Unser internes Netz
03         match-clients { 192.168.1.0/24; };
04         // Rekursion für interne Hosts
05         recursion yes;
06         // Interne Ansicht der Zone
07         zone "test.ch" {
08                 type master;
09                 file "primary/internal-test.ch.zone";
10         };
11 };
12
13 view "external" {
14         // Alle anderen Hosts
15         match-clients { any; };
16         // Keine Rekursion für externe Hosts
17         recursion no;
18         // Externe Ansicht der Zone
19         zone "test.ch" {
20                 type master;
21                 file "primary/external-test.ch.zone";
22         };
23 };
```

Listing 13: TSIG: »named.conf«

```
01 // Schlüssel bekannt machen
02 key frodo-zeus. {
03         algorithm hmac-md5;
04         secret "nXtCC4k7bkmflqOR7rJxpA=";
05 };
06
07 // IP vom anderen Server
08 server 192.168.1.1 {
09         keys { frodo-zeus. ;};
10 }
```

Dynamische Updates

Da im vorherigen Kapitel TSIG installiert wurde, können wir eines der häufigsten Features davon jetzt einsetzen: Man kann Zonen auf einem sicheren Weg dynamisch updaten.

Den besten Einblick gewährt die Manpage zu »nsupdate(8)«. Den Key übergibt man »nsupdate« mit den Optionen »-y« oder »-k«. Mit »-k« kann man die Key-Datei (sie hieß im vorherigen Beispiel »Kfrodo-zeus.+157+32808.private«) angeben. Die Datei mit der Endung ».key« (die auch erzeugt wurde) wird ebenfalls benötigt.

Da es recht umständlich ist, die beiden Dateien aufzubewahren, ist »*-y*« die bessere Alternative:

```
nsupdate -y frodo-zeus.\
         :nXtCC4k7bkmflqOR7rJxpA=
```

Und schon kann man (wenn TSIG richtig konfiguriert wurde) Updates an Zonen vornehmen.

Bei dynamischen Updates muss man allerdings in Kauf nehmen, dass die Informationen im Zonenfile nicht mehr aktuell sein können. BIND macht nur alle 15 Minuten einen Dump der Informationen. Das ist viel effektiver, da sonst viele Updates an einer Zone zu lange dauern würden. Zur Sicherheit schreibt BIND aber alle Änderungen in ein Journal, um sie nach einem Crash wieder nachziehen zu können.

Wer allerdings ganz sicher sein will, dass das Zonenfile auf der Festplatte aktuell ist, kommt um ein »rndc stop« nicht herum.

DNSSEC

Die kryptografische Authentifizierung von DNS-Informationen ist dank DNSSEC möglich. DNSSEC ist eine Erweiterung, die in RFC 2535 [10] genauer beschrieben ist. Bevor wir weitermachen, noch eine kurze Anmerkung: Die Tools, die im Folgenden Anwendung finden, erwarten, dass die Schlüssel im aktuellen Verzeichnis liegen.

Mit »dnssec-keygen« erstellen wir Keys. Eine Secure-Zone muss einen oder mehrere Schlüssel enthalten, die verwendet werden, um die Records in der Zone sowie die Zone-Keys der delegierten Zonen zu signieren. Dieser Zonenschlüssel muss denselben Namen wie die Zone sowie den Namenstyp »ZONE« haben. Ferner muss er für Authentifizierung geeignet sein.

Folgender Befehl wird einen 768-Bit-DSA-Key für »test.ch« erzeugen:

```
dnssec-keygen -a DSA -b 768 \
    -n ZONE test.ch.
```

Dabei werden zwei Dateien erzeugt. Die Datei mit der Endung ».key« enthält den Namen des Schlüssels, den Algorithmus sowie das Key-Tag. Die Datei mit der Endung ».private« wird dazu benutzt, um die Signaturen zu erzeugen – sie enthält den privaten Schlüssel. Mit dem öffentlichen Schlüssel in der ».key«-Datei kann man anschließend die Signaturen noch verifizieren.

Der öffentliche Schlüssel sollte in das Zonen-File integriert werden. Am besten lässt sich das mit dem »$INCLUDE«-Statement erledigen, das die ganze ».key«-Datei einliest.

Nun müssen wir ein Key-Set erstellen. Das Key-Set ist für den Administrator der übergeordneten Zone (in der so genannten Parent-Zone) bestimmt, der das Key-Set signieren kann. Das folgende Kommando generiert ein solches Key-Set. Die TTL (Time To Live) wird dabei 3600 (Sekunden) und die Signatur wird zehn Tage gültig sein.

```
dnssec-makekeyset -t 3600 -e +864000
    Ktest.ch.+003+12345
```

Nachdem wir dieses Kommando gestartet haben, erhalten wir die Datei »keyset-test.ch.«. Diese Datei muss nun an den Administrator der Parent-Domain gesendet werden. Der signiert das Key-Set dann folgendermaßen:

```
dnssec-signkey keyset-test.ch.
Kchild.example.+003+12345
```

Die jeweils kursiv gesetzten Dateinamen sind durch den Namen und den Tag des Keys für die Zone (in diesem Fall sprechen wir von der Parent-Zone) zu ersetzen. Wir bekommen eine Ausgabedatei: »signedkey-test.ch.«. Diese Datei sollte an den Administrator der Subdomain geschickt und dort aufbewahrt werden.
Nun können wir eine Zone signieren. Dazu müssen alle »signedkey«-Dateien, die zu sicheren Subzonen gehören, präsent sein. Das folgende Kommando signiert die Zone »test.ch«, die im Zonen-File »test.ch.zone« liegt:

```
dnssec-signzone -o test.ch test.ch.zone
```

Wir erhalten dabei »test.ch.zone.signed«. Diese Datei muss noch in der »named.conf« als Datei für die Zone referenziert werden.
Damit BIND Schlüssel als echt annimmt, gibt es das »trusted-keys«-Statement. Im Prinzip kann man einfach den Inhalt des ».key«-Files (das mit »dnssec-keygen« erzeugt wurde) hineinkopieren und umformatieren:

```
trusted-keys {
  test.ch 256 3 3 "langer Base64-String";
};
```

Ganz wichtig ist es, denn Base64-kodierten String in doppelte Anführungszeichen einzuschließen, da BIND sonst aufgrund der Leerzeichen Schwierigkeiten beim Parsen hat.

Diverse nützliche Sachen

Während des Betriebs oder bei der Wartung eines DNS-Servers kann es leicht dazu kommen, dass man einmal aus Versehen die Serial-Nummer einer Zone falsch berechnet. Wenn sich außerdem auch noch die Slave-Server bereits upgedatet haben, lässt sich die Nummer nachträglich eigentlich nicht mehr kleiner machen.
Aber eben nur eigentlich, denn es gibt durchaus eine elegante Lösung dafür. Wenn man 2147483647 (2^{31} -1) als Serial-Nummer angibt und wartet, bis sich alle Slaves damit aktualisiert haben, kann man anschließend wieder eine beliebige Seriennummer wählen. Das ist möglich, weil BIND die Seriennummer nur als vorzeichenbehaftete 32-Bit-Zahl sieht, die mit dieser Zahl an ihrer obersten Grenze angelangt ist.
Wie jedem anderen laufenden Programm kann man dem »named« auch Signale schicken. Obwohl sich damit weniger Möglichkeiten als mit »rndc« ergeben, kann es manchmal doch nützlich sein, sie zu kennen. Die in Tabelle 5 definierten Signale werden von BIND 9 interpretiert.
Wie es bei vielen anderen Programmen so ist, ist auch BIND 9 insgesamt recht kompliziert und es gäbe noch 1000 andere Fähigkeiten und Möglichkeiten, die der Artikel hätte vorstellen können. Wer noch mehr über die Konfiguration von BIND wissen will, dem sei vor allem das BIND-9-ARM (Administrator Reference Manual) [12] empfohlen. Darin ist so ziemlich jede Option zu finden, die man einstellen kann.
Für eher allgemeine Informationen über Domain Name Service ist dagegen das DNS Resource Directory [11] eine gute Hilfe. (*hmi*)

Tabelle 5: Signale für »named«

Signal	Zweck
SIGHUP	»named.conf« und Zonenfiles neu einlesen
SIGTERM	aufräumen und beenden
SIGINT	wie SIGTERM

Infos

[1] BIND-Homepage: [http://www.isc.org/products/BIND/]
[2] Homepage des ISC: [http://www.isc.org/]
[3] DNS-Rootdatei: [ftp://ftp.rs.internic.net/domain/named.root]
[4] Liste der Resource Records: [http://www.dns.net/dnsrd/rr.html]
[5] Format of DNS files, Paul Vixie: [http://www.dns.net/dnsrd/docs/style.txt]
[6] RFC 1982, Serial Number Arithmetic: [ftp://ftp.is.co.za/rfc/rfc1982.txt]
[7] RFC 1035, Domain Names – Implementation and Specification: [ftp://ftp.is.co.za/rfc/rfc1035.txt]
[8] DNS-HOWTO: A simple domain: [http://www.linuxdoc.org/HOWTO/DNS-HOWTO-5.html]
[9] RFC 1996, Notify: A mechanism for prompt notification of authority zone changes: [ftp://ftp.is.co.za/rfc/rfc1996.txt]
[10] RFC 2535, Domain Name System Security Extension: [ftp://ftp.is.co.za/rfc/rfc2535.txt]
[11] DNS Resource Directory: [http://www.dns.net/dnsrd/]
[12] BIND 9 Administrator Reference Manual: [http://www.crt.se/dnssec/bind9/]
[13] RFC 830: [http://asg.web.cmu.edu/rfc/rfc830.html]

Der Autor

Thomas Bader beschäftigt sich schon ziemlich lange mit Unix und den diversen Derivaten. In seiner Freizeit ist er Sysadmin und Beirat von Trash.net und hilft beim deutschsprachigen OSS-Newsportal Symlink mit.

Tabelle 6: Glossar

TTL	Time To Live. Gibt an, ab wann Informationen im Cache ungültig werden und neu angefordert werden müssen.
Zonentransfer	So bezeichnet man den Vorgang, bei dem alle Informationen einer Zone heruntergeladen werden.
HMAC-MD5	Ein Hash-Algorithmus (Einwegfunktion).
Query	Die Abfrage einer Information.
Base64	Ein Kodierungs-Algorithmus.
chroot	Prozesse mit einem anderen Wurzelverzeichnis (»root«, »/«) laufen lassen.
BIND	Berkeley Internet Name Domain.
Daemon	Programm, das im Hintergrund läuft und auf bestimmte Ereignisse wartet und reagiert.
Jail	Verzeichnisbaum, in dem ein Prozess durch den Unix-Systemaufruf »chroot(2)« eingesperrt wird, auch Chroot-Jail.