Kapitel 8

DNS

8.1 DNS-översikt

DNS betyder Domain Name System och är en global, distribuerad och hierarkisk databas för, bland annat, översättning från domännamn till IP-nummer.

Det finns nästan 4 miljarder, 2^{32}, möjliga IP-adresser. I början av 1980-talet när Internet bestod av några tusen datorer hade alla unika namn som listades i en fil vid namn HOSTS.TXT. Detta system blev ohanterligt, så Paul Mockapetris med flera uppfann 1983 DNS. Första förslaget finns beskrivet i RFC 882 och 883. DNS är hierarkiskt, som ett träd med en gemensam rot och förgreningar där varje organisation sköter sina respektive delar. Människor har lättare att minnas namn än siffror, så via DNS kan datorer veta att ftp.df.lth.se har adressen 194.47.250.18 och att en av adresserna till www.google.com är 74.125.43.147.

På Internet finns bara ett DNS, men det består av tusentals namnservrar som är auktoritativa för en eller flera zoner. 13 namnservrar har hand om roten i trädet (som brukar ritas överst och ha en tom sträng som namn). Alla andra namnservrar vet adressen till rotservrarna. Varje domän separeras av en punkt och domännamnen förgrenas från höger (bakifrån). T.ex. datorn ftp.df.lth.se tillhör Sverige, Lunds tekniska högskola, dess datorförening och namnet är ftp. Att DNS är distribuerat innebär att i fallet ftp.df.lth.se så är det de 13 rotservrarna som vet vilka namnservrar som

sköter .se, dessa vet vilka som sköter lth.se och de på lth.se vet vilka namnservrar som sköter df.lth.se. Namnservern på DF vet vilket IP-nummer som ftp.df.lth.se har. Trädets olika grenar (zoner) har alltså olika administratörer. Det finns bara ett träd, och ansvar delegeras neråt.

Figur 8.1: *Förgrenad namnrymd*

Den vanligaste programvaran för namnservrar heter BIND (Berkeley Internet Name Daemon). Men det finns numera även andra alternativ. Namnservrar kan delas in i olika typer. En indelning är master och slave. En master läser in sin information från lokala filer och en slavserver får sina uppgifter från en annan namnserver. Men för datorn som frågar är det ingen skillnad på master och slave, båda är auktoritativa för de zoner de kan besvara. En annan indelning är iterativa och rekursiva. En rekursiv server kan ta reda på vilken adress som helst, men en iterativ besvarar endast frågor för de zoner den själv har. Rekursiva servrar sparar de uppgifter som de frågas om i sin cache, så nästa gång den får en fråga om samma domän behöver den inte börja med roten. Iterativa servrar kan ses som innehållsservrar och tillhör ofta den organisation som äger respektive domän. En namnserver har ofta flera olika roller, den kan vara master för vissa zoner och slave för andra, den kan vara iterativ ut mot Internet och rekursiv in mot det lokala nätverket.

DNS är ett klient/server-system. Alla datorer som ska använda Internet är DNS-klienter. Man anger en lista med 1–3 namnservrar som de ska fråga, dessa IP får man ofta från sin internetleverantör. De servrar som klienter frågar måste vara rekursiva, det vill säga de måste kunna ta reda på svaret till vilken fråga som helst. Om min dator är inställd på att fråga namnservern med IP 87.96.254.52 (tillhör Alltele som jag får min internetanslutning från) och jag vill besöka www.google.se så börjar min dator att fråga denna namnserver. Vi antar att den inte har det eller andra svar sparade. Min dator frågar alltså Alltele om namnet www.google.se. Den vet inte svaret, men vet vem man kan fråga. Den börjar med att fråga någon av de 13 rotservrarna. Den svarar att

den inte vet vilken adress www.google.se har, men vet IP till de servrar som sköter se. Allteles namnserver frågar en av dem, de vet inte heller vilken adress www.google.se har, men de vet IP till namnserver för google.se. Alltele frågar till exempel ns2.google.com, och den svarar att en av adresserna till www.google.se är 74.125.43.147. Då har Allteles namnserver fått det rätta svaret och lämnar det till min dator. Och nästa gång någon frågar kan den svara direkt så länge som svaret finns kvar i minnet. Även steg på vägen sparas, så ifall någon annan kund vill gå till t.ex. www.lu.se behöver inte den börja med att fråga rotservrarna, från min förfrågan om google.se vet Allteles namnserver minst ett IP-nummer till servrar som har hand om .se-domänen.

DNS kan ses som en stor katalog över alla datorer på nätet där informationen är utspridd och länkad från respektive föräldrazon till barnen. Detta gör att respektive företag eller organisation sköter sin lilla del av hela namnrymden, och systemet blir skalbart. Rotservrarna svarar på ca 10 000 frågor i sekunden, så ifall inget skulle cachas skulle det inte fungera. Varje zon innehåller uppgifter hur länge information ska sparas, ofta ett par dygn.

Zonfilerna är ofta vanliga textfiler, en för varje domän eller subdomän. Det finns olika slags records. Varje zon har en SOA (Start of authority). Där sparas olika time-out-värden och ett serienummer som anger för slavservrar om zonen har ändrats. Det record som anger mappning mellan namn och ip heter A. Varje dator, t.ex. en webbserver, har minst ett A-record. Ofta väljer man ett annat namn än det som används mest, t.ex. så heter www.expertmaker.com egentligen expert.expertmaker.com. Dess A-record kan vara `expert IN A 85.235.1.37`. För att ange alias kan man använda CNAME-records, man kan t.ex. ha `www IN CNAME expert` för den nyss nämnda datorn. Fördel med CNAME är att man knyter namnet för en tjänst som www eller ftp till ett annat namn och lättare kan byta vid behov. Men man kan även ha flera A-records (vilket är det som används för www.expertmaker.com). För mappning åt andra hållet (reverse, från IP till namn) används ett record som heter PTR. Detta kan ligga på en annan namnserver och sköts ofta av internetleverantören. För att ange namnservrar används NS-records.

DNS är ett system som har ca en miljard användare, många dagligen, de flesta utan att veta hur det fungerar.

Och för att ha uppfunnits på 80-talet har det klarat tidens tand väl. Gör man rätt från början behöver man sällan ändra.

8.2 Innehåll i zoner – Resursposter

De enskilda posterna i DNS kallas Resource Records (RR), på svenska kan de kallas resursposter. De vanligaste är SOA, NS, A, AAAA, PTR, CNAME och TXT. Jag beskrev kort några av dem i inledningen till detta kapitel, men de är så viktiga att de förtjänar att upprepas. Först i en tabell och sen med beskrivning.

Tabell 8.1: Resursposter

SOA	Start Of Authority	Beskriver en zon
NS	Name Server	Pekar på namnservrar
A	Adress	Från namn till IPv4
AAAA	v6-adress	Från namn till IPv6
PTR	Pointer	Från IP till namn
MX	Mail Exchange	För e-post
CNAME	Canonical Name	Alias, annat namn
TXT	Text	Diverse text

Det finns flera, men dessa är de vanligaste. Jag berättar lite om var och en.

8.2.1 A

A-poster är hjärtat av DNS. De pekar från namn till IP. Till exempel: `google.com. 300 IN A 209.85.149.99`
Syntaxen för A-poster är:
`namn [TTL] [Klass] A IP-adress`
Time to live är frivilligt att ange för varje post, enklare är att använda $TTL i början av zonfilen. Klass är nästan alltid IN=Internet så även den kan man hoppa över. Det viktigaste är namnet i början på raden, typen A och IP-numret. Exempel på andra giltiga versioner av A-post är: `www A 194.47.250.11`

`ftp A 194.47.250.18`
Där är TTL och klass överhoppad, och namnet är inte fullständigt. Zonfiler fungerar så att om man inte avslutar ett namn med en punkt så används standardzonen (den anges med $ORIGIN i början av zonfilen eller i namnserverns konfiguration). Anger man inget namn alls så används det från föregående rad. Man kan även ha flera $ORIGIN i en zonfil. Dessa två A-poster gör att URL:er som
http://www.df.lth.se/ och http://ftp.df.lth.se/ hittar till rätt datorer (vad som gör att det blir rätt svar beskrivs i andra kapitel). Alltså den mappning från namn till IP som gör att vi slipper minnas alla IP-nummer utantill.

8.2.2 CNAME

CNAME används för att ge smeknamn på datorer, ofta för att ge datorn själv ett alias men även kalla den för samma sak som tjänsten den serverar. Till exempel: `klump A 194.47.250.18`
`ftp CNAME klump`
Där heter datorn egentligen klump, men går man till namnet ftp hamnar man också på klump. Det mesta som går att göra med CNAME kan man även göra med multipla A-poster, men fördel med CNAME är att det är lättare att till exempel byta IP på en maskin. Då byter man bara en A-post och låter CNAME var kvar. Nackdel med CNAME är att det blir två namnuppslag.

8.2.3 AAAA

AAAA är motsvarigheten till A, men för IPv6. Till exempel:
`f.root-servers.net. 604800 IN AAAA 2001:500:2f::f`
Man kan som vanligt utelämna nollor på ett ställe i adressen, se stycke 5.7.

8.2.4 PTR

PTR är det omvända till A och AAAA, alltså från IP till namn. Men en IP-adress är mest signifikant från vänster och en domän från höger, till exempel är 130.235.x.y ett IP som tillhör Lunds universitet och bör därmed heta z.lu.se (där x och y är valfria oktetter och z valfritt namn). Lösningen på detta

är den påhittade domänen in-addr.arpa. Den kan ses som ett eget träd enbart till för omvända uppslag. Om en dator har IP 130.235.102.10 så kastar man om ordningen på talen för att få motsvarande namn, alltså 10.102.235.130.in-addr.arpa och därmed har man fått den minst signifikanta siffran (datorn på LAN:et) först och LU-nätet närmast in-addr. Läser man från höger, det vanliga hållet i domännamn ser man först 130, sen 235, 102 och 10 och det blir ett träd med förgreningar vilket DNS behöver. Då kan namnservrarna på LU sköta domänen 235.130.in-addr.arpa och i den ha en dator som heter 10.102. Eller kan man dela upp det ett steg till och separat administration av 102.235.130.in-addr.arpa. Dock om nätgränser inte är på jämna oktetter (CIDR-subnät, se 5.3) blir det lite svårare, men det finns ett snyggt hack där man använder CNAME för det (läs i Nemeth eller RFC2317).

Har man blivit tilldelad ansvar för domänen 102.235.130.in-addr.arpa. så består reverse-zonen av rader som:

```
1 PTR gateway
10 PTR www
11 PTR mail
12 PTR ftp
```

Rent tekniskt är egentligen första fältet fortfarande namn, men de är ganska lika siffror. Eftersom de inte avslutas med punkt är de relativa standarddomänen för zonen, vilken i det här fallet ska vara 102.235.130.in-addr.arpa.

Man får/ska bara ha en PTR-post för varje IP. Ibland sköter någon annan PTR-mappning (reverse mapping), till exempel internetleverantören, men har man möjlighet så är den bra att sköta själv. Bland annat är det lägre risk att bli spamsorterad om din mejlserver heter mail.domän.se än att den heter dhcp238.isp.se, och mest mejl tappar man om den inte har något bakåtnamn alls.

8.2.5 SOA

De poster jag har tagit upp hittills består främst av namn, typ och värde. SOA är lite krångligare, men jag kan trösta med att det bara behövs en för varje zon och de går bra att kopiera och modifiera från annat ställe. Ordningen mellan

poster spelar egentligen ingen roll, men man bör alltid börja med SOA (följt av NS och sen andra). SOA betyder som sagt Start Of Authority och kan se ut så här:

```
@ IN SOA ns.admin.com. hostmaster.admin.com. (
             2011080401       ; serial
                     3H       ; refresh
                     15       ; retry
                     1W       ; expiry
                     8H)      ; negative ttl
```

@ är en förkortning av aktuell zon. IN är som vanligt Internet. SOA är typen. Det som ska följa direkt därefter är namnet på din huvudsakliga namnserver, och en kontaktmejladress där snabel-a är utbytt mot punkt (eftersom @ betyder zon). Båda dessa namn ska avslutas med punkt. Det ska givetvis vara din egen domän i stället för admin.com, och ange helst ett alias som hostmaster eller root i stället för din vanliga mejl i kontaktuppgiften.

Därefter följer fem tal, varav det första kan anses som viktigast. Dessa tal skriver man oftast under varandra och därför har man parenteser innan och efter dem. Först kommer serienumret, det ska alltid öka vid varje ändring i zonfilen för att slavservrar ska kunna jämföra med sina zonfiler. Man kan numrera serial med 1, 2, 3 osv men jag tycker det är bäst att följa konventionen år-månad-dag-nr. Serial måste inte öka med ett, men det får aldrig minska. Och det är lätt att glömma att öka det, gör man det så kan olika namnservrar ge olika svar. De sista två siffrorna ger möjlighet till 99 ändringar varje dag, det brukar räcka.

De andra siffrorna är i ordningsföljd. Refresh: hur ofta slavar ska fråga master. Retry: hur ofta slavar ska prova igen om inte mastern svarar. Expire: Hur länge som en slav ska tycka att den är giltig om master är nere. Och sist negativ TTL, hur länge svar av typen ”vet/finns ej” ska sparas i minnet. Värdena ovan är ganska rimliga, och normalt sett behöver man inte ändra dessa värden. Vid ändringar skickas oftast notifieringar till slavservrar så refresh och retry är numera mest som reservmekanism. Det sista siffervärdet hade annan betydelse förr (bra att veta om ni läser gammal bok eller info).

8.3 Konfiguration av BIND

8.3.1 Installation av BIND på RH

På Red Hat installerar man BIND med:
`sudo yum install bind`

För en lite säkrare installation kan man välja att köra i chroot, det paketet installeras med:
`sudo yum install bind-chroot`

Då skapas en katalog `/var/named/chroot/` som en ny rotkatalog för BIND. Om någon utnyttjar eventuellt säkerhetshål i BIND så ska de inte kunna komma utanför den katalogen. Denna lösning medför lite mer administration, som längre sökvägar och filer med samma namn på olika ställen. Det kan förvirra, så radera de som inte ligger under `/var/named/chroot/` Jag brukar skapa symlänk för `/etc/named.conf` som pekar på:
`/var/named/chroot/etc/named.conf`

En annan nackdel är att enbart root kan läsa många av filerna, så det blir lätt att man kör `su -` istället för `sudo`.

8.3.2 Installation av BIND på Debian

På Debian installerar man BIND med

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

chroot får man sätta upp själv.

8.3.3 named.conf

I `named.conf` ska man lägga till minst fyra rader för varje domän som ska skötas av namnservern. De kan se ut så här:

```
zone "expertmaker.com" IN {
        type master;
        file "master/expertmaker.com";
};
```

Det som står efter zone är domännamnet. Typen är master. Raden med file anger namnet på zonfilen. Jag brukar lägga dem i kataloger som heter `master` och `slave`, men det är godtyckligt, även vad filerna ska heta kan man välja fritt. Men ganska logiskt är samma som domännamnet. Sökvägen till file är den som anges med directory under options i `named.conf`, ofta så här:

```
        directory       "/var/named";
```

så det blir alltså `/var/named/chroot/var/named/master` som filen `expertmaker.com` ska ligga i.

På en rekursiv namnserver ska man ha ”recursion yes;” under options, och en entry för rotzonen:

```
zone "." IN {
        type hint;
        file "named.ca";
};
```

Däremot för en innehållsserver, som den som har zonfilen expertmaker.com här ovanför, ska man ha:

```
recursion no;
```

Du bör även lista IP-nummer till slavarna och enbart tillåta `allow-transfer` för dem.

För en slavserver blir varje zons konfiguration så här:

```
        zone "expertmaker.com" {
                type slave;
                file "slaves/extern_expertmaker.com";
                masters { 85.235.7.58; };
        };
```

Skillnad är förstås ”type slave”. Det efter ”zone” ska vara samma som på mastern, men filnamn och katalog har valfria namn. Man ska även lista masterns IP-nummer. Och på en slav bör man ha

```
notify no;
```

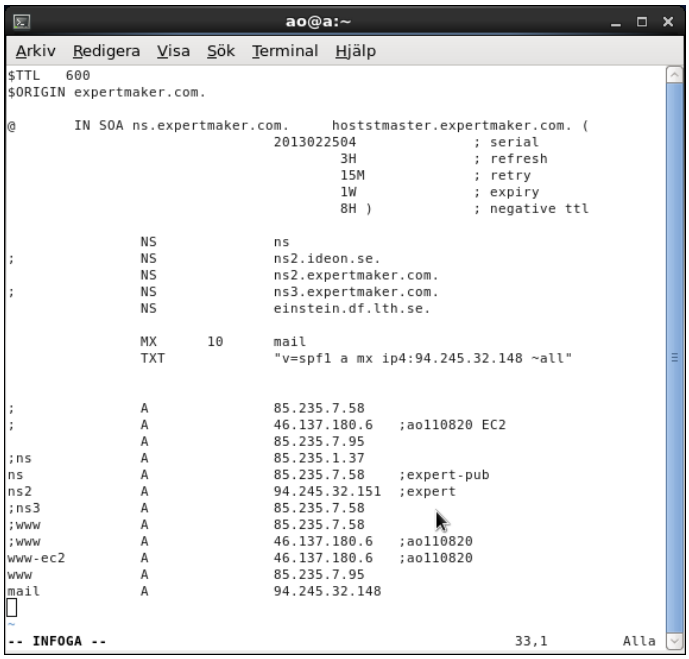
Allra sist i `named.conf` bör man ha `include "/etc/rndc.key";` och generera en nyckel med `rndc-confgen -a` vid installationen.

8.3.4 Zonfiler

Jag har tidigare nämnt de olika posterna i en zonfil. Det är dessa filer som är själva databasen för DNS och BIND. Du ska redigera dem på den server som är master för domänen, och köra `rndc reload` efteråt. Så här:

```
sudo vi /var/named/chroot/var/named/master/expertmaker.com
sudo rndc reload
```

Glöm inte att uppdatera serienumret. Och glöm inte avslutande punkter där de behövs. I figur 8.2 ser du exempel på en zonfil.



Figur 8.2: *Zonfil*

Lägg märke till att kommentarer inleds med semikolon, och hur jag skriver serienumret. Att ha högerkanten rakt ovanför H:et i raden nedanför gör att man inte missar en siffra när man ändrar det. Denna version är alltså fjärde ändringen som gjordes 2013-02-25. (Filen är längre än 33 rader i verkligheten).

Att ha 600 sekunder (tio minuter) på $TTL är inte alltid lämpligt, bättre värde är t.ex. 3600, men man kan sänka om man planerar att göra flera ändringar.

8.4 Lästips

Här är två böcker om DNS:

- Liu, Cricket & Albitz, Paul (2006) *DNS and BIND*. O'Reilly
- Aitchison, Ron (2005) *Pro DNS and BIND*. Apress

Jag rekommenderar även att läsa kapitel 17 i Nemeth et al (2010) *UNIX and Linux System Administration Handbook* (Jag har nämnt den boken förut, men deras beskrivning av DNS är mycket bra.)