DNS Basics

Oscar Buse

08 Juni 2001

Samenvatting

Dit is een document over DNS, met name bedoeld voor de servicedesk van InterNLnet.

De bedoeling van dit document is niet om te beschrijven hoe DNS technisch (op unix systemen) geïmplementeerd is. Hoewel dit onvermijdelijk is ligt de nadruk op een uitleg van DNS zonder ons (al te) druk te maken over de technische implementatie.

Inhoudsopgave

Inleiding

In dit document wordt het Domain Name System, oftewel DNS, beschreven. Kort gezegd verzorgt DNS de naam $< - >$ adres vertaling tussen computers onderling. Met *naam* worden namen bedoeld als: `neerbosch.nijmegen.internl.net`, `www-.internl.net` etc.. Met *adres* wordt IP-adres bedoeld. Elke computer die in een netwerk hangt beschikt over één (of meerdere) IP-adres(sen). IP-adressen bestaan uit de zgn. 'dotted-decimal'-notatie: bijvoorbeeld: `193.143.24.34`.

Computers kunnen gemakkelijk nummers onthouden en elkaar ook met behulp van die IP-adressen vinden. Maar namen daarentegen zijn weer gemakkelijker voor de mens: vandaar DNS.

In hoofdstuk 1, "Waarom DNS", wordt de geschiedenis van DNS beschreven en wordt ook toegelicht waarom DNS is zoals het is. Ook wordt een kleine inleiding gegeven op begrippen als domeinen en zones. Verder wordt benadrukt bij welke instanties de verantwoordelijkheid m.b.t. DNS over het algemeen liggen.

In hoofdstuk 2, "Hoe werkt DNS", zal preciezer gekeken worden naar de (theoretische) werking van DNS. Hierbij spelen o.a. begrippen als *resolver*, *name-server* en *caching* een rol. Dan wordt ook uitgelegd hoe het komt dat als iemand, ergens op de wereld, in zijn browser `http://www.internl.net` typt dat uiteindelijk onze www-server wordt gevonden!

In hoofstuk 3, "De omgekeerde wereld", zal een vaak minder begrepen maar ook zeer essentieel onderdeel van DNS: de *reverse lookup*, oftewel adres naar naam vertaling worden uitgelegd.

In hoofstuk 4, "DNS data" komt de data-structuur aan bod welke door DNS gebruikt wordt. Dan wordt ook duidelijk wat DNS en email met elkaar te maken hebben.

Als de theorie rond DNS grotendeels is behandeld worden in Hoofdstuk 5, "Tools", een aantal tools (commando's) behandeld die kunnen helpen bij het verkrijgen van de juiste DNS-informatie en bij het opsporen van problemen.

In hoofdstuk 6, "Configuratie van BIND"[1] wordt globaal bekeken hoe DNS (bij InterNLnet) technisch geïmplementeerd is.

In hoofdstuk 7, "Dokter DNS" wordt het tijd om "Dokter DNS" te consulten met een aantal mogelijk voorkomende problemen.

Uiteindelijk worden in Hoofdstuk 8, "Nieuwe ontwikkelingen", enkele nieuwe ontwikkelingen aangeduid.

In de Literaturtlijst tenslotte staan enkele boeken en URL's om te gebruiken als naslag cq. studie.

Veel (lees)plezier!

[1] Berkeley Internet Name Daemon, de verreweg populairste implementatie van DNS, op zo'n beetje iedere versie van Unix maar er is ook een versie voor NT.

Hoofdstuk 1

Waarom DNS

In reeds lang vervolgen tijden, toen DNS nog niet bestond, beschikte iedere computer die in het (toen veel kleinere) internet hing over een file (`HOSTS.TXT`) met daarin de naam-adres koppeling van iedere andere computer in het internet. Dit file werd centraal bijgehouden op een computer (met naam SRI-NIC) van SRI's[1] Network Information Center (de "NIC"). Voor een systeembeheerder was het zaak om regelmatig een nieuwe versie van dit bestand op te halen. Naarmate het aantal hosts toenam groeide ook de onmogelijkheid om deze constructie voort te zetten:

- de belasting op SRC-NIC werd onwerkbaar.
- omdat alle namen opgeslagen waren in één enkel file (*flat name-space*) werd het ondoenlijk om te garanderen dat namen uniek bleven.
- het werd steeds lastiger om bij te blijven: als je eindelijk een nieuwe versie van `HOSTS.TXT` had verkregen was deze eigenlijk direct alweer verouderd.

Duidelijk dus dat er iets moest komen dat de nadelen van een centraal file moest opheffen. De oplossing ligt in het aanbrengen van structuur in de naamgeving. Door het aanbrengen van structuur kan een hiërarchie worden gecreëerd waarmee makkelijker te verwezenlijken is dat namen uniek zijn en dat het beheer en opslag van de data gedecentraliseerd worden. Hierdoor werden de belangrijkste eisen van het nieuw te verzinnen systeem:

- de naamgeving voor hosts op het internet moest hiërarchisch (i.t.t. de flat namespace van `HOSTS.TXT`) worden.
- het bijhouden en opslaan van alle host-informatie moest niet meer komen te liggen bij één centrale instantie maar moest kunnen worden gedecentraliseerd.

Rond 1984 zijn hiertoe de eerste publicaties (in de vorm van RFC's[1]) van de hand

[1] Stanford Reasearch Institute, California

[1] RFC staat voor "Request for Comments": documenten die de, nogal informale, basis vormen van de technische details van nieuwe technieken m.b.t. het internet

van een zekere Paul Mockapetris verschenen. Het nieuwe systeem werd DNS gedoopt: Domain Name System. Biede bovengenoemde eisen zijn verwerkt in DNS.

1.1 hiërarchie

DNS toont qua hiërarchie veel overeenkomsten met een (unix) filesysteem. Om tot zulk een hiërarchische indeling te komen werd het internet opgedeeld in *domeinen*. Aan de top van de hierarchie staat het *root-domein*. Strikt genomen heeft dit domein geen naam. Direct onder het root-domain bevinden zich de zgn. *toplevel domeinen*. In het begin werd de volgende indeling gemaakt:

- het com-domein (com): met name bedoelt om commerciële bedrijven in onder te brengen, denk bv. aan `microsoft.com`.
- het edu-domain (edu): vooral voor educatieve instellingen (universiteiten).
- het gov-domein (gov): voor overheidsinstellingen (`whitehouse.gov`).
- het mil-domain (mil): voor militaire instellingen, b.v. voor het Amerikaanse leger (`army.mil`).
- het net-domein (net): voor *netwerk*-organisaties, denk bv. aan `internic.net`.
- het org-domein (org): voor niet-commerciële instellingen, bv. `linux.org`.
- het int-domein (int): voor internationale organisaties als bv. de North Atlantic Treaty Organisation (NATO).

Zoals je ziet was er niet voor gekozen om van zogenaamde *landen-domeinnamen* uit te gaan. Dit is ook niet zo gek: niemand had toen verwacht dat het oorspronkelijke, in de Verenigde Staten gelegen, ARPAnet[2] zou uitgroeien tot het hedendaagse internet.

Om toch de internationale groei van het internet te kunnen beantwoorden zijn later de landen-domeinnamen aan DNS toegevoegd. Elk land kreeg zijn eigen 2-letterige domainnaam:

- uk: voor Groot-Brittanië (officieël (volgens ISO standaard 3166) is dit gb maar in de praktijk wordt meestal toch uk gebruikt.
- at: voor Oostenrijk.
- au: Australië.
- aw: Aruba

[2]het ARPAnet (ARPA komt van "The U.S. Department of Defence's Advanced Research Project Agency") is het prille begin van het internet. Het eigenlijke doel was om door computers met elkaar te verbinden op die manier te bezuinigen op (toenmalige) dure *recources* als geheugen- en rekencapaciteit.

- nu: Niue eiland.

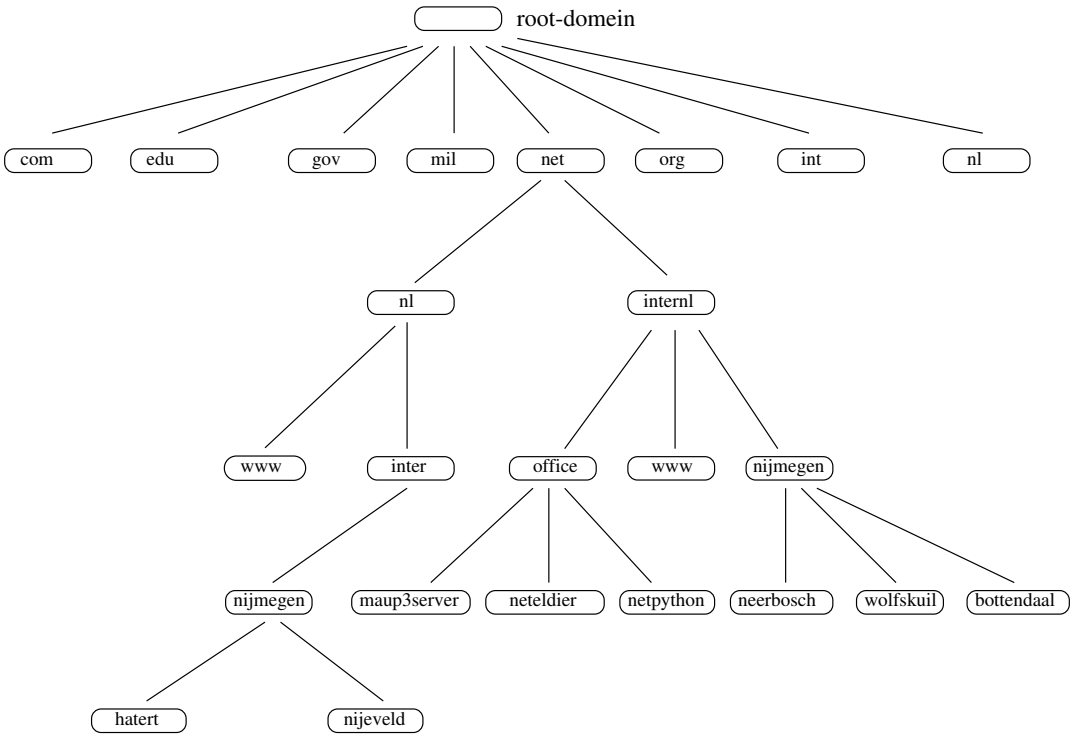
om er slechts enkele te noemen.

Om aan de stijgende vraag naar domeinnamen te kunnen voldoen en om meer diversiteit in de naamgevingen te kunnen aanbrengen zijn (of worden in de nabije toekomst) nieuwe toplevel domeinen geïntroduceerd[3]. Om er enkele te noemen:

.aero	Voor de luchtvaart.
.biz	Voor bedrijven (waarbij `.com` een meer internationaal karakter heeft).
.coop	Voor 'coöperatieve' bedrijven.
.info	Voor elk doel.
.museum	Voor musea.
.name	Voor particulieren.
.pro	Voor accountants, advocaten en doctoren.

Onder elk toplevel-domain (ook wel: *first-level domain*) bevinden zich weer *second-level domains* etc...

In een plaatje ziet dit er als volgt uit:



[3]zie: `www.icann.org/tlds/`

Zoals uit het plaatje blijkt lijkt dit sterk op een omgekeerde *boom* met bij de wortels het root-domein. Een domein-*naam* is een aaneenschakeling van domeinen in de boom aan elkaar geknoopt met een punt. Bijvoorbeeld: `neerbosch.nijmegen-.internl.net`. Host `neerbosch` is lid van het domein `nijmegen`, wat weer lid is van het domein `internl` wat zelf weer lid is van het toplevel domein `nl`. Een domein met daaronder (sub)domeinen wordt ook wel een *parent* genoemd en vice versa kun je die subdomeinen *child* van dat domein noemen. Het hoeft geen betoog dat een domein zowel parent als child kan zijn. Strikt genomen eindigt een volledige domeinnaam (FQDN: *Fully Qualified Domain Name*) op `.<lege-string>` maar die lege string (van het root-domein) wordt voor het gemak maar weggelaten zodat een FQDN eindigt op een punt.

Duidelijk uit deze boomstructuur blijkt dat aan de eis van hiërarchie wordt voldaan: er zijn verschillende niveau's. Hierdoor is ook het probleem m.b.t. unieke namen verdwenen: alleen domeinen op hetzelfde niveau met dezelfde parent kunnen niet dezelfde naam hebben. In alle andere gevallen kunnen er geen naam-conflicten optreden. Maar hoe zit het nu met die tweede eis: decentralisatie van het bijhouden en opslag van de domein-informatie?

1.2 decentralisatie van autoritiet

Door de hiërarchische indeling van DNS kan de verantwoordelijkheid m.b.t. domeinen eenvoudig gedelegeerd worden naar afzonderlijke takken in de gehele boom.

Een voorbeeld:
De instantie verantwoordelijk voor het beheer van het toplevel domein `net` kan de bevoegdheid voor het bijhouden van alle domein-informatie voor het domein `internl.net` leggen bij de instantie die behoort bij `internl.net`. In dit geval mag het duidelijk zijn: dat zijn wij. Met delegatie wordt bedoeld dat, in dit geval, InterNLnet volledig vrij is in het kiezen van namen *onder* de tak `internl.net.`. De instantie die het domein `net.` beheerd hoeft niets te weten van alle namen onder `internl.net.`. Het enige wat bij `net.` nog wel bekend moet zijn zijn de *name-servers* van `internl.net.`. Hiermee is tevens een belangrijk nieuw begrip geïntroduceerd: de name-server. Simpel gezegd is dit een computer die verstand heeft van alle namen in een domein. Dat dit inderdaad een beetje simpel gezegd is blijkt als je het verschil begrijpt tussen domeinen en *zones*.

1.2.1 domeinen en zones

Een domein bestaat uit één tak, met alle bijbehorende vertakkingen, van de DNS-boom. Zo bestaat het domein `internl.net.` uit het domein `internl.net.` zelf met alle namen daaronder.

Een zone is hetzelfde maar dan zonder de gedelegeerde delen. Over een zone spreek je ook eerder vanuit het oogpunt van de name-server. In het voorbeeld hierboven heeft de name-server van `.net.` geen weet van alle namen onder `internl.net.`.

Het enige wat ze wel weten zijn de namen (en IP-adressen) van de name-servers van `internl.net.`. Men zegt ook wel dat `net.` de autoriteit m.b.t. alle namen onder `internl.net.` gedelegeerd heeft naar `internl.net.`. Dit scheelt de beheerder voor het domein `net.` een hoop werk! Ook duidelijk wordt dan dat de administratie van de *allerhoogste* name-servers in de hierarchie, de zgn. *root naam-servers*, een stuk eenvoudiger wordt: de meeste autoriteit is gedelegeerd naar de name-servers een niveau lager, wat met name rest is de administratie van de namen (en weer IP-adressen) van die name-servers.

1.3 autoriteiten

Duidelijk uit bovenstaande blijkt dat de autoriteit m.b.t. tot alle namen op het internet niet meer bij één centrale instantie berust maar is uitgesmeerd over allerlei instanties autoritair voor hun stukje in de DNS-boom. Men zegt ook wel dat elk domein zijn eigen *naming authority* kent. Zo is bijvoorbeeld de naming authority voor het domein `nl.` de Stichting Internet Domeinregistratie Nederland[4]. De Internet Corporation for Assigned Names and Numbers[5] is de naming authority op het hoogste niveau en kan dus nieuwe toplevel domeinnamen verzinnen. Hoe gemakkelijk domeinnamen worden toegekent verschilt per naming authority. De één kijkt of de aangevraagde domeinnaam alleen syntactisch correct. Zo moet een domainnaam in elk geval voldoen aan de volgende criteria:

- alleen hoofd- en kleine letters, cijfers en de *hyphen* (-) zijn toegestaan. Er is overigens geen onderscheid tussen hoofd- en kleine letters.
- het eerste teken is bij voorkeur een letter en het laatste teken mag geen hyphen zijn.

Andere naming authorities kijken of de naam niet ook bv. een handelsmerk is. Zo kan de naam `grolsch` geweigerd worden als deze wordt aangevraagd door een bedrijf wat niets van doen heeft met het betreffende biermerk.

Naming-authorities hebben dus autoriteit over het uitgeven van namen onder een bepaalde tak in de DNS-boom. Zoals reeds is opgemerkt is de naming-authority voor het domein `nl.` (SIDN) bevoegd onder dit domein namen uit te delen. Maar wie is dan verantwoordelijk voor de *name-space* voor deze namen, m.a.w. wie is verantwoordelijk voor de vertaling van de namen naar IP-adressen? Gelukkig voor SIDN dragen zij die verantwoordelijkheid niet alleen. Ook is het lang niet altijd zo dat elk bedrijf met zijn eigen domeinnaam verantwoordelijk is voor de name-space in dat domein. In de praktijk komt de verantwoordelijkheid voornamelijk op rekening van een combinatie van bedrijven, internet providers (zoals bv. InterNLnet) en netwerk-organisaties (bv. internic, netwerk-solutions, RIPE[6]).

[4]SIDN, `www.sidn.nl`
[5]ICANN, `www.icann.org`
[6]Réseaux IP Européens, `www.ripe.net`

Hoe dit allemaal met elkaar samenwerkt zodat toch vrij eenvoudig namen naar IP-adressen kunnnen worden vertaald (daar is het uiteindelijk vooral om te doen) wordt in het volgende hoofdstuk uitgelegd.

Hoofdstuk 2

Hoe werkt DNS

In het vorige hoofdstuk is uitgelegd wat DNS is en hoe DNS is opgebouwd. Door de hiërarchische structuur bleek dat het vrij eenvoudig is de kennis m.b.t. de namen op het internet uit te smeren over allerlei instanties (lees: name-servers) , elk verantwoordelijk (autoritair) voor hun stukje van de boom.

Maar hoe werkt nu in de praktijk het achterhalen van het juiste IP-adres, gegeven een bepaalde naam? Dit is het best te verduidelijken aan de hand van een voorbeeld. Echter, voordat dit voorbeeld uitgewerkt wordt, eerst een paar begrippen:

2.1 de resolver

DNS werkt volgens het client-server principe. Hierbij is de resolver de client en is de name-server (waarvan straks meer) de server. Maar wat is nu een resolver? Dit zijn niet meer dan een aantal *library-functies ingebakken* in de meeste software zoals browsers (bv. Netscape of IE), email-clients (bv. Thunderbird), telnet-clients, ftp-clients etc.... Meestal kunnen deze ingebakken functies niet meer dan:

- een vraag formuleren (wat is het IP-adres van deze FQDN?).
  Merk op dat een resolver altijd een naam *aanvult* tot een FQDN.
- een name-server benaderen en de vraag overhandigen.
- een antwoord ontvangen en interpreteren.

Er zijn wel intelligentere resolvers (bv. *adns*) maar deze zijn niet gebruikelijk.

2.1.1 configuratie van de resolver

Hoe weet de resolver nu welke name-server te vragen? Dit is geconfigureerd op het Operating Systeem (OS): bij Windows geef je dit bv. op in tabblad "DNS"

onder tabblad "Network" van het tabblad "TCPIP-instellingen" in het configuratiescherm van "Jouw computer". Onder unix zijn het enkele regeltjes in het file `/etc/resolv.conf`.

In `/etc/resolv.conf` kan met het woord `domain` het default domein worden opgegeven. Dit domein wordt dan gebruikt om niet-FQDN's aan te vullen. Met het keyword `nameserver` kunnen dan één of meerdere name-servers worden opgegeven welke door de resolver geraadpleegd kunnen worden. Tegenwoordig kan i.p.v. het keyword `domain` ook het keyword `search` gebruikt worden. Hiermee kunnen meerdere domeinen opgegeven worden waarbij het eerste domein geldt als het default domein. Het gebruik van `search` maakt het gebruik van `domain` dan ook overbodig. Het gebruik van search kan handig zijn indien veel gebruik wordt gemaakt van een aantal domeinen. Voorbeeld van een `/etc/resolv.conf` voor een machine in het domein `office.internl.net`:

```
search internl.net nijmegen.internl.net
nameserver 202.219.13.156
nameserver 202.67.14.144
nameserver 203.168.10.120
```

Er is al opgemerkt dat een resolver aan de name-server altijd een FQDN overhandigt. Maar hoe vult de resolver de naam aan indien geen FQDN wordt opgegeven door een gebruiker? De regels hiervoor zijn als volgt:

- bij een enkele naam vult de resolver deze aan met het default domein.
- indien de naam uit 2 of meer labels bestaat (dus in elk geval 1 punt bevat) wordt deze naam beeindigt met een punt. Levert dit niets op dan wordt alsnog de naam aangevuld met het default domein.

Indien gebruik wordt gemaakt van het keyword `search` i.p.v. het keyword `domain` dan zullen achtereenvolgens alle domeinen achter `search` geprobeerd worden waarbij begonnen wordt met het default (eerste) domein. Als een gebruiker bv. typt:
`ssh neerbosch`
Dan wordt eerst geprobeerd aan te vullen met het eerste default domein `internl-.net`. Dit zal mislukken (`neerbosch.internl.net` is geen bestaande host). De tweede aanvulling met `nijmegen.internl.net` match in dit geval wel: `neerbosch-.nijmegen.internl.net` is een bestaande naam.

Merk hierbij op dat het meestal niet zo'n goed idee is om bv. een toplevel domainnaam te kiezen voor een domein een stuk lager in de hiërarchie: Als iemand wil connecten op `www.org.internl.net` maar alleen `www.org` typt wordt de host `www.org.` gezocht i.p.v. de bedoelde `www.org.internl.net`...

2.2 name-server

Het werkpaard voor DNS is de name-server: deze doet alle mogelijke moeite achter het IP-adres van een naam te komen een geeft het uiteindelijke antwoord. Zo-

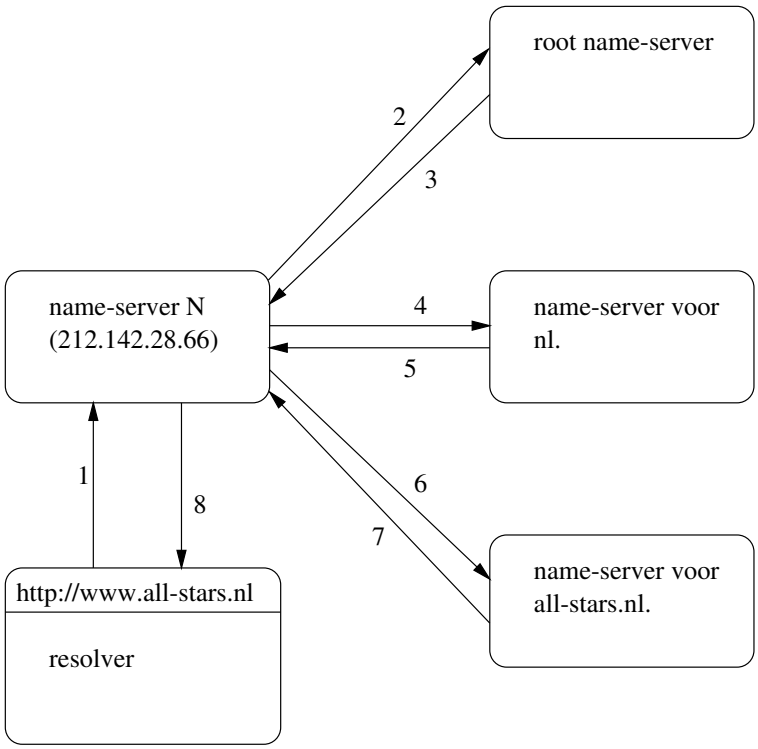
als reeds vermeld zijn verreweg de meeste name-servers op het internet computers voorzien van de BIND-software. In tegenstelling tot de resolver is de name-server gewoon zichtbaar: op unix is het een proces wat continu draait, met als naam: `named`.

2.3 resolutie

Een vraag m.b.t. DNS kan bijvoorbeeld zijn: wat is het IP-adres van `www.waarheenwaarvoor.nl`? Het proces waarbij DNS hierop tracht antwoord te geven heet *resolutie*.

2.4 een voorbeeld

Om het proces van resolutie beter te begrijpen volgt hier een voorbeeld waarbij van het volgende plaatje wordt uitgegaan.



Verder wordt aangenomen dat de gebruiker op zijn linux-doos het volgende in `/etc/resolv.conf` heeft staan:

```
search arnhem.chello.nl chello.nl
nameserver 212.142.28.66
nameserver 212.142.28.130
```

Stel deze gebruiker typt in zijn browser `http://www.all-stars.nl` in. De volgende stappen worden dan doorlopen (de nummers 1 tm 8 in het plaatje komen overeen met deze stappen):

1 De browser bevat de resolver en zal aan een name-server vragen wat het IP-adres is van `www.all-stars.nl.`. Let op de FQDN: de laatste punt wordt dus door de resolver toegevoegd. Het is echter niet verstandig, bv. omdat je denkt de resolver wat werk te besparen, om zelf die laatste punt toe te voegen: bij bv. uri's zullen veel *redirects* (met Javascript, PHP of iets anders) geen rekening houden met die laatste punt...maar dit terzijde.

   In eerste instantie wordt dan iets gevraagd aan de name-server die luistert[1] op de machine met IP-adres `212.142.28.66`. Mocht die geen thuis geven dan kan alsnog de tweede (`212.142.28.130`) geprobeerd worden. Geeft die ook niet thuis dan zal de vraag in elk geval mislukken. Duidelijk waarom de name-servers niet met hun naam maar met hun IP-adres vernoemd zijn?

2 Stel de eerste name-server (name-server N) luistert en ontvangt dus de vraag: geef mij het IP-adres dat hoort bij `www.all-stars.nl.`? In eerste instantie zal deze name-server kijken of hij zelf het antwoord weet. Zelf zal hij namelijk ook wel over wat zone-data beschikken en misschien zelfs wel voor `all-stars.nl.`. In dat geval kan hij onmiddelijk antwoord geven. Is dit echter niet zo, de name-server weet niets van `all-stars.nl.`, dan zal de name-server om hulp gaan vragen. Maar bij wie? Ervan uitgaande dat ook de name-server van `nl.` (zeer waarschijnlijk weet die meer!) bij deze name-server niet bekend is, is er nog maar één mogelijkheid: de vraag doorsturen naar één van de *root-servers*[2] Bovendien wordt gebruik gemaakt van it anycast: een techniek om meerdere servers naar 1 IP-adres te laten luisteren. Dus in feite zijn er veel meer dan 13 root name-servers. Hierbij moet je weten dat **elke** name-server **altijd** beschikt over de IP-adressen van de root-servers. Zoals reeds verteld zijn dit de name-servers met autoriteit bovenin de dns-boom: elke root-server weet tenminste de onderliggende name-servers voor de diverse top-level domeinen (`com.`, `org.`, etc..) te bereiken.

3 We zijn dus weer wat verder: de vraag komt nu te liggen bij 1 van de 13 root name-servers. Deze zal, waarschijnlijk, niet direct het antwoord weten maar kan de vragende name-server wel een stukje op weg helpen. Dit door het IP-adres te geven van de name-server welke meer zal moeten weten. Dit is de name-server autoritair voor het domein `nl.`. Merk hierbij op dat autoriteit voo de zone `nl.` door de root-servers is weggedelegeerd.

4 Met de informatie van één van de root name-servers kan onze name-server N een name-server autoritair voor het domein `nl.` raadplegen.

[1] met een *name-server die luistert* wordt bedoeld: het BIND-proces op die computer luistert in afwachting van vragen (requests)

[2] het *meervoud* is hier wel op z'n plaats: om een zekere redundantie en spreiding van load op de root-servers te verkrijgen zijn er, op dit moment, 13 root name-servers verspreid over de wereld. Tien in Amerika, twee in Europa en één in Japan.

5 Ook de name-server van `nl.` zal, weer waarschijnlijk, het antwoord niet direct weten maar zal, weer via delegatie, wel de vragende nameserver N weer een stukje verder kunnen helpen. Namelijk door het IP-adres van de name-server te geven autoritair voor het domein `all-stars.nl.`.

6 Met de informatie van één van de name-servers autoritair voor het domein `nl.` kan onze name-server N een name-server autoritair voor het domein `all-stars.nl.` raadplegen.

7 Deze zal uiteindelijk weten wat het IP-adres is van de host met naam `www-.all-stars.nl.` en geeft dit antwoord aan de vragende name-server.

8 Deze geeft op zijn beurt het antwoord door aan de resolver. De browser kan vervolgens, met het IP-adres in handen, de juiste machine benaderen en vragen naar de webpagina voor `www.all-stars.nl` (let op dat nu de punt weer weg is: dit was tenslotte het oorspronkelijk verzoek: geef mij de webpagina welke behoort bij `www.all-stars.nl`).

Bovenstaand mechanisme roept natuurlijk om het herinvoeren van `HOSTS.TXT`! Dit is dan ook een *worst case scenario* en de praktijk is gelukkig meestal heel wat minder omslachtig. Vaak weten name-servers eerder de juiste name-server te benaderen of ze weten zelf het antwoord al (zonder dat ze autoritair zijn!). Dit komt door een mechanisme genaamd: *caching*.

2.5 caching

Name-servers kunnen informatie die ze verkrijgen gedurende een bepaalde tijd onthouden: dit wordt caching genoemd. Als in bovengenoemd voorbeeld nogmaals het verzoek voor het IP-adres van `www.all-stars.nl.` binnen komt zal dezelfde name-server het antwoord waarschijnlijk onthouden hebben (*in z'n cache hebben*) en dus direct kunnen antwoorden ondanks het feit dat de name-server niet autoritair is voor `all-stars.nl.`. Zo'n antwoord is dan ook een *non-authoritive* antwoord (maar niet-te-min een antwoord). Nog beter: niet alleen het IP-adres van `www.all-stars.nl.` zal bewaard worden maar ook alle andere informatie verkregen tijdens de resolutie door de DNS-hierarchie: de namen en IP-adressen van alle name-servers welke onderweg passeerden. De tijd hoelang een name-server deze informatie bewaard kan ingesteld worden en wordt uitgelegd in Hoofdstuk 4 ("DNS data").

Nu is caching niet het enige mechanisme wat de performance van DNS behoorlijk opschroeft. Een name-server staat er meestal niet alleen voor maar heeft vaak de hulp van andere name-servers welke voor ruwweg dezelfde informatie autoritair zijn. Tijd om de begrippen *master* en *slave* name-servers te introduceren.

2.6 master en slave name-servers

De verantwoordelijkheid voor autoriteit m.b.t. een zone ligt vaak niet bij één name-server. Meestal is er sprake van één master name-server en één of meerdere slave name-servers[3]. De master bevat de *hard-copy's* van alle zone-data: voor nu is het genoeg te onthouden dat dit bestanden zijn met daarin de koppelingen van alle namen in de zone aan IP-adresssen. In hoofdstuk 4 ("DNS data") worden deze files en de bijbehorende data nader bekeken. Een slave heeft zelf geen originale copy van de *zone-data* maar verkrijgt zijn data van de master (overigens bewaard de slave deze data niet alleen in z'n cache maar ook in files). Om de zoveel tijd (in te stellen op de master, wordt ook in in Hoofdstuk 4 behandeld) zal een slave bij de master aankloppen om verouderde data te verversen. Denk echter, mede door de naamgeving, niet dat een slave minder autoritair is dan de master: beide zijn autoritair voor de namen waarvoor vastgelegd is dat ze daarvoor autoritair moeten zijn. De voordelen van het bestaan van een master en één of meerder slaves mag duidelijk zijn:

- doordat een slave zijn informatie opvraagt bij de master is de administratie van alle zone-date op één plek (de master) gecentraliseerd. Dit bevordert het *up-to-date* en correct houden van de zone-data.
- het feit dat meerdere name-servers autoritair zijn voor een domein (of meerdere domeinen) bevordert de bereikbaarheid: als één name-server uitvalt is niet gelijk dat domein (of meerdere) onbereikbaar.
- ook zal het feit dat meerder name-servers antwoord kunnen geven op dezelfde vraag zorgen voor een spreiding van de *load* op zo'n name-server wat weer de performance ten goede komt.

2.7 caching only name-server

Er is nog een derde soort name-server: de caching only name-server. Deze name-server is (bijna) nergens autoritair voor maar doordat een name-server automatisch een cache opbouwt met reeds eerder verkregen informatie kan een caching name-server handig zijn om andere name-servers te ontlasten. Een caching only name-server wordt ook wel een resolving name-server genoemd.

[3]dit werden vroeger resp. de primary master en secondary masters genoemd. Alleen Microsoft houdt hardnekkig vast aan deze namen.

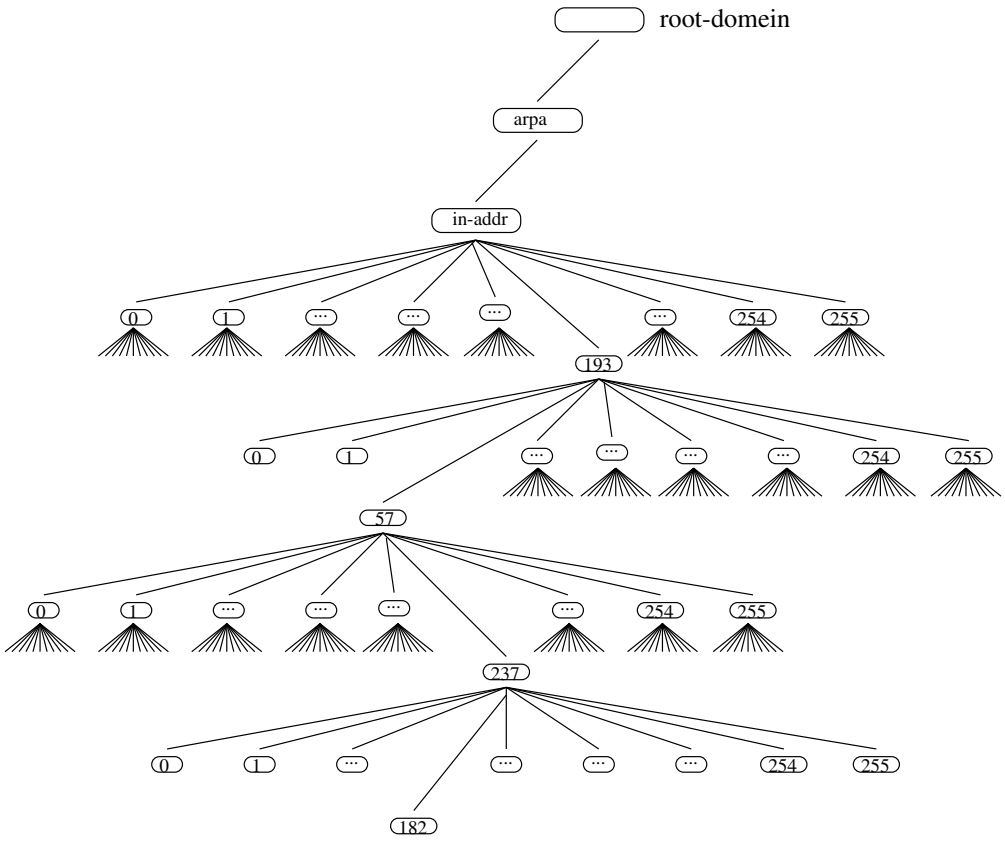
Hoofdstuk 3

De omgekeerde wereld

Tot nu toe is uitgelegd hoe DNS werkt als gevraagd wordt om het IP-adres behorende bij een naam. De omgekeerde vraag: "geef mij de naam behorende bij dit IP-adres?" is echter ook goed mogelijk en zal dus ook beantwoord moeten worden. Denk bv. aan logfiles welke een stuk leesbaarder (en de performance van de web-server een stuk trager!) worden als daarin namen staan in plaats van IP-adressen. Ook kijken sommige authorisatie-mechanismes of iemand afkomstig is vanaf een bepaald domein in plaats van afkomstig van een IP-adres (of IP-*range*) Voor de ingewijden: denk bv. aan de inhoud van `.rhosts`, `/etc/hosts/allow` en `/etc/hosts.equiv`.

Hiertoe is een even simpele als geniale oplossing bedacht: de DNS-boom werd uitgebreid met een bijzondere tak welke alleen bestaat uit nummers. Deze tak wordt ook wel de `in-addr.arpa` tak genoemd. Deze naamgeving is niet geheel willekeurig: Zoals reeds vermeld heette het prille internet nog ARPAnet. In het begin van DNS werden de hosts van het ARPAnet ondergebracht onder het domein `arpa`. Echter samen met de invoering van de top-level domeinnamen werden meer en meer namen vanuit het arpa-domain naar deze, meer geschikte, top-level domainnamen verhuisd. Dit had een uiteindelijke leegloop onder het arpa-domein tot gevolg zodat dit een prachtige bestemming werd voor een boom met alleen maar nummers[1].
Het idee is als volgt: een IP-adres bestaat uit 4 nummers van elk tussen de 0 en 255. Onder het domein `in-addr.arpa.` vinden zich 256 subdomeinen genummerd 0 t/m 255. Dit herhaald zich voor elk subdomein en dat in totaal 4 keer. Een plaatje zal de boel verduidelijken. Extra aangegeven is het pad `182.237.67.193,in-addr.arpa`:

[1] wel is nog extra het domein in-addr toegevoegd om juist daaronder alleen maar nummers te hangen.



Elk domein (behalve de domeinen helemaal onderaan) heeft zelf weer 256 subdomeinen. Het mag duidelijk zijn dat dit verreweg de grootste tak is van de DNS-boom: iedere host op het internet is terug te vinden in deze boom!

Merk in de naamgeving op dat, net zoals bij de gewone namen, de naamgeving begint *onderaan* in de boom. Echter IP-adressen worden juist geformuleerd van bovenaf: het IP-adres `193.67.237.182` heeft als tak `182.237.67.193,in-addr.arpa`. Dit heeft een praktische reden: aangezien autoriteit meestal gedelegeerd wordt naar netwerken (bv. naar het netwerk `193.67.237.0`) is de hierarchie in de boom analoog aan de formulering van IP-adressen. Bij IP-adressen beschrijven de eerste zoveel bytes het netwerk-deel en de rest een uiteindelijke host. De uiteindelijke hosts bevinden zich dan ook onderaan in de boom en de paden worden net zo gelezen als de gewone namen. Het zijn alleen de IP-adressen die *andersom* uitgesproken worden.

Uit bovenstaand verhaal blijkt dat er ook, net zoals voor de *name-space* ook autoriteit moet bestaan m.b.t. tot de *IP-space*. Dit komt ook, net zoals voor de *name-space*, voornamelijk op rekening van een combinatie van bedrijven, internet providers (zoals bv. InterNLnet) en netwerk-organisaties.

Hoofdstuk 4

DNS data

4.1 resource-records

Nu is beschreven hoe DNS in elkaar steekt wordt het tijd om eens wat preciezer te gaan kijken naar de data die is opgeslagen in DNS. Tot nu toe is voornamelijk geroepen dat DNS namen aan nummers koppelt en andersom. Maar er is meer aan de hand. In DNS zijn meer soorten informatie opgeslagen dan bovennoemde twee. Alle informatie is vastgelegd in zgn. *resource-records*. Voor nu is het genoeg te weten dat deze resource-records zijn opgeslagen in files op de master name-server. Alle resource-records die behoren bij één domein worden ook wel de *zone-data* van dat domein genoemd. In hoofdstuk 6 wordt dit verder uitgewerkt.

De structuur van een resource-record is als volgt:

[naam] [TTL] klassetype data

Bijvoorbeeld:

```
all-stars.nl.  86400   IN   NS   ns.internl.net.
```

Het eerste veld is een domeinnaam, hostnaam of nummer. Wordt dit eerste veld weggelaten dan geldt dezelfde naam als het vorige record. In dit voorbeeld gaat het om het domein `all-stars.nl.`. In het tweede veld geeft de time-to-live (TTL) aan hoelang andere nameservers deze informatie in hun cache mogen hebben. Als deze weggelaten wordt geldt de default TTL (hoe deze bekend is volgt nog). Als DNS niet voor Internet aangelegenheden wordt gebruikt (wat maar heel zelden voorkomt) staat er in het derde veld iets anders dan IN (van INternet). In het vierde veld staat het type record, in dit geval is er sprake van een NS (Name Server)-record. In dit geval geeft dan het vijfde veld aan wat de naam van de name-server (`ns.internl.net.`) is voor het domein `all-stars.nl`.

Met name het type record geeft aan om wat voor informatie het gaat. Straks worden deze typen verder uitgewerkt maar hier alvast een overzicht:

SOA Start Of Authority record. Dit type record bestaat altijd voor een domein (of het nu om een naam- of nummer-domein gaat) en bevat administratieve gegevens voor dat domein.

NS Name Server record. Dit duidt aan dat het gaat om een name-server.

MX Mail eXchange record. Dit duidt een mail-server aan.

A Address record. Geeft het IP-adres behorende bij een naam.

CNAME Canonical NAME record. Geeft de *kanonieke* (echte) naam van een host.

HINFO Hardware Info record. Een record met hardware-informatie m.b.t. een host. Bijvoorbeeld welk type OS gebruikt wordt.

PTR PoinTeR record. het *omgekeerde* record voor vertaling van IP-adres naar naam.

Om de afzonderlijke typen records te verduidelijken ga ik uit van de volgende zonedata voor het domein `all-stars.nl.`:

```
all-stars.nl.   IN  SOA  ns.internl.net.  hostmaster.inter.nl.net. (
                2001041701d; Serial
                28800      ; Refresh (8 hours)
                7200       ; Retry (2 hours)
                604800     ; Expire (7 days)
                3600 )     ; Minimum (1 hour)
;
                IN      NS      ns.internl.net.
                IN      NS      ns2.internl.net.
                IN      MX 0    goalie.all-stars.nl.
                IN      MX 40   grootstal.nijmegen.inter.nl.net.
                IN      MX 100  fallback.nl.uu.net.
; Hosts / Aliases
localhost       IN      A       127.0.0.1
www             IN      CNAME   aanleun.nijmegen.inter.nl.net.
goalie          IN      A       62.163.53.199
```

Zoals je kunt zien bevat de zone `all-stars.nl.` 9 resource-records.

4.1.1 Het SOA-record

Dit record bevat de administratieve gegevens voor een domein en zal daardoor altijd als eerste veld een domein hebben. Voor `all-stars.nl.` is het SOA-record:

```
all-stars.nl.   IN  SOA  ns.internl.net.  hostmaster.inter.nl.net. (
                2001041701d; Serial
                28800      ; Refresh (8 hours)
                7200       ; Retry (2 hours)
                604800     ; Expire (7 days)
                3600 )     ; Minimum (1 hour)
```

Merk hierbij op dat alles wat in DNS-data achter een punt-comma (`;`) staat commentaar is en dus niet wordt geïnterpreteerd.

Het data-veld van een SOA-record bevat:

Origin	De naam van de master nameserver voor het domein. De machine dus die over de hard-copy van de zone-data beschikt.
Person	Het email-adres van de verantwoordelijke persoon voor het domein: degene die de zone-data bijhoudt. Dit is meestal het adres van de hostmaster voor de organistatie die autoritair is voor dat domein (bv. `hostmaster.inter.nl.net`). Merk op: het @-teken in dit email-adres is hier vervangen door een punt (.).
Serial	Het volgnummer (versienummer) van de zone-data. Na iedere wijziging moet deze worden opgehoogd. Op deze manier kan een slave nameserver checken of zijn informatie nog up-to-date is. Dit nummer (maximaal 32 bits) is groot genoeg om ook datum-informatie te bevatten[1].
Refresh	Dit veld geeft aan hoe vaak de slave de master zal checken. (in het voorbeeld 28000 seconden).
Retry	Hoe lang te wachten indien het niet lukt om een zone-transfer te doen (in het voorbeeld 7200 seconden).
Expire	Dit zegt hoe lang de slave van zijn gegevens mag uitgaan als hij niet in staat is de primary te benaderen. In de regel wordt deze waarde vrij hoog gekozen om tijd genoeg te hebben om evt. problemen met de master te kunnen verhelpen.
Minimum	De default waarde voor de TTL indien dit veld in andere records leeg is[2].

[1] Als je de datum consequent ten tijde wijziging in het format yyyymmdd wegschrijft verkrijg je automatisch een hoger nummer.

[2] in nieuwere versies van BIND geeft deze waarde de tijd voor *negative caching* aan. Dan bepaald deze waard hoe lang name-servers foutieve informatie (bv. een naam die niet bestaat) kunnen bewaren. De ouderwetse TTL wordt dan geset met de *variabele* $TTL

4.1.2 Het NS-record

Met dit record worden de namesevers welke autoritair zijn voor een zone aangeduid. Heel belangrijk: deze records moeten ook voorkomen in de zone-data van de nameserver(s) van een domain hoger[3]. Voor `all-stars.nl.` zijn de NS-records:

```
IN      NS      ns.internl.net.
IN      NS      ns2.internl.net.
```

Merk op dat het eerste en tweede veld leeg zijn. De naam wordt dan de naam van het vorige record (en dat is het SOA-record). Als het TTL-veld wordt weggelaten dan geldt de default TTL en die staat vermeld in het SOA-record (in dit geval 3600 seconden). In dit voorbeeld is sprake van twee NS-records. Zoals uit het SOA-record blijkt is de `ns.internl.net.` de master. Er is verder dus één slave: `ns2.internl.net.`.

4.1.3 Het MX-record

Met MX-records kan voor hosts of domeinen aangeven worden waar mail voor die specifieke machine of dat domein afgeleverd dient te worden. Voor `all-stars.nl.` zijn de MX-records:

```
IN      MX 0    goalie.all-stars.nl.
IN      MX 40   grootstal.nijmegen.inter.nl.net.
IN      MX 100  fallback.nl.uu.net.
```

Merk ook hier op dat het eerste en het tweede veld leeg zijn gelaten. Het laatste veld in een MX-record geeft de host aan die mail voor het betreffende domein kan verwerken. Het is gebruikelijk dat er meedere MX-records voor een domein bestaan. Indien een host niet bereikbaar is voor het afleveren van de mail zijn er nog alternatieven. De regel hierbij is dat een lagere waarde achter het MX-veld een hogere prioriteit aanduid. In dit voorbeeld zal dus altijd eerst mail geprobeerd te worden afgeleverd op de host `goalie.all-stars.nl.`. Lukt dit niet dan komt de host `grootstal.nijmegen.inter.nl.net.` in aanmerking. Lukt ook dat niet dan is er altijd nog de host `fallback.nl.uu.net.`.

4.1.4 Het A-record

Deze records koppelen een naam aan een IP-adres. Voor `all-stars.nl.` heb je bv. het volgende A-record:

```
goalie          IN      A       62.163.53.199
```

Dit record geeft het IP-adres van de host `goalie` in het domein `all-stars.nl.`. M.a.w.: de FQDN `goalie.all-stars.nl` is gekoppeld aan het IP-adres: `62.163.53.199`

[3] die records die voorkomen in de zone-data van het domein hoger worden ook wel glue-records genoemd. Dit vanwege het feit dat door deze constructie de name-servers elkaar kunnen vinden zoals omschreven in het uitgebreide voorbeeld over resolutie in Hoofdstuk 3

4.1.5 Het CNAME-record

CNAME-records worden gebruikt om *logische* namen te koppelen aan *kanonieke* (echte) namen. Voor `all-stars.nl.` is er één CNAME-record:

```
www             IN      CNAME   aanleun.nijmegen.inter.nl.net.
```

Deze koppelt dus de logische naam `www` aan de echte naam `aanleun.nijmegen-.inter.nl.net.`. Dit is overigens niet helemaal juist: ook de naam `aanleun.nijmegen.inter.nl.net.` is een CNAME (voor `vip-52.nijmegen.inter.nl.net.` welke uiteindelijk met een A-record gekoppeld is aan het IP-adres: `193.67.237.182`). Dat mag: CNAMES die verwijzen naar CNAMES: als dit uiteindelijk maar wel een IP-adres oplevert.

In de praktijk kom je vaak tegen dat servers die een specifieke taak hebben gebruik maken van CNAME-records, bv.:

```
www.bla.nl.     IN      CNAME   asterix.bla.nl.
mail.bla.nl.    IN      CNAME   obelix.bla.nl.
ns.bla.nl.      IN      CNAME   idefix.bla.nl.
```

Dit voorkomt dat je allerlei A-records krijgt die gekoppeld zijn aan hetzelfde IP-adres (dus dat bv. `www.bla.nl.` en `asterix.bla.nl.` allebei een A-record hebben met hetzelfde IP-adres) wat de onderhoudbaarheid van de zone-data een stuk onoverzichtelijker zou maken. Eventuele wijzigingen in IP-adressen zijn op deze manier gemakkelijker door te voeren.

4.1.6 Het HINFO-record

Deze komt niet voor in het voorbeeld-domein. Het is voor de juiste werking van DNS ook geen verplicht domein, het verschaft enkel wat additionele informatie. Wegens security reden (redelijk obscuur: er zijn wel andere methoden om achter het OS en de hardware van een host te komen, maar alle beetjes helpen) wordt dit record vaak niet gebruikt. Hier toch een voorbeeld:

```
wolfskuil.nijmegen.inter.nl.net. 86400 IN HINFO "SUN-ULTRA-10" "SOLARIS7"
```

Hieruit blijkt dat de host `wolfskuil.nijmegen.inter.nl.net.` een Sun Ultra-10 is met OS Solaris 7.

4.1.7 Het PTR-record

Deze records verzorgen de *omgekeerde* koppeling: tussen IP-adressen en namen zoals omschreven in Hoofdstuk 3 ("De omgekeerde wereld"). Voor ons voorbeeld bestaat bv. het volgende pointer-record:

```
182.237.67.193.in-addr.arpa.  86400  IN  PTR  vip-52.nijmegen.inter.nl.net.
```

Merk op dat dit record *niet* terug te vinden is in de zone-data van `all-stars.nl.`. Dit record hoort in de `in-addr-arpa` tak van de DNS-boom en valt onder het domein: `237.67.193.in-addr.arpa.`.

Hoofdstuk 5

Tools

Nu de theorie voldoende is uitgelegd wordt het tijd om naar wat tools te kijken die kunnen helpen bij het verkrijgen van informatie m.b.t. DNS-data en bij het opsporen van problemen. De tools die aan bod komen zijn: `host`, `nslookup`, `dig` en `traceroute`. Voor een volledige beschrijving van de syntax en opties van elke tool zie de manpages. Ik zal hier met name een aantal voorbeelden uitwerken.

5.1 host

Deze tool lijkt heel erg veel op `dig` die later wordt behandeld. Meestal wordt `host` alleen gebruikt om even het IP-adres van een host op te zoeken:

```
$ host www.littleplanet.nl
www.littleplanet.nl. is an alias for aanleun.nijmegen.inter.nl.net.
aanleun.nijmegen.inter.nl.net. is an alias for vip-52.nijmegen.inter.nl.net.
vip-52.nijmegen.inter.nl.net. has address 193.67.237.182
```

Maar er kan veel meer mee:

Om de MX-records te verkrijgen doe je:

```
$ host -t mx all-stars.nl
all-stars.nl. mail is handled by 100 fallback.nl.uu.net.
all-stars.nl. mail is handled by 0 goalie.all-stars.nl.
all-stars.nl. mail is handled by 40 grootstal.nijmegen.inter.nl.net.
```

Meer informatie doe je met het vlaggetje `-t any`:

```
$ host -t any all-stars.nl
all-stars.nl. mail is handled by 0 goalie.all-stars.nl.
all-stars.nl. mail is handled by 40 grootstal.nijmegen.inter.nl.net.
all-stars.nl. mail is handled by 100 fallback.nl.uu.net.
```

```
all-stars.nl. SOA ns.internl.net. hostmaster.inter.nl.net. 2001041701
      28800 7200 604800 3600
all-stars.nl. name server ns.internl.net.
all-stars.nl. name server ns2.internl.net.
```

Nog meer informatie krijg je als je `-v` toevoegt. Dit vlaggetje geeft de output (net als `dig` standaard doet) volgens het officiële master file format (het format van de zone-data van de master voor dat domein zoals die als hard-copy in een file staat):

```
$ host -v -t any all-stars.nl
Trying "all-stars.nl."
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61017
;; flags: qr rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 2, ADDITIONAL: 4

;; QUESTION SECTION:
;all-stars.nl.                  IN      ANY

;; ANSWER SECTION:
all-stars.nl.           64477   IN      MX      40 grootstal.nijmegen.inter.nl.net.
all-stars.nl.           64477   IN      MX      100 fallback.nl.uu.net.
all-stars.nl.           64477   IN      MX      0 goalie.all-stars.nl.
all-stars.nl.           64477   IN      SOA     ns.internl.net. hostmaster.inter.nl.net.
                                                2001041701 28800 7200 604800 3600
all-stars.nl.           64477   IN      NS      ns2.internl.net.
all-stars.nl.           64477   IN      NS      ns.internl.net.

;; AUTHORITY SECTION:
all-stars.nl.           64477   IN      NS      ns.internl.net.
all-stars.nl.           64477   IN      NS      ns2.internl.net.

;; ADDITIONAL SECTION:
goalie.all-stars.nl.    64477   IN      A       62.163.53.199
grootstal.nijmegen.inter.nl.net. 86400 IN A     193.67.237.11
ns.internl.net.         86400   IN      A       193.67.237.4
ns2.internl.net.        86400   IN      A       194.229.132.5

Received 313 bytes from 194.229.132.196#53 in 6 ms
```

Let op dat je alleen de *zone*-informatie ziet. Om ook alle hosts in die zone te zien is er het `-l` vlaggetje. Als dit echter gevraagd wordt aan een name-server welke niet autoritair is (bv. de eerste regel in `/etc/resolv.conf` op unix bevat vrijwel zeker een name-server die niet autoritair is) mislukt het commando met de foutmelding `NOAUTH`:

```
$ host -l all-stars.nl
Host all-stars.nl. not found: 9(NOTAUTH)
; Transfer failed.
```

Om dit te vermijden kan een name-server welke wel autoritair is (in dit geval de `ns.internl.net` of de `ns2.internl.net`) achter het commando gezet worden (dit kun je overigens altijd doen als je specifiek een name-server wilt benaderen):

```
$ host -v -l all-stars.nl ns.internl.net
Trying "all-stars.nl."
Using domain server:
Name: ns.internl.net
Address: 193.67.237.4#53
```

```
Aliases:

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23490
;; flags: qr aa ra; QUERY: 1, ANSWER: 11, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;all-stars.nl.                  IN      AXFR

;; ANSWER SECTION:
all-stars.nl.           86400   IN      SOA     ns.internl.net. hostmaster.inter.nl.net.
                                                2001041701 28800 7200 604800 3600
all-stars.nl.           86400   IN      NS      ns.internl.net.
all-stars.nl.           86400   IN      NS      ns2.internl.net.
all-stars.nl.           86400   IN      MX      0 goalie.all-stars.nl.
all-stars.nl.           86400   IN      MX      40 grootstal.nijmegen.inter.nl.net.
all-stars.nl.           86400   IN      MX      100 fallback.nl.uu.net.
goalie.all-stars.nl.    86400   IN      A       62.163.53.199
localhost.all-stars.nl. 86400   IN      A       127.0.0.1
www.all-stars.nl.       86400   IN      CNAME   aanleun.nijmegen.inter.nl.net.
all-stars.nl.           86400   IN      SOA     ns.internl.net. hostmaster.inter.nl.net.
                                                2001041701 28800 7200 604800 3600

Received 15 bytes from 193.67.237.4#53 in 6 ms
```

Wat je nu ziet is een complete *zone-transfer* zoals die ook tussen een slave en een master om de zoveel tijd plaatvindt zodat de slave weer up-to-date wordt.

Overigens staan lang niet alle name-servers een complete zone-transfer toe. Dit vanwege security-redenen (je gooit al je hosts en IP-adressen wel erg makkelijk op straat) en vanwege performance redenen (vooral een grote zone eist een complete transfer wat nogal wat computer-resources opeist wat uiteraard ten koste gaat van de performance van de machine).

5.2 dig

Dit commando lijkt erg veel op `host`, ik zal het wat summier toelichten en de *host-equivalenten* ernaast laten zien.

Een zone transfer doe je met:

```
dig axfr <domein> @<autoritive name-server>
```

Ter vergelijking met host:

```
host -v -l <domein> autoritive name-server
```

Een vertaling van IP-nummer naar naam gaat met:

```
dig -x IP-nummer [@name-server]
```

Ter vergelijking met host:

```
host [-v] -t ptr IP_nummer [name-server]
```

5.3 nslookup

Nog een tool om de DNS te raadplegen. Het opvallende van `nslookup` is dat deze tool ook een interactieve modus kent. Je komt in deze mode als je geen argumenten geeft of als het eerste argument een *hyphen* (-) is en het tweede argument de naam of het IP-adres van een name-server. De niet-interactieve modus wordt gebruikt als het eerste argument de naam of het IP-adres is van een host. Optioneel kan dan ook nog een name-server als tweede argument worden meegegeven.

Het voert wat ver om hier dieper op het gebruik van `nslookup` in te gaan. Met `host` en/of `dig` kom je al een heel eind, voor meer informatie over `nslookup` zie de manual pagina. Overigens is het sowieso verstandig je te beperken tot één tool en die goed te leren gebruiken: allemaal leveren ze op hun manier zo'n beetje dezelfde informatie. Pas als je dingen gaat automatiseren (op unix: in *scripts* gaat zetten) is het raadzaam de daarvoor meest geschikte tool te kiezen.

5.4 traceroute

Deze tool is in dit verhaal een beetje vreemde eend in de bijt. Deze levert namelijk niet zozeer veel DNS-informatie maar met name toont deze tool de bereikbaarheid van hosts en/of netwerken aan. Aangezien het een onmisbaar hulpmiddel is bij het achterhalen van problemen verdient `traceroute` hier toch wat aandacht.

Wat `traceroute` doet is elke hop[1] weergeven zoals die door een IP-pakketje onderweg wordt tegengekomen. Mocht het verkeer ergens ophouden dan is in de output van traceroute goed te zien waar het stokt.
Bekijk als voorbeeld eens de output van een `traceroute` naar `www.ozemail.com.au`:

```
$ traceroute -m 40 www.ozemail.com.au
traceroute to www.ozemail.com.au (203.108.7.50), 40 hops max, 40 byte packets
 1  goalie (192.168.10.11)  0.724 ms  0.642 ms  0.632 ms
 2  a53193.upc-a.chello.nl (62.163.53.193)  19.735 ms  18.800 ms  17.811 ms
 3  Vlan50.ah00rs01.net.upc.nl (212.142.25.65)  38.862 ms  21.632 ms  25.464 ms
 4  Gig6-0.ah00rt04.brain.upc.nl (212.142.28.122)  21.925 ms  22.498 ms  21.270 ms
 5  srp8-0.ah00rt02.brain.upc.nl (212.142.32.50)  23.675 ms  22.073 ms  22.973 ms
 6  srp0-0.am00rt01.brain.upc.nl (212.142.32.1)  28.901 ms  25.921 ms  23.769 ms
 7  srp0-0.am00rt06.brain.upc.nl (212.142.32.44)  29.047 ms  24.062 ms  23.578 ms
 8  nl-ams-rd-01-pos-4-0.chellonetwork.com (213.46.161.53)  23.972 ms  24.989 ms  25.091 ms
 9  nl-ams-rc-01-pos-0-1.chellonetwork.com (213.46.160.13)  28.314 ms  25.916 ms  27.097 ms
10  uk-lon-rc-02-pos-0-0.chellonetwork.com (213.46.160.10)  31.601 ms  34.586 ms  34.122 ms
11  uk-lon-rc-01-pos-1-0.chellonetwork.com (213.46.160.45)  36.289 ms  31.212 ms  35.835 ms
12  uk-lon-rd-01-pos-1-0.chellonetwork.com (213.46.160.54)  36.266 ms  38.492 ms  35.386 ms
13  212.187.151.161 (212.187.151.161)  36.034 ms  34.252 ms  37.161 ms
14  so-2-0-0-0.mp2.lon2.level3.net (212.187.128.53)  36.788 ms  34.772 ms  51.352 ms
15  so-2-0-0-0.mp2.lon2.level3.net (212.187.128.53)  38.851 ms  37.419 ms  35.801 ms
16  212.187.128.162 (212.187.128.162)  105.486 ms  101.059 ms  102.332 ms
17  pos8-0.core2.NewYork1.level3.net (209.247.10.38)  101.590 ms  100.928 ms  104.456 ms
18  uunet-level3-oc12.NewYork1.Level3.net (209.244.160.162)  105.321 ms  102.027 ms  102.465 ms
19  0.at-6-1-0.XL1.NYC9.ALTER.NET (152.63.22.226)  102.612 ms  105.193 ms  101.436 ms
20  0.so-7-0-0.XR1.NYC9.ALTER.NET (152.63.23.138)  102.362 ms  101.537 ms  103.580 ms
21  0.so-3-0-0.TR1.NYC9.ALTER.NET (152.63.22.98)  101.707 ms  101.912 ms  102.845 ms
```

[1] een hop is reis van een IP-pakketje van de ene host naar een direct bereikbare host, zonder tussenkomst van een andere host.

```
22  125.at-5-0-0.TR1.LAX9.ALTER.NET (152.63.5.73)  191.585 ms  192.953 ms  196.896 ms
23  131.ATM5-0.IR1.LAX9.ALTER.NET (152.63.10.238)  192.353 ms  192.586 ms  193.555 ms
24  POS3-0.IR1.LAX12.ALTER.NET (137.39.31.225)  197.717 ms  194.627 ms  196.902 ms
25  342.ATM11-0-0.TR1.SYD2.Alter.Net (210.80.48.138)  435.841 ms  437.373 ms  437.091 ms
26  So-3-3-2.XR1.SYD2.Alter.Net (210.80.48.134)  439.908 ms  438.919 ms  435.586 ms
27  So-3-3-0.XR2.SYD2.Alter.Net (210.80.3.82)  437.195 ms  435.934 ms  437.845 ms
28  So-0-0-0.XR2.SYD3.ALTER.NET (210.80.33.10)  439.819 ms  438.652 ms  463.047 ms
29  412.ATM8-0-0.GW1.SYD3.Alter.Net (210.80.3.158)  436.597 ms  437.091 ms  439.089 ms
30  500.ATM1-0.BORDER3.SYD.OZEMAIL.NET.AU (203.166.16.38)  440.234 ms  438.908 ms  438.484 ms
31  core1-fe4-0-0.syd.ozemail.net.au (203.108.190.153)  443.670 ms  440.247 ms  442.838 ms
32  panda.online.ozemail.net (203.108.7.50)  441.089 ms  440.445 ms  444.205 ms
```

De 3 laatste velden laten de (*roundtrip*) tijd zien die een IP-pakketje over de route tot die host doet. Merk het verschil op in tijd tussen stap 24 en stap 25: je gaat dan ook van Los Angeles naar Sydney (waarschijnlijk via satelliet). Merk ook op dat hier het vlaggetje `-m 40` nodig was om het maximaal aantal hops wat traceroute doet te verhogen naar 40 (default is dit 30)
Dit gaat kennelijk niet mis maar als bv. de host `nl-ams-rd-01-pos-4-0.chello-network.com` down is zul je het volgende zien:

```
[oscar@perl ~]$ traceroute -m 40 www.ozemail.com.au
traceroute to www.ozemail.com.au (203.108.7.50), 40 hops max, 40 byte packets
 1  goalie (192.168.10.11)  0.724 ms  0.642 ms  0.632 ms
 2  a53193.upc-a.chello.nl (62.163.53.193)  19.735 ms  18.800 ms  17.811 ms
 3  Vlan50.ah00rs01.net.upc.nl (212.142.25.65)  38.862 ms  21.632 ms  25.464 ms
 4  Gig6-0.ah00rt04.brain.upc.nl (212.142.28.122)  21.925 ms  22.498 ms  21.270 ms
 5  srp8-0.ah00rt02.brain.upc.nl (212.142.32.50)  23.675 ms  22.073 ms  22.973 ms
 6  srp0-0.am00rt01.brain.upc.nl (212.142.32.1)  28.901 ms  25.921 ms  23.769 ms
 7  srp0-0.am00rt06.brain.upc.nl (212.142.32.44)  29.047 ms  24.062 ms  23.578 ms
 8  * * *
 9  * * *
10  * * *
11  * * *
```

Hier is duidelijk een probleem met de routering na de 7-de hop.

Hoofdstuk 6

Configuratie van BIND

Zoals reeds is opgemerkt is BIND op unix verreweg de meest voorkomende implementatie van DNS. In dit hoofdstuk wordt summier toegelicht wat erbij de configuratie van BIND komt kijken. Hierbij ga ik uit van BIND op een unix systeem. Meer info over BIND is te vinden op: `http://www.isc.org/products/BIND/`.

Alle informatie voor DNS (dus met name de resource records) bevindt zich in asciifiles. Centraal hierbij is het bootfile. In dit file staat o.a. waarvoor de name-server master dan wel slave is, waar de zonedata (resource records) zijn te vinden en waar de root-servers te vinden zijn. Bij het starten van de nameserver (het programma `named`) kan een locatie (en naam) van de boot- file meegegeven worden:

```
# named -b bootfile
```

Indien dit niet wordt gedaan wordt uitgegaan van `/etc/named.boot` als default bootfile.

Voor de name-server `ns.internl.net` (onze master name-server voor de meeste klant domeinen) is de inhoud ongeveer als volgt:

```
# bootfile for named

options {
        directory "/etc/named";
        pid-file        "slave/named.pid";
};

logging {
        channel logging_syslog {
                syslog daemon;
                severity info;
                print-severity yes;
        };
        category default { logging_syslog; };
};
```

```
# ------------------------------------------------------------
# STANDARD ZONES each server should have (in /etc/named)
# ------------------------------------------------------------

zone "." {
        type hint;
        file "cache";
};

zone "localhost" in {
        type master;
        file "localhost";
};

zone "0.0.127.in-addr.ARPA" in {
        type master;
        file "0.0.127.in-addr.ARPA";
};

## ------------------------------------------------------------
## MASTER-ZONES FOR THIS SERVER (in /etc/named)
## ------------------------------------------------------------

zone "132.229.194.in-addr.ARPA" in {
        type master;
        file "132.229.194.in-addr.ARPA";
};

zone "all-stars.nl" in {
        type master;
        file "all-stars.nl";
};

zone "littleplanet.nl" in {
        type master;
        file "littleplanet.nl";
};

etc...

## ------------------------------------------------------------
## SLAVE-ZONES FOR THIS SERVER (in /etc/names/aq.names)
## ------------------------------------------------------------

zone "237.67.193.in-addr.ARPA" in {
        type slave;
        file "aq.names/237.67.193.in-addr.ARPA";
        masters {
```

```
                193.67.237.6;
        };
};

zone "nijmegen.inter.nl.net" in {
        type slave;
        file "aq.names/nijmegen.inter.nl.net";
        masters {
                193.67.237.6;
        };
};

etc...
```

De eerste *directive* `options` zegt: De directory waaronder zich de zone-data bevindt is `/etc/named`. Alle filenamen in de rest van het file zijn dan relatief t.o.v. deze directory. Het proces-id (pid: voor het stoppen van de name-server) wordt bewaard in het file `/etc/named/slave/named.pd`.

De tweede directive `logging` zegt hoe `named` moet loggen. In dit voorbeeld wordt o.a. gezegd dat de logging via `syslog` (een standaard logging programma op unix) moet gebeuren. Er zijn nog veel meer directives en opties maar het voert wat te ver om dat hier uit te werken. Met name het boek van O'Reilly (DNS en BIND, zie literatuurverwijzing) geeft uitgebreide beschrijvingen van alle mogelijkheden.

Hierna volgen, zoals het commentaar (alles achter een #-je is commentaar in `/etc-/named.conf`) al aangeeft,A de zone-data waar iedere name-server standaard master voor is (het commentaar in het voorbeeld is niet helemaal overeenkomstig de gebruikelijke terminologie...). Het file `cache` bevat de namen en IP-adressen van de 13 root name-servers. Iedere name-server moet over dit file beschikken. Hier is deze name-server natuurlijk niet master of slave voor, vandaar het uitzonderlijke type `hint`. Merk ook de `in-addr.arpa` zone-data die verplicht is voor *localhost*.

Dan volgt een lijst met alle zones waar deze name-server master voor is. Op dit moment bevat de `ns.internl.net` ongeveer 2900 van dergelijke entries. Uit de voorbeelden valt op te maken dat de zone-data voor `all-stars.nl` zich bevindt in het file `/etc/named/all-stars`. De inhoud van dit file is zoals weergegeven in Hoofdstuk 4: de voorbeeld zone-data van `all-stars.nl` met de 9 resource-records.

Als laatste volgt nog een lijst met alle zones waar deze name-server slave voor is. Merk hier op dat het IP-adres van de master vermeld wordt. Standaard beschikt deze name-server niet over de zone-data van die domeinen (hij is immers slave) maar na een zone-transfer vanaf de master wordt de zone-data geplaatst in het genoemde file. Voor bv. `nijmegen.inter.nl.net` is dit dus het file `/etc/named/aq.names-/nijmegen.inter.nl.net`.

Hoofdstuk 7

Dokter DNS

In dit hoofdstuk zal ik een aantal veel voorkomende situaties scheppen waarin wat kennis van DNS een en ander kan verduidelijken.

Q: "We willen ons domein graag verhuizen maar blijft onze web-site dan bereikbaar?"

A: In principe is dit geen probleem. Als de oude DNS wordt omgezet zullen uiteindelijk alle name-servers op internet dit gaan snappen. Zolang de web-site die tijd ook nog op het oude IP-adres luistert is er dus weining aan de hand.

Q: "Maar hoelang duurt het dan voordat alle name-servers het nieuwe IP-adres weten?"

A: Vanwege caching is het niet zo dat als er een wijziging in de zone-data plaats vindt dat dan gelijk iedereen op de hoogte is. Wel kun je zien hoelang van een *non-authoritive* antwoord (meestal is een cache-antwoord non-authoritive) nog gebruik gemaakt kan worden. Bijvoorbeeld:

```
$ dig www.all-stars.nl

;; ANSWER SECTION:
www.all-stars.nl.       13h57m11s IN CNAME  aanleun.nijmegen.inter.nl.net.
aanleun.nijmegen.inter.nl.net.  23h59m7s IN CNAME  vip-52.nijmegen.inter.nl.net.
vip-52.nijmegen.inter.nl.net.  23h59m7s IN A  193.67.237.182
```

Hier zie je dat de CNAME nog 13 uur, 57 minuten en 11 seconden uit de cache van de name-server waaraan dit gevraagd wordt komt. Als die tijd op is zal de name-server het weer opnieuw moeten gaan vragen waarna hij dus zal worden voorzien van de nieuwe informatie. Gebruikelijk is het om een TTL van ongeveer 8 uur tot een dag te hanteren. Het kan dus geen kwaad

om, indien een wijziging van tevoren bekend is, dan ook alvast de TTL flink omlaag te schroeven (bv. naar 5 minuten).

Q: "Wij hebben een eigen mail-server maar we ontvangen geen mail!"

A: Een simpele check op de MX-records (`idig mx <domein>`) moet hier voldoende zijn. Om er zeker van te zijn dat de boel niet net verhuisd is kan het geen kwaad om het ook nog even met een authoritive name-server te checken. Dan moet je met `dig ns` wel eerst even achter een authoritive name-server komen:

```
dig ns <domain>
```

en dan:

```
dig mx <domain> @<authoritive name-server>
```

Dit moet in elk geval wel hetzelfde opleveren als de eerdere check. Indien de bewuste mail-server niet in de MX-records voorkomt of niet de laagste MX-waarde heeft zal het inderdaad misgaan. Indien de mailserver wel vermeld is kun je nog even met een `telnet` op de poort waarop de mailserver normaal gesproken luister (25) checken of dat ding überhaupt iets doet:

```
telnet <mail-server> 25
```

Als je wat ziet wat lijkt op een reactie van een mail-server (zie je wel in de output) dan luistert ie in elk geval. Dus dan lijkt er met de DNS en die mailserver niet zoveel mis. Dan wordt het tijd om je af te vragen of die mail wel richting die mail-server gestuurd wordt ("gerelayed"). Dan zal de fout in de config van een andere mailserver liggen waarvan gebruik wordt gemaakt. Is dat er bv. één van InterNLnet dan kan in de logfiles gekeken worden of er bv. regels voorkomen met de phrase "Relaying Denied". Het commando `grep` is dan je vriend.

Komt de mail bv. niet vanaf onze mail-server maar vanaf een andere dan is het in feite niet ons probleem maar het kan geen kwaad nog ff te kijken met een `telnet` op poort 25 van die andere mailserver:

```
$ telnet <onbekende-mail-server> 25
Trying 192.168.10.11...
Connected to onbekende-mail-server
Escape character is '^]'.
220 onbekende-mail-server ESMTP SendMail 8.2.4b
helo me
```

```
250 Ok
MAIL FROM: clinton@whitehouse.gov
250 Ok
RCPT TO: oscar@all-stars.nl
554 <oscar@all-stars.nl>: Relay access denied
```

Dan zit het 'm dus in de config van die mail-server of die mailserver behoort die mail nooit te krijgen (bv. een verkeerde instelling van een smtp-server van een klant).

Q: "Ik kan pingen naar de `login5.inter.nl.net` maar zodra ik ernaar probeer te ftp-en wordt de verbinding direct verbroken?"

A: "Hier kan het zijn dat de reverse lookup mis gaat. De ftp-*daemon* kan checken of het IP-adres waarvandaan geconnect wordt wel *terug geresolved* (IP-adres -> naam) kan worden. Indien dit niet kan weigert de ftp-daemon de connectie. Controle:

```
dig -x <IP-adres>
```

Dat moet dan een antwoord geven.

Q: "Ik ben bekend met A-records maar wat is een AAAA-record??"

A: Dit is een record wat een IPv6-adres (IP versie 6) aan een naam bindt. IP versie 6 is nog in ontwikkeling maar zal uiteindelijk de huidige adressering van 32 bits (IP versie 4) vervangen door een adressering met 128 bits.

Hoofdstuk 8

Nieuwe ontwikkelingen

De ontwikkelingen rond DNS staan niet stil. Belangrijke nieuwe ontwikelingen zijn o.a.:

- Dynamic updates. Tot BIND versie 8 was het altijd noodzakelijk dat een systeembeheerder met de hand de zone-data moest wijzigen. Met behulp van *dynamic updates* is dit proces te automatiseren. Dit is met name handig bij het gebruik van IP-adressen verkregen met DHCP voor hosts met vaste hostnamen. Aangezien met dynamic updates DNS-wijzigingen via een netwerk gaan schuilt hierin wel een security-risico. In RFC 2137 wordt dan ook een versleutelde beveiliging van dynamic updates voorgesteld.
- Notify mechanisme. Vanaf versie BIND 8.1.2 stelt default de master slaves op de hoogte van wijzigingen in de zone-data door een *notify* naar de slaves te sturen.
- Secure DNS (DNSSEC). Al jaren wordt geconstateerd dat gewoon DNS kwetsbaar is. Een gebruiker wil graag zekerheid dat de verkregen DNS informatie ook werkelijk de juiste is. Op dit moment is de ontwikkeling van *secure* DNS, oftewel DNSSEC, volop aan de gang. Met als voornaamste doel het versleutelen van de DNS informatie en authenticatie. Het is waarschijnlijk dat DNSSEC ooit de opvolger wordt van het huidige DNS maar er zijn nog wel wat hindernissen te nemen. Zo zijn een aantal heikele punten:
  - Om DNSSEC te impplementeren is er een behoorlijke uitbreiding van het aantal soorten resource-records nodig. Nieuwe resource-records zijn o.a.:
    * Een SIG-record. Dit record bevat een cryptografische signature voor resource records.
    * Een KEY-record. Dit is een record met public keys van SIG-records.

    De complexiteit van de resource records neemt met DNSSEC dermate toe dat het ondoenlijk wordt om dit niet-geautomatiseerd bij te houden. Er moeten dus tools komen die resource records maken en bijhouden.

  - De performance van DNSSEC laat nog te wensen over. Op dit moment is gewoon DNS ongeveer een factor 3 sneller.

- Incremental zone-transfers. Aangezien, bij grote zones, de belasting van een zone-transfer nogal groot kan zijn is het mogelijk vanaf Bind 8.2 te werken met zgn. "incremental zone-transfers". Alleen de zone-data die gewijzigd is wordt dan opgehaald door de slaves. Dit is echter in een nogal prematuur stadium en er zijn nog niet veel succes verhalen van bekend.

Het mag duidelijk zijn dat hier slechts hele summiere informatie staat over nieuwe ontwikkelingen in DNS. Diegene die nieuwsgierig is geworden en meer wil weten verwijs ik naar de literatuurlijst.

Hoofdstuk 9

Literatuurverwijzing

9.1 boeken

- Titel: DSN en BIND
  Auteur(s): Paul Albitz & Cricket Liu
  Uitgevers: O'Reilly & Associates
  3^e druk.
  ISBN: 1-56592-512-2

- Titel: Het Internet Handboek voor netwerkbeheerders
  Auteur(s): Jeroen Vanheste
  Uitgevers: Addison & Wesley
  ISBN: 90-6789-919-4

9.2 URL's

Hieronder enkele handige URL's om je op weg te helpen:

```
http://www.google.com :)
http://www.icann.net (alles m.b.t. tot namen en nummers).
www.ripe.net (bevat o.a. een whois-database waar naar verantwoordelijke
        instanties voor ip-adressen kan worden gezocht.
http://www.sidn.nl (veel, nederlandse, informatie over domeinen).
http://e-zine.nluug.nl/hold.html?cid=132 (Leerzaam artkel over met
        name DNSSEC).
http://www.nominum.com/resources/whitepapers/bind-white-paper.html
        (de naam zegt al, tevens een aantal handige links).
http://www.isc.org/ (voor de BIND software en nog veel meer
        informatie over BIND).
```