

Linux für Einsteiger

Teil4 - Fortsetzung

DNS

Domain Name Service

Wer soll sich alle IP-Adressen merken (es gibt rund 4 Mrd Ipv4-Adressen bzw. 32Bit und demnächst noch erheblich mehr Ipv6 128Bit etwa $6 \cdot 10^{24}$ IP-Adressen pro m^2 !!!)?

Also schafft man sich eine Möglichkeit die Zahlen durch Namen zu ersetzen, und genau da setzt der DNS ein.

Dabei ist das gesamt Netz hierarchisch aufgebaut.

Ganz oben stehen die ROOT Name-Server (zu finden in / **var/named/named.ca**).

Diese sind die letzten Instanzen, wenn alle anderen Nameserver nicht mehr weiter wissen.

Im Grunde läuft es so, dass wenn eine DNS keine Auflösung machen kann, erst einmal ein anderer gefragt wird (dieser kann festgelegt werden). Wenn dieser es auch nicht weiß, dann fragt der einen übergeordneten DNS und so weit bis zu den ROOT Name-Servern.

Wenn keiner die aufgerufene Seite kennt, gibt es im Browser, oder wo auch immer man einen externen Rechner über einen Namen anruft, den es nicht gibt, eine Fehlermeldung.

Allerdings muss nicht hinter jeder IP-Adresse ein Namen stehen. So sind z.B. gewisse Server, die geklaute Softwareprogramme und/oder Filme usw. anbieten, oft nur über eine IP-Adresse zu erreichen.

Welchen DNS man gebrauchen soll ?

Normalerweise braucht man sich darüber keine Gedanken machen, wenn man sich über eine Modem einwählt, da der Provider alles zur Verfügung stellt, was man zur Teilnahme am Internet braucht. Manchmal stellt man aber fest, dass es lange dauert bis eine Seite beginnt sich aufzubauen (dies dann aber sehr schnell). Dies kann an dem DNS liegen, den der Provider zur Verfügung stellt. Dessen DNS wußte möglicherweise nicht welche IP-Adresse hinter einem gesuchten Namen steckte und musste seinerseits erst einmal einen anderen DNS fragen, was zu einer Verzögerung geführt hat.

Den DNS kann man aber in der Datei **/etc/resolv.conf** ändern. Um sich einen neuen DNS zu wählen, kann man mal versuchen den DNS herauszufinden, der eine viel besuchte Seiten auflöst oder der von einer Uni verwendet wird.

Z.B. auf einer Konsole mal

dig www.uni-goettingen.de

eingeben und man erhält folgende (gekürzte) Ausgabe.

;; AUTHORITY SECTION:

```
uni-goettingen.de. 86400 IN NS gwdu01.GWDG.de.  
uni-goettingen.de. 86400 IN NS ws-kar1.win-ip.dfn.de.
```

;; ADDITIONAL SECTION:

```
gwdu01.GWDG.de. 86400 IN A 134.76.10.46  
ws-kar1.win-ip.dfn.de. 63922 IN A 193.174.75.154
```

Hier steht im wesentlichen, dass die Domain **uni-goettingen.de** durch den NS (Nameserver) **gwdu01.GWDG.de** aufgelöst wird, der die IP-Adresse **134.76.10.46** besitzt. Wenn man diesen Server erreicht, was mit dem Befehl

ping 134.76.10.46

geprüft werden kann, kann man diesen für sich als Nameserver verwenden. Der Vorteil ist, dass dieser NS sehr wahrscheinlich von vielen Mitarbeitern der Uni verwendet wird und somit über einen großen Cache verfügt. DNS verwenden diesen Cache als einen Speicher für Seiten, die sie bereits früher einmal aufgelöst haben, um sofort eine Antwort zu haben, und nicht erst lange zu suchen.

Ach ja übrigens, der Port der dabei benutzt wird, ist der Port 53 in UDP (also ohne zu testen ob die Antwort tatsächlich angekommen ist oder nicht, wie das bei TCP/IP der Fall ist. DNS muß schnell sein. Es geht zwar auch über Port 53 TCP/IP , was aber länger dauert.)

Für kleine Netzte gibt es, statt einen ausgewachsenen Name-Server zu betreiben, die Möglichkeit in einer Datei der **/etc/hosts** eine einfache Zuordnung zwischen Namen und IP-Adresse zu machen. Diese Einträge sehen dann etwa wie folgt aus:

192.168.0.15 laptop laptop.matthias-riepe.de

Diese Datei muß allerdings auf jeden Rechner innerhalb des Netzwerkes vorliegen, was den administrativen Aufwand bei einer kleinen Anzahl noch überschaubar läßt, aber so ab 10 Rechnern ist es wohl schon o.k. einen Nameserver aufzubauen. Hier braucht man dann nur einmalig auf allen Rechnern die **/etc/resolv.conf** anpassen und dann den Nameserver aufbauen, in dem nur noch eine Datei angepasst werden muß.

Den Aufbau dieses Servers wollen wir uns jetzt mal etwas genauer ansehen.

Unter RedHat oder FedoraCore-Systemen sind dabei die Dateien **/etc/named.conf** und das Verzeichnis **/var/named** entscheidend.

Die Datei **/etc/named.conf** :

```
options                                {    directory "/var/named";
                                        forwarders { 134.76.10.46; };
                                        forward only;
};
zone "."                                {    type hint;
                                        file "named.ca";
};
zone "localhost"                       {    type master;
                                        file "localhost.zone";
};
zone "0.0.127.in-addr.arpa"             {    type master;
                                        file "named.local";
};
zone "15.0.168.192.in-addr.arpa"       {    type master;
                                        file "db.15.0.168.192.in-
addr.arpa"; };
zone "matthias-riepe.de"                {    type master;
                                        file "db.matthias-riepe.de";
};
```

Dabei sagt :

- options** welchen DNS soll ich fragen wenn ich keine Antwort kenne
- forward only** verhindert das die ROOT-NS gefragt werden (die haben wichtigere Aufgaben als irgendwelche Fragen eine Internetusers zu beantworten)
- zone "."** schau auch wenn es nötig wird in den ROOT-Servern nach
- zone "localhost"** brauch ich, um zu entscheiden wer ich eigentlich bin Name --> IP
- zone "0.0.127.in-addr.arpa"** und natürlich auch umgekehrt IP --> Name
- zone "matthias-riepe.de"** siehe den vorletzten Eintrag
- zone "15.0.168.192.in-addr.arpa"** siehe den vorletzten Eintrag

jeder der letzten 5 Einträge zielt auf eine Datei im Verzeichnis /
var/named mit dem Eintrag

file "...";

und zu denen kommen wir jetzt.

named.ca ist der ROOT-File, der nicht verändert werden sollte.

localhost.zone und **named.local** werden bei der Installation erzeugt und lösen localhost auf und umgekehrt. Aber jetzt wird es interessant wenn wir eine Domäne verwalten wollen und zwar in diesem Falle **matthias-riepe.de**. Sehen wir uns diese Datei mal genauer an...

\$TTL 86400

@ IN SOA laptop.matthias-riepe.de. root.laptop.matthias-riepe.de. (

42 ; serial (d. adams)

3H ; refresh

15M ; retry

1W ; expire

©BaLiSta Hamburg – Matthias Riepe

	1D)			; minimum
@		IN	NS	laptop.matthias-riepe.de.
@		IN	A	192.168.0.15
laptop.matthias-riepe.de.		IN	A	192.168.0.15
www		IN	A	192.168.0.15

ganz wichtig und immer als erstes **\$TTL** (TimeToLive <SEKUNDEN>) für Name-Server, die cachten, wie lange diese, diesen RR (Resource Record) behalten sollen. Das @ steht hier für die Zone **matthias-riepe.de** in der SOA (Start of Authority) **laptop.matthias-riepe.de**, wobei bei Beschwerden doch bitte root@laptop.matthias.de angemacht werden soll. Ja der erste Punkt in

root.laptop.matthias-riepe.de. ist eigentlich ein @, das aber schon eine andere Bedeutung hat. Der Punkt am Ende von **de.** ist ebenfalls richtig.

Jeder RR hat eine Seriennummer, hier 42, danach folgt ein Kommentar. Diese Zahl ist für Slave DNS wichtig, die sich nur anhand diese Nummer orientieren, um festzustellen, ob sich ein RR geändert hat, also wenn man intelligent ist, hier am besten ein Datum hinterlegen.

dann **3H**(ours) Kommentar : refresh --> erst nach dessen Ablaufen sollen Slave NS wieder beim Master nachfragen, ob sich irgendetwas geändert hat oder aber auch nicht (Seriennummer)

15M(inutes) Kommentar : retry --> Lieber Slave NS warte doch bitte so lange nachdem ein refresh schief gegangen ist.

1W(eek) Kommentar : expire --> Lieber Slave NS nutze doch bitte so lange den Eintrag auch wenn alle refreshs schief gegangen sind erst danach lösche ihn.

1D) Kommentar : minimum --> Zeit, die der NS die Antwort "No such host" speichern soll, um es danach erneut zu versuchen.

So und nun kommt die eigentlich Auflöse-Arbeit:

@		IN	NS	laptop.matthias-riepe.de.	
@		IN	A	192.168.0.15	
laptop.matthias-riepe.de.			IN	A	192.168.0.15
www			IN	A	192.168.0.15

das @ hatte wir schon

also übersetzen wir doch mal

matthias-riepe.de. ist **IN N(ame)S(erver) laptop.matthias-riepe.de.** zu finden. Der . am Ende ist richtig.

matthias-riepe.de. ist **IN** der **A(dresse) 192.168.0.15** zu finden
oder auch nur **laptop** ist **IN** der **A(dresse) 192.168.0.15** zu finden

www oder auch **www.matthias-riepe.de.** ist **IN** der **A(dresse) 192.168.0.15** zu finden

merkwürdig hinter der Adresse **192.168.0.15** ist **laptop.matthias-riepe.de** und **www.matthias-riepe.de** zu finden. Ja das geht. So ist es z.B. möglich sogenanntes Multihosting zu veranstalten und unter einer IP mehrere Domains zu hosten (es lebe HTTP 1.1). Hätte ich jetzt einen Web-Server laufen könnte ich ihn über einen Browser erreichen.

Natürlich gibt es auch noch andere Optionen, die man setzen kann, z.B. einen einfachen Loadbalancer einrichten. Dieser arbeitet aber nur nach dem Round Robin Prinzip

(1. Anfrage bearbeitet 1.Server , 2. Anfrage der 2.Server , die 3.Anfrage wieder der 1.Server usw.)

Anfragen, die an Mailserver gerichtet werden, können zielgerichtet über sogenannte **MX** Einträge weitergeleitet werden.

nun schauen wir uns mal die Datei **db.15.0.168.192.in-addr.arpa** an

\$TTL 86400

@ IN SOA laptop.matthias-riepe.de. root.laptop.matthias-riepe.de. (

46 ; serial (d. adams)

3H ; refresh

15M ; retry

1W ; expiry

1D) ; minimum

@ IN NS laptop.matthias-riepe.de.

15.0.168.192.IN-ADDR.ARPA. IN PTR laptop.matthias-riepe.de.

(PoinTerRecords)

Interessant ist hier eigentlich nur der letzte Eintrag, der eine IP-Adresse in einen Namen verwandelt.

So jetzt das ganze gespeichert und die **/etc/resolv.conf** noch angepasst, so dass sie jetzt so aussieht:

```
search matthias-riepe.de
nameserver 192.168.0.15
```

den letzten Eintrag kann man auch weglassen oder nur ihn verwenden und den Server gestartet

mit einem
dig matthias-riepe.de
erhält man nun folgende Antwort (verkürzt):

```
:: ANSWER SECTION:
```

```
matthias-riepe.de. 86400 IN A 192.168.0.15
```

```
:: AUTHORITY SECTION:
```

```
matthias-riepe.de. 86400 IN NS laptop.matthias-riepe.de.
```

```
:: ADDITIONAL SECTION:
```

```
laptop.matthias-riepe.de. 86400 IN A 192.168.0.15
```

```
:: Query time: 49 msec
```

```
:: SERVER: 127.0.0.1#53(127.0.0.1)
```

Hurra der Nameserver läuft.