Die Namensauflösung im Internet

Michael Dienert

7. März 2012

Inhaltsverzeichnis

1 Namen und Adressen im Internet

Der Datenverkehr im Internet arbeitet mit auf dem Internet-Protocoll basierenden Diensten wie z.B. TCP, UDP, FTP usw.

Hostrechner werden dabei über ihre 32-bit lange IPv4-Adresse adressiert. In Zukunft werden vermehrt 128-bit lange IPv6-Adressen hinzukommen.

Da man sich solch lange Zahlen nur schwer merken kann und man ihnen auch absolut nicht ansieht, was für Dienste der entsprechende Rechner anbietet und wo er steht,

werden schon immer aussagekräftige Namen als *Hostnamen* (Name eines Rechners) vergeben.

Einordnung ins OSI-Schichtenmodell:

IP-Adressen - Mit den numerischen 32-Bit (IPv4) und zukünftig 128-Bit (IPv6) - Adressen werden *Pakete* auf der **Schicht 3** zugestellt.

> IP-Adressen gehören zur Schicht 3 (Network)

Namen - Namen werden bei der *Bedienung der Anwendungssoftware* (z.B. Mozilla) verwendet, um eine bestimmte **Internet-Resource** (z.b. einen http-Server) anzusprechen.

> Namen gehören zur Schicht 7 (Application)

Vorteile der Namen:

- leichte Merkbarkeit
- aussagekräftiger als Zahlen. Beispiel: **www.wara.de** → Webserver der Walther-Rathenau-Schule in Deutschland
- Namen sind langlebiger. Wird z.B. bei einem Providerwechsel eine neue IP-Adresse vergeben, kann der Namen dennoch beibehalten werden.

2 Hostnamen und DNS

In den Urzeiten des Internets wurden die Zuordnungen von **Rechnernamen** und IP-Adresse in einer zentralen Datei gespeichert, die auf allen Rechnern vorhanden sein musste und die ständig abgeglichen wurde.

Hier ein Beispiel dieser Datei (/etc/hosts):

```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1        localhost
255.255.255.255        broadcasthost
::1             localhost
141.31.147.114  msv.wara.de msv
141.31.147.113  caponova.wara.de caponova capo
```

Ab einer gewissen Anzahl an Hostrechnern im Netz ist wird dieses Verfahren jedoch äusserst aufwändig und wurde daher 1983 von *Paul Mockapetris* durch das **Domain Name System** erweitert. Damit wurde die Datei /etc/hosts jedoch nicht abgeschafft. Wie man im obigen Beispiel sehen kann, kann man sie immer noch zur Auflösung von Namen in IP-Adressen verwenden.

Vorteile von /etc/hosts gegenüber DNS:

- funktioniert auch ohne Netzwerkverbindung, z.b. beim Starten eines Rechners
- schnelle Antwort auf Anfragen
- einfacher Dateiaufbau, leicht zu konfigurieren

3 Namensauflösung auf einem Hostrechner, Name Resolver

`DNS` und `/etc/hosts` werden gemeinsam für die Namensauflösung verwendet.

Sie werden allerdings nicht direkt von den Anwendungsprogrammen abgefragt, sondern die Anfragen werden von einem Zwischenprogramm erledigt, dem sog. *Name Resolver* oder einfach nur *Resolver*.

> Auf Unix-Systemen ist der Resolver eine *C-Programmbibliothek*. Diese enthält Funktionen, die von einem Anwendungsprogramm aufgerufen werden, für das Programm die Anfrage erzeugen und die Antwort an das Programm zurückliefern. Der Resolver kann so konfiguriert werden, dass Anfragen sowohl an `/etc/hosts` als auch an einen bestimmten Nameserver gerichtet werden.

Dabei muss die IP-Adresse des Nameservers auf dem lokalen Rechner hinterlegt sein. Sie steht auf Unix-Systemen in der Datei `/etc/resolv.conf`.

Hier ein Beispiel dieser Datei:

```
#Datei /etc/resolv.conf

nameserver 10.16.1.1
nameserver 129.143.2.4
nameserver 129.143.2.1
```

4 DNS

Das DNS ist eine **baumförmig strukturierte, verteilte Datenbank**.

4.1 Struktur des DNS

Der DNS-Namensraum hat eine Baumstruktur wie in Abb. 1 dargestellt. Dabei entsprechen die Blätter des Baums den Hostnamen.

Dargestellt sind nur 3 Hierarchieebenen, aber es kann auch noch weitere Unterebenen geben (Beispiel: www.example.co.uk.)

4.2 Syntax von DNS-Namen

Die Knoten des Baums werden *Label* genannt. Es gelten folgende Regeln:

- das Label der Root-Domain ist eine **leere Zeichenkette**.

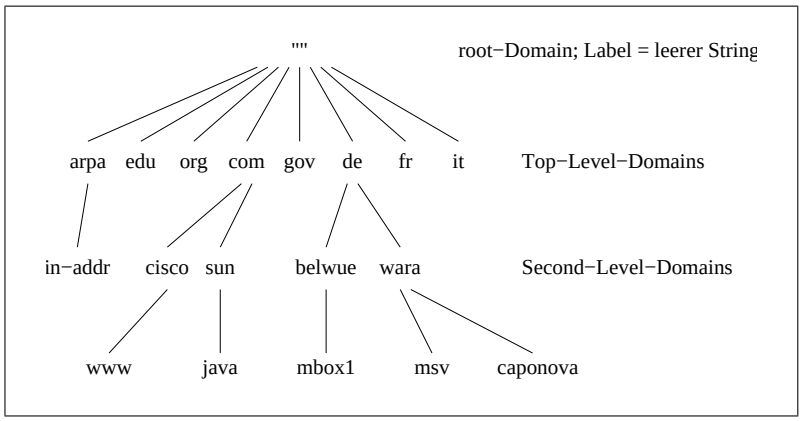

Abbildung 1: DNS-Baum

- ein Label besteht aus maximal 63 Zeichen aus dem 7-bit ASCII-Zeichensatz. Dabei dürfen nur **Buchstaben**, **Zahlen** und das **Minuszeichen** verwendet werden! Gross- und Kleinschreibung wird nicht unterschieden.

  Der Unterstrich darf nicht verwendet werden!

- ein Label darf nicht mit dem Minuszeichen beginnen oder enden.

- folgende Top-Level-Domain Labelnamen sind erlaubt:

  ISO-3166 Ländercodes : bestehen aus 2 Zeichen; `de, fr, it, pl, ...`
  Ausnahme: der ISO-Ländercode von Grossbritannien ist GB, verwendet wird im DNS jedoch uk. Ausserdem werden in GB nur bestimmte Second-Level-Domains vergeben (org.uk., co.uk., usw.)

  generische TLDs : bestehen aus 3 und mehr Zeichen; z.B. `com, edu, gov, mil, net, org, int, arpa, biz`

- Ein *vollqualifizierter Domain-Name* (full qualified domain-name, **FQDN**) gibt den **absoluten Pfad** von der root-Domain bis zur gewünschten Domäne an. Die Labels werden dabei durch Punkte getrennt. Beispiel:

  ```
  msv.wara.de.
  ```

  Der Punkt vor dem root-Domain-Label (also der Punkt nach "de") darf weggelassen werden, da die Angabe auch ohne Punkt eindeutig ist. Strenggenommen gehört der Punkt vor dem root-Domain-Label jedoch zu einem FQDN und die Programme, die den Namen in eine Internet-Adresse auflösen ergänzen den Punkt automatisch.

Ein FQDN darf nicht länger als 255 Zeichen sein!

4.3 Root-Server

Die wichtigsten DNS-Server im Internet sind die sog. **Root-Server**. Davon gibt es insgesamt 13 Stück.

> Die Root-Server enthalten die Namen und Adressen **aller** Top-Level-Domains.

6 dieser Root-Server sind weltweit auf mehrere Maschinen verteilt und werden über *Anycast* erreicht. Die restlichen Server stehen in den USA.

Bei Anycast haben mehrere Rechner die gleiche IP-Adresse (!). Welcher der Rechner erreicht wird hängt einzig und allein vom Routing-Protokoll ab.

Die Root-Server haben folgendes Namensschema:

`<buchstabe a-k>.root-servers.net.`

Die Adressen der Root-Server kann man mit dem Kommando dig ermitteln.

Versuch: ermittle die Adressen aller root-server:

```
dig a.root-servers.net

; <<>> DiG 9.3.5-P2 <<>> a.root-servers.net
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39725
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;a.root-servers.net.            IN      A

;; ANSWER SECTION:
a.root-servers.net.     65976   IN      A       198.41.0.4

;; Query time: 12 msec
;; SERVER: 192.168.2.1#53(192.168.2.1)
;; WHEN: Mon Mar  2 01:25:47 2009
;; MSG SIZE  rcvd: 52
```

4.4 Zone und Domäne

4.4.1 Domäne

> Eine Domäne ist die Menge aller Rechner (Hosts), die unter einem *gemeinsamen Namen* zusammengefasst sind.
> D.h. zur Root-Domain gehören **alle** Rechner im Internet, zur Domäne `wara.de.` gehören alle Rechner, deren FQDN mit `wara.de.` endet.

Im Sprachgebrauch wir mit Domäne meistens eine Second-Level-Domain gemeint (`wara.de.` , `fachinformatiker.de.`).

4.4.2 Zone

> Eine Zone ist die Menge aller Rechner, bei denen die Zuordnung IP-Adresse - FQDN in einer gemeinsamen Datenbank gespeichert ist.
> Diese Datenbank liegt auf speziellen Rechnern, die man **Nameserver** nennt.
> Man kann Zone nun auch so definieren: alle Rechner, für deren Namensauflösung ein Nameserver zuständig ist.

Zone und Domäne stimmen meistens überein. Man kann in einer Domäne, für eine evtl. vorhandene Unterdomäne jedoch einen eigenen Nameserver einsetzen, so dass man in einer Domäne zwei oder mehr Zonenen hat. Hat die Domäne keine Unterdomänen, sind Zone und Domäne identisch (Abb. 2).

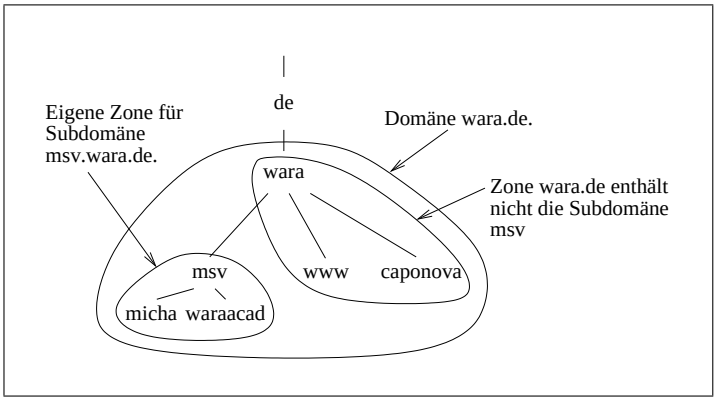

Abbildung 2: Eine Domäne mit zwei Zonen

4.5 Nameserver

Da für die Funktion des DNS die Nameserver mit ihrer Datenbank entscheidend sind, betreibt man pro Zone meistens zwei Rechner als Nameserver. Der **Primary Nameserver** liest die Datenbank mit den Namen und den Adressen aus einer sog. **Zonendatei** ein.

Der Secondary Nameserver holt sich die Datenbank über eine Netzwerkverbindung vom Primary Nameserver.

4.6 Die Zonendatei

In RFC-1035 ist der Aufbau einer Zonendatei beschrieben. Der Nameserver `named` arbeitet z.B. mit einer Zonendatei, die so aufgebaut ist. Windows-Versionen mit ADS verwenden ein nicht lesbares Binärformat.

4.6.1 Das @-Zeichen

In einer Zonenedatei steht das @-Zeichen als Platzhalter für die aktuelle Zone, wenn die Variable `$ORIGIN` nicht gesetzt wurde.

Enthält also die Datei `named.conf` z.B. folgenden Eintrag:

```
zone "ip6uebung.de"{
        type master;
        file "/etc/bind/db.ip6uebung";
};
```

hat das @-Zeichen *innerhalb* der Datei `db.ip6uebung` den Wert `ip6uebung.de` .
Wird die Variable `$ORIGIN` gesetzt wird immer da, wo das @-Zeichen auftritt ihr Wert eingefügt.

4.6.2 Resource Records

Eine Zonendatei nach RFC-1035 ist eine Liste von Datensätzen, die man **Resource Records** nennt.
Hier ein Beispiel:

```
$ORIGIN .
$TTL 3600       ; 1 hour
wara.de                 IN SOA  caponova.wara.de. postmaster.wara.de. (
                                2004031298 ; serial
                                10800      ; refresh (3 hours)
                                3600       ; retry (1 hour)
                                604800     ; expire (1 week)
                                38400      ; minimum (10 hours 40 minutes)
                                )
                        NS      caponova.wara.de.
                        MX      10 caponova.wara.de.
$ORIGIN wara.de.
$TTL 7200       ; 2 hours
caponova                A       10.16.1.1
msv                     A       10.16.1.101
r025-01                 A       10.20.25.101
r025-02                 A       10.20.25.102
r025-03                 A       10.20.25.103
r025-04                 A       10.20.25.104
r025-05                 A       10.20.25.105
r025-06                 A       10.20.25.106
r025-07                 A       10.20.25.107
r025-08                 A       10.20.25.108
r025-09                 A       10.20.25.109
r025-10                 A       10.20.25.110
```

Die einzelnen Resource-Record-Einträge haben folgendes Format:

```
name [timeToLive/s] [class] type data
```

- `$ORIGIN`: um nicht jedesmal den kompletten FQDN schreiben zu müssen, wird die Variable `$ORIGIN` gesetzt. Mit dem Wert von `$ORIGIN` werden alle Namen, **die nicht mit einem Punkt enden** zu einem FQDN ergänzt. Taucht irgendwo in der Zonendatei das Zeichen @ auf, wird es durch den aktuellen Wert von `$ORIGIN` ersetzt. Wurde `$ORIGIN` noch nicht gesetzt, hat es den Wert, der in der Datei `named.conf` für die Zone vergeben wurde (aktuelle Zone). Vergleiche auch Kap. 4.6.1

- Die *time-to-live* (TTL) gibt die Zeit in Sekunden an, die dieser Eintrag von einem Caching-Nameserver im Cache gehalten werden daraf. Nach Ablauf von TTL

muss der Caching-Nameserver solche Einträge bei einem authoritativen Nameserver neu anfragen.

- Class hat fast immer den Wert `IN` , das steht für InterNet.
- Die wichtigsten Typen sind:

  SOA - Start of Authority = Beginn einer Zone

  NS - verweist auf die Nameserver der Zone oder auf die Nameserver einer Sub-Zone

  A - Abbildung Name zu IPv4-Adresse

  AAAA - Abbildung Name zu IPv6-Adresse

  CNAME - Canonical Name, das ist der **eigentliche** Name eines Hosts. Der Name, der am Anfang eines CNAME-RRs steht ist somit ein Alias-Name.

  MX - Mail Exchange; Name des Mailservers, der für diese Zone die Mail verwaltet.

  PTR - wird bei der Auflösung Name in IP-Adresse verwendet

4.7 Suchvorgang bei der Namensauflösung

Es gibt mehrere Varianten bei der Suche nach IP-Adressen:

iterative Suche - bei der iterativen Suche wird von oben nach unten gesucht: der für den Client zuständige Nameserver fragt zunächst beim Root-Server an, dieser gibt die Adresse des zuständigen Top-Level-Domain-Servers zurück, eine Anfrage dort wiederum liefert die Adresse des zuständigen Second-Level-Domain-Servers usw., bis schliesslich der für die angefrage Domäne zuständige Server die IP-Adresse liefert.

Hier ein Beispiel:

```
; <<>> DiG 9.3.5-P2 <<>> +trace msv.wara.de.
;; global options:  printcmd
.                       70013   IN      NS      a.root-servers.net.
.                       70013   IN      NS      b.root-servers.net.
.                       70013   IN      NS      c.root-servers.net.
.                       70013   IN      NS      d.root-servers.net.
.                       70013   IN      NS      e.root-servers.net.
.                       70013   IN      NS      f.root-servers.net.
.                       70013   IN      NS      g.root-servers.net.
.                       70013   IN      NS      h.root-servers.net.
.                       70013   IN      NS      i.root-servers.net.
.                       70013   IN      NS      j.root-servers.net.
.                       70013   IN      NS      k.root-servers.net.
.                       70013   IN      NS      l.root-servers.net.
.                       70013   IN      NS      m.root-servers.net.
;; Received 500 bytes from 192.168.2.1#53(192.168.2.1) in 64 ms

de.                     172800  IN      NS      A.NIC.de.
de.                     172800  IN      NS      L.DE.NET.
de.                     172800  IN      NS      Z.NIC.de.
de.                     172800  IN      NS      S.DE.NET.
de.                     172800  IN      NS      C.DE.NET.
de.                     172800  IN      NS      F.NIC.de.
;; Received 287 bytes from 198.41.0.4#53(a.root-servers.net) in 224 ms

wara.de.                86400   IN      NS      dns1.belwue.de.
wara.de.                86400   IN      NS      dns3.belwue.de.
;; Received 106 bytes from 194.0.0.53#53(A.NIC.de) in 92 ms

msv.wara.de.            86400   IN      CNAME   msv2.wrg.fr.bw.schule.de.
msv2.wrg.fr.bw.schule.de. 86400 IN      A       141.31.147.114
bw.schule.de.           86400   IN      NS      dns3.belwue.de.
bw.schule.de.           86400   IN      NS      arbi.Informatik.Uni-Oldenburg.de.
bw.schule.de.           86400   IN      NS      dns1.belwue.de.
bw.schule.de.           86400   IN      NS      artemis.rus.uni-stuttgart.de.
;; Received 258 bytes from 129.143.2.10#53(dns1.belwue.de) in 64 ms
```

rekursive Suche - bei der rekursiven Suche fragt der Nameserver des Clients den nächsten, höhergelegenen Nameserver. Von diesem erhält er die Antwort. Dabei sind 2 Fälle möglich:

1. der angefrage NS kennt die IP des angefragten Namens und liefert diese zurück
2. der angefrage NS kennt die IP nicht und fragt deshalb seinerseits beim nächsthöheren NS nach. Sobald er von diesem Antwort erhält, liefert er den Namen zurück.

Caching - da das rekursive und iterative Abfragen langsam ist und Resourcen verbraucht, werden einmal angefragte Namen in einem Cache gespeichert, so dass bei neuerlicher Anfrage sofort geantwortet werden kann. Damit bei sich ändernden IP-Adressen-Namenszuordungen der Cache keine falschen Informationen enthält, ist die Lebensdauer der Cache-Einträge durch die TTL (s.o.) begrenzt.

4.8 Rückwärts-Auflösung: Name zu Adresse suchen

4.8.1 Beispiel einer Zonenedatei für die Auflösung IP-Adresse → DNS-Name (Reverse Zone):

```
$ORIGIN .
$TTL 3600       ; 1 hour
10.in-addr.arpa         IN SOA  caponova.wara.de. wara.de. (
                                2004031238 ; serial
                                3600       ; refresh (1 hour)
                                1800       ; retry (30 minutes)
                                604800     ; expire (1 week)
                                3600       ; minimum (1 hour)
                                )
                        NS      caponova.wara.de.
$ORIGIN 1.16.10.in-addr.arpa.
$TTL 7200       ; 2 hours
1                       PTR     caponova.wara.de.
101                     PTR     msv2.wara.de.
254                     PTR     ipcop.wara.de.
9                       PTR     rztest-01.wara.de.

--------------------
```

4.8.2 Testen der Rückwärtsauflösung mit dem Kommando `dig`

```
dig @129.143.2.10 -x 141.31.147.114 ptr

; <<>> DiG 9.3.5-P2 <<>> @129.143.2.10 114.147.31.141.in-addr.arpa. ptr
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57781
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 2

;; QUESTION SECTION:
;114.147.31.141.in-addr.arpa.   IN      PTR

;; ANSWER SECTION:
114.147.31.141.in-addr.arpa. 86400 IN   PTR     msv2.wrg.fr.bw.schule.de.

;; AUTHORITY SECTION:
147.31.141.in-addr.arpa. 86400  IN      NS      dns3.belwue.de.
147.31.141.in-addr.arpa. 86400  IN      NS      amadeus.informatik.ba-stuttgart.de.
147.31.141.in-addr.arpa. 86400  IN      NS      dns1.belwue.de.

;; ADDITIONAL SECTION:
dns1.belwue.de.         86400   IN      A       129.143.2.10
dns3.belwue.de.         86400   IN      A       131.246.119.18

;; Query time: 65 msec
;; SERVER: 129.143.2.10#53(129.143.2.10)
;; WHEN: Mon Mar  2 01:36:52 2009
;; MSG SIZE  rcvd: 206
```