Domain Name Service

Kathleen Haucke
Communication Networks Seminar - Wintersemester 2004 / 2005
Hasso-Plattner-Institut für Softwaresystemtechnik
kathleen.haucke@student.hpi.uni-potsdam.de

Abstract

Von den experimentellen Anfängen des ARPANET bis heute waren die Konsistenz der gespeicherten Namensinformationen, sowie die Überschaubarkeit der Verwaltungseinheiten stets die Hauptkriterien für die Bewertung eines Netzwerk - Namensdienstes. Während man diesen Anforderungen in den siebziger Jahren noch durch den Einsatz einer einzigen Datei gerecht werden konnte, werden heute tausende von Name Servern überall auf der Welt benötigt, um den mittlerweile hierarchisch aufgebauten Domain Name Space zu verwalten. Mit enormem Weitblick wurde das Domain Name System bereits 1984 von P. Mockapetris entworfen und seine Idee, sowohl Daten als auch Verantwortung zu delegieren hat sich durchgesetzt, obwohl dadurch der Ablauf der Namensauflösung auf den ersten Blick umständlich erscheint. Das vorliegende Paper soll dazu beitragen auch jene Abläufe und Strukturen zu verstehen, die dem Internetnutzer sonst verborgen bleiben.

Keywords

Domain, Domain Name Space, Namensauflösung, Name Server, Resolver, Ressource Record

Inhaltsverzeichnis

1 Einleitung und Motivation

Trotzdem es vom technischen Standpunkt her nahe liegend ist, einen Computer über seine physikalische Adresse (die MAC – Adresse der Netzwerkkarte) innerhalb eines Netzwerkes anzusprechen, wurden bereits in den Anfängen der Computervernetzung einfachere Namen für die Hosts verwendet. Denn selbst als es noch sehr wenige vernetzte Computer gab, konnten Namen wie, „Annas Computer“ oder „Station XYZ“ besser vom Menschen erinnert werden als lange Zahlenfolgen (wie bei einer IP – Adresse) oder zufällige Kombinationen aus Ziffern und Buchstaben. Um diesen Komfort für die Benutzer eines Netzwerkes anbieten zu können ist es notwendig einen Mechanismus zu implementieren der Namen (ASCII – Strings) auf physikalische Adressen abbildet. Im ARPANET, dem Vorläufer des heutigen Internet, wurde vom Network Information Center (NIC) zu diesem Zweck eine zentrale Datei gepflegt, die alle Zuordnungen von Hostnamen zu ihren physikalischen Adressen enthielt. Diese Datei wurde HOSTS.TXT genannt und nach ihr auch der Mechanismus der Namensauflösung. Für die Organisation eines relativ kleinen Netzwerkes, mit nur einigen hundert, technisch versierten, Nutzern war dieses Verfahren ausgesprochen effizient, da es nur notwendig war die aktuelle Version der HOSTS.TXT zu laden, um an die Adressen aller anderen Teilnehmer des Netzwerks zu gelangen. Mit der rasant ansteigenden Zahl von Nutzern ergaben sich allerdings gravierende Probleme im Umgang mit der HOSTS.TXT:

- Da Änderungen nur zentral durch das NIC eingepflegt wurden und es keinen Zwang zum regelmäßigen Herunterladen der Datei gab, kam es vor, dass veraltete Versionen der HOSTS.TXT benutzt wurden.
- Die Belastung für den zentralen Server des NIC wuchs mit der immer größeren Zahl neuer Netzwerkteilnehmer auf ein unkontrollierbares Maß an.
- Namenskonflikte traten immer häufiger auf, da der flache Namenraum nur begrenzte Möglichkeiten bot und generell das Prinzip „first come – first served“ eingesetzt wurde.
- Der Wunsch einiger Organisationen, ihren lokalen Namensraum selbst zu verwalten, um Änderungen schneller umzusetzen, konnte durch den HOSTS.TXT - Mechanismus nicht erfüllt werden.

2 Entwurfsziele des DNS

Um die vorgenannten Probleme zu lösen gab es im Laufe der Zeit viele Ansätze, die im Großen und Ganzen einen gemeinsamen Nenner hatten: die Organisation der Daten in einem hierarchisch aufgebauten, und somit verteilt administrierbaren, Namensraum mit dem „.“ als Trennzeichen zwischen den Hierarchieebenen. Für den Einsatz im Internet hat sich letztlich das Domain Name System (DNS) durchgesetzt. Beim Entwurf des DNS wurden von P. Mockapetris unter anderen die folgenden Ziele verfolgt, die im RFC1034 ([MOC1987/1]) nachzulesen sind:

- Es sollte in erster Linie ein konsistenter Namensraum geschaffen werden, mit Domainnamen, die keinerlei Hinweise auf Netzwerkadressen, mögliche Routen zur Ressource oder Architekturen enthalten.
- Der hierarchische Aufbau des Namensraumes sollte zur Vermeidung von Namenskonflikten und zur Vergrößerung des potentiell nutzbaren Namensraumes führen.
- Der Einsatz einer verteilten Datenbank, zur Speicherung der Namensinformationen, ergab sich direkt aus dem immensen, zu erwartenden, Datenumfang und sollte außerdem dazu beitragen, notwendige Aktualisierungen möglichst zeitnah einpflegen zu können.
- Zur Performancesteigerung sollten bereits erfragte Daten redundant in lokalen Zwischenspeichern vorgehalten werden.
- Die Elemente des DNS sollten unabhängig vom eingesetzten Kommunikationssystem oder der Art des Hosts (z.B. PCs oder Mainframes) nutzbar sein.
- Flexibilität in der Adressierung verschiedenster Ressourcen sollte über die Einführung von Typen erreicht werden.

3 Elemente des DNS

Die grundsätzliche Funktionsweise des DNS ergibt sich aus dem Zusammenspiel der von Mockapetris definierten Elemente: „Domain Name Space“, „Ressource Records“, „Name Server“ und „Resolver“.

3.1 Domain Name Space

Unter einer Domain wird i. A. eine Gruppe von hierarchisch zusammengeschlossenen Computern verstanden, die jeweils über ihren obersten Knoten, die so genannte ROOT identifiziert werden. Im Sinne des DNS entspricht allerdings auch ein einzelner, im Netzwerk registrierter, Host bereits einer Domain. Jede Domain ist über ihren Namen identifizierbar. Dieser Name wird Label genannt, er darf weder Punkte, noch Sonderzeichen enthalten und maximal 63 Zeichen haben. Es ist charakteristisch für eine Domain vollständiges Wissen über alle ihre Subdomains zu enthalten (mehr dazu unter 4.2). Der Domain Name Space umfasst alle angeschlossenen Domains eines Netzwerkes, wobei man sich die Struktur des Domain Name Space wie einen Baum vorstellen kann. Die Blätter des Baumes sind Domains, die keine weiteren Subdomains enthalten, sie können aus nur einem oder aber beliebig vielen Hosts bestehen. Um einen Knoten innerhalb des Baumes adressieren zu können wird seinem Namen Pfadinformation hinzugefügt. Man liest dabei „baumaufwärts bis zur Wurzel“ und nutzt den „.“ als Trennzeichen zwischen den Labels. Im Domain Name Space ist die eindeutige Adressierbarkeit eines jeden Knotens gewährleistet, da auf jeder Hierarchieebene für die Einzigartigkeit von Namen gesorgt wird. Für eine konkrete Vorstellung vom Domain Name Space ist es notwendig zu wissen, dass die Wurzel des gesamten Domain Name Space ein

leeres Label besitzt, es wird auch das NULL – Label genannt und darf an keiner anderen Stelle im Domain Name Space verwendet werden. Erst ab der ersten Hierarchieebene unterhalb der ROOT ist es daher notwendig ein explizites Label anzugeben, um einen Teilbaum des Domain Name Space anzusprechen (mehr dazu unter 4.1). Man spricht in diesem Fall von einem relativen Namen oder Pfad und nimmt üblicherweise an, dass der Pfad direkt unterhalb der Wurzel des gesamten Baumes beginnt. Wird am Ende des Pfades jedoch das NULL - Label in Form eines Punktes explizit angegeben, beispielsweise: host.sample.com., nennt man das einen absoluten oder vollständig qualifizierten Namen. Die Konzepte „Domain“ und „Domain Name Space“ lassen sich sowohl für lokale Netze, als auch für das Internet anwenden, indem man die Vorstellung von kleineren Teilbäumen innerhalb eines Gesamtbaumes beibehält. Abbildung 1 verdeutlicht schematisch die baumartige Struktur des Domain Name Space.

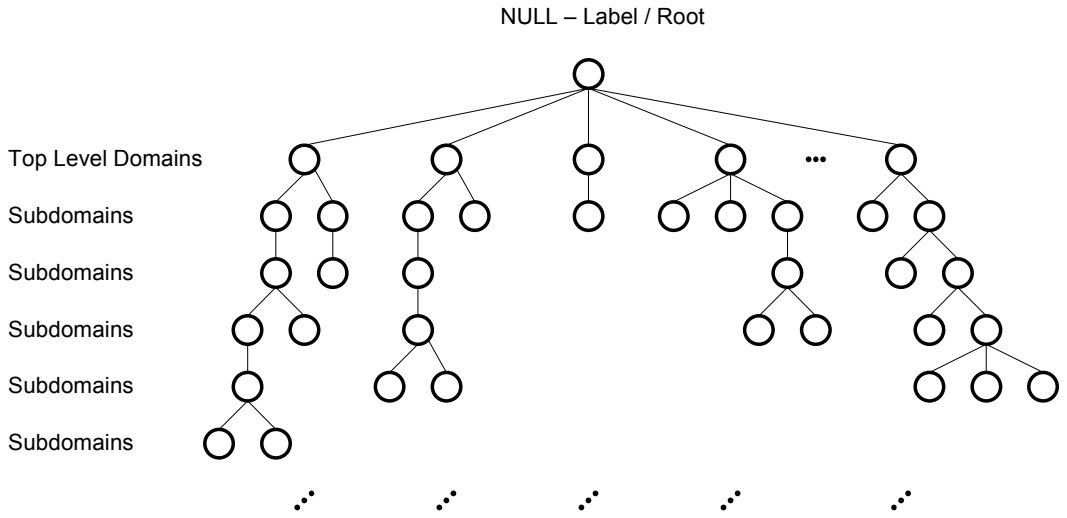


Abbildung 1 – generelle Struktur des Domain Name Space (Schema)

3.1.1 Zonen

Das Konzept einer Domain könnte jedoch nur mit sehr hohem Aufwand technisch umgesetzt werden, da hierfür wie unter 3.1 bereits erwähnt vollständiges und stets aktuelles Wissen über alle vorhandenen Subdomains notwendig ist. *„Dies würde bei der Root - Domain dazu führen, dass der Server den kompletten Internet - Namensraum verwalten müsste.“* [PH2003/2004, Seite 6] Um diesen immensen Aufwand zu vermeiden, werden Domains in autonome, überschaubare und sich nicht überlappende Verwaltungseinheiten, die so genannten Zonen aufgeteilt. *„Für alle Knoten, die sich innerhalb einer Zone befinden, wird diese Zone als autoritär oder gültig bezeichnet.“* [PH2003/2004, Seite 7] Um eine Zone eindeutig und vollständig zu beschreiben werden folgende Informationen benötigt:

- gültige Zonendaten (authoritative data), zur Identifikation aller Ressourcen / Hosts innerhalb der Zone
- Informationen über den obersten Zonenknoten
- Adressen von Name Servern, die delegierte Subzonen verwalten

Eine Zone ist jeweils mit anderen Zonen durch Delegation verbunden. Unter Delegation wird hier im Wesentlichen die Möglichkeit eines Name Servers verstanden, Verantwortung für bestimmte Teilzonen an weitere Name Server abzugeben und nur die Referenz auf diese Server, nicht aber die gültigen Zonendaten der Teilzonen zu verwalten. Abbildung 2 zeigt beispielhaft, wie die Aufteilung eines Domainnamensraumes in Zonen aussehen kann.

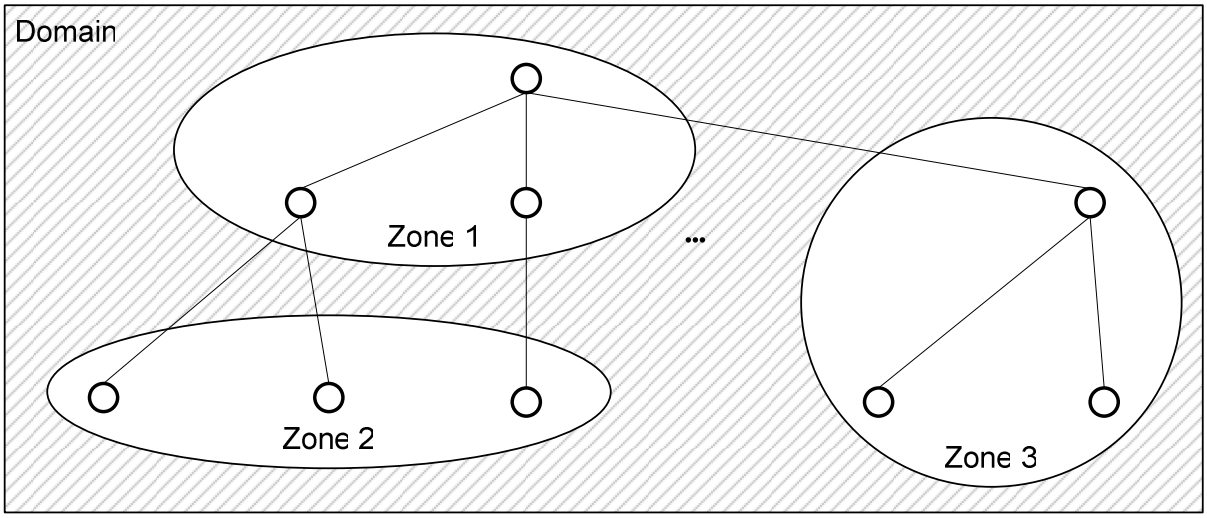


Abbildung 2 – Abgrenzung der Begriffe Domain und Zone

3.1.2 Zonentransfer

Informationen über Zonen sind im Allgemeinen redundant auf mehreren Name Servern vorhanden. Werden Änderungen am Datenbestand des autoritären (primären) Name Servers durchgeführt, müssen diese auch auf den sekundären Servern verfügbar gemacht werden. Zu diesem Zweck prüfen die sekundären Name Server in regelmäßigen Abständen, ob ein Zonentransfer notwendig ist. Mehr dazu unter 3.3.2.

3.2 Ressource Records

Um Informationen über die Knoten im Domain Name Space zu verwalten, wurden von Mockapetris die so genannten Ressource Records (RRs) vorgeschlagen. Sie enthalten sowohl die üblichen Zuordnungen von Namen zu Adressen, als auch Angaben über den Typ der gespeicherten Information. RRs werden ausgewertet, um Knoten innerhalb des Domain Name Space zu identifizieren. Den generellen Aufbau eines RR verdeutlicht Tabelle 1.

Eintrag	Beschreibung
owner / name	enthält den Domainnamen, auf den sich der RR bezieht; primärer Suchschlüssel
class	ein 16-Bit-Wert, der die genutzte Protokollfamilie benennt
ttl	„time to live“ der 32-Bit-Wert bestimmt die Lebensdauer des RR, angegeben in Sekunden
type	identifiziert den Typ des RR
rdlength	gibt die Länge der nachfolgenden Nutzdaten an
rdata	in diesem Feld werden die Nutzdaten gespeichert

Tabelle 1 – genereller Aufbau eines Ressource Records

Um im Kontext einer Domain verschiedene Ressourcen zu unterscheiden nutzt man typisierte RRs. Im Internet werden am häufigsten die in Tabelle 2 aufgeführten RRs verwendet.

Typ	Beschreibung
A	enthält die IPv4 - Adresse eines Hosts
CNAME	Zuordnung eines kanonischen Namens zu einem Alias
HINFO	enthält Informationen über einen Host
MX	identifiziert einen Host, der an die Domäne adressierte Mails akzeptiert
NS	enthält die Adresse eines Name Servers
PTR	enthält einen Zeiger auf einen Domainnamen
SOA	identifiziert den Anfang einer Autoritätszone
WKS	enthält eine Beschreibung bekannter Dienste

Tabelle 2 – häufig genutzte Ressource Records und ihre Verwendung

Auf Grundlage der, in RFC1035 durch Mockapetris veröffentlichten, Beschreibung der Ressource Records wird ihre Struktur im Folgenden näher beschrieben und jeweils anhand eines Beispiels verdeutlicht.

- **A – adress**

Ein RR vom Typ **A** beinhaltet für die Klasse **IN** (Internet) eine IPv4 - Adresse, die zum Auffinden des angegebenen owners – der Domain – genutzt werden kann.

Struktur	<owner>	<class>	<ttl>	A	<adress>
Beispiel	host.sample.com.	IN		A	127.0.0.1

- **CNAME – canonical name**

Um einen Alias über das DNS zu verwalten, wird ein **CNAME** RR benötigt. Für eine tatsächlich vorhandene Ressource können beliebig viele Aliase hinterlegt werden, die größtenteils genutzt werden, weil sie mnemonischer sind, als der tatsächliche Domainname.

Struktur	<owner>	<class>	<ttl>	CNAME	<canonical-dname>
Beispiel	localhost.sample.com.	IN		CNAME	host.sample.com.

- **HINFO – host information**

In einem **HINFO** RR können Informationen zu einem Host hinterlegt werden. Es werden die CPU und das Betriebssystem angegeben. Die Verwendung der **HINFO** RRs ist jedoch umstritten, da diese Informationen auch für einen unerwünschten Zugriff auf den Host genutzt werden können.

Struktur	<owner>	<class>	<ttl>	HINFO	<cpu> <os>
Beispiel	host.sample.com.	IN		HINFO	VAX-11/780 UNIX

➢ MX – Mail Exchanger

MX – Datensätze identifizieren einen Host der, an die Domäne adressierte, Mails akzeptiert.

Struktur	<owner>	<class>	<ttl>	MX	<exchange-dname>
Beispiel	host.sample.com.	IN		MX	postbox.host.sample.com.

➢ NS – Name Server

NS – Datensätze beschreiben existierende Name Server für die angegebene Domain. In Kapitel 3.3 wird näher auf den Einsatz und die Unterscheidung von Name Server – Typen eingegangen.

Struktur	<owner>	<class>	<ttl>	NS	<name-server-dname>
Beispiel	host.sample.com.	IN		NS	nameserver.sample.com.

➢ PTR – pointer

Für die inverse Namensauflösung, d.h. für die Zuordnung eines Domainnamens zu einer bekannten Adresse, wird ein RR vom Typ **PTR** verwendet. Es handelt sich dabei um einen Pointer auf eine Stelle im Domain Name Space.

Struktur	<owner>	<class>	<ttl>	PTR	<dname>
Beispiel	127.0.0.1.in-addr.arpa.	IN		PTR	host.sample.com.

➢ SOA – start of authority

Um auf einem autoritären Name Server den Anfang einer Autoritätszone zu kennzeichnen wird ein **SOA** RR genutzt, der verschiedene Informationen für die Verwaltung der Domaindaten enthält; welche das genau sind zeigt Tabelle 3 auf der nächsten Seite.

Struktur	<owner>	<class>	<ttl>	SOA	<source-dname> <mbox> (<Tabelle 3>)
Beispiel	host.sample.com.	IN		SOA	nameserver.sample.com. admin.sample.com. (<Tabelle 3>)

➢ WKS – well known service

Die **WKS** RRs werden eingesetzt, um die Verfügbarkeit von Diensten zu vermerken, die von einem bestimmten Protokoll an einer angegebenen Internetadresse angeboten werden.

Struktur	<owner>	<class>	<ttl>	WKS	<adress> <protocol> <service-list>
Beispiel	host.sample.com.	IN		WKS	127.0.0.1 TCP (telnet smtp ftp)

Eintrag	Beschreibung
serial	enthält die Versionsnummer der originalen Kopie dieser Zone
refresh	ein 32-Bit-Zeitintervall, das angibt in welchen Abständen die Zone aktualisiert werden sollte
retry	ein 32-Bit-Zeitintervall, das angibt wie lange nach einem fehlgeschlagenen refresh mit dem erneuten Zonentransfer gewartet werden soll
expire	gibt an, wie viel Zeit verstreichen darf, bis die Zone nicht mehr autoritär ist
minimum	gibt die minimale time to live für alle Ressource Records der Zone an

Tabelle 3 – Informationen zur Verwaltung von Domaindaten

3.3 Name Server

Wie bereits mehrfach angesprochen, werden die zur Namensauflösung genutzten Ressource Records durch Name Server (NS) verwaltet. Sie sind für die Zuordnung von Adressen zu Namen zuständig und teilweise auch für die Weiterleitung von Anfragen an weitere NS, sofern eine Beantwortung auf Grundlage des jeweils eigenen Datenbestandes nicht möglich ist. Name Server werden auch als aktive Komponenten des DNS bezeichnet. Die bereits angesprochene Delegation von Verantwortungsbereichen wird implementiert, indem jeweils Teilbereiche der Domaindatenbank auf verschiedenen Name Servern gespeichert werden; man unterscheidet Primary - und Secondary Name Server. Name Server enthalten stets die Adressen der NS, die für delegierte Subzonen zuständig sind, sowie die Adressen der Root - Server. Zur Absicherung von Serverausfällen sollten die Domaindaten stets redundant auf mehreren Name Servern vorgehalten werden.

3.3.1 Primary Name Server

Der bereits unter 3.1.2 angesprochene primäre Name Server verwaltet die Ressource Records einer Zone autoritär und ist somit verantwortlich für die Konsistenz der Daten. Durch den Administrator werden die Namensinformationen der Zone manuell in so genannte Masterfiles eingetragen. Der Aufbau eines solchen Masterfiles ist spezifiziert im RFC1035 ([MOC1987/2]) und Beispiele können nachgelesen werden ab Seite 84 ff. [QCM1994]. Aus den Masterfiles bezieht der primäre Name Server seine gültigen Zonendaten. Lädt der primäre Name Server nach der Aktualisierung eines Masterfiles die neuen Ressource Records in seinen Speicher, so vermerkt er stets noch zusätzliche Informationen über Art und Umfang der Veränderung. Diese protokollierten Informationen werden später für den Zonentransfer genutzt werden. Die Versionsnummer des Datenbestandes gibt Aufschluss über die Aktualität der Zonendaten.

3.3.2 Secondary Name Server

Ein sekundärer Name Server trägt im Gegensatz zum primären Name Server keine Verantwortung für die Konsistenz der Zonendaten. Um die Last auf dem autoritären Name Server zu begrenzen enthält er jeweils eine Kopie des Datenbestandes mit der Versionsnummer, die zum Zeitpunkt des letzten Zonentransfers auf dem primären Name Server aktuell war. Bei jedem Start des sekundären NS und in regelmäßigen Abständen, die aus der refresh – Angabe des SOA RR für diese Zone zu entnehmen sind, ruft der sekundäre Name Server die aktuelle Versionsnummer des primären Name Servers ab und vergleicht sie mit der eigenen.

Sollten die Versionsnummern nicht mehr übereinstimmen, muss der sekundäre Name Server einen Zonentransfer anregen, bei dem jedoch nur diejenigen Daten übertragen werden, die sich geändert haben. Aus den unter 3.3.1 beschriebenen protokollierten Informationen über die letzten Aktualisierungen kann der primäre NS erkennen, welche RR er dem sekundären NS neu übermitteln muss. Diese so genannte **inkrementelle Methode** des Zonentransfers wurde im RFC1995 definiert, um die gewaltige Netzlast zu reduzieren, die bei einem kompletten Zonentransfer auftritt.

3.3.3 Caching

Um die Antwortzeiten zu erhöhen und gleichzeitig die Netzlast zu reduzieren werden die Ergebnisse erfolgreich beantworteter DNS - Anfragen zwischengespeichert. Meist übernehmen die Name Server diese Aufgabe und verringern dadurch die Anzahl von Anfragen, die sie bei eintreffenden Queries an andere NS weiterleiten müssen. Somit erweitert ein Name Server sein Spektrum direkt beantwortbarer Anfragen. Wenn ein Name Server bei der Namensauflösung andere NS kontaktiert erhält er außerdem Informationen über die Zuständigkeit dieser Name Server für bestimmte Zonen und diese Daten können ebenso bei späteren Namensauflösungen genutzt werden. Einige Implementierungen von DNS unterstützen sogar das so genannte **negative caching** beim dem auch negative Antworten von Name Servern gespeichert werden – bei einer wiederholten Anfrage weiß ein NS also welche anderen Name Server er nicht mehr kontaktieren muss. Selbstverständlich können die zwischengespeicherten Namensinformationen nicht unbegrenzt genutzt werden, da sonst Veränderungen des autoritären Datenbestandes ignoriert werden. Daher enthalten RRs eine Art „Verfallsdatum“, die so genannte time to live (ttl). Bei der Erstellung der originalen RRs schätzt der Domainadministrator die potentielle Lebensdauer der Namensinformation und setzt daraufhin die ttl. Hat nun ein Name Server einen RR in seinem Cache gespeichert, muss er vor der Verwendung der Daten prüfen ob die ttl bereits abgelaufen ist. In einem solchen Fall werden die, nicht mehr gültigen, RRs aus dem Speicher des Name Servers entfernt und müssen neu erfragt werden.

3.4 Resolver

Resolver sind Programme, die beim Host laufen und in dessen Auftrag Daten von Name Servern abfragen. In der Regel sind Resolver Systembibliotheken, die beispielsweise durch den Browser eines Hosts mit der Generierung einer DNS - Query beauftragt werden. Jeder Resolver muss mindestens einen Name Server ansprechen können, um Anfragen mit dessen Rückmeldungen direkt zu beantworten oder weiterleiten zu können. Ein Resolver muss neben der Erstellung von Anfragen auch in der Lage sein, erhaltene Antworten von Name Servern zu interpretieren, sowie an den Host zurückzugeben. Die Interpretation von Antworten umfasst i. A. die Unterscheidung zwischen RRs und Fehlermeldungen bzw. die Reaktion auf nicht erhaltene Antworten. In diesem Fall wäre das erneute Versenden der Anfrage an den bisher genutzten oder einen weiteren Name Server notwendig. Neben dieser recht einfachen Art kann es auch umfangreichere Resolver - Implementierungen geben, die weitere Mechanismen z.B. zur Zwischenspeicherung von bereits erhaltenen Daten (Caching) enthalten können. Zumeist wird diese Funktionalität jedoch durch Name Server realisiert.

4 Domain Hierarchie

Warum der Domain Name Space hierarchisch aufgebaut ist, wurde bereits an mehreren Stellen verdeutlicht. Im folgenden Abschnitt soll auf diese Hierarchie näher eingegangen werden.

4.1 Top Level Domains

Innerhalb der beschriebenen Baumstruktur kommt den Top Level Domains eine besondere Bedeutung zu. Sie zerlegen den gesamten Namensraum unterhalb der ROOT in erste Teilbereiche. Die Top Level Domains (TLDs) werden unterschieden in generische und geographische TLDs. In den USA wurden in den Anfängen des Internets nur die generischen TLDs genutzt, wie sie in Abbildung 3 dargestellt sind. Auch heute werden sie vorzugsweise von amerikanischen Organisationen verwendet, während Internetteilnehmer aus anderen Staaten die geographischen TLDs favorisieren. Die geographischen TLDs wurden später hinzugefügt um der andauernden Internationalisierung gerecht zu werden. Die Zuordnung einer geographischen Top Level Domain zu jedem Staat der Welt wurde im Standard ISO 3166 festgeschrieben. Damit wurde jeweils eine Abkürzung aus zwei Buchstaben für jeden Staat reserviert. Während ein Großteil der Staaten ihre zugeordneten TLDs nutzt, gibt es Ausnahmen, die häufig vom ISO – Standard abweichen (beispielsweise die USA, wo .com weitaus verbreiteter ist, als .us) oder ihre TLD gar nicht nutzen. Auch die generischen TLDs wurden erweitert, z.B. um die TLDs **.info, .biz** oder die TLD **.tv**, die ursprünglich als geographische TLD des Inselstaates Tuvalu vorgesehen war und jetzt für Medienunternehmen in aller Welt eingesetzt wird. Die originalen generischen Top Level Domains waren im Einzelnen für die folgenden Verwendungszwecke vorgesehen, zum Teil werden die hier aufgeführten Beschränkungen in der Realität jedoch nicht eingehalten:

- COM

Jede kommerzielle Organisation hat die Möglichkeit eine **COM** Domain zu etablieren, jedoch wird die Länge der Label hier auf 12 alphanumerische Zeichen begrenzt. Dies schränkt den Namensraum stark ein, sodass es bereits einer guten Begründung bedarf, wenn eine Organisation eine zweite **COM** Domain beantragt.

- EDU

Obwohl die **EDU** TLD ursprünglich alle Arten von Bildungseinrichtungen umfassen sollte, wird sie heutzutage nur von amerikanischen Colleges und Universitäten genutzt.

- GOV

Die Benutzung der TLD **GOV** wird im RFC2146 geregelt, sie steht für ausgewählte amerikanische Regierungsbehörden zur Verfügung.

- MIL

Einen Domainnamen innerhalb der **MIL** TLD können ausschließlich militärische Organisationen registrieren.

- NET

Provider von Netzwerkdiensten, wie die InterNIC oder die DENIC registrieren ihre Domains in der **NET** TLD.

- ORG

Im Gegensatz zur **COM** TLD werden Domainnamen in der **ORG** TLD den nicht-kommerziellen Organisationen, wie Vereinen, zugeordnet.

- INT

Organisationen, wie z.B. die NATO oder die WHO, die internationale Aufgaben staatenüber-greifend erfüllen, können eine Domain in der **INT** TLD beantragen. Um eine Genehmigung zu erhalten müssen jedoch eine Reihe von Anforderungen erfüllt werden, die im RFC1591 aufgeführt sind.

- (ARPA)

Die TLD **ARPA** wurde nur kurzeitig genutzt, um alle Hosts des ehemaligen ARPNET nach und nach in die generischen Top Level Domains zu transferieren.

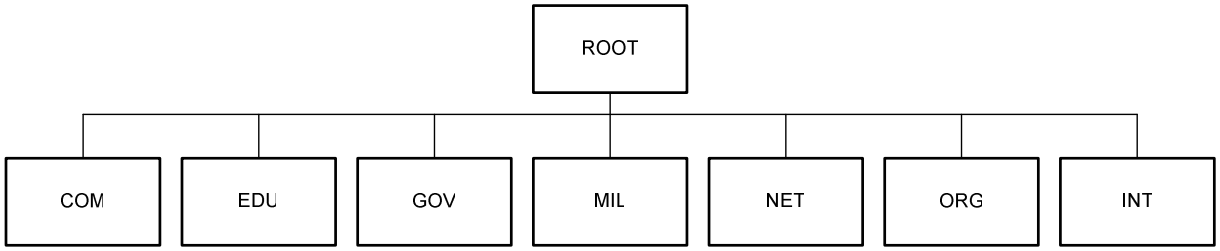


Abbildung 3 – Übersicht über die ursprünglich festgelegten generischen TLDs

4.2 Subdomains

Ist eine Domain einmal innerhalb des Domain Name Space; z.B. unterhalb einer TLD registriert kann sie unterteilt werden, um Teilbereiche des Netzwerkes voneinander zu trennen. Diese Unterteilung ist allerdings eine rein logische und sie hat zumeist organisatorische Gründe, z.B. Delegation der Verantwortung für die Administration. Beispielsweise unterteilen Unternehmen ihre Domain oft in Subdomains für verschiedene Standorte oder Fachbereiche. Für die Zerlegung in Subdomains benötigt der Inhaber einer Domain keine Genehmigung, er muss allerdings für die Eindeutigkeit aller Namen unterhalb seiner Domain sorgen, damit die Namensauflösung mittels DNS funktioniert. Um Navigation innerhalb der eingerichteten Subdomains zu gewährleisten, werden RRs vom Typ NS benötigt. Sie zeigen auf die autoritären NS, bei denen Daten über die jeweilige Subdomain abgefragt werden können. Bei der Einteilung von Domains in Subdomains ergibt sich ein generelles Adressschema für diese Subdomains, dass in Abbildung 4 beispielhaft dargestellt ist.

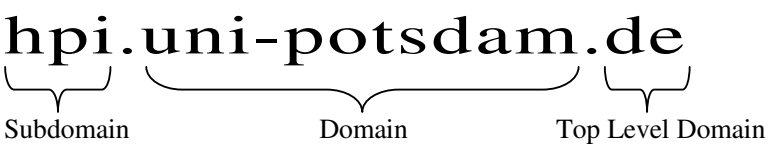


Abbildung 4 – Beispiel für das Namensschema im Domain Name Space

5 Namensauflösung

Um die Adresse zu einer Ressource zu ermitteln wird ein Verfahren eingesetzt das üblicherweise „Namensauflösung“ genannt wird. Vereinfacht handelt es sich dabei um die Beantwortung einer Anfrage durch einen Name Server. Allerdings können diese vom Resolver generierten Anfragen nur in seltenen Fällen direkt durch den primären Name Server der Zone beantwortet werden, da oft auf Ressourcen zugegriffen werden soll, die außerhalb der Zone liegen. Um die Namensauflösung für diese Ressourcen durchzuführen, kann sowohl ein iterativer, als auch ein rekursiver Ansatz verwendet werden. Grundsätzlich wird durch die Spezifikation des DNS nur die Implementierung des iterativen Verfahrens gefordert. Name Server bieten dennoch meist das rekursive Verfahren mit an, sie müssen dazu allerdings einen lokalen Resolver enthalten.

5.1 iterative Namensauflösung

Bei der iterativen Namensaulösung wird ein großer Teil der Arbeit durch den Resolver erledigt. Er generiert eine DNS - Anfrage und sendet diese an einen ihm bekannten autoritären Name Server der Zone. Dieser durchsucht daraufhin seine Ressource Records und den lokalen Speicher nach zutreffenden Namensinformationen. Findet der Name Server einen Ressource Record, der die Anfrage beantwortet, reicht er die Information an den Resolver zurück. Kann der Name Server die Anfrage jedoch nicht zufriedenstellend beantworten durchsucht er seinen Datenbestand erneut nach einem Eintrag für einen Name Server, der zur Namensauflösung beitragen kann und sendet dessen Adresse an den Resolver, der wiederum diesen neu benannten Name Server bezüglich seiner Anfrage kontaktiert. So spricht der Resolver nach und nach verschiedene Name Server an, die jeweils diejenigen Informationen aus ihrem Datenbestand zurückliefern, die am besten zur Namensauflösung beitragen. Dabei „hangelt“ sich der Resolver innerhalb des Domain Name Space „baumaufwärts“, gegebenenfalls bis zur Root. Abbildung 5 zeigt den generellen Ablauf einer erfolgreichen iterativen Namensauflösung. Im folgenden Beispiel wird gezeigt welche Schritte notwendig sind, um die von einem Host gestellte Anfrage nach der Domain **hpi.uni-potsdam.de** zu beantworten. Es wird angenommen, dass der Host sich außerhalb der Domain **uni-potsdam.de** befindet und die Anfrage zum ersten Mal an den lokalen Name Server gestellt wird.

- der lokale Resolver des Hosts generiert eine DNS – Query für **hpi.uni-potsdam.de** und sendet diese an einen ihm bekannten autoritären Name Server
- der Name Server durchsucht seine Ressource Records und seinen Cache nach Typ A – Einträgen für **hpi.uni-potsdam.de**
- der Name Server durchsucht seine Ressource Records und seinen Cache ein zweites Mal nach Einträgen vom Typ NS für **uni-potsdam.de**

- der Name Server durchsucht seine Ressource Records und seinen Cache ein drittes Mal nach Einträgen vom Typ NS für **de**
- der Name Server durchsucht seine Ressource Records und seinen Cache ein viertes Mal nach einem Einträgen vom Typ NS für einen Root Name Server
- der Name Server sendet dem Resolver einen NS – Eintrag für einen Root Name Server
- der lokale Resolver des Hosts sendet daraufhin die Query für **hpi.uni-potsdam.de** an den Root - NS
- der Root Name Server durchsucht seine Ressource Records und seinen Cache nach Typ A – Einträgen für **hpi.uni-potsdam.de**
- der Root Name Server durchsucht seine Ressource Records und seinen Cache ein zweites Mal nach Einträgen vom Typ NS für **uni-potsdam.de**
- der Root Name Server durchsucht seine Ressource Records und seinen Cache ein drittes Mal nach Einträgen vom Typ NS **de**
- der Root Name Server sendet dem Resolver einen NS – Eintrag für den **de** - Name Server
- der lokale Resolver des Hosts sendet daraufhin die Query für **hpi.uni-potsdam.de** an den **de** – Name Server
- der **de** - Name Server durchsucht seine Ressource Records und seinen Cache nach Typ A – Einträgen für **hpi.uni-potsdam.de**
- der **de** - Name Server durchsucht seine Ressource Records und seinen Cache ein zweites Mal nach Einträgen vom Typ NS für **uni-potsdam.de**
- der **de** - Name Server sendet dem Resolver einen NS – Eintrag für den Name Server der Domain **uni-potsdam.de**
- der lokale Resolver des Hosts sendet daraufhin die Query für **hpi.uni-potsdam.de** an diesen Name Server
- der NS der Domain **uni-potsdam.de** durchsucht seine Ressource Records nach Typ A – Einträgen für **hpi.uni-potsdam.de** und sendet die Antwort auf die Anfrage an den Resolver des Hosts zurück

5.2 rekursive Namensauflösung

Bei der rekursiven Methode dagegen gibt der Resolver nur den Anstoß zur Namensauflösung, während die Name Server aktiv an der Beantwortung der Anfrage arbeiten. Da nicht jeder DNS Name Server rekursive Anfragen bedienen kann sendet der Resolver vorab eine Anfrage an den Name Server, um zu prüfen, ob

dieser eine rekursive Namensauflösung durchführen kann. Ist die Antwort des Name Servers positiv, sendet er die eigentliche rekursive Anfrage an den Name Server, im Falle einer negativen Antwort wird eine iterative Anfrage an den Name Server übermittelt. Enthält der Name Server die notwendigen Ressource Records zur Namensauflösung, gibt er dem Resolver direkt das Ergebnis zurück. Muss jedoch ein weiterer Name Server kontaktiert werden, übernimmt dies der Name Server. An dieser Stelle muss vom Name Server wiederum entschieden werden, ob die weiterführende Anfrage rekursiv oder iterativ gestellt werden soll. Abbildung 6 auf Seite 16 verdeutlicht, dass auf dem Weg zum gesuchten Ressource Record mehrere Name Server, bis hin zu einem Root Name Server, kontaktiert werden können. Sie geben als Antwort auf die Anfrage jeweils entweder die gesuchten Informationen zurück oder sprechen einen, besser für die Namensauflösung qualifizierten, Name Server an. Letztlich wird das Ergebnis der Namensauflösung zurückgereicht und es bleibt vor dem Resolver verborgen, wie viele Schritte für die Namensauflösung notwendig waren. Mit der leichten Abwandlung, dass nicht der Resolver die weiterführenden Kontakte zu Name Servern initiiert, sondern jeder beteiligte Name Server selbst, kann das unter 5.1 aufgeführte Beispiel für **hpi.uni-potsdam.de** auch hier zur Verdeutlichung herangezogen werden.

5.3 inverse Namensauflösung

Im Gegensatz zur bisher betrachteten Namensauflösung, bei der generell die Ermittlung einer Adresse zu einem gegebenen Namen gefordert war, kann es notwendig sein, einen gültigen Domainnamen zu einer bekannten IP – Adresse zu finden. Genutzt wird dieses Verfahren beispielsweise von Systemen ohne Festplattenspeicher, die bei einem Neustart zwar ihre physikalische Adresse ermitteln können, jedoch keine Information über ihren Namen haben. Diese Form der Namensauflösung nennt man **invers**, genau genommen ist es auch keine Namens- sondern eine Adressauflösung. Zur Beantwortung inverser Anfragen werden von Name Servern Ressource Records vom Typ PTR ausgewertet, jedoch gibt DNS für die erfolgreiche Beantwortung einer inversen Anfrage keinerlei Garantien, die DNS Definition fordert die Implementierung des Mechanismus nicht einmal. Ein NS, der mit einer inversen Anfrage konfrontiert wird, kann diese nur beantworten, wenn die gesuchte Ressource innerhalb seiner Zone liegt, denn nur für Ressourcen der eigenen Zone enthält der Name Server solche PTR - Einträge. Andernfalls, hat der Name Server keine Möglichkeit einen anderen Name Server mit der weiterführenden Datensuche zu beauftragen. Um trotzdem von IP – Adressen auf Namen abbilden zu können wird die Domain **in-addr.arpa** genutzt. Alle hier vorhandenen Subdomains werden mit bis zu vier Labels benannt, die jeweils ein Oktett aus einer IP – Adresse darstellen. Für einen einzelnen Host werden genau die vier Oktette seiner IP – Adresse als genutzt, für die Repräsentation von Gateways werden entsprechend weniger Label benötigt. Innerhalb dieses Namensraumes könnte ein Name Server demnach eine DNS – Query auflösen, bei der der Domainname zu einer IP – Adresse gesucht wird. Abbildung 7 auf der Seite 17 zeigt beispielhaft wie der Namensraum der **in-addr.arpa** Domain strukturiert ist.

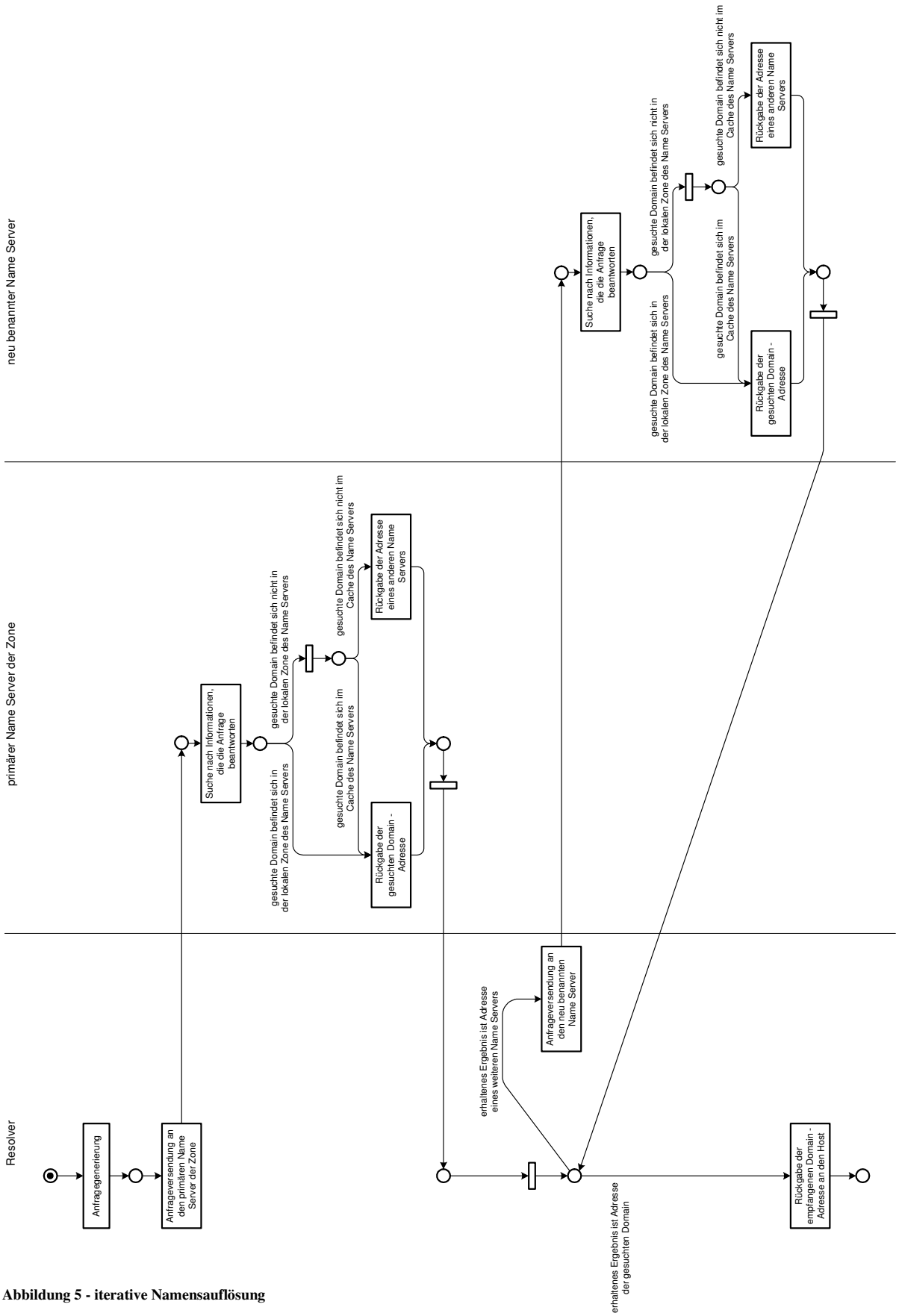


Abbildung 5 - iterative Namensauflösung

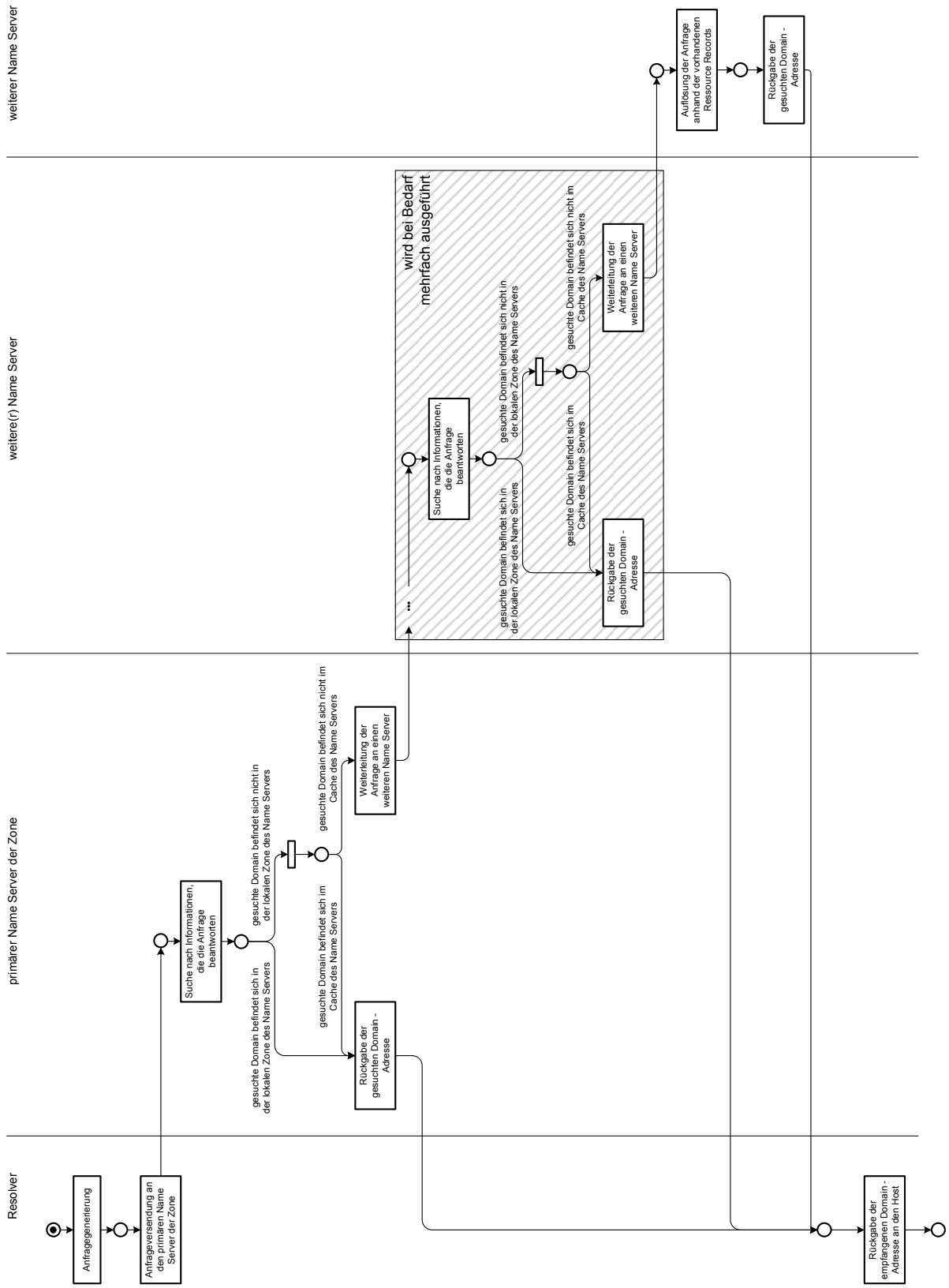
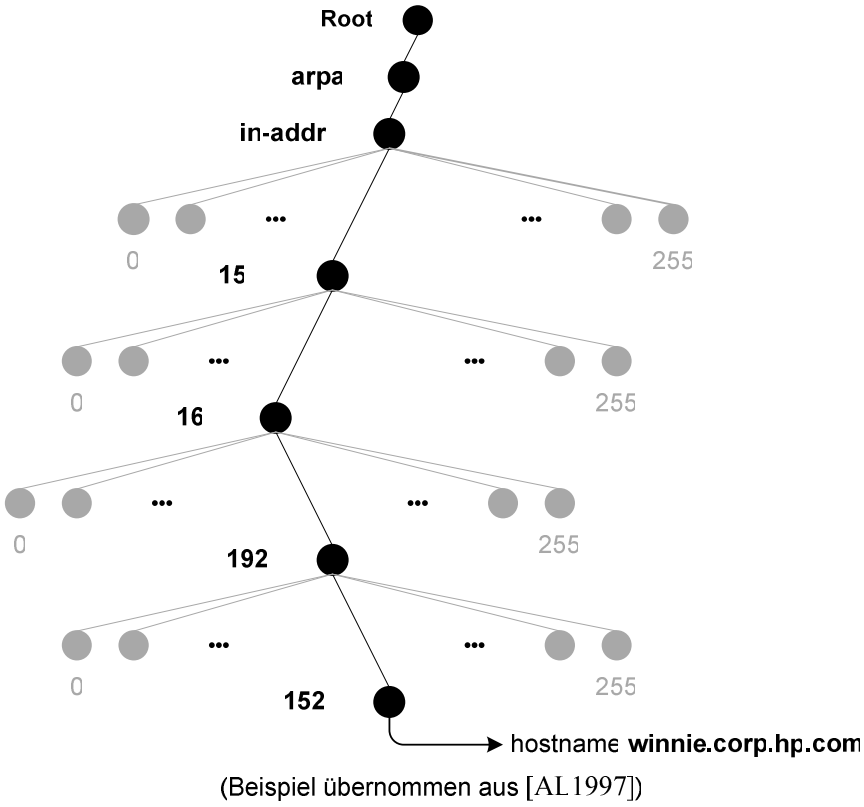


Abbildung 6 – rekursive Namensauflösung (mit rekursiven Aufrufen auf allen Hierarchieebenen)



(Beispiel übernommen aus [AL1997])

Abbildung 7 – Struktur der in-addr.arpa Domain

6 Grenzen und Schwachstellen des DNS

Trotzdem der DNS seit seiner Definition beständig an die neuesten Anforderungen des Internets angepasst wurde, weist das System gewisse konzeptionelle Schwachstellen auf. Ein wichtiger Kritikpunkt ist die langwierige Namensauflösung, die nur teilweise durch die intensive Nutzung von Caching beschleunigt werden kann, da Namensinformationen, wie bereits unter 3.3.3 erläutert, nicht unbegrenzt im Cache eines NS gehalten werden können. Ein anderes Problem tritt auf, wenn ein DNS Name Server ausfällt. Sollte ein NS, aus welchem Grund auch immer, nicht zur Verfügung stehen, wird ein redundanter Name Server benötigt, um die Aufgaben des ausgefallenen zu übernehmen. Die Anforderung, zumindest einen redundanten Name Server vorzuhalten, ist bereits in der Definition des DNS festgeschrieben. Fraglich ist jedoch, ob ein zusätzlicher NS bei der heutigen Netzlast und dem Anfragevolumen des Internet noch angemessen ist. Selbst der Einsatz ausreichend vieler Name Server genügt allerdings nicht, um alle Schwachstellen des DNS zu umgehen. Beispielhaft sollen hier zwei Varianten des so genannten DNS spoofing erläutert werden. Die erste Variante wird **cache pollution** genannt und baut auf der Tatsache auf, dass Name Server ihre lokalen Cache – Speicher verwalten. Bei einem **cache pollution** - Angriff werden einem NS, zusätzlich zu den tatsächlich angefragten Daten noch weitere, manipulierte Einträge geschickt. Diese werden durch den Name Server ebenfalls im lokalen Cache abgelegt und können für nachfolgende Anfragen verwendet werden. Diese Angriffsmöglichkeit ist jedoch bereits seit 1997 bekannt und wurde mittlerweile behoben. Indem NS mit heuristischen Verfahren die Glaubwürdigkeit der empfangenen Daten

prüfen werden erfolgreiche **cache pollution** – Angriffe vermieden. Informationen werden demnach nur dann durch den NS genutzt, wenn sie auch einen Zusammenhang zur gestellten Anfrage aufweisen. Betreibt ein Angreifer etwas mehr Aufwand kann er, mittels einer Methode die **query-ids erraten** genannt wird, einem Name Server gefälschte Einträge „unterjubeln“. Hierbei wird gezielt die Umleitung von Anfragen auf, durch den Angreifer, wählbare Ressourcen angestrebt. Dazu muss erfolgreich die aktuelle Query - ID des betroffenen NS erraten werden, da der Name Server nur eine Antwort mit der richtigen ID akzeptiert. Die folgenden Abbildungen zeigen den erfolgreichen Ablauf eines solchen Angriffs. Weiterführende Informationen zu dieser Art des DNS spoofing können in den Quellen [KM2002] und [MW1997] nachgelesen werden.

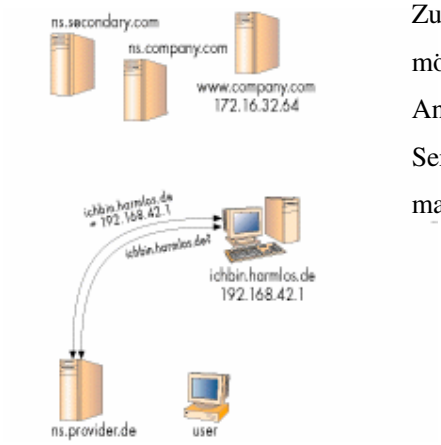


Abbildung 8 – DNS spoofing – Schritt 1

Zuerst stellt der Angreifer dem Name Server, den er manipulieren möchte, eine rekursive Anfrage nach seiner eigenen Domain. Aus der Antwort auf diese Anfrage kann er die derzeitige Query-ID des Name Servers ermitteln, die später benötigt wird, um dem NS vorzuspielen, man antworte auf seine Anfrage.

Im zweiten Schritt versendet der Angreifer eine Anfrage an den NS, bezüglich der Domain für die er manipulierte Einträge erstellen will. Gleich darauf schickt er falsche Antwortpakete an den Name Server. Da viele Name Server sequentielle Query - IDs benutzen, kann hier mit einer Reihe aufsteigender Query - IDs gearbeitet werden, die alle größer als die im Schritt 1 ermittelte sind.

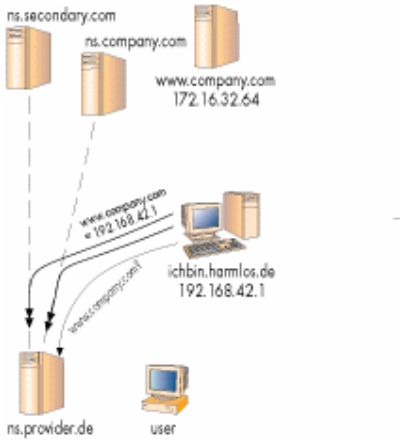


Abbildung 9 – DNS spoofing – Schritt 2

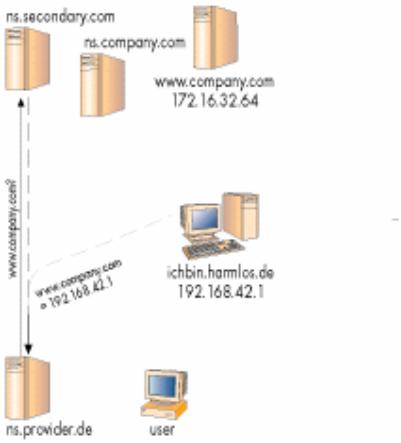


Abbildung 10 – DNS spoofing – Schritt 3

Sofort nachdem der Name Server die Anfrage an den tatsächlich zuständigen N weitergeleitet hat, erhält er auch schon die gefälschten Antworten des Angreifers und hält dasjenige Paket mit der korrekt erratenen Query – ID für die legitime Antwort des befragten Name Servers. In Wirklichkeit nimmt der Name Server an dieser Stelle manipulierte Namensinformationen in seinen lokalen Cache auf.

Erreicht später das wirkliche Antwortpaket den Name Server verwirft er es, da es unerwartet eintrifft. "*Die Kommunikation im DNS erfolgt aus Effizienzgründen nicht über TCP, sondern verbindungslos über das paketorientierte UDP (User Datagram Protocol). Da UDP-Pakete unterwegs verlorengehen können, beinhalten die Server und Clients spezielle Logik zum Wiederholen unbeantworteter Anfragen; unerwartete Antwortpakete werden stillschweigend ignoriert. Während TCP-Pakete durch 32-Bit-Sequenznummern recht brauchbar gegen Fälscher geschützt sind, die nicht auf der Leitung mitlesen können, fehlt dem UDP-Protokoll jegliche Sicherung gegen eingestreute Pakete.*" ([MW1997])

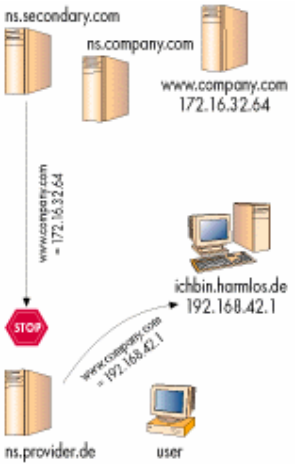


Abbildung 11 – DNS spoofing – Schritt 4

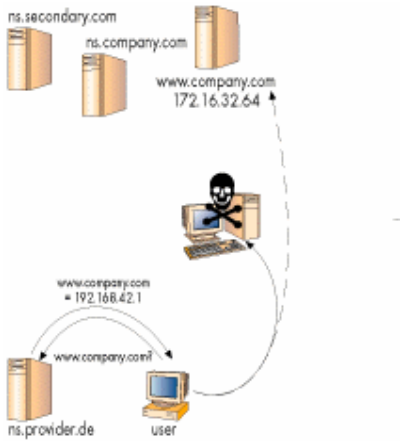


Abbildung 12 – DNS spoofing – Schritt 5

Schließlich werden Anfragen bezüglich der umgeleiteten Domain durch den Name Server so lange mit den gefälschten Daten beantwortet, wie die ttl des Eintrages noch gültig ist.

Abschließend werden einige Hinweise gegeben, die helfen können einem DNS spoofing - Angriff dieser Art zu entgehen:

- Es sollten keine aufeinander folgenden Query – Ids genutzt werden, um das Erraten zu erschweren.
- Die zusätzliche Nutzung von digitalen Unterschriften erhöht ebenfalls die Sicherheit.
- Die Versendung der Anfragen von einem zufällig ausgewählten Port macht es dem Angreifer schwerer dem Name Server erfolgreich ein manipuliertes Datenpaket zu übermitteln, da in diesem Fall Query – ID und Portnummer richtig erraten werden müssen.

7 Tabellenverzeichnis

8 Abbildungsverzeichnis

9 Quellenverzeichnis

- **[AL1997]**

  Paul Albitz und Cricket Liu

  DNS and Bind, 2nd Edition, O`Reilly 1997

- **[CDK2002]**

  G. Coulouris, J. Dollimore und T. Kindberg

  Verteilte Systeme - Konzepte und Design, 3. überarbeitete Auflage, Addision Wesley 2002

- **[IWA2004]**

  Dr. Sebastian Iwanowski

  FH Wedel – Vorlesungsunterlagen der Veranstaltung „Verteilte Systeme“, SS2004

  http://www.fh-wedel.de/~iw/Lehrveranstaltungen/SS2004/VS/VS7.pdf

- **[KM2002]**

  Frank Kühnlenz und Nicolas Michael

  Spoofing – Seminarvortrag im Rahmen des Seminars „Der Fehler im System“

  http://www.informatik.hu-berlin.de/~kuehnlen/spoofing/spoofing.pdf

- **[MEI2001]**

  Prof. Dr. Christoph Meinel

  Universität Trier– Vorlesungsunterlagen der Veranstaltung „Sicherheit in offenen Netzen“, SS2001

  http://www.telematik-institut.org/publikationen/online-vorlesungen/weitere-vorlesungen-und-skripte/SION-kap-2.6.pdf

- **[MOC1987/1]**

  P. Mockapetris

  RFC1034, Domain Name System - Concepts and Facilities, 1987

  http://www.ietf.org/rfc/rfc1034.txt

- **[MOC1987/2]**

  P. Mockapetris

  RFC1035, Domain Name System - Implementation and Specification, 1987

  http://www.ietf.org/rfc/rfc1035.txt

- **[MW1997]**

  Viktor Mraz und Klaus Weidner

  Falsch Verbunden – Gefahr durch DNS – Spoofing, c't 0/97, S.286: Internet Sicherheit

  http://www.heise.de/ct/97/10/286/default.shtml

- **[PH2003/2004]**

  Robert Porscha und Jörn Hartwig

  DNS - Domain Name Service, Seminar Communication Networks, HPI Universität Potsdam, WS2003

  http://www-ks.hpi.uni-potsdam.de/docs/engl/teaching/cns/ws2003-04/cns-seminar-webpage.html

- **[POS1994]**

  J. Postel

  RFC1591, Domain Name System Structure and Delegation, 1994

  http://www.isi.edu/in-notes/rfc1591.txt

- **[QCM1994]**

  John S. Quarterman und Smooth Carl-Mitchell

  The Internet Connection – System Connectivity and Configuration, Addison Wesley 1994

- **[RG2000]**

  S. Ratnasamy und J.M. González

  A Trace-Based Study of DNS, University of California, Berkley, May 19, 2000

- **[RS2003]**

  V. Ramasubramanian und E.G. Sirer

  The Design and Implementation of a Next Generation Name Service for the Internet, Cornell University

- **[TB2003]**

  A.S. Tanenbaum

  Computer Networks, Prentice Hall, March 17, 2003