

2 Grundlagen des DNS

Die Standardisierung des Domain Name Systems (DNS) liegt über 30 Jahre zurück. Zwar richtet sich das heute verwendete System immer noch nach den ursprünglichen Standards, es wurden jedoch im Laufe der Zeit zahlreiche Anpassungen und Konkretisierungen der Spezifikation vorgenommen, deren Dokumentation sich derzeit über mehr als 50 Dokumente (sog. „*Requests for Comments*“, abgek. RFCs) verteilt. Die operativen Eigenschaften des Systems sind diesen Dokumenten allerdings nicht zu entnehmen. Zudem haben sich neue Anwendungen auf Basis des DNS herausgebildet, die über die ursprünglich vorgesehene Aufgabe, die Auflösung von Domainnamen in IP-Adressen, hinausgehen. Die Grundlagen des Systems werden zwar in einer Vielzahl von Werken (u. a. [Ste93; CDK01; Nat05; LA06; BH07; AB09; Ait11; MS12a]) ausführlich behandelt; allerdings hat sich bislang noch keine einheitliche Terminologie etabliert. Zudem fehlt ein umfassender Überblick zum aktuellen Stand der Technik.

Das Ziel dieses Kapitels besteht daher darin, die Grundlagen des DNS zu erläutern, den aktuellen Stand der Technik darzustellen und die in dieser Arbeit verwendete Terminologie einzuführen. Dazu wurden umfangreiche Recherchen in den Standards, der Sekundärliteratur sowie wissenschaftlichen Veröffentlichungen angestellt. Aus Gründen der Übersichtlichkeit werden weder alle bisher formulierten, teilweise miteinander konkurrierenden Denkansätze wiedergegeben, noch gehen die Betrachtungen an jeder Stelle bis ins letzte Detail. Stattdessen wird eine für das Verständnis angemessene Beschreibungstiefe gewählt, und die Darstellung umfasst nur jene Aspekte, welche im Zusammenhang mit der Untersuchung der Beobachtungsmöglichkeiten von Bedeutung sind.

Wesentliche Inhalte In diesem Kapitel wird u. a. aufgezeigt, dass rekursive Nameserver eine zentrale Rolle bei der Namensauflösung spielen und dass die dort existierenden Beobachtungsmöglichkeiten in Zukunft an Bedeutung gewinnen werden.

Aufbau des Kapitels Nach einem Überblick in Abschnitt 2.1 werden in Abschnitt 2.2 die Hintergründe der Entstehung des DNS vorgestellt, um die getroffenen Entwurfsentscheidungen nachvollziehbar zu machen. Anschließend wird in Abschnitt 2.3 der Aufbau des hierarchischen Namensraums erläutert, der (wie sich in Abschnitt 5.2.1 zeigen wird) die datenschutzfreundliche Nutzung erschwert.

In Abschnitt 2.4 wird die Architektur des Systems dargestellt. Dabei werden die einzelnen Systemkomponenten beschrieben und insbesondere der Unterschied zwischen rekursiven Nameservern, auf deren Beobachtungsmöglichkeiten der Fokus dieser Arbeit liegt, und

autoritativen Nameservern herausgearbeitet. In Abschnitt 2.5 werden die Datenstrukturen vorgestellt, die zur Speicherung und beim Nachrichtentransport verwendet werden, bevor in Abschnitt 2.6 die Protokolle erläutert werden, die für den Nachrichtentransport eingesetzt werden.

In Abschnitt 2.7 wird das mehrstufige Caching-Konzept des DNS vorgestellt, welches Effizienz, Performanz und Verfügbarkeit (vgl. Abschnitt 3.5.3) des Systems positiv beeinflusst. Das Caching-Konzept ist im Zusammenhang mit den Mechanismen zum Schutz vor Verkettung von Interesse, auf die später eingegangen wird (s. Abschnitt 7.3).

Im letzten Teil des Kapitels werden in Abschnitt 2.8 die für die Betrachtung der Beobachtungsmöglichkeiten relevanten Veränderungsprozesse beschrieben, die sich seit der Standardisierung des DNS ergeben haben. Die dabei identifizierten Trends, u. a. die Verwendung alternativer rekursiver Nameserver, erhöhen die Relevanz der Beobachtungsmöglichkeiten. Den Abschluss bildet eine Aufstellung der existierenden und geplanten DNS-Anwendungen, welche sich die Protokolle und Datenstrukturen des DNS für eigene Zwecke zunutze machen, jedoch teilweise zu zusätzlichen Beobachtungsmöglichkeiten führen (s. Abschnitt 2.9). Das Kapitel schließt mit einem Fazit in Abschnitt 2.10.

2.1 Einordnung und Überblick

Das Domain Name System (DNS) ist ein auf die Anforderungen des Internets spezialisierter Namensdienst. Ein Namensdienst ist ein System, das für eine Menge von Objekten, die über einen Textnamen identifiziert werden, ein oder mehrere Attribute speichert [CDK01, S. 417]. Die wichtigste Funktion eines Namensdienstes ist die Namensauflösung, d. h. die Ermittlung der Attributwerte, welche für einen Namen hinterlegt sind [CDK01, S. 417].

Ohne Namensdienst setzt die Kommunikation im Internet die Kenntnis der IP-Adresse des zu kontaktierenden Endgeräts voraus. Diese wird zum Routing von Datenpaketen im Internet verwendet. Die IP-Adresse erlaubt einen unmittelbaren Zugriff, da sie die genaue Position des Geräts im Netz bezeichnet. Ändert sich die Position des Geräts im Netz, dann ändert sich allerdings auch seine Adresse. Adressen sind daher für die Identifizierung von Geräten nicht geeignet [CDK01, S. 414]. Dieses Problem wird durch das DNS gelöst: Endgeräten kann im DNS ein Name zugewiesen werden, der erst bei Bedarf in die jeweils aktuelle Adresse des Geräts übersetzt (oder: „aufgelöst“) wird. Es erlaubt somit die Identifizierung von Ressourcen unabhängig von ihrem tatsächlichen Standort.

Die im DNS benannten Objekte sind einzelne Rechner (**Hosts**); bei den Attributen handelt es sich meistens um deren IP-Adresse [CDK01, S. 425]. Der Domainname *www.example.net* wird z. B. in die IP-Adressen 192.0.43.10 bzw. 2001:500:88:200::10 (IPv6) aufgelöst. Die Benutzer können Ressourcen somit über einprägsame Namen ansprechen und müssen sich nicht die jeweiligen IP-Adressen merken. Abbildung 2.1 stellt den grundsätzlichen Ablauf beim Abruf einer Ressource von einem Server schematisch dar.

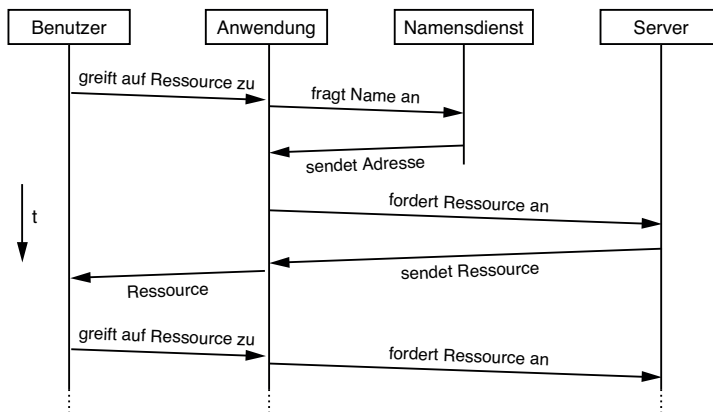


Abbildung 2.1: Zugriff auf eine Ressource mit einem Namensdienst (Darstellung nach [MS12a, S. 727])

Um Leistungsengpässe zu vermeiden und Fehlertoleranz zu bieten, wurde das DNS als *verteiltes System* implementiert [CDK01, S. 39 f.]. Die Verwaltung der Domainnamen erfolgt dabei innerhalb eines hierarchischen Namensraums, um eine dezentrale Administration zu ermöglichen [BH07, S. 160]. Der hierarchische Namensraum gibt zudem die Struktur des verteilten Systems vor, d. h. die Verteilung der Datensätze auf verschiedene DNS-Server [MS12a, S. 727 f.]. Um die Belastung des Systems durch die Namensauflösung zu beherrschen, kommt ein Caching-Mechanismus zum Einsatz, der wiederholte Anfragen vermeidet (vgl. Abschnitt 2.7).

Das DNS hat sich zu einer unerlässlichen Infrastruktur-Technologie entwickelt. Anwendungen wie das World Wide Web (WWW) sowie die Kommunikation mittels E-Mails sind erst mit einem Namensdienst benutzerfreundlich realisierbar. Das DNS wird daher mitunter als die „wichtigste und meistbenutzte (Hintergrund-)Anwendung des Internet [sic]“ bezeichnet [BH07, S. 158].

2.2 Entwicklung und Entwurfsziele

Um die Architektur des DNS und die Design-Entscheidungen nachvollziehen zu können, ist es hilfreich, seine Entwicklungsgeschichte zu betrachten. Im ARPANET, aus dem sich schließlich das Internet entwickelte, erfolgte die Namensauflösung mittels einer *Host-Table* bzw. einer *Hosts-Datei*, die ab 1971 auf jedem Rechner manuell gepflegt wurde [MS12a, S. 728]. Dabei handelt es sich um eine Textdatei, in der die Zuordnung zwischen Rechnernamen und Adressen hinterlegt ist. Die Namensauflösung erfolgte auf jedem System lokal

durch Nachschlagen eines Hostnamens in dieser Datei. Da der manuelle Abgleich von Änderungen fehlerträchtig und aufwendig war, strebte man eine Zentralisierung der Pflege an [Kud74]. Die Verwaltung des Namensraums wurde dem Stanford Research Institute (SRI), einer Einrichtung an der Universität Stanford in Kalifornien, übertragen. Das SRI stellte die Hosts-Datei fortan per FTP zur Verfügung [Fei+82; Sch10, S. 195].

Mit dem zunehmenden Wachstum des Internets stieß die Namensauflösung auf Basis der Hosts-Datei an ihre Grenzen. Es gab drei Problemfelder (nach [CDK01, S. 424]): Die Lösung wies eine **schlechte Skalierbarkeit** auf. Die Pflege aller Hostnamen in einer Datei durch das SRI stellte einen organisatorischen und personellen Flaschenhals dar; die Verteilung der Hosts-Datei an alle Endgeräte war ineffizient und erforderte eine hohe Bandbreite. Der flache Namensraum erlaubte zudem **keine Autonomie**: Um Namenskonflikte zu vermeiden, mussten die Rechner aller beteiligten Organisationen gemäß eines einheitlichen Namensschemas benannt werden. Die Administratoren hatten keine Möglichkeit, ein für ihre eigenen Rechner aussagekräftigeres, abweichendes Schema zu verwenden. Der dritte Kritikpunkt war die **Unflexibilität** des Konzepts: Die Hosts-Datei eignete sich lediglich zur Übersetzung von Hostnamen in IP-Adressen. Es konnten keine zusätzlichen Attribute für Hosts hinterlegt werden.

Ausgehend von den Erfahrungen mit der Hosts-Datei wurden bei Entwurf und Entwicklung des DNS vier zentrale Ziele verfolgt (in Anlehnung an [MS12a, S. 729] und [Ait11, S. 4]):

Entwurfsziel 1 Verteilung der Last der Namensauflösung auf mehrere Server, um Engpässe zu vermeiden,

Entwurfsziel 2 Delegation der Administration von Teilen des Namensraums, um Autonomie und Flexibilität zu erreichen,

Entwurfsziel 3 Delegation des Betriebs der Nameserver, um das DNS als verteiltes System zu implementieren, und

Entwurfsziel 4 Erweiterbarkeit um zusätzliche Attribute, um einen möglichst universell nutzbaren Namensdienst zu schaffen.

Das Realisierungskonzept für das DNS wurde zwischen 1980 und 1987 entwickelt. Die Entwicklungsgeschichte des DNS lässt sich anhand der relevanten Requests for Comments (RFC)¹ nachvollziehen. Mills schlug 1981 in RFC 799 ein dezentrales System vor, welches eine Auflösung der Hostnamen von E-Mail-Servern ermöglichte [Mil81]. Diese Idee wurde 1982 von Su und Postel in RFC 819 für beliebige Arten von Hosts generalisiert [SP82]. RFC 819 sieht unter anderem vor, anstelle eines flachen Namensraums zusammengesetzte Hostnamen (Domainnamen) zu verwenden, welche in einem hierarchischen Namensraum organisiert werden. Die sich dadurch ergebende Aufteilung des Namensraums in

¹Bei den RFCs handelt es sich um eine Serie von Dokumenten, in denen Techniken, Verhalten und Definitionen, welche das Internet betreffen, veröffentlicht werden. Ein Teil der RFCs wird schließlich zu Internet-Standards. Details zur RFC-Serie, den beteiligten Organisationen sowie dem Veröffentlichungsprozess werden in den RFCs 2223 [PR97] und 4844 [DB07] beschrieben.

Bereiche (Domains) ermöglicht zum einen die dezentrale Namensauflösung (adressiert Entwurfsziel 3). Zum anderen verfügen die Administratoren der einzelnen Domains somit über einen hohen Grad an lokaler Autonomie (adressiert Entwurfsziel 2).

Mockapetris konkretisierte das Konzept 1983 in den RFCs 882 und 883 [Moc83a; Moc83b]. Eine wesentliche Neuerung seines Vorschlags bestand in der Einführung von *Resource-Records*, um neben der IP-Adresse weitere Attribute für einen Domainnamen zu hinterlegen (adressiert Entwurfsziel 4). Weiterhin führte Mockapetris dedizierte Nameserver für die Namensauflösung und ein Caching-Konzept ein (adressiert Entwurfsziel 1). In RFC 920 schlugen Postel und Reynolds eine praktische Implementierung dieses Systems vor [PR84], das ab 1985 in der Praxis verwendet wurde [Rad01, S. 8]. Auf Basis der Erfahrungen mit ersten Implementierungen überarbeitete Mockapetris seine Spezifikation noch einmal: 1987 wurden schließlich seine beiden RFCs 1034 und 1035 veröffentlicht [Moc87a; Moc87b], die alle vorherigen ersetzen und bis heute den grundlegenden DNS-Standard bilden.

Das ursprüngliche Konzept der Hosts-Datei wird in reduzierter Form auch heute noch von den gängigen Betriebssystemen unterstützt. In dieser Datei, welche auf Unix-Systemen unter */etc/hosts* gespeichert wird, kann ein Benutzer unabhängig vom DNS eigene Hostnamen definieren und diesen IP-Adressen zuweisen. Diese zusätzlichen Einträge sind dann nur seinem System bekannt. Üblicherweise werden die in der Hosts-Datei hinterlegten Einträge bei der Namensauflösung bevorzugt.

2.3 Organisation des Namensraums

Das DNS definiert einen hierarchisch organisierten Namensraum, der als Baum implementiert ist.² Ausgehend von einem einzigen Wurzelknoten, der schlicht als **Root** bezeichnet wird, unterteilt sich der Namensraum in mehrere Bereiche, die als **Domains** bezeichnet werden. Domains können entweder weitere (Sub-)Domains enthalten oder eine Menge von Name-Wert-Paaren, in denen die Daten für einen Host hinterlegt werden. Auf diese **Resource-Records** wird in Abschnitt 2.5 noch genauer eingegangen. Eine Domain beinhaltet also ihren eigenen Knoten sowie den gesamten Teilbaum, der sich aus diesem Knoten entwickelt. Die Namen der Endgeräte, die als **Hosts** bezeichnet werden, befinden sich in den Blättern des Baums. Jeder Knoten und jedes Blatt wird durch ein **Label** bezeichnet, ein innerhalb der Domain eindeutiger Bezeichner, der eine maximale Länge von bis zu 63 Zeichen hat. Labels bestehen üblicherweise ausschließlich aus Buchstaben, Ziffern sowie dem Bindestrich. Die Groß-/Kleinschreibung ist dabei unerheblich. Sie soll beim Transport einer DNS-Anfrage nicht verändert werden (s. [Moc87b, S. 9 f.] und [Eas06a]), da RFC 1034 zukünftigen Anwendungen die Möglichkeit einräumt, auch nichtdruckbare Zeichen bzw. Binärdaten in einem Label zu verwenden [Moc87a, S. 7 f.].

Der vollständige Name eines Hosts ergibt sich durch die Konkatenation der Labels beginnend ab dem **Hostnamen** (erstes Label) bis zur Wurzel, die ein leeres Label hat. Die

²Die Darstellung in diesem Abschnitt orientiert sich an [MS12a, S. 732 ff.] und [BH07, S. 159 ff.].

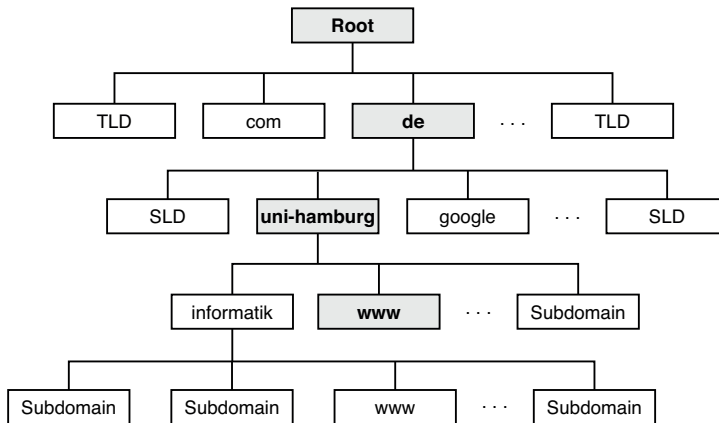


Abbildung 2.2: Die Domain *www.uni-hamburg.de* im hierarchischen Namensraum

Labels werden jeweils durch einen Dezimalpunkt voneinander getrennt, wobei auf den abschließenden Dezimalpunkt, der vor dem leeren Label der Root-Domain steht, in den meisten Fällen verzichtet werden kann. Der vollständige Name eines Hosts wird **Fully Qualified Domain Name (FQDN)** genannt, da er einen Host eindeutig bezeichnet. In der Praxis wird diese Terminologie jedoch nicht konsequent verwendet: Üblicherweise werden die unpräzisen Begriffe **Domainname** oder einfach nur **Domain** verwendet, wenn eigentlich ein FQDN gemeint ist. Ein FQDN darf inklusive des abschließenden Punktes eine maximale Länge von 255 Zeichen haben [EB97, S. 13].

Abbildung 2.2 verdeutlicht die Struktur des Namensraums und den Aufbau eines Domainnamens am konkreten Beispiel. Der obere Teil des DNS-Baums weist eine festgelegte Struktur auf. Diese korrespondiert mit der dezentralen Organisation der Namensvergabe. Die Domains der ersten Ebene, welche unmittelbar unterhalb der Root angesiedelt sind, werden als **Top Level Domains (TLD)** bezeichnet, die Domains der zweite Ebene als **Second Level Domains (SLD)**. Für Domains in tieferen Ebenen gibt es keine besonderen Bezeichner.

Dezentralisierung und lokale Autonomie werden durch die Prinzipien **Domain-Autorität** und **Delegation** erreicht. Die Ebene der TLDs wird von der IANA (Internet Assigned Numbers Authority) bzw. der von ihr beauftragten ICANN (Internet Corporation for Assigned Names and Numbers) verwaltet. Die Verwaltung der SLDs unterhalb der TLDs wird an **Network Information Centers (NICs)**, die auch als **Registrare** bezeichnet werden, delegiert. Jedes NIC hat die sog. **Autorität (authority)** über die ihm zugewiesene TLD, d. h. es kann eigenmächtig SLDs vergeben. Wird eine neue SLD registriert, **delegiert** das NIC die Autorität für diese an den Inhaber der SLD. Das Konzept der Delegation ist

nicht auf die ersten beiden Ebenen beschränkt: Der Inhaber einer SLD kann seinerseits die Autorität für einzelne Subdomains an Dritte delegieren.

Trotz lokaler Autonomie über eine Domain kann der **Domain-Inhaber** (der im Zusammenhang mit der Domain-Registrierung auch als „Registered Name Holder“ bezeichnet wird [Int13a]) jedoch nicht völlig uneingeschränkt darüber verfügen. Vielmehr ist er von der Kooperation des Betreibers der übergeordneten Domain abhängig, da die jeweils höhere Domain die Autorität über die darunter liegenden hat, also entscheidet, welche Sub-Domains existieren und wer für diese zuständig ist. Durch Löschung eines Eintrags einer delegierten Domain kann der Betreiber erreichen, dass diese inklusive aller Sub-Domains in tieferen Ebenen im DNS nicht mehr aufgefunden werden kann [LHD05, S. 16].

Neben generischen TLDs, zu denen u. a. *.com*, *.net*, *.org*, *.edu* und *.gov* zählen, gibt es länderspezifische TLDs, die als Label die zweistellige Landeskennung nach ISO 3166 tragen. Hierzu zählt auch *.de*, die TLD von Deutschland, die an die DeNIC eG delegiert wird. Eine Sonderrolle nimmt die generische TLD *.arpa* ein, die für Infrastrukturdienste, etwa die Rückwärtsauflösung von IP-Adressen in Domainnamen reserviert ist [MS12a, S. 735, 746 f.].

2.4 Architektur des verteilten Systems

Die Architektur des DNS ist an seinen zwei zentralen Aufgaben ausgerichtet, der verteilten Speicherung und Bereitstellung der Namensinformationen sowie dem effizienten Zugriff darauf, der Namensauflösung. Die nachfolgende Darstellung der Architektur fokussiert zunächst auf die Speicherung der Namensinformationen. Im Anschluss daran werden die Komponenten beschrieben, die an der Namensauflösung beteiligt sind. An beiden Prozessen sind Server-Komponenten beteiligt, die in der Literatur oft abstrakt als *Nameserver* bezeichnet werden (etwa von [SS01, S. 231]). Da das Verhalten eines Nameservers jedoch in erheblichem Maße davon abhängt, welche Rolle er wahrnimmt, kann die Verwendung dieses allgemeinen Begriffs zu Mehrdeutigkeiten führen.³ Im Rahmen der Erläuterungen werden daher präzise Begriffe eingeführt, um im weiteren Verlauf der Arbeit die jeweilige Rolle eines Nameservers auszuweisen.

Im Zusammenhang mit den Beobachtungsmöglichkeiten sind insbesondere die sog. *rekursiven Nameserver* von Bedeutung, deren Rolle in diesem Abschnitt erläutert wird.

2.4.1 Verteilte Speicherung der Namensinformationen

Wie in Abschnitt 2.2 bereits angedeutet, wurde das DNS als verteiltes System konzipiert, um eine ausreichende Skalierbarkeit und Ausfallsicherheit zu erreichen. Die Architektur

³Die unpräzise Begriffsbildung ist darauf zurückzuführen, dass die verbreiteten Nameserver-Implementierungen beide Rollen gleichzeitig ausüben können. In der Praxis wird von dieser Möglichkeit häufig Gebrauch gemacht.

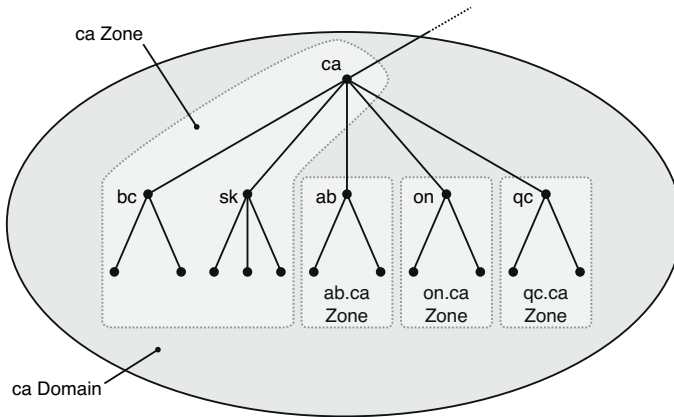


Abbildung 2.3: Beispielhafte Veranschaulichung des Unterschieds zwischen Domains und Zonen (Abb. nach [LA06, S. 24])

des Systems orientiert sich dabei am hierarchischen Namensraum, indem die Konzepte der Autorität und der Delegation von Autorität technisch umgesetzt werden [Ait11, S. 8]. Der Namensraum wird dazu in überlappungsfreie **Zonen** eingeteilt, die einen Teilbereich des Namensraums abdecken. Auf jedem autoritativen Nameserver gibt es eine sog. **Zonendatei**, in der die Resource-Records (s. Abschnitt 2.5) hinterlegt sind, für die er autoritativ ist.

Zwischen Zonen und Domains gibt es einen wesentlichen Unterschied: Eine Domain bezeichnet den gesamten Teilbaum des Namensraums, der sich unterhalb eines bestimmten Knotens aufspannt. Eine Zone enthält hingegen genau diejenigen Knoten, für die ein bestimmter Nameserver autoritativ ist. Knoten, die er an andere Nameserver delegiert, sind in seiner Zone nicht enthalten.

Beispiel 2.1. Abbildung 2.3 veranschaulicht den Unterschied zwischen Domains und Zonen anhand der „ca“-ccTLD von Kanada: Die „ca“-Domain enthält mehrere Subdomains. Für die Subdomains „ab.ca“, „on.ca“ und „qc.ca“ sind in der Zonendatei der „ca“-Zone, die auf dem autoritativen Nameserver der „ca“-Domain hinterlegt ist, Delegationen zu anderen autoritativen Nameservern eingetragen. Bei eingehenden Anfragen für Domainnamen in diesen Subdomains verweist der „ca“-Nameserver den Anfragenden dadurch auf die jeweils dafür autoritativen Nameserver. Bei den Subdomains „bc.ca“ und „sk.ca“ verhält es sich anders: Diese sind Teil der „ca“-Zone, d. h. Anfragen für diese Domainnamen beantwortet der Nameserver der „ca“-Domain selbst. □

Für jede Zone gibt es mindestens einen **autoritativen Nameserver**, der für die Namensauflösung innerhalb der jeweiligen Domain zuständig ist. Zur Sicherstellung der Verfügbarkeit der hinterlegten Daten werden üblicherweise mehrere autoritative Nameserver eingesetzt

(s. Abschnitt 3.5.3). Für jede Zone gibt es genau einen **primären Nameserver**, der auch als **Master** bezeichnet wird, und die maßgeblichen Daten enthält. Auf dem primären Nameserver hinterlegt der Betreiber der Zone die Resource-Records (s. Abschnitt 2.5). Die übrigen autoritativen Nameserver der Zone, welche eine Kopie der Zoneninformationen vom Master beziehen, werden als **sekundäre Nameserver** bzw. **Slaves** bezeichnet [Ait11, S. 20 f.].⁴ Die Slaves erhalten die Kopie vom Master mittels eines **Zonentransfers** (s. Abschnitt 2.6.4).

Neben Resource-Records, die u. a. die Zuordnung von Domainnamen zu IP-Adressen erlauben und vom Betreiber des Nameservers verwaltet werden, enthält eine Zone auch Hinweise zu den Subdomains der Zone, die an andere Nameserver delegiert werden. Für jede delegierte Subdomain wird mindestens ein für diese Zone autoritativer Nameserver hinterlegt. Ein autoritativer Nameserver kann dadurch zu jeder Subdomain innerhalb seiner Zone eine Auskunft erteilen: Entweder kann er die Anfrage unmittelbar selbst beantworten oder er beantwortet sie mit einem Verweis (**Referral**) auf den Nameserver, der dafür zuständig ist [Moc87a, S. 19 f.].⁵

Eine besondere Rolle kommt den **Root-Servern** zu, die für die **Root-Zone** zuständig sind. In der Root-Zone sind für alle TLDs die jeweils zuständigen TLD-Nameserver hinterlegt. Die IP-Adressen der Root-Server sind *auf allen Nameservern hinterlegt*. Liegen einem Nameserver bei einer Namensauflösung keine weiteren Informationen vor, richtet er die Anfrage an einen Root-Server.⁶ An die Root-Server werden daher sehr hohe Verfügbarkeitsanforderungen gestellt, die durch eine stark verteilte Speicherung der Root-Zone adressiert werden. Es gibt 13 logische Root-Server, die über die Namen *a.root-servers.net* bis *m.root-servers.net* direkt angesprochen werden können.

Die meisten Root-Server verfügen über mehrere weltweit verteilte *physische Instanzen*, die durch den Einsatz von **Anycast-Routing** alle unter derselben IP-Adresse erreichbar sind (s. S. 113 bzw. [Ait11, S. 9]). Anycast-Routing bewirkt, dass die DNS-Anfragen eines Clients zur netzwerktopologisch „nächstgelegenen“ physischen Instanz weitergeleitet werden. Da die Verfügbarkeit der TLD-Nameserver für eine funktionierende Namensauflösung ebenfalls von zentraler Bedeutung ist, setzen auch viele Registrare diese Technik ein (vgl. hierzu auch die quantitative Studie von Fan et al. [Fan+12]).

⁴Die Verwendung und Bedeutung dieser Bezeichnungen hat sich im Zeitverlauf verändert. Da die Einzelheiten für diese Arbeit nicht relevant sind sei für eine präzisere und detailliertere Darstellung auf [Ait11, S. 63 ff.] verwiesen.

⁵Falls ein autoritativer Nameserver eine Anfrage für einen Domainnamen erhält, der außerhalb seiner *Domain* liegt, kann er grundsätzlich mit einem Referral auf die Root-Server antworten. Allerdings wird durch die vergleichsweise große Antwort eine Möglichkeit für Amplification-Angriffe (s. S. 23) geschaffen. Autoritative Nameserver reagieren daher in solchen Fällen üblicherweise mit einer REFUSED-Antwort (s. Abschnitt 2.6.3). Rekursive Nameserver verhalten sich in einem solchen Fall hingegen völlig anders (s. Abschnitt 2.6.2).

⁶Durch den Einsatz von Caching, das in Abschnitt 2.7 behandelt wird, kennt ein Nameserver häufig bereits den zuständigen TLD-Nameserver. Der Großteil der DNS-Anfragen kann daher ohne Beteiligung der Root-Server beantwortet werden.

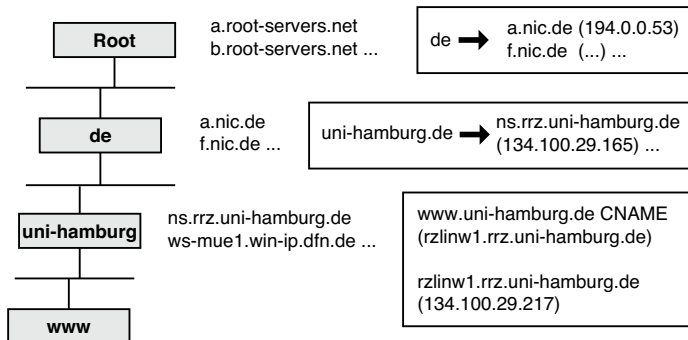


Abbildung 2.4: Verteilung des Namensraums und Delegation von Autorität am Beispiel der Domain `www.uni-hamburg.de`

Beispiel 2.2. Die Verteilung des Namensraums und die Delegation von Autorität wird in Abbildung 2.4 an einem konkreten Beispiel, dem Namen `www.uni-hamburg.de`, verdeutlicht. Die Namensauflösung erfolgt von rechts beginnend beim Label `de`. In der Root-Zone sind die autoritativen Nameserver für diese ccTLD hinterlegt: `a.nic.de`, `f.nic.de`, `l.nic.de` und `z.nic.de`. Da die autoritativen Nameserver für die `de`-Zone selbst wiederum innerhalb der `de`-Zone liegen, liefern die Root-Server neben den Domainnamen dieser TLD-Nameserver zusätzlich deren aktuelle IP-Adressen mit. Ohne diese sog. *Glue-Records*, die in [Ait11, S. 166] detaillierter beschrieben werden, wäre das weitere Traversieren des Namensbaums nicht möglich. Die ccTLD-Server kennen ihrerseits die autoritativen Nameserver, die für die Zone `uni-hamburg.de` zuständig sind. Neben dem Server `ns.rrz.uni-hamburg.de`, der als primärer Nameserver fungiert, gibt es noch fünf sekundäre Nameserver. Die Mehrzahl wird vom Regionalen Rechenzentrum der Universität Hamburg betrieben bzw. vom Rechenzentrum des Fachbereichs Informatik. Zur Erhöhung der Ausfallsicherheit gibt es neben diesen Nameservern, die alle über das IP-Netzsegment der Universität Hamburg ans Internet angebunden sind, einen Nameserver (`ws-mue1.win-ip.dfn.de`), der an einem anderen Ort (München) betrieben wird. Diese autoritativen Nameserver sind in der Lage, Anfragen zu Domainnamen innerhalb der Zone `uni-hamburg.de` zu beantworten. Für die Domainnamen `www.uni-hamburg.de` ist ein CNAME-Eintrag hinterlegt, der auf den Domainnamen `rzlinw1.rrz.uni-hamburg.de` verweist. Da für die Domain `rrz.uni-hamburg.de` ebenfalls der Server `ns.rrz.uni-hamburg.de` autoritativ ist, ist dessen IP-Adresse dort hinterlegt. □

2.4.2 Systemkomponenten für die Namensauflösung

Im vorherigen Abschnitt wurde die Infrastruktur beschrieben, mit welcher die Speicherung und Bereitstellung der Namensinformationen erfolgt. In diesem Abschnitt steht die

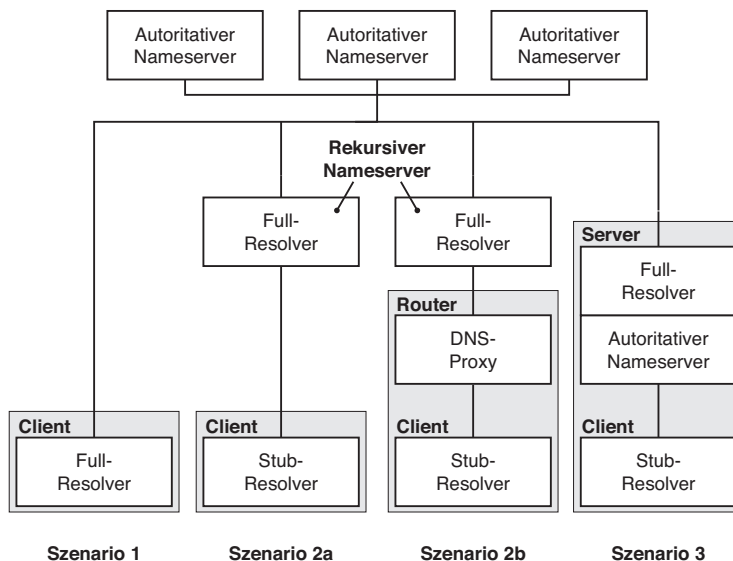


Abbildung 2.5: Szenarien zur Namensauflösung und dabei beteiligte Komponenten

Namensauflösung im Vordergrund. Die RFCs 1034 und 1035 sehen verschiedene Möglichkeiten zum Zugriff auf autoritative Nameserver vor (vgl. [Moc87a, S. 29 ff.] und [Moc87b, S. 4 ff.]).

Im Folgenden werden die typischerweise an der Namensauflösung beteiligten Komponenten anhand von vier Szenarien vorgestellt.

Abbildung 2.5 zeigt die Szenarien im Überblick. Sie greifen auf verschiedene Art und Weise auf die autoritativen Server zu. Dabei ist unter einem **Client** ein beliebiges Endgerät im Netz zu verstehen, welches DNS-Anfragen durchführt, also z. B. ein Arbeitsplatz-PC, ein Server oder Netzkomponten wie Router oder Firewalls. Auf dem Client gibt es eine Software-Komponente, welche die Namensauflösung für die Anwendungen übernimmt. Diese Komponente wird in RFC 1034 als **Resolver** bezeichnet [Moc87a, S. 29] und ist typischerweise Bestandteil des Betriebssystems. Der grau umrandete Bereich deutet die Komponenten an, die sich jeweils in der Verfügungsgewalt des Nutzers befinden.

Aus den hier dargestellten Szenarien lassen sich weitere, teilweise wesentlich komplexere Szenarien ableiten. Diese basieren jedoch im Kern stets auf den hier dargestellten Komponenten. Für eine ausführliche Übersicht sei auf [Ait11, S. 129 ff.] verwiesen.

2.4.2.1 Szenario 1: Full-Resolver

Im diesem Fall interagiert der Resolver unmittelbar mit den autoritativen Nameservern. Zur Auflösung eines Domainnamens muss er u. U. mit mehreren autoritativen Servern interagieren, um den DNS-Baum von der Wurzel abwärts zu traversieren bis er die Anfrage beantworten kann. Der Full-Resolver ist bei allen Schritten, die zur Namensauflösung nötig sind, involviert. Der Resolver muss dazu die IP-Adressen der Root-Server kennen und in der Lage sein, die Antworten der kontaktierten autoritativen Server, insbesondere die Referrals (s. S. 19) auf andere Server, zu interpretieren. Ein solcher Resolver wird auch als **Full-Resolver** bezeichnet [BH07, S. 162]. Eine Implementierung eines Full-Resolvers für Linux ist das Programm *dnscache* von Daniel J. Bernstein.⁷ Die populäre Implementierung BIND beinhaltet ebenfalls einen Full-Resolver.

2.4.2.2 Szenario 2a: Stub-Resolver und rekursiver Nameserver

Die gängigen Betriebssysteme (Windows, Linux sowie BSD- und UNIX-basierte Systeme) sowie eingebettete Systeme wie DSL-Router enthalten keinen Full-Resolver [Ait11, S. 46]. Zum einen soll dadurch die DNS-Implementierung auf Client-Systemen möglichst einfach und fehlerfrei implementiert werden können, zum anderen stehen auf eingebetteten Systemen mitunter nicht die für die Ausführung eines Full-Resolvers erforderlichen Ressourcen zur Verfügung.

Daher kommt auf den meisten Client-Systemen ein sog. **Stub-Resolver** zum Einsatz, der lediglich dazu in der Lage ist, eingehende DNS-Anfragen an einen Full-Resolver weiterzuleiten [BH07, S. 162]. Das Betriebssystem stellt die Funktionalität des Stub-Resolvers über verschiedene Systemfunktionen zur Verfügung. Auf den gängigen Systemen (Windows, viele Unix- und Linux-Derivate sowie Mac OS X) wurde dazu ursprünglich die im POSIX-Standard POSIX.1-2001 (IEEE Std 1003.1-2001 [IEE01]) definierte Funktion **gethostbyname** verwendet. Wegen verschiedener Einschränkungen (u. a. der fehlenden IPv6-Unterstützung) gilt *gethostbyname* seit dem Standard POSIX.1-2008 (IEEE Std 1003.1-2008 [IEE08]) als obsolet. Zur Namensauflösung sollen stattdessen die Funktionen **getaddrinfo** bzw. **getnameinfo** verwendet werden.⁸ Auf Windows-Systemen existiert weiterhin die Win32-Funktion **WSAAsyncGetHostByName**, von deren Verwendung jedoch inzwischen abgeraten wird [Mic12b].

Der vom Stub-Resolver verwendete Full-Resolver wird auf einem dedizierten Server betrieben, der in der Praxis häufig *auch* als „Nameserver“ oder „Resolver“ bezeichnet wird.⁹

⁷*dnscache* ist Teil von Bernsteins *djbdns*-Distribution (vgl. <http://cr.yp.to/djbdns.html> bzw. [Bera]). Weitere Informationen zu *dnscache* sind unter <http://cr.yp.to/djbdns/dnscache.html> verfügbar [Berb].

⁸Implementierungsdetails zu den Funktionen können den Man-Pages entnommen werden, welche im Internet u. a. unter <http://linux.die.net/man/3/gethostbyname> und <http://linux.die.net/man/3/getaddrinfo> abrufbar sind.

⁹Zum Beispiel wird der Full-Resolver bei Linux- bzw. Unix-Systemen in der Datei */etc/resolv.conf* mit dem Parameter „nameserver“ festgelegt.

Dieser Nameserver verwaltet jedoch keine eigenen Zonen, sondern wird ausschließlich zur Auflösung beliebiger Domainnamen verwendet. Der Stub-Resolver richtet an den Full-Resolver dazu sog. **rekursive DNS-Anfragen**, mit denen er zum Ausdruck bringt, dass der Full-Resolver den Referrals der autoritativen Nameservern so lange folgen soll, bis er den Resource-Record (s. Abschnitt 2.5) für den angefragten Namen zurückliefern kann. Eine genauere Beschreibung der DNS-Anfragetypen folgt in Abschnitt 2.6.2.

Zur Differenzierung der Bedeutung des unscharfen Begriffs „Nameserver“ haben sich in der Literatur unterschiedliche Nomenklaturen etabliert, u. a. werden die Begriffe „Cache-Server“ bzw. „Proxy-Server“ [BH07, S. 162], „DNS Resolver“ [Ait11, S. 17 f.] sowie „nicht-autoritativer Nameserver“ [MS12a, S. 739] verwendet, um einen Nameserver zu bezeichnen, der von einem Stub-Resolver zur vollständigen Abwicklung der Namensauflösung genutzt wird. Zur Abgrenzung werden autoritative Nameserver mitunter auch „Content-Server“ [BH07, S. 163] genannt.

In dieser Arbeit wird zur Differenzierung konsequent auf die Terminologie der RFCs 1034 und 1035 zurückgegriffen: Dort wird zwischen **autoritativen Nameservern**, welche die Namensinformationen speichern und bereitstellen, und **rekursiven Nameservern**, die einen Full-Resolver enthalten und von den Clients zur Namensauflösung verwendet werden, unterschieden.¹⁰

Die Verwendung von rekursiven Nameservern kann die Bearbeitungsdauer einer DNS-Anfrage erheblich reduzieren, da der Full-Resolver die Antworten der autoritativen Server zwischenspeichert. Eine detailliertere Beschreibung der Caching-Mechanismen im DNS folgt in Abschnitt 2.7. Rekursive Nameserver werden entweder innerhalb eines Unternehmensnetzes oder vom Internet-Zugangsanbieter betrieben. Sie stehen üblicherweise lediglich einer geschlossenen Benutzergruppe zur Verfügung.

Offene Resolver Im Internet gibt es allerdings zahlreiche öffentlich verfügbare rekursive Nameserver, die Anfragen von beliebigen Clients entgegennehmen. Diese Server werden als „**offene Resolver**“ (engl. „**Open Resolvers**“) bezeichnet und können mitunter für Denial-of-Service-Angriffe (sog. „amplification attacks“) missbraucht werden [US-13]. Die Zahl der offenen Resolver ist rückläufig, verbleibt jedoch auf hohem Niveau: Laut Untersuchungen des Dienstleisters „The Measurement Factory“ existierten am 20. Oktober 2013 insgesamt 78 399 offene Resolver, während ein Jahr zuvor noch 124 501 offene rekursive Nameserver gelistet waren.¹¹

¹⁰Der Begriff „rekursiver Nameserver“ stellt als Differenzierungsmerkmal die im Vergleich zu den autoritativen Servern wesentliche Eigenschaft heraus, nämlich rekursive Anfragen entgegenzunehmen. Der bei rekursiven Nameservern optional zur Performanzsteigerung existierende Cache wird dabei als nachrangige Eigenschaft angesehen. Im Gegensatz dazu erscheint der u. a. in [BH07, S. 162] verwendete Begriff „Cache-Server“ weniger geeignet, da er eine Einschränkung auf rekursive Server, die einen Cache verwenden, vornimmt. Der Begriff „DNS Resolver“ [Ait11, S. 17 f.] erscheint ebenfalls wenig geeignet, da dann keine begriffliche Unterscheidung zwischen der zur Namensauflösung verwendeten Softwarekomponente, dem Resolver, und dem im Netz platzierten Nameserver möglich ist.

¹¹Ausführliche Ergebnisse unter <http://dns.measurement-factory.com/surveys/openresolvers/ASN-reports/>.

Bei einem **Amplification-Angriff** sendet ein Angreifer gezielt DNS-Anfragen für Domainnamen, welche bei der Namensauflösung in einer großen DNS-Antwort resultieren, an einen rekursiven Nameserver. Zudem trägt der Angreifer die IP-Adresse des Opfers im UDP-Paket (s. Abschnitt 2.6.1), mit dem er die DNS-Anfrage an den Nameserver sendet, als Absender ein (sog. *IP-Spoofing*), sodass der Nameserver die große DNS-Antwort an das Opfer sendet. Der Angreifer kann dadurch mit einer relativ geringen Senderate den Nameserver dazu bringen, mit einer erheblich höheren Rate an das Opfer zu senden. Das Verstärkungsverhältnis (engl. „amplification factor“) kann nach Angaben in [Kam+07] bis zu 60 betragen. Durch die parallele Verwendung mehrerer offener Resolver kann der Angreifer die Verfügbarkeit des Opfers erheblich beeinträchtigen. Geeignete Techniken zur Verhinderung von Amplification-Angriffen bestehen zum einen in der Beschränkung des Zugriffs auf rekursive Nameserver, zum anderen in der Filterung des (ausgehenden) Datenverkehrs, um IP-Spoofing zu unterbinden.

2.4.2.3 Szenario 2b: Stub-Resolver, DNS-Proxy und rekursiver Nameserver

Die Kommunikation zwischen Stub-Resolver und rekursivem Nameserver erfolgt nicht immer auf direktem Wege. Mitunter tritt an die Stelle des rekursiven Nameservers auch ein **DNS-Proxy**, auch als **DNS-Forwarder** [Bel09, S. 1] bezeichnet, der eingehende DNS-Anfragen an einen rekursiven Nameserver weiterleitet [Ait11, S. 18].

DNS-Proxys kommen vor allem auf den im Heimbereich eingesetzten DSL-Routern zum Einsatz (vgl. die Untersuchungsergebnisse in [BP08]). Die DSL-Router fungieren üblicherweise als NAT-Gateway und teilen den daran angeschlossenen Endgeräten ihre Netzwerkkonfiguration per DHCP (Dynamic Host Configuration Protocol; spezifiziert in [Dro97]) mit. Den Clients wird dadurch die interne IP-Adresse des Routers (vgl. RFC 1918 [Rek+96, S. 4]) als rekursiver Nameserver bekanntgemacht. Dieses Vorgehen ermöglicht es dem Router, auch dann DNS-Anfragen zu bearbeiten, wenn ihm der rekursive Nameserver des Internet-Providers (noch) nicht bekannt ist. Weiterhin eröffnet dies dem Router die Möglichkeit, einen speziellen Domainnamen (z. B. *fritz.box* bei Routern der AVM GmbH, vgl. [AVM14, S. 38]) in seine eigene interne IP-Adresse aufzulösen, um einen komfortablen Zugriff auf seine Benutzeroberfläche anzubieten.

Ein DNS-Proxy ist äußerlich nicht von einem rekursiven Nameserver zu unterscheiden. Im Gegensatz zu einem rekursiven Nameserver, auf dem DNS-Anfragen mit Hilfe eines Full-Resolvers aufgelöst werden, fehlt DNS-Proxys, auch da sie häufig auf leistungsschwacher Hardware ausgeführt werden, diese Funktionalität. Sie sind üblicherweise nicht dazu in der Lage, selbst mit Nameservern zu interagieren bzw. DNS-Nachrichten zu interpretieren oder zu erzeugen. Stattdessen beschränken sie sich auf das Weiterleiten von DNS-Nachrichten. Im Idealfall erfolgt diese Weiterleitung völlig *transparent*, also für die beteiligten Komponenten (Stub-Resolver und rekursiver Nameserver) nicht erkennbar. Fehlerhafte Implementierungen können jedoch zu Kommunikationsproblemen führen, wenn die Verwendung des Transportprotokolls TCP oder der Protokollerweiterungen

EDNS oder DNSSEC nicht unterstützt wird, weswegen RFC 5625 spezielle Anforderungen an DNS-Proxys stellt [Bel09].

2.4.2.4 Szenario 3: Autoritativer Nameserver mit aktivierter Rekursion

Dieses Szenario unterscheidet sich von den vorherigen Szenarien dadurch, dass der zur Namensauflösung verwendete Nameserver nun selbst für mindestens eine Zone autoritativ ist. Gleichzeitig nimmt er rekursive Anfragen für beliebige Domainnamen entgegen, die er wie ein rekursiver Nameserver mit Hilfe fremder autoritativer Nameserver beantwortet. Auf eine solche **hybride Konfiguration** [Ait11, S. 129] wird zurückgegriffen, wenn der getrennte Betrieb von autoritativen und rekursiven Nameservern aus ökonomischen Erwägungen nicht in Frage kommt. Im Hinblick auf die Gewährleistung von Integrität und Verfügbarkeit weisen hybride Konfigurationen jedoch Nachteile auf. Eine strikte Aufteilung der Aufgaben auf getrennte Nameserver wird empfohlen. Für detailliertere Erläuterungen zum Betrieb hybrider Konfigurationen sei auf [Ait11, S. 70,129], [Bus+00] sowie [Berc] verwiesen.

Bei konkreten Implementierungen wird die Gruppe der Clients, von denen rekursive Anfragen entgegengenommen werden, üblicherweise anhand der Absender-IP-Adresse eingeschränkt, z. B. durch eine Firewall oder durch die Definition einer entsprechenden *Whitelist*. Bei der Nameserver-Implementierung BIND wird diese etwa durch den Parameter „*allow-recursion { address_match_list }*“ konfiguriert [Ait11, S. 461].

Gestaltungsmöglichkeiten gibt es auch hinsichtlich Inhalt und Erreichbarkeit der autoritativ verwalteten Zone: diese kann entweder Teil des öffentlichen DNS-Namensraums sein oder davon unabhängig einen eigenen, z. B. nur im internen Unternehmensnetz verwendeten Namensraum definieren. Weiterhin können z. B. IP-basierte Zugriffsbeschränkungen für die Zone implementiert werden, um z. B. zu verhindern, dass unternehmensinterne Domainnamen aus dem öffentlichen Netz aufgelöst werden können (vgl. Abschnitt 3.7.2).

Der Betrieb solcher **Split-Konfigurationen** [Ait11, S. 71 f.] bringt jedoch zusätzliche Einschränkungen mit sich: Clients müssen zur korrekten Auflösung interner Namen zwingend den internen rekursiven Nameserver verwenden. Bei Verwendung eines anderen rekursiven Nameservers, etwa eines Nameservers, der von einem Drittanbieter betrieben wird (vgl. Abschnitt 2.8.4), können die internen Namen nicht mehr aufgelöst werden.

2.5 Resource-Records

Die **Resource-Records (RRs)** bilden die „atomaren Einheiten des DNS“ [BH07, S. 166]. Sie erlauben es, für einen Domainnamen mehrere Attribute unterschiedlichen Typs zu hinterlegen. RRs werden im DNS universell verwendet: die RFCs 1034 und 1035 geben nicht nur vor, wie RRs auf den autoritativen Nameservern zu hinterlegen sind, sondern sie definieren auch, wie RRs innerhalb von DNS-Anfragen und -Antworten übertragen werden [BH07, S. 166].

In diesem Abschnitt werden der Aufbau und die wichtigsten Typen von RRs beschrieben. Die folgende Darstellung orientiert sich an der Syntax, die in den RFCs 1034 und 1035 verwendet wird [Moc87a].

RRs bestehen gemäß RFC 1035 aus den folgenden Feldern [Moc87b, S. 11 f.]:

<NAME> [<TTL>] <CLASS> <TYPE> <RDATA>

Die einzelnen Bestandteile haben folgende Bedeutung: *NAME* ist der Name (FQDN), für den dieser RR Informationen enthält. Dabei handelt es sich entweder um einen *Domainnamen*, für den der Nameserver autoritativ ist, einen *Hostnamen*, der sich in der Domain befindet oder den *Namen einer Subdomain*, die an einen anderen Nameserver delegiert wird. Das Feld *TTL* (*time-to-live*) ist ein optional anzugebender Integerwert, der festlegt, wie viele Sekunden ein RR maximal in einem Cache aufbewahrt werden darf (vgl. Abschnitt 2.7). Das Feld *CLASS* war dafür vorgesehen, die Netzklasse, zu der der Eintrag gehört, auszudrücken. RFC 1035 definiert Klassen für das Internet, das CSNET, das CHAOS-Netz und Hesiod. Heute wird hier stets der Wert *IN* (Internet) eingetragen. *TYPE* ist der eigentliche Typ des RRs und *RDATA* (*Resource Data*) enthält die Daten, die für den RR hinterlegt werden, z. B. eine IP-Adresse oder einen Domainnamen. Der genaue Aufbau des Felds *RDATA* hängt davon ab, welcher *TYPE* verwendet wird. Im Folgenden werden die am häufigsten verwendeten RR-Typen kurz beschrieben. Eine detailliertere Übersicht findet sich etwa in [BH07, S. 167 ff.].

In jeder Zone gibt es einen RR vom Typ **SOA (Start of Authority)**, in dem der primäre Nameserver, die E-Mail-Adresse einer Kontaktperson sowie Zeitintervalle, die u. a. für den Zonentransfer relevant sind, definiert werden. **A-Records** enthalten die IPv4-Adresse für einen Host, die später hinzugefügten **AAAA-Records** die IPv6-Adresse [Tho+03].

PTR-Records erlauben eine sog. **Rückwärtsauflösung** (engl. „reverse lookup“) von IP-Adressen in Domainnamen mittels der speziellen Domains *in-addr.arpa* [BH07, S. 166] bzw. *ip6.arpa* [Tho+03, S. 3]. Dabei werden die Oktette einer IP-Adresse in umgekehrter Reihenfolge angeordnet: Um den Domainnamen zu erhalten, der für die Adresse 43.10.2.15 hinterlegt ist, wird also eine Anfrage für den PTR-Record der Domain *15.2.10.43.in-addr.arpa* durchgeführt.

In jeder Zone müssen die Nameserver angegeben werden, die für diese Zone autoritativ sind. Diese werden in einzelnen **NS-Records** aufgeführt, die als Namen jeweils den Domainnamen enthalten und im Feld *RDATA* den Namen eines Nameservers. NS-Records dienen auch dazu, die Nameserver zu definieren, die für eine delegierte Subdomain zuständig sind. In diesem Fall entspricht der *NAME* dem Domainnamen der Subdomain.

Weiterhin gibt es **CNAME-Records**, mit denen ein Name als Alias für einen anderen Domainnamen, der im Feld *RDATA* angegeben wird, definiert werden kann. Der Domainname, auf den der Alias zeigt, darf sich auch außerhalb der Domain befinden, für die der CNAME-Record definiert wird [Ait11, S. 38]. **MX-Records** weisen den bzw. die Mailserver einer Domain aus. Das Feld *RDATA* enthält hier ein Tupel bestehend aus einer Prioritätszahl und dem Domainnamen eines Mailservers, wobei Mailserver mit einem

niedrigeren Prioritätswert bevorzugt für die Zustellung verwendet werden. Mit den bereits in RFC 1035 definierten **TXT-Records** können für einen Domainnamen beliebige Texte (Strings) hinterlegt werden [Moc87b, S. 20], um die Flexibilität des Systems zu erhöhen.

Später wurden Record-Typen hinzugefügt, welche das Auffinden von Diensten (Service-Discovery, s. u. a. [CK13]) mittels DNS-Anfragen adressieren. So kann ein Administrator mit den in RFCs 2782 und 3958 standardisierten **SRV-Records** bekannt geben, welcher Host für einen bestimmten Dienst innerhalb einer Domain zuständig ist [GVE00; DN05]. SRV-Records werden etwa dazu verwendet, um den SIP-Server aufzufinden, welcher eingehende Voice-over-IP-Anrufe für die Nutzer einer Domain entgegennehmen soll [RS02]. Einen weitergehenden Ansatz stellt der in RFC 2915 standardisierte **NAPTR-Record-Typ** (Name Authority Pointer) dar [MD00a]. NAPTR-Records enthalten reguläre Ausdrücke, mit denen anfragende Clients einen Domainnamen in einen anderen Domainnamen bzw. in einen URI (Uniform Ressource Identifier; [BLFM05]) umwandeln können. Nach der Umwandlung kann die Namensauflösung ggf. in einem anderen Namensraum bzw. durch einen anderen Namensdienst fortgesetzt werden. NAPTR-Records werden u. a. zur Realisierung der DNS-Anwendungen „ENUM“ und „ONS“ eingesetzt (s. Abschnitt 2.9.4 und Abschnitt 2.9.5). Die Konzepte zur Service-Discovery mittels DNS-Anfragen wurden später konsolidiert. Sie wurden in den RFCs 3401 bis 3405 standardisiert [Mea02a; Mea02b; Mea02c; Mea02d; Mea02e].

Neben den oben erwähnten NS- und MX-Records gibt es noch weitere Record-Typen, die für einen Domainnamen mehrfach auftreten können. Das Vorhandensein mehrerer A- bzw. AAAA-Records drückt aus, dass ein Host unter mehreren IP-Adressen erreicht werden kann bzw. mehrere Hosts unter einem Domainnamen erreichbar sind. Die in der Praxis verwendeten Nameserver-Implementierungen beantworten Anfragen in diesem Fall entweder mit einer Teilmenge oder mit allen verfügbaren RRs. Die Reihenfolge kann dabei zufällig bzw. durch zyklisches Vertauschen (sog. *Round-Robin-Verfahren*) variiert werden, um eine Lastverteilung zu erreichen. Als komplementäre Situation kann der Fall angesehen werden, dass mehrere unterschiedliche Domainnamen mittels A-Records auf dieselbe IP-Adresse zeigen. Von dieser Möglichkeit wird beim Virtual-Hosting Gebrauch gemacht, bei dem mehrere Webseiten mit unterschiedlichen Domains von einem einzigen Webserver ausgeliefert werden [Apa12]. Im DNS gibt es also keine eindeutige Zuordnung zwischen Domainnamen und IP-Adressen.

Beispiel 2.3. Die Definition von RRs soll anhand einer konkreten Zonendatei (für die Nameserver-Software BIND) illustriert werden, die in Abbildung 2.6 abgebildet ist. Dieses Beispiel ist eine Weiterentwicklung der Erläuterungen in [Ait11, S. 27]. In Zeile 1 wird als Standard-Wert für die TTL ein Tag festgelegt. Dieser Wert gilt für alle RRs, bei denen die TTL nicht explizit angegeben wird. In Zeile 2 wird die **Basis-Domain** der Zone definiert. Alle in der Zonendatei verwendeten Domainnamen, die am Ende keinen abschließenden Punkt aufweisen, werden um die Basis-Domain zu FQDNs erweitert.

Es folgt der SOA-Record (Zeilen 4–10), in dem *ns1.example.com* als primärer Nameserver ausgewiesen wird. Die Seriennummer und die drei Intervalle dienen der Koordination

des Zonentransfers an sekundäre Nameserver. Mit der *NXDOMAIN-TTL* wird festgelegt, wie lange rekursive Nameserver eine *NXDOMAIN*-Antwort (s. Beschreibung der Fehlerbehandlung in Abschnitt 2.6.3) im Cache aufbewahren dürfen.

In den Zeilen 11 und 12 werden die autoritativen Nameserver der Zone *example.com* definiert. Der erste Nameserver befindet sich innerhalb der eigenen Zone, weswegen seine IP-Adresse mit einem A-Record in Zeile 18 definiert wird. Die Subdomain *sub.example.com* wird an zwei andere Nameserver (*ns1.sub.example.com* und *ns2.sub.example.com*) delegiert. Diese werden in den Zeilen 13 und 14 ausgewiesen. Da sich die beiden autoritativen Nameserver von *sub.example.com* selbst innerhalb der delegierten Domain befinden, sind in der *example.com*-Zone entsprechende Glue-Records erforderlich (Zeilen 19 und 20). Die TTL der Nameserver wird auf einen hohen Wert (eine Woche) gesetzt, damit rekursive Nameserver bei den meisten Anfragen für Domains unterhalb von *example.com* die autoritativen Nameserver bereits aus ihrem Cache entnehmen können

In den Zeilen 15 und 16 werden die Mailserver der Domain *example.com* definiert. In den verbleibenden Zeilen folgen die Hosts, die innerhalb der Zone bekannt sind. Hervorzuheben ist die Tatsache, dass die A-Records für *uriel* und *mail* auf dieselbe IP-Adresse zeigen. Der Host mit der Adresse 192.168.1.3 kann also über beide Namen erreicht werden. Weiterhin gibt es zwei Einträge für *web*, die auf unterschiedliche IP-Adressen zeigen. Die kurze TTL von einer Minute kommt zum Einsatz, um über die DNS-Anfragen eine einfache Lastverteilung auf zwei Server zu realisieren. Der Name *ftp.example.com* wird mittels eines CNAME-Records als Alias definiert, der auf eine andere Domain zeigt. Das freistehende „@“ in Zeile 27 wird durch den Domainnamen ersetzt, der als *ORIGIN* definiert ist [Ait11, S. 30], d. h. dieser Eintrag legt fest, wie Anfragen für den A-Record der Domain *example.com* selbst zu beantworten sind. Im Beispiel wird dieselbe IP-Adresse zurückgegeben, auf die auch *www.example.com* zeigt, ein in der Praxis übliches Verhalten.

Bei Zeile 28 handelt es sich um einen sog. **Wildcard-Eintrag** [Ait11, S. 197 f.]. Ist dieser vorhanden, so wird er zur Beantwortung aller DNS-Anfragen verwendet, für die es keine passenden RRs gibt. □

2.6 Verwendete Protokolle

Im DNS gibt es ein einheitliches Protokoll, das den Ablauf der Kommunikation zwischen Nameservern regelt und das Datenformat der dabei übermittelten Nachrichten vorgibt. Es wird für die Kommunikation zwischen Stub-Resolver und rekursivem Nameserver, die Kommunikation zwischen rekursivem Nameserver und autoritativen Nameservern sowie die Kommunikation zwischen autoritativen Nameservern untereinander (sog. Zonentransfer) verwendet.

Das DNS-Protokoll ist ein Anfrage-Antwort-Protokoll, das stets von einem sog. **DNS-Client** initiiert wird, indem dieser eine **DNS-Anfrage** (engl. „query“ oder „request“) an

```

1 $TTL 1d ; Standard-Wert für die TTL der Zone
2 $ORIGIN example.com. ; Basis-Domain der Zone
3
4 @           IN      SOA  ns1.example.com. hostmaster.example.com. (
5                               2012030700 ; Seriennummer
6                               12h ; Refresh-Intervall
7                               15m ; Refresh-Retry-Intervall
8                               3w ; Expiration-Intervall
9                               2h ; NXDOMAIN-TTL
10                              )
11           1w  IN      NS  ns1.example.com.
12           1w  IN      NS  ns2.example.net.
13 sub       1w  IN      NS  ns1.sub.example.com.
14 sub       1w  IN      NS  ns2.sub.example.com.
15           IN      MX  10 mail.example.com.
16           IN      MX  20 mail.example.net.
17
18 ns1        IN      A  192.168.1.2
19 ns1.sub    IN      A  192.168.2.1
20 ns2.sub    IN      A  192.168.2.2
21 mail       IN      A  192.168.1.3
22 uriel      IN      A  192.168.1.3
23 web        60  IN      A  192.168.1.11
24 web        60  IN      A  192.168.1.12
25 www        IN      A  192.168.1.21
26 ftp        IN      CNAME ftp.example.net.
27 @          IN      A  192.168.1.21
28 *          IN      A  192.168.1.11

```

Abbildung 2.6: Inhalt der Zonendatei für die Domain *example.com* (s. Beispiel 2.3)

einen Nameserver sendet. Der kontaktierte Nameserver reagiert darauf mit einer **DNS-Antwort** (engl. „reply“ oder „response“). Ein und derselbe Nameserver kann gleichzeitig in der Rolle eines DNS-Clients und in der Rolle eines DNS-Servers auftreten. Ein Stub-Resolver tritt hingegen immer als DNS-Client auf.

Dieser Abschnitt ist wie folgt aufgebaut: In Abschnitt 2.6.1 werden zunächst die Transportprotokolle betrachtet, die für die Übermittlung von DNS-Nachrichten verwendet werden. Im Anschluss daran werden in Abschnitt 2.6.2 die zwei existierenden Varianten der Namensauflösung beschrieben. Dabei wird deutlich, dass insbesondere die Beobachtungsmöglichkeiten auf den rekursiven Nameservern von Interesse sind. Weiterhin wird auf die Fehlerbehandlung eingegangen (s. Abschnitt 2.6.3), die im Zusammenhang mit der später behandelten Manipulation von NXDOMAIN-Antworten (s. Abschnitt 2.8.3) von Bedeutung ist. Abschließend wird in Abschnitt 2.6.4 der Ablauf des Zonentransfers erläutert, der bei der Zusammenfassung der bisherigen Sicherheitsbetrachtungen in Kapitel 3 relevant wird.

2.6.1 Verwendete Transportprotokolle

Der Transport von DNS-Anfragen kann sowohl mit dem verbindungslosen Protokoll **UDP** (*User Datagram Protocol*, spezifiziert in RFC 768 [Pos80]) als auch mit dem verbindungsorientierten Protokoll **TCP** (*Transmission Control Protocol*, spezifiziert in RFC 793 [Pos81b]) erfolgen [Moc87b, S. 32]. In beiden Fällen wird der Server-Port 53 verwendet. TCP wird vor allem für den Zonentransfer vom primären Nameserver zu den sekundären Nameservern verwendet. Für DNS-Anfragen und -Antworten wird hingegen üblicherweise UDP verwendet [BH07, S. 175].

Trotz der geringeren Zuverlässigkeit wird UDP für die Übermittlung von DNS-Anfragen bevorzugt. Dafür gibt es zwei Gründe: Bei Verwendung von TCP steigt zum einen die Antwortzeit, also die Zeit zwischen Absenden der Anfrage und Eintreffen der Antwort. Zum anderen ist der Protokoll-Overhead von TCP im Vergleich zu UDP größer, da das DNS-Protokoll vorsieht, jede DNS-Anfrage in einer separaten TCP-Verbindung zu übermitteln. Die primäre Ursache für die höhere Antwortzeit und die niedrigere Effizienz von TCP ist der *3-Way-Handshake*, der beim Verbindungsaufbau durchlaufen wird [BH07, S. 121]. Ein allgemeiner Wechsel zu TCP wird auch deswegen kritisch gesehen, weil dies zu einer erhöhten Belastung stark frequentierter Nameserver führen würde. Diese müssten zusätzlich den Zustand aller aktiven TCP-Verbindungen verwalten. Da die Bearbeitung von DNS-Anfragen an sich völlig zustandslos ist und in einem einzigen Schritt erfolgen kann, erschien den Entwicklern UDP besser geeignet.

Da UDP im Gegensatz zu TCP die Zustellung einer Nachricht nicht garantiert, können Anfragen oder Antworten auf dem Transportweg verloren gehen. DNS-Clients senden Anfragen daher erneut ab (*Retransmission*), wenn sie innerhalb eines festgelegten Zeitintervalls (Timeout) keine Antwort empfangen [Moc87b, S. 32]. Je nachdem wie die Retransmission-Strategie umgesetzt ist, kommt es zu mehr oder weniger starken Verzögerungen, wenn DNS-Pakete über stark ausgelastete Verbindungen bzw. Verbindungen mit einer hohen Paketverlustrate übertragen werden. Wie sich in Abschnitt 4.3.5 zeigen wird, ergibt sich dadurch eine möglicherweise unerwünschte Beobachtungsmöglichkeit: Ein Beobachter kann Unterschiede im Retransmission-Verhalten zur Identifikation von Betriebssystemen und Web-Browsern ausnutzen.

2.6.1.1 Größenbegrenzung für Übermittlung per UDP

Eine weitere Einschränkung ergibt sich aus der Fragmentierung von IP-Paketen auf dem Transportweg. Die Vorgaben des IPv4-Standards bezüglich der Mindestanforderungen an Implementierungen sind aus heutiger Sicht sehr konservativ. Für die Entwicklung des DNS war insbesondere entscheidend, dass IPv4-Endgeräte lediglich dazu in der Lage sein müssen, „Datagramme“ (gemeint sind hier IP-Pakete [Pos81a, S. 1]) mit einer Größe von 576 Byte zu empfangen bzw. diese aus einzelnen IP-Paket-Fragmenten wieder zusammen-

zusetzen [Pos81a, S. 13, 25].¹² IPv4-Implementierungen müssen zur Umsetzung dieser Anforderung über einen entsprechend dimensionierten Empfangspuffer verfügen, in dem die einzelnen Fragmente eines IP-Paketes wieder zusammengesetzt werden, bevor sie an die nächsthöhere Schicht weitergereicht werden. Übersteigt eine PDU (*Protocol Data Unit*, in diesem Fall DNS-Nachricht) die Größe des Empfangspuffers, wird sie von der IPv4-Implementierung in Fragmenten an die nächsthöhere Schicht weitergereicht. Da das DNS-Protokoll keine Möglichkeit vorsieht, fragmentierte PDUs wieder zusammenzusetzen, müssen DNS-Anfragen und -Antworten von der Transportschicht jedoch stets unfragmentiert an die in der Anwendungsschicht befindliche Resolver-Implementierung übergeben werden. Daher wurde in RFC 1035 für DNS-Nachrichten, die über UDP übertragen werden, eine **maximale Größe von 512 Byte** festgelegt, um inklusive des IP-Headers (typischerweise 20 Byte) und des UDP-Headers (8 Byte) unterhalb der zugesicherten IP-Paketgröße von 576 Byte zu bleiben [Moc87b, S. 32].

Für *Anfragen* stellt die Limitierung der Nachrichtenlänge keine Einschränkung dar, da diese durch die Begrenzung der Länge eines Domainnamens auf 255 Byte (s. Abschnitt 2.3) die maximale Nachrichtenlänge ohnehin nicht überschreiten können. DNS-Antworten können in der Praxis jedoch durchaus größer werden als 512 Byte, wenn eine entsprechend große Anzahl an RRs übermittelt werden soll. Tritt dieser Fall ein, antwortet der Nameserver dem Resolver mit einem UDP-Datagramm, das lediglich einen Teil der DNS-Antwort enthält. In dieser Antwort ist das *Truncation-Flag* gesetzt, das ausdrückt, dass die Antwort unvollständig ist [BH07, S. 175]. Der DNS-Client muss die Anfrage dann per TCP wiederholen, wodurch sich die Antwortzeit für die Namensauflösung erheblich erhöht.

2.6.1.2 Absehbares Wachstum der DNS-Antworten durch neue Entwicklungen

Sowohl durch die Einführung von IPv6 als auch DNSSEC (s. Abschnitt 3.6.3) wird die Länge vieler DNS-Antworten die Größenbeschränkung von 512 Byte überschreiten [Ait11, S. 41]. Würden etwa anstelle von IPv4-Adressen (32 bit) in Zukunft nur noch IPv6-Adressen (128 bit) übertragen werden, stiege der Platzbedarf jedes A-Records um 12 Byte; würden neben den bereits existierenden A-Records zusätzliche AAAA-Records aufgenommen, stiege der Platzbedarf pro Adresse sogar um 28 Byte [Rik+04].

Um abzuschätzen, welcher Anteil von DNS-Antworten durch die Migration auf IPv6 die Größenbeschränkung überschreiten würde, haben Rikitake et al. im Jahr 2003 anhand von aufgezeichneten DNS-Anfragen eine Untersuchung durchgeführt [Rik+04]. Sie fügten bei den beobachteten DNS-Antworten für jeden A-Record zusätzlich einen AAAA-Record ein und ermittelten die daraus resultierende Nachrichtenlänge. In ihrem Experiment stieg

¹²Diese Anforderung impliziert, dass auch Datagramme mit einer Größe von weniger als 576 Byte auf dem Transportweg fragmentiert werden können. Die maximale Paketgröße, die garantiert ohne Fragmentierung weitergeleitet wird, wird in RFC 791 mit 68 Byte angegeben [Pos81a, S. 25]. Diese Anforderung ergibt sich aus der maximalen Größe eines IPv4-Headers (60 Byte) und der minimalen Länge eines Fragments (8 Byte).

der Anteil der Antworten, welche über 512 Byte lang waren, ausgehend von 0,04 % bei IPv4 auf 4,65 %.

Im Vergleich zu IPv6 hat die DNSSEC-Einführung noch wesentlich stärkere Auswirkungen auf die Größe der DNS-Antworten. Zusätzlich zu den tatsächlich angefragten Records sieht DNSSEC die Übermittlung einer zugehörigen digitalen Signatur in einem RRSIG-Record vor. Ein RRSIG-Record weist gemäß RFC 4034 bei Verwendung des derzeit bei DNSSEC üblicherweise eingesetzten RSA-Verfahrens eine Länge von $32 + |\text{Name der Zone}| + |\text{Schlüssellänge}|$ auf (siehe [Are+05c, S. 7] und die Erläuterung der Länge einer RSA-Signatur in RFC 5702 [Jan09, S. 5])¹³. Bei einer Schlüssellänge von 1024 bit ergibt sich somit ein Wachstum der Antworten um mindestens 160 Byte. Der zu erwartende Anteil der Antworten mit einer Länge von mehr als 512 Byte wurde nie empirisch ermittelt, da diese Größenbeschränkung durch die Einführung von EDNS0 (s. Abschnitt 2.6.1.3) zwischenzeitlich obsolet wurde. In Simulationen haben Ager et al. jedoch anhand von im Jahr 2002 aufgezeichneten DNS-Anfragen gezeigt, dass bei Verwendung von RSA-Signaturen zwischen 20 und 30 % der DNS-Antworten eine Länge von mehr als 1228 Byte aufweisen [ADF06].

2.6.1.3 Extension Mechanisms for DNS (EDNS0)

In RFC 2671 [Vix99], der im Jahr 1999 publiziert wurde, wird eine abwärtskompatible Erweiterung des DNS-Protokolls definiert, mit der DNS-Nachrichten auch dann per UDP transportiert werden können, wenn sie größer sind als 512 Byte. Dabei wird ausgenutzt, dass die meisten Endgeräte inzwischen auch IP-Pakete mit einer Länge von mehr als 576 Byte empfangen und ggf. aus Fragmenten zusammensetzen können [Vix99, S. 2].

Das Verfahren basiert auf einem bereits im November des Jahres 1997 vorgestellten Internet-Draft (*draft-ietf-dnsind-udp-size-00*; die letzte Revision des Dokuments datiert auf den Juni des Jahres 1998 [Eas98]). In diesem Dokument wird vorgeschlagen, das bislang unbenutzte *RCODE-Feld* (4 Bit) in DNS-Anfragen zu nutzen, um die von einem DNS-Resolver maximal unterstützte Nachrichtenlänge zu kodieren. Im Gegensatz zu einer generellen Anhebung der maximalen Nachrichtenlänge wird dadurch eine höhere Flexibilität und insbesondere Abwärtskompatibilität erreicht. Wegen der geringen Größe des *RCODE-Felds* wurden allerdings nur acht verschiedene Nachrichtengrößen spezifiziert, die einen Bereich von 512 bis 12000 Byte abdecken.

Die in RFC 2671 spezifizierten *Extension Mechanisms for DNS*, welche zur Unterscheidung von künftigen Erweiterungen häufig mit der Abkürzung *EDNS0* bezeichnet werden, greifen die Idee aus [Eas98] auf: EDNS0-fähige DNS-Clients übermitteln in ihren Anfragen einen neu eingeführten *OPT-Pseudo-Resource-Record* an den Nameserver. Im *CLASS-Attribut* (16 Bit) dieses OPT-RRs wird dem Nameserver die **Sender-UDP-Payload-Size**, die maximale Länge (in Byte) der UDP-Nutzdaten, welche der Client aus einzelnen Fragmenten wieder

¹³ Abweichend hiervon geben Ager et al. in [ADF06] eine Größe von $46 + |\text{Name der Zone}| + |\text{Schlüssellänge}|$ Byte für einen RRSIG-Record an. Sie erläutern jedoch nicht, wie sie diesen Wert ermittelt haben.

zusammensetzen kann, mitgeteilt [Vix99, S. 3 f.]. Die Verwendung des CLASS-Attributs erlaubt im Gegensatz zum RCODE-Feld eine wesentlich feingranularere Festlegung der Nachrichtenlänge.

Bei der Verwendung von EDNS0 können die Nutzdaten einer DNS-Antwort theoretisch eine maximale Größe von 65507 Byte erreichen.¹⁴ Eine in der Praxis wichtige Schwelle liegt jedoch bereits bei 1472 Byte: wegen der üblicherweise in IP-Netzen geltenden MTU (Maximum Transfer Unit) von 1500 Byte (vgl. [ADF06, S. 4] und [Bel10, S. 4]) müssen größere DNS-Anfragen auf jeden Fall in mehreren Fragmenten (IP-Paketen) übertragen werden. Durch die Fragmentierung sinkt die Zuverlässigkeit der Übertragung und es kann zu Verzögerungen bei der Namensauflösung kommen: geht bei der Übertragung der DNS-Antwort *ein einziges Fragment* verloren, muss der DNS-Client nach einem Timeout eine Retransmission der Anfrage anstoßen. Zudem wird im EDNS0-Standard die Problematik beschrieben, dass Netzknoten oder Firewalls auf dem Transportweg möglicherweise nicht mit fragmentierten Paketen umgehen können und ihre Weiterleitung verweigern [Vix99, S. 3 f.]. Dass fragmentierte DNS-Nachrichten in der Praxis zu erwarten sind, zeigt die Simulation von Ager et al.: Bei Verwendung von DNSSEC überschritten im Experiment etwa 5 % der DNS-Antworten die kritische Größe von 1472 Byte. Ob die Verwendung von EDNS0 zu einer unzuverlässigen und somit verzögerten Namensauflösung führt, hängt u. a. von der Paketverlustrate auf dem Übertragungsweg ab. Während im November 2011 bei vollständig drahtgebundenen Verbindungen in Nordamerika und Europa Verlustraten von weniger als 0,1 % beobachtet wurden [CMZ12], stellte eine Studie im Jahr 2010 bei Smartphones erheblich höhere Werte von ca. 3,5 % fest [Fal+10]. Gerade in drahtlosen Netzen könnte die Übermittlung von DNSSEC-Signaturen an den Stub-Resolver also zu spürbaren Verzögerungen führen.

2.6.1.4 Übertragung mittels TCP

Im vorigen Abschnitt wurde deutlich, dass die zuverlässige Übermittlung von großen DNS-Anfragen mittels EDNS0 nicht sichergestellt werden kann. Als Alternative verbleibt dann lediglich die Übertragung von DNS-Anfragen mittels TCP. Dementsprechend wird erwartet, dass der Anteil der über TCP übertragenen Anfragen in Zukunft steigen wird. RFC 5966 [Bel10] trägt dieser Entwicklung Rechnung. Darin wird die Unterstützung von TCP, die ursprünglich lediglich empfohlen wurde, nun zum verpflichtenden Bestandteil von standardkonformen Resolver- und Nameserver-Implementierungen [Bel10, S. 2]. Während die Verwendung von TCP bislang lediglich als Notlösung angesehen wurde, falls eine vorherige Übermittlung einer Anfrage per UDP fehlschlug, ist das Protokoll nun gleichberechtigt: Resolver dürfen bei Anfragen, bei denen eine große Antwort zu erwarten ist, von vornherein TCP verwenden.

¹⁴Diese Größe ergibt sich aus der maximalen Größe der Nutzdaten eines IPv4-Pakets (65535 Byte) abzüglich der Länge des IP-Paketheaders (20 Byte) und des UDP-Paketheaders (8 Byte). Diese Beschränkung gilt auch für IPv6-Pakete, solange die „Jumbo Payload“-Option nicht verwendet wird [BDH99].

RFC 5966 spricht zudem die Möglichkeit der Verwendung persistenter TCP-Verbindungen an, innerhalb derer mehrere DNS-Anfragen übermittelt werden können. Bei persistenten Verbindungen entfällt der sonst übliche 3-Way-Handshake, welcher jede einzelne Namensauflösung verzögert. Da Nameserver nur eine beschränkte Menge von Verbindungen gleichzeitig offen halten können, kann diese Funktionalität jedoch für Denial-of-Service-Angriffe ausgenutzt werden [Bel10, S. 5]. Es wird daher empfohlen, dass Nameserver offene Verbindungen bei Inaktivität nach wenigen Sekunden schließen. RFC 1035 hatte hierfür noch ein Intervall von etwa zwei Minuten vorgeschlagen [Moc87b, S. 33]. In der Praxis schließen viele Nameserver die TCP-Verbindung allerdings sofort wieder. Es bleibt daher abzuwarten, inwiefern sich persistente Verbindungen in der Praxis etablieren werden.

In der Forschung gibt es zudem erste Vorschläge, welche die verzögerungsarme Übermittlung von DNS-Anfragen auf Basis von TCP zu erreichen suchen, etwa mittels TCP for Transactions (T/TCP) bzw. „Stateless TCP“. **TCP for Transactions** [Rik+03] erlaubt es einem DNS-Client, die DNS-Anfrage bereits im ersten IP-Paket des 3-Way-Handshakes zu übermitteln und dadurch eine mit UDP vergleichbare Antwortzeit zu erreichen. T/TCP wurde in RFCs 1379 und 1644 spezifiziert und zeitweise in SunOS, FreeBSD und Linux implementiert. Da das Konzept jedoch Sicherheitsprobleme aufweist, wurde die Unterstützung des Protokolls wieder aus den Betriebssystemen entfernt. Bei **Stateless TCP** [Hay+11], das im Jahr 2009 von Houston für die Nutzung in Verbindung mit DNS vorgeschlagen wurde [Hou09], soll hingegen eine modifizierte TCP-Implementierung zum Einsatz kommen, welche Datenpakete auch dann entgegennimmt, wenn überhaupt kein 3-Way-Handshake durchlaufen wurde. Da der Server bei dieser Technik keine Zustandsinformationen vorhält, gibt es Einschränkungen hinsichtlich der maximal erlaubten Nachrichtenlänge: DNS-Anfragen müssen in ein TCP-Paket passen und DNS-Antworten dürfen nicht die Größe des Empfangspuffers des Clients überschreiten. Zudem gibt es bislang keine Sicherheitsanalyse für Stateless TCP. Zwar wird in den beiden Publikationen anhand von Prototypen und Experimenten die Eignung der Verfahren im Praxiseinsatz demonstriert, eine flächendeckende Einführung erscheint angesichts der erforderlichen Änderungen an Betriebssystemen bzw. DNS-Implementierungen sowie der genannten Unzulänglichkeiten jedoch unwahrscheinlich.

2.6.2 Rekursive und iterative Namensauflösung

Das DNS unterscheidet zwischen *rekursiver* und *iterativer* Namensauflösung [Ait11, S. 42 ff.]. Die gewünschte Art der Namensauflösung wird in einer DNS-Anfrage vom anfragenden Resolver durch Setzen des *Recursion-Desired-Flags* (**RD-Flag**) kodiert.

2.6.2.1 Rekursive Namensauflösung

Aus der Perspektive eines Clients ist die rekursive Namensauflösung zu bevorzugen, da sie mit geringem Aufwand für den Client verbunden sind. Der Client sendet dazu eine einzige

rekursive Anfrage, d. h., eine Anfrage, in der das *RD-Flag* gesetzt ist, an einen Nameserver. Der Client erwartet daraufhin eine Antwort, die entweder die gewünschten Informationen (den angefragten RR) oder einer Fehlermeldung (s. Hinweise zur Fehlerbehandlung in Abschnitt 2.6.3) enthält, d. h. der kontaktierte Nameserver darf den Client nicht an einen anderen Nameserver verweisen. Die in den gängigen Betriebssystemen verwendeten Stub-Resolver erzeugen stets rekursive Anfragen. Diese senden sie an den Nameserver, den der Nutzer in den Netzwerkeinstellungen des Betriebssystems eingetragen hat (bzw. an den Nameserver, der per DHCP übermittelt wird).

Mit einer rekursiven Anfrage beauftragt der Client den kontaktierten rekursiven Nameserver mit der Namensauflösung. Liegen dem kontaktierten Nameserver die gewünschten Informationen nicht in seiner Zonendatei oder im Cache (s. Abschnitt 2.7) vor, muss er den Namensraum so lange – in einem rekursiven Prozess – traversieren, bis ihm der gewünschte RR vorliegt oder ein Fehler auftritt. Hierzu muss der Nameserver u. U. mit mehreren autoritativen Nameservern kommunizieren.

Rekursive Nameserver sind dadurch gekennzeichnet, dass sie den Wunsch des Clients, für ihn eine rekursive Namensauflösung durchzuführen, akzeptieren. Autoritative Server, insbesondere die Root-Server und die TLD-Server, weigern sich hingegen, den Namensraum im Auftrag des Anfragenden zu traversieren. Sie beantworten eine eingehende rekursive Anfrage in der gleichen Weise wie eingehende iterative Anfragen.

Rekursive Nameserver setzen in ihren Nachrichten an autoritative Server dennoch häufig das RD-Flag. Falls ein kontaktierter autoritativer Nameserver ausnahmsweise rekursive Anfragen akzeptiert, dann soll *er* sich um die Namensauflösung kümmern [Ait11, S. 45].¹⁵

2.6.2.2 Iterative Namensauflösung

Das gerade beschriebene in der Praxis zu beobachtende opportunistische Verhalten der rekursiven Nameserver hat zwar keine negativen Konsequenzen, in RFC 1034 ist jedoch ein defensiver Ansatz vorgesehen [Moc87a, S. 23]: Ein anfragender Nameserver soll demnach bei Anfragen an einen anderen Nameserver grundsätzlich davon ausgehen, dass dieser keine rekursive Namensauflösung anbietet, der Anfragende also selbst die Hierarchie des DNS-Namensraums traversieren muss.

Dieser Ansatz wird als iterative Namensauflösung bezeichnet. Dabei werden **iterative Anfragen** gesendet, bei denen das *Recursion-Desired-Flag* nicht gesetzt ist. Der Anfragende erwartet in diesem Fall nicht den angeforderten RR, sondern lediglich die bestmögliche Antwort, die der kontaktierte Nameserver auf Basis seiner Zonendatei bzw. der Einträge in seinem Cache liefern kann. Ein autoritativer Nameserver beantwortet iterative (und wie in Abschnitt 2.6.2.1 angedeutet auch rekursive) Anfragen typischerweise

- mit dem angeforderten RR, falls dieser in seiner Zonendatei enthalten ist, d. h., wenn er für den angeforderten Domainnamen autoritativ ist,

¹⁵Nicht alle in der Praxis verwendeten Implementierungen weisen dieses Verhalten auf (vgl. [AM01], wonach rekursive Nameserver *stets iterative Anfragen stellen*, wenn sie autoritative Nameserver kontaktieren).

- mit einem Referral (s. S. 19), also einem Verweis auf einen anderen autoritativen Nameserver, falls der Nameserver für ein Suffix der angefragten Domain autoritativ ist und in seiner Zonendatei eine zur angefragten Domain passende Delegation enthalten ist, oder
- mit einem REFUSED-Fehler (s. Abschnitt 2.6.3), falls keiner der obigen Fälle zutrifft.

Erhält der Anfragende ein Referral als Antwort, muss er sich mit seiner Anfrage an den darin genannten Nameserver wenden. Bei der iterativen Namensauflösung muss der anfragende Nameserver daher üblicherweise mehrere Anfragen stellen, bis ihm der gewünschte RR vorliegt. Stub-Resolver sind im Gegensatz zu den auf rekursiven Nameservern eingesetzten Full-Resolvoren üblicherweise nicht dazu in der Lage, Referral-Antworten zu verarbeiten.

Zur Verdeutlichung wird im Folgenden das Zusammenwirken der rekursiven und der iterativen Namensauflösung an einem Beispiel skizziert, das sich an [Ait11, S. 43 ff.] orientiert.

Beispiel 2.4. In Abbildung 2.7 sind die Interaktionen dargestellt, die zur Namensauflösung erforderlich sind. An der Namensauflösung sind ein Arbeitsplatz-PC (Client mit Stub-Resolver), ein rekursiver Nameserver, der z. B. vom Internetzugangsanbieter betrieben wird, und mehrere autoritative Nameserver beteiligt. Der nachfolgend beschriebene Ablauf ist üblicherweise bei Breitband-Internetzugängen von Privatkunden zu beobachten (Szenario 2a in Abschnitt 2.4.2).

Im Beispiel wird unterstellt, dass dem Stub-Resolver und dem rekursiven Nameserver keine zur Auflösung benötigten Informationen im Cache vorliegen bzw. der lokalen Hosts-Datei entnommen werden können. Weiterhin sei angenommen, dass der rekursive Nameserver das Recursion-Desired-Flag setzt, wenn er Anfragen an die autoritativen Nameserver übermittelt (also sich wie in Abschnitt 2.6.2.1 beschrieben opportunistisch verhält).

Bei der Namensauflösung werden folgende Schritte durchlaufen:

1. Ein Anwendungsprogramm auf dem Arbeitsplatz-PC, etwa ein Web-Browser, ruft die Funktion zur Namensauflösung für den Domainnamen *www.uni-hamburg.de* auf, die von der lokalen Stub-Resolver-Bibliothek angeboten wird.
2. Der Stub-Resolver sendet eine rekursive Anfrage für den A-Record der Domain *www.uni-hamburg.de* an den im Betriebssystem eingetragenen rekursiven Nameserver.
3. Der rekursive Nameserver sendet eine rekursive Anfrage für den A-Record von *www.uni-hamburg.de* an einen Root-Server.
4. Der Root-Server unterstützt lediglich iterative Anfragen und behandelt die rekursive Anfrage daher wie eine iterative Anfrage: Er antwortet dem rekursiven Nameserver mit der Liste der TLD-Nameserver (Domainnamen sowie IP-Adressen), welche für die Domain „*de*“ autoritativ sind (Referral).

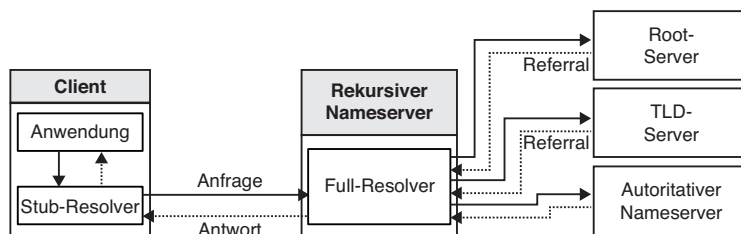


Abbildung 2.7: Prozess der Namensauflösung (nach [Ait11, S. 45])

5. Der rekursive Nameserver sendet eine rekursive Anfrage für den A-Record von *www.uni-hamburg.de* an einen der ihm nun bekannten TLD-Nameserver.
6. Da auch der TLD-Server keine rekursiven Anfragen unterstützt, antwortet er mit einem weiteren Referral: Er übermittelt dem rekursiven Nameserver die Liste der Nameserver, welche für die Domain *uni-hamburg.de* autoritativ sind.
7. Der rekursive Nameserver sendet eine rekursive Anfrage für den A-Record von *www.uni-hamburg.de* an einen der autoritativen Nameserver, die er in Schritt 6 erfahren hat. Im Beispiel wählt er den Nameserver *ns.rrz.uni-hamburg.de*.
8. Der Nameserver *ns.rrz.uni-hamburg.de* ist für die Domain *uni-hamburg.de* autoritativ. Er akzeptiert zwar keine rekursiven Anfragen, er verfügt jedoch in seiner Zonendatei über die angeforderten Informationen: Für *www.uni-hamburg.de* ist ein CNAME-Record eingetragen, der auf den Namen *rzlinw1.rrz.uni-hamburg.de* verweist. Für diesen Domainnamen ist *ns.rrz.uni-hamburg.de* ebenfalls autoritativ. Er generiert daher eine Antwort, in welcher sowohl der CNAME-Record von *www.uni-hamburg.de* als auch der A-Record von *rzlinw1.rrz.uni-hamburg.de* enthalten sind. Die Antwort sendet er an den rekursiven Nameserver.
9. Der rekursive Nameserver generiert eine Antwort mit den vom autoritativen Nameserver erhaltenen CNAME- und A-Records. Diese Antwort sendet er an den Stub-Resolver.
10. Die im Stub-Resolver implementierte Funktion zur Namensauflösung liefert an den Web-Browser die IP-Adresse zurück, die im A-Record von *rzlinw1.rrz.uni-hamburg.de* enthalten ist. □

Beobachtungsmöglichkeiten Wie am Ablauf der Namensauflösung erkennbar wird, erfährt der **rekursive Nameserver** sowohl die IP-Adresse des Nutzers als auch alle Domainnamen, die ein Nutzer anfragt. Daraus ergeben sich weitreichende Beobachtungsmöglichkeiten, wie in Kapitel 4 aufgezeigt wird.

Der **autoritative Nameserver**, der die RRs für den angefragten Domainnamen vorhält, sieht hingegen – sofern der Nutzer einen rekursiven Nameserver verwendet, wovon im

Folgenden stets ausgegangen werden soll – lediglich die IP-Adresse des rekursiven Nameservers. Auf die Beobachtungsmöglichkeiten der autoritativen Nameserver wird im folgenden Abschnitt noch näher eingegangen.

2.6.2.3 Beobachtungsmöglichkeit auf autoritativen Nameservern

Nicht nur die rekursiven Nameserver erfahren die angefragten Domainnamen; auch die autoritativen Nameserver können die bei ihnen eingehenden DNS-Anfragen beobachten. Beachtenswert ist in diesem Zusammenhang, dass der rekursive Nameserver bei *allen* Anfragen, die er an autoritative Server sendet, stets den FQDN (s. Abschnitt 2.3) übermittelt, also den vollständigen Domainnamen. Auch die Root-Server und die TLD-Nameserver erfahren also den FQDN, den der Nutzer auflösen will. Autoritative Server an der Spitze der Namensraum-Hierarchie haben daher die Möglichkeit, das Nutzungsverhalten aller Internetnutzer zu beobachten.

Eigentlich benötigen die Nameserver an der Spitze der Hierarchie zur Erfüllung ihrer Aufgabe den FQDN gar nicht: Sie antworten auf Anfragen (z. B. für *www.google.com*) ohnehin stets mit einem Referral auf den autoritativen Server in der nächstniedrigeren Ebene – und um ein solches Referral auszustellen, benötigt ein autoritativer Server lediglich das Suffix des FQDNs, welches die Domain in der nächstniedrigeren Ebene ausweist. Bei der Anfrage an einen Root-Server würde also die Angabe der Domain „*com*“ ausreichen. Das Präfix des Domainnamens, also im Beispiel die Subdomain „*www.google*“, ist zur Ermittlung des NS-RRs des autoritativen Nameservers der „*com*“-Domain für den Root-Server irrelevant. Die in der Praxis anzutreffenden Resolver-Implementierungen übermitteln dennoch stets den vollständigen FQDN. Ein neuer Internet-Draft schlägt vor, von dieser Tradition abzurücken ([Bor13], s. auch S. 227).

Von den Beobachtungsmöglichkeiten auf den autoritativen Nameservern geht im Vergleich zu den Möglichkeiten auf den rekursiven Nameservern eine geringere Bedrohung für die Privatsphäre einzelner Nutzer aus, da die Beobachtung auf den autoritativen Servern zwei Einschränkungen unterliegt:

1. Der autoritative Nameserver sieht nicht die IP-Adresse des Nutzers, der die Anfrage ursprünglich gestellt hat, sondern lediglich die IP-Adresse des rekursiven Nameservers, der sie für den Nutzer anonym an den autoritativen Nameserver übermittelt. Üblicherweise wird ein rekursiver Nameserver von einer Vielzahl von Nutzern verwendet; anhand der IP-Adresse des rekursiven Nameservers kann der autoritative Nameserver den Nutzer, der die Anfrage gestellt hat, daher nicht identifizieren. Unter Umständen kann er anhand des verwendeten rekursiven Nameservers allerdings darauf schließen, dass der Nutzer zu einer bestimmten Benutzergruppe gehört, etwa dass er Kunde bei einem bestimmten Internetzugangsanbieter ist oder in einer bestimmten Organisation arbeitet.
2. Nicht jede Anfrage, die ein Nutzer stellt, wird an einen Root-Server oder TLD-Server übermittelt. Das in Abschnitt 2.7 erläuterte Caching-Konzept wirkt wie ein

Mechanismus zum Schutz vor Beobachtung: Erfährt ein rekursiver Nameserver im Zuge der Namensauflösung von einem Root-Server die IP-Adresse eines TLD-Servers (z. B. des „com“-Servers) bzw. eines SLD-Servers (z. B. des Nameservers, der für *google.com* autoritativ ist), wird diese üblicherweise einen oder mehrere Tage lang zwischengespeichert (TTL des NS-RRs). Geht in diesem Zeitraum beim rekursiven Nameserver eine Anfrage für einen Domainnamen ein, von dessen SLD-Server der rekursive Nameserver die IP-Adresse bereits kennt, nutzt er diese Information, um die Anfrage direkt an den SLD-Server zu richten. Die Root-Server bzw. die TLD-Server (hier: der „com“-Server) können also nur einen Bruchteil der von den Nutzern gestellten Anfragen beobachten.

Eine Beobachtung von Nutzern und deren Verhalten ist auf autoritativen Nameservern daher nur eingeschränkt und unzuverlässig möglich. Im weiteren Verlauf dieser Arbeit stehen daher die Beobachtungsmöglichkeiten auf rekursiven Nameservern im Fokus.

2.6.3 Fehlerbehandlung

Bei den in Abschnitt 2.6.2 skizzierten Abläufen wird davon ausgegangen, dass während der Namensauflösung keine Fehler auftreten. Ob eine Anfrage erfolgreich beantwortet werden konnte, lässt sich anhand des RCODE-Felds (von engl. „response code“) in der DNS-Antwort erkennen [Moc87b, S. 27 f.]. Der Erfolgsfall wird durch den Wert **NOERROR** angezeigt. Eine vollständige Aufstellung aller definierten RCODEs findet sich in [Ait11, S. 595]. Im Folgenden werden lediglich die RCODEs genannt, die im weiteren Verlauf von Bedeutung sind.

Eine der häufigsten Fehlerursachen besteht Untersuchungen von Jung et al. [Jun+02] zufolge darin, dass für den angefragten Domainnamen überhaupt keine Resource-Records existieren. Man spricht in diesem Fall davon, dass der Domainname nicht existiert [Ait11, S. 595]. Der autoritative Nameserver, der für die *übergeordnete* Domain zuständig ist, reagiert auf die Anfrage mit dem RCODE **NXDOMAIN** (von engl. „non-existent domain“). Der rekursive Nameserver beantwortet die Anfrage des Stub-Resolver dann ebenfalls mit einer NXDOMAIN-Antwort. Eine Anfrage für den nicht-existierenden Domainnamen „*nicht-existierende-seite.com*“ wird also von einem der Nameserver, die für die „com“-Domain autoritativ sind, mit NXDOMAIN beantwortet, eine Anfrage für den wegen eines Tippfehlers nicht-existierenden Namen „*eee.uni-hamburg.de*“ wird von einem Nameserver, der für *uni-hamburg.de* autoritativ ist, beantwortet, und eine Anfrage für „*www.uni-hamburg.de*“ wird von einem der Root-Server mit NXDOMAIN beantwortet.

Anders verhält es sich, wenn der angefragte Domainname existiert, jedoch für den angefragten RR-Typ keine Daten hinterlegt sind. Im Unterschied zum vorherigen Fall gibt es nun auf jeden Fall einen Nameserver, der für die angefragte Domain autoritativ ist. Dieser beantwortet die Anfrage mit einer NOERROR-Antwort, in der keinerlei RRs enthalten sind.

Weitere in der Praxis häufig zu beobachtende RCODEs sind **SERVFAIL**, mit dem ein Nameserver bekannt gibt, dass er die Anfrage (etwa aufgrund einer temporären Störung oder eines Konfigurationsfehlers) nicht beantworten kann, sowie **REFUSED**, mit dem ein Nameserver ausdrückt, dass die Beantwortung auf Grund von Zugriffskontrollmechanismen verweigert wird. Sendet ein Client eine rekursive Anfrage an einen autoritativen Nameserver, reagiert dieser typischerweise mit einer REFUSED-Antwort.

2.6.4 Zonentransfer

Wie in Abschnitt 2.4.1 erläutert werden die Resource-Records einer Zone auf mehreren autoritativen Nameservern vorgehalten. Um Inkonsistenzen durch unterschiedliche Daten zu vermeiden, besteht die gängige Vorgehensweise darin, alle Änderungen auf dem Master-Server vorzunehmen und diese von dort aus auf die Slaves zu kopieren. Die Verteilung der Zonendaten muss nicht manuell durchgeführt werden, sondern kann von den Nameservern automatisch in Form eines sog. Zonentransfers vorgenommen werden. Hierfür existieren zwei Techniken:

Polling Die Slave-Nameserver fragen in regelmäßigen Abständen beim Master-Server nach, ob sich die Zonendaten geändert haben und führen ggf. einen Zonentransfer durch

Notification Der Master-Nameserver benachrichtigt bei einer Änderung alle Slave-Server, die dann ihrerseits einen Zonentransfer anstoßen.

Ursprünglich waren lediglich Polling-Techniken vorgesehen. In RFC 1034 ist das ursprüngliche **AXFR-Protokoll** (von engl. „authoritative transfer“) definiert [Moc87a, S. 28 f.]; RFC 5936 enthält eine detailliertere und überarbeitete Spezifikation [LH10]. Das Protokoll sieht vor, dass jeder Slave in regelmäßigen Abständen den SOA-Record des Masters abruft und überprüft, ob die dort hinterlegte Zonen-Seriennummer mit der Seriennummer übereinstimmt, die in der Zonendatei des Slaves steht. Falls die Seriennummer beim Master inkrementiert wurde, fordert der Slave mittels einer DNS-Anfrage vom Typ AXFR einen Zonentransfer an. Als Antwort erhält der Slave sämtliche Resource-Records vom Master. Im Gegensatz zu normalen DNS-Anfragen wird ein Zonentransfer mittels TCP übertragen, um die Größenbeschränkung von UDP (s. S. 30) zu überwinden.

Gerade bei großen Zonen wie sie etwa bei den autoritativen Nameservern von SLDs auftreten ist das AXFR-Protokoll ineffizient, da es bei jeder Änderung immer eine vollständige Replikation der Zonendaten vornimmt. Eine Weiterentwicklung stellt das **IXFR-Protokoll** dar (von engl. „incremental zone transfer“), das in RFC 1995 spezifiziert ist [Oht96]. Auch hier bestimmt der Slave durch den Polling-Mechanismus, wann ein Zonentransfer durchzuführen ist. Im Unterschied zum AXFR-Protokoll übermittelt der Slave bei einer IXFR-Anfrage den SOA-Record mit der aus seiner Sicht aktuellen Seriennummer. Anhand der Seriennummer bestimmt der Master die geänderten RRs und übermittelt nur diese an den Slave.

Bei der Polling-Technik kommt es bei der Verbreitung von Änderungen zu unerwünschten Verzögerungen. Abhilfe schafft der Einsatz der Notification-Technik, die in RFC 1996 beschrieben wird [Vix96]. Dabei sendet der Master bei Änderungen an seinen Zonendaten an alle Slave-Nameserver, die für die Zone autoritativ sind, eine DNS-Anfrage vom Typ **NOTIFY**. Empfängt ein Slave-Server eine NOTIFY-Anfrage, stößt er umgehend einen Zonentransfer mit dem AXFR- oder IXFR-Protokoll an.

Zonentransfers bedrohen die Vertraulichkeitsinteressen der autoritativen Nameserver. Ursachen und Sicherheitsmechanismen werden in Abschnitt 3.7.2 erläutert.

2.7 Caching

Die Zuordnung zwischen Domainnamen und IP-Adressen ändert sich in der Praxis nur vergleichsweise selten. DNS-Anfragen für denselben Namen, die innerhalb einer kurzen Zeitspanne ausgeführt werden, führen dadurch sehr häufig zu identischen Antworten. Es liegt nahe, in solchen Fällen auf eine wiederholte DNS-Anfrage zu verzichten und stattdessen die kürzlich erhaltene Antwort erneut zu verwenden. Eine konsequente und strukturierte Umsetzung dieser Idee stellt das im DNS enthaltene *Caching-Konzept* dar, welches spezifiziert, wie Informationen auf den an der Namensauflösung beteiligten Komponenten in einem Zwischenspeicher, dem *Cache*, zwischengespeichert werden können. Die Motivation für das Zwischenspeichern besteht insbesondere darin, Wartezeiten zu verringern, die sich durch die zahlreichen Interaktionsschritte bei der Auflösung eines Domainnamens ergeben [Moc87a, S. 29].

Die Auseinandersetzung mit dem Caching-Konzept des DNS ist aus zwei Gründen erforderlich. Zum einen ist es zur Einschätzung der Beobachtungsmöglichkeiten von Bedeutung, wie sich bereits bei der Betrachtung der Beobachtungsmöglichkeiten von autoritativen Nameservern in Abschnitt 2.6.2.3 gezeigt hat. Darüber hinaus ergeben sich durch das Caching-Konzept Gestaltungsmöglichkeiten zur Konstruktion von Mechanismen zum Schutz vor Beobachtung, wie das in Abschnitt 7.3 vorgestellte Verfahren demonstriert.

Dieser Abschnitt ist wie folgt aufgebaut: In Abschnitt 2.7.1 wird zunächst der Zielkonflikt dargestellt, der sich aus der Zwischenspeicherung von Daten ergibt. Im Anschluss daran wird in Abschnitt 2.7.2 das Konzept der Time-to-Live erläutert, mit dem Nameserver-Betreiber individuell einen Kompromiss zwischen Konsistenz und Verfügbarkeit finden können. In Abschnitt 2.7.3 wird schließlich erläutert, welche Möglichkeiten der Zwischenspeicherung in den einzelnen Komponenten des DNS vorgesehen sind und wie diese in der Praxis umgesetzt werden. Dazu wurden Quellcode-Analysen und praktische Versuche mit gängigen Betriebssystemen und Anwendungen durchgeführt.

2.7.1 Zielkonflikt zwischen Konsistenz und Verfügbarkeit

Die Zwischenspeicherung von Informationen hat sowohl Vor- als auch Nachteile. Diese werden im Folgenden kurz erläutert.

Der Nachteil der Zwischenspeicherung betrifft bei der Namensauflösung die **Konsistenz** der Informationen. Im Zusammenhang mit verteilten Systemen wird damit üblicherweise die Anforderung verbunden, dass sich redundant vorgehaltene Informationen nicht widersprechen [Tra+82]. Ein damit verwandtes Schutzziel, auf das in Abschnitt 3.2 noch eingegangen wird, ist die „Integrität“, die sich auf die *Korrektheit* der Informationen bzw. den Schutz vor beabsichtigten und unbeabsichtigten Modifikationen bezieht.

Solange keine Informationen zwischengespeichert werden, sind die von der Namensauflösung gelieferten DNS-Antworten stets *konsistent*, d. h. die Anwendung, welche die Namensauflösung angefordert hat, erhält auf jeden Fall die Informationen, die zu diesem Zeitpunkt in der jeweiligen Zone hinterlegt sind. Durch die Einführung von Caching kann diese Konsistenz nicht mehr gewährleistet werden. Wenn der Zonenadministrator RRs in einer Zone verändert, etwa weil sich die IP-Adresse eines Webserver nach einem Hardware-Ausfall geändert hat, spiegeln die zwischengespeicherten Kopien nicht mehr die korrekten Daten wider. Der autoritative Server kann weder das Ausmaß dieser Inkonsistenz bestimmen noch die betroffenen Komponenten über die Aktualisierung informieren, da er nicht weiß, welche Komponenten inkonsistente Daten im Cache vorhalten.

Inkonsistenzen können beim Einsatz von Caching im DNS zwar nicht vermieden werden, sie werden jedoch im Laufe der Zeit automatisch beseitigt, was auch als „**eventual consistency**“ bezeichnet wird [Vog08; ÖV11; BG13, S. 461 f.]. Das Caching-Konzept sieht vor, dass zwischengespeicherte Daten nach einer festgelegten Zeitspanne aus dem Cache gelöscht werden müssen. Je größer diese maximal tolerierte Zeitspanne ist, desto länger dauert es, bis Clients nach einer Aktualisierung eines RRs die neuen Daten abrufen. Die maximal erlaubte Zeitspanne (*time-to-live*, abgekürzt *TTL*) wird im DNS nicht universell festgelegt. Sie kann vom Zonenadministrator für jeden RR den eigenen Anforderungen entsprechend individuell gewählt werden [Moc87a, S. 3].

Aus der Zwischenspeicherung ergeben sich jedoch auch Vorteile. Zum einen sinken dadurch die Antwortzeiten bei der Namensauflösung. Zum anderen wird die **Verfügbarkeit** der Informationen verbessert. Ein kurzzeitiger Ausfall aller autoritativen Nameserver einer Zone führt aus Sicht der Nutzer nicht unmittelbar zur Unverfügbarkeit aller darin gespeicherten Informationen; solange die zuletzt angefragten Informationen noch in den Zwischenspeichern der rekursiven Nameserver vorliegen, können entsprechende Anfragen beantwortet werden. Die Überbrückungszeit ist umso größer, je länger Informationen zwischengespeichert werden.

Anwendung des CAP-Theorems Konsistenz und Verfügbarkeit stehen im DNS also miteinander in Konflikt: Eine Erhöhung der Verfügbarkeit durch längere Zwischenspeicherung geht dabei stets zu Lasten der Konsistenz. Dieser Zielkonflikt tritt bei verteilten Systemen grundsätzlich auf, wie das von Brewer vermutete [Bre00] und später axiomatisch bewiesene [GL02] **CAP-Theorem** verdeutlicht: Demnach kann ein verteiltes System lediglich zwei der folgenden drei Anforderungen gleichzeitig erfüllen:

- **Konsistenz** (engl. „consistency“): Alle Nutzer sehen zu einem bestimmten Zeitpunkt dieselben Informationen.
- **Verfügbarkeit** (engl. „availability“): Jede Anfrage an das System wird beantwortet.
- **Partitionstoleranz** (engl. „partition tolerance“): Ein Knoten kann Anfragen beantworten, wenn Kommunikationsverbindungen unterbrochen sind, die verhindern, dass er mit allen Komponenten des Systems Kontakt aufnimmt (sog. Partition des Netzes).

Wie Brewer in [Bre00] ausführt, sind im DNS die Anforderungen Verfügbarkeit und Partitionstoleranz erfüllt, die Konsistenz hingegen nicht: Kann ein DNS-Client einen rekursiven Nameserver erreichen und ihm eine Anfrage für einen bestimmten Domainnamen übermitteln, die der Nameserver mit den Informationen in seinem Cache beantworten kann, dann kann der rekursive Nameserver die Anfrage des Clients auf jeden Fall beantworten, selbst wenn das Netz durch eine Störung partitioniert ist, sodass der rekursive Nameserver keinen der autoritativen Nameserver erreichen kann. Die zurückgelieferten Informationen sind allerdings unter Umständen inkonsistent.

2.7.2 Der Time-to-live-Wert (TTL)

Der angesprochene Zielkonflikt kann zwar nicht umgangen werden, die daraus resultierenden Auswirkungen lassen sich allerdings beeinflussen. Das Konzept der „**Time to Live**“ (TTL) erlaubt es jedem Zonenverwalter, seine Präferenzen bezüglich der Verfügbarkeit und der Konsistenz zu formulieren. Der Zonenverwalter kann den Wert der TTL für jeden RR separat festlegen. Der Wert der TTL gibt die maximale Lebensdauer eines RRs in einem Cache eines Resolvers in Sekunden an. Die TTL wird nach RFC 1035 [Moc87b, S. 10] als vorzeichenbehafteter Integer mit einer Breite von 32 Bit kodiert, wobei gemäß RFC 2181 lediglich Werte im Bereich zwischen 0 und $2^{31} - 1 = 2\,147\,483\,647$ zulässig sind [EB97, S. 10 f.]. Ein Wert von 0 drückt dabei aus, dass die Antwort überhaupt nicht zwischengespeichert werden soll [Moc87a, S. 13].

Zur Vereinfachung der Konfiguration kann im SOA-Record einer Zone ein Standard-Wert für die TTL definiert werden, der innerhalb der Zone für alle RRs gilt, für die der Administrator keinen TTL-Wert eingetragen hat. Viele Implementierungen, die zum Betrieb von autoritativen Nameservern verwendet werden, nutzen hierfür abweichend vom ursprünglich vorgesehenen Zweck das in RFC 1035 als *Minimum-TTL* bezeichnete Feld [Moc87b, S. 19 f.]. RFC 2308 kritisiert diese Umwidmung und führt eine neue dedizierte *\$TTL-Direktive* ein, mit welcher der Standard-Wert innerhalb einer Zonendatei festgelegt werden soll [And98, S. 8 f.]. Der Minimum-TTL-Wert wird in RFC 2308 umgewidmet und soll nun die TTL für das in diesem Standard definierte **negative Caching** angeben [And98, S. 9]. Negatives Caching betrifft die Antworten für Domainnamen, die nicht existieren und mit dem Fehlercode *NXDOMAIN* (s. Abschnitt 2.6.3) beantwortet werden.

RFC 1034 nennt als Richtwert für die TTL eine Größenordnung von einigen Tagen [Moc87a, S. 13]. Diese Empfehlung wird in RFC 1912 für RRs, welche sich selten än-

dern, auf einen Wert von ein bis zwei Wochen angehoben [Bar96, S. 5]. Vergleichbare Werte finden sich auch in der Sekundärliteratur [Ait11, S. 28]. In der Praxis werden inzwischen jedoch vor allem für A-Records **wesentlich niedrigere TT-Werte verwendet**, die im Bereich von wenigen Stunden oder sogar wenigen Sekunden liegen: Gao et al. haben eine groß angelegte Studie durchgeführt und berichten, dass 90 % der von ihnen untersuchten A-Records eine TTL von weniger als 60 Minuten aufwiesen [Gao+13]. Die Belastung der oberen Hierarchie-Ebenen des DNS ist dennoch vergleichsweise gering, da für die NS-Records von TLD- und SLD-Nameservern weiterhin sehr große TTL-Werte im Bereich von mehreren Tagen verwendet werden.

Mit dem TTL-Wert kann der Betreiber einer Zone zwar ausdrücken, wie lange die übermittelten Informationen gültig sind, das DNS kann jedoch nicht verhindern, dass nach dem Ablauf der TTL die in den RRs enthaltenen Informationen noch genutzt werden. Eine solche **Missachtung der TTL-Werte durch Clients** kann bei einem Adresswechsel zu längeren Unverfügbarkeitsperioden führen. Darunter leidet insbesondere die Benutzbarkeit neuartiger Dienste. So weisen bereits die Entwickler des im Jahr 2002 von IBM vorgestellten Dienstes „YouServ“, welcher Nutzern das dezentrale Anbieten von Inhalten ermöglichen soll, darauf hin, dass der Dienst durch veraltete Cache-Einträge in den Browsern u. U. nicht zuverlässig funktioniert [Bay+02]. Spätere Untersuchungen machen das Ausmaß dieses Problems deutlich: In einer Studie betrachten Pang et al. die Verlagerung des Datenverkehrs ab dem Zeitpunkt eines IP-Adress-Wechsels in den RRs von Webservern bzw. autoritativen Nameservern [Pan+04]. Wie ihre Ergebnisse zeigen, wandert zwar ein Großteil des Datenverkehrs nach der Änderung der RRs innerhalb der in der TTL angegebenen Zeitspanne zu den neuen Servern; die alten Server werden jedoch auch mehrere Stunden nach der Umstellung noch von einer Vielzahl von Clients verwendet. Diese in der Praxis verbreitete Missachtung der TTL-Werte lässt sich u. a. auf die mehrstufigen Caching-Hierarchie des DNS zurückführen, die in den folgenden Abschnitten näher betrachtet wird.

2.7.3 Mehrstufige Caching-Hierarchie

Das DNS sieht Caching auf mehreren Ebenen vor. Abbildung 2.8 verdeutlicht die verschiedenen Ebenen anhand der in der Praxis anzutreffenden Gestaltungsoptionen. Die autoritativen Nameserver fehlen in dieser Abbildung, da dort keinerlei Caching-Mechanismen vorgesehen sind. Die auf den autoritativen Nameservern eingehenden DNS-Anfragen können schließlich ohnehin unmittelbar, also ohne Kommunikation mit weiteren Nameservern beantwortet werden.

Die folgenden drei Unterabschnitte gehen im Detail auf diejenigen Komponenten ein, die einen Cache verwenden bzw. verwenden können. Zunächst werden in Abschnitt 2.7.3.1 die rekursiven Nameserver beschrieben, die in der Praxis immer mit einem Cache ausgestattet sind. Bei den auf den Clients angesiedelten Stub-Resolvern ist der Cache hingegen ein optionaler Bestandteil (s. Abschnitt 2.7.3.2). Einige Anwendungen verfügen zudem über

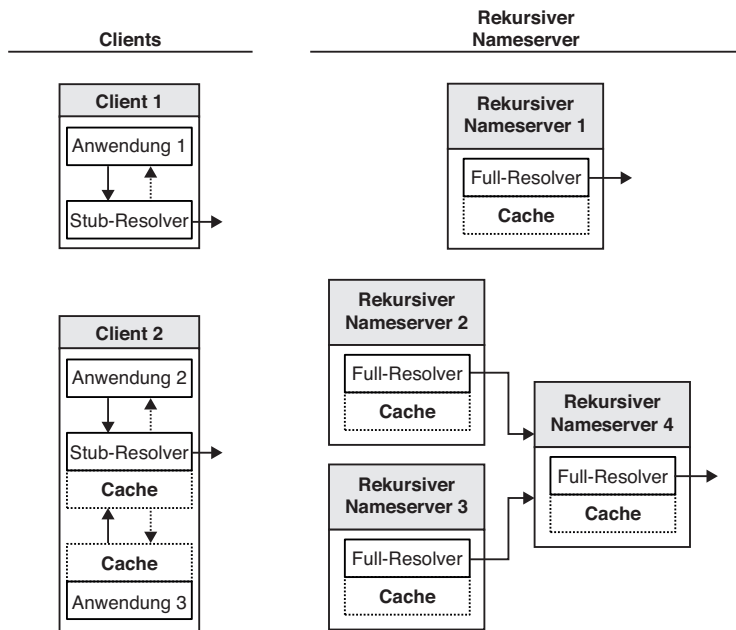


Abbildung 2.8: Illustration der mehrstufigen Caching-Hierarchie

einen vom Stub-Resolver unabhängigen dedizierten Cache (s. Abschnitt 2.7.3.3). Eine Sonderrolle nehmen Anwendungen ein, die den Stub-Resolver des Betriebssystems vollständig umgehen.

2.7.3.1 Caching in rekursiven Nameservern

Ein rekursiver Nameserver speichert nicht nur die endgültige Antwort auf eine Anfrage in seinem Cache, sondern auch alle während der Namensauflösung von autoritativen Nameservern erhaltenen Zwischenergebnisse. Bei den Zwischenergebnissen handelt es sich um die NS- und A-Records der TLD- und SLD-Nameserver, die zur Auflösung eines Namens kontaktiert werden müssen (vgl. Abschnitt 2.6.2) sowie ggf. CNAME-Records. Diese Informationen stehen dann für später eintreffende DNS-Anfragen zur Verfügung, deren Beantwortung dadurch weniger Kommunikationsaufwand erzeugt. Im Idealfall findet der rekursive Nameserver die IP-Adresse des zuständigen SLD-Nameservers bzw. zumindest des zuständigen TLD-Nameservers in seinem Cache vor.

Wird ein rekursiver Nameserver von mehreren Clients zur Namensauflösung verwendet, profitieren diese untereinander von dessen Cache, wie die Untersuchungen zur Cache-

Hit-Rate von Jung et al. [Jun+02] zeigen. Den Großteil der DNS-Anfragen, der auf eine vergleichsweise kleine Anzahl von populären Domainnamen entfällt, kann ein rekursiver Nameserver daher unmittelbar aus seinem Cache beantworten [Jun+02]. Um vom Cache eines anderen hochfrequentierten rekursiven Nameservers zu profitieren, kann ein rekursiver Nameserver so konfiguriert werden, dass er alle Anfragen an diesen weiterleitet (vgl. die rekursiven Nameserver 2 und 3 in Abbildung 2.8, die den rekursiven Nameserver 4 benutzen) und somit zu einem DNS-Forwarder mit Cache wird (s. auch Abschnitt 2.4.2.3).

Die in der Praxis verbreiteten Software-Implementierungen, die auf rekursiven Nameservern eingesetzt werden, berücksichtigen grundsätzlich die von den autoritativen Servern übermittelten TTL-Werte (im Gegensatz zu den Komponenten, die in den folgenden Abschnitten vorgestellt werden). Die Lebenszeit der Einträge im Cache kann jedoch bei einigen Implementierungen durch Konfigurationsparameter über den durch die TTL vorgegeben Zeitpunkt hinaus verlängert bzw. verkürzt werden (Parameter *cache-min-ttl* bzw. *cache-max-ttl* beim Nameserver Unbound¹⁶).

Die Implementierungen stellen lediglich sicher, dass Einträge nach Ablauf der TTL nicht mehr zur Namensauflösung herangezogen werden. Es gibt jedoch keine Zusicherung, dass die Einträge auf jeden Fall bis zum Ablauf der TTL im Cache vorgehalten werden: da die Größe des Zwischenspeichers üblicherweise beschränkt ist, kann es passieren, dass Einträge vorzeitig aus dem Cache entfernt werden.

Das DNS sieht vor, dass ein Eintrag von anfragenden Clients, die ihrerseits wiederum einen Cache verwenden, nicht länger verwendet werden soll, als es der Betreiber des autoritativen Nameservers in seiner Antwort vorgesehen hat. Der spätestmögliche Ablaufzeitpunkt ergibt sich aus der gegenüber dem anfragenden rekursiven Nameserver geäußerten Gültigkeitsdauer. Der rekursive Nameserver liefert in seinen DNS-Antworten einen entsprechend dekrementierten Wert der TTL zurück; in Abbildung 2.8 gilt also, dass die TTL eines Eintrags auf den rekursiven Nameservern 2 und 3 nie größer sein kann als auf Nameserver 4.

2.7.3.2 Caching in Stub-Resolvern

Unter dem Stub-Resolver werden wie in Abschnitt 2.4.2.2 ausgeführt die vom Betriebssystem bereitgestellten Funktionen zur Namensauflösung verstanden. Aus den Antworten des rekursiven Nameservers kann der Stub-Resolver anhand des übermittelten TTL-Werts die noch verbleibende Gültigkeitsdauer einer Antwort ersehen. Eventuell übermittelte Zwischenergebnisse (etwa die NS-Records der autoritativen Nameserver) sind für den Stub-Resolver unerheblich, da er ohnehin ausschließlich mit dem im Betriebssystem hinterlegten rekursiven Nameserver kommuniziert.

Grundsätzlich ist der Stub-Resolver also dazu in der Lage, DNS-Einträge in einem Zwischenspeicher aufzubewahren, um wiederkehrende Anfragen für einen Namen ohne

¹⁶vgl. Dokumentation von Unbound unter <http://www.unbound.net/documentation/unbound.conf.html>

Tabelle 2.1: Caching-Verhalten der Stub-Resolver gängiger Betriebssysteme

Betriebssystem	Verhalten	Quelle(n)
bis Windows 2000	verwendet keinen Cache	[Mic07]
Windows XP/Vista	berücksichtigt TTL des rekursiven Nameservers; $TTL_{max} = 86400$	[Mic07; Dav06]
Windows 7	berücksichtigt TTL des rekursiven Nameservers; $TTL_{max} = 86400$	eigene Versuche
Mac OS X 10.7.3	orientiert sich an der TTL des rekursiven Nameservers; $TTL_{min} = 15$	eigene Versuche [Jac+07]
Linux	distributionsabhängig; glibc (Standard) verwendet <i>keinen</i> Cache; nsd verwendet $TTL_{default} = 900$	eigene Versuche [Wei08; BA11]

weiteren Kommunikationsaufwand bearbeiten zu können. Die gängigen Betriebssysteme unterscheiden sich jedoch erheblich hinsichtlich des Umsetzungsgrades solcher Caching-Mechanismen im Stub-Resolver. Tabelle 2.1 fasst das in der Literatur dokumentierte Verhalten einiger Stub-Resolver zusammen. Eine Auswahl von Systemen wurde in einer Versuchsreihe, die am Ende dieses Abschnittes erläutert wird, genauer betrachtet (vgl. Vermerk „eigene Versuche“ in der Tabelle). Im Folgenden werden die wesentlichen Unterschiede der Stub-Resolver der gängigen Betriebssysteme beschrieben.

Während Mac OS X und alle Windows-Betriebssysteme seit Windows XP über einen Cache im Stub-Resolver verfügen, ist in der Standard-Installation vieler Linux-Distributionen kein Zwischenspeicher vorgesehen. Zudem blockierte in frühen Implementierungen der Aufruf der *gethostbyname*- bzw. *getaddrinfo*-Funktionen die Programmausführung während der Namensauflösung, was eine parallele Verarbeitung mehrerer Anfragen erschwerte und durch die Serialisierung der Anfragen zu erheblichen Wartezeiten führte [HN00].¹⁷ Aufgrund dieser Unzulänglichkeiten wurden Lösungen entwickelt, mit denen Linux-Clients mit einem DNS-Cache nachgerüstet werden können, z.B. der Systemdienst *nsd*¹⁸, der auf einigen Linux-Distributionen bei der Installation automatisch eingerichtet wird. Neben dedizierten Caching-Diensten besteht unter Linux auch die Möglichkeit, einen Full-Resolver wie zum Beispiel BIND auf einem Client zu installieren und als DNS-Forwarder zu betreiben. Da der DNS-Cache auf Windows-Systemen über eine beschränkte Kapazität verfügt (die genauen Eigenschaften des Zwischenspeichers sind nicht dokumentiert), wurden auch für dieses Betriebssystem Programme zur Implementierung eines vom

¹⁷ Aktuelle Implementierungen bieten üblicherweise auf *getaddrinfo* basierende Funktionen an, die eine asynchrone, also nichtblockierende Namensauflösung ermöglichen, z. B. die Funktion *getaddrinfo_a* in der glibc (s. http://linux.die.net/man/3/getaddrinfo_a).

¹⁸ Für ausführlichere Informationen zu nsd sei auf <https://www.kernel.org/doc/man-pages/online/pages/man8/nsd.8.html> verwiesen.

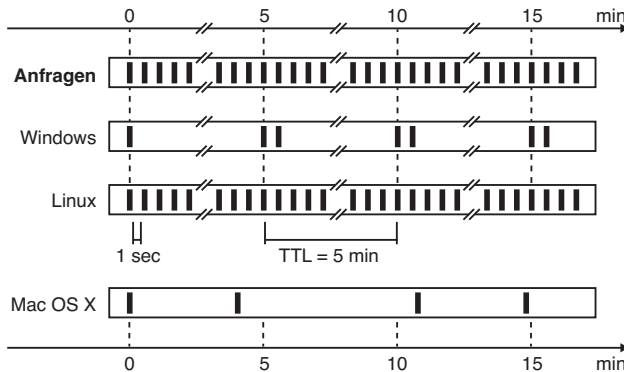


Abbildung 2.9: Illustration des Anfrageverhaltens der Stub-Resolver gängiger Betriebssysteme beim Aufruf der Funktion *gethostbyname* im Abstand von einer Sekunde für eine Domain mit einer TTL von fünf Minuten

Stub-Resolver unabhängigen Zwischenspeichers entwickelt.¹⁹ Eine Besonderheit stellt der Stub-Resolver des Betriebssystems Mac OS X dar, der RRs mit sehr kurzen TTLs eine minimale Lebensdauer von 15 Sekunden zuweist.

Versuch zum Caching im Stub-Resolver Mitunter weicht das in der Praxis zu beobachtende Caching-Verhalten der Stub-Resolver von der in der Literatur dokumentierten Funktionsweise ab. Dies lässt sich an einem Versuch mit einer dafür vorbereiteten Domain verdeutlichen. Die Domain sollte Außenstehenden nicht bekannt sein, um Störeinflüsse zu vermeiden. Der verwendete Domainname sollte zu Beginn des Versuchs nicht im Cache des rekursiven Nameservers vorliegen, der von den getesteten Systemen verwendet wird. Im RR des Domainnamens wird ein kleiner TTL-Wert T in der Größenordnung einiger Minuten eingetragen. Im Versuch wird nun auf verschiedenen Betriebssystemen die Routine zur Namensauflösung in regelmäßigen Abständen Δt aufgerufen, wobei gilt $\Delta t \ll T$. Das tatsächliche Anfrageverhalten des Stub-Resolvers kann dann mit einem Netzwerk-Sniffer (z. B. Wireshark) beobachtet werden: der Sniffer protokolliert die Zeitpunkte der DNS-Anfragen, welche der Stub-Resolver an den rekursiven Nameserver richtet.

Abbildung 2.9 illustriert die Ergebnisse für $T = 5$ Minuten und $\Delta t = 1$ Sekunde bei den Betriebssystemen Windows 7, Mac OS X 10.7.3 sowie Ubuntu Linux 11.10, das von Haus aus keinen DNS-Cache verwendet. Die Namensauflösung erfolgte im Versuch durch den Aufruf der *gethostbyname*-Funktion, die für diesen Zweck ausreichend ist. In der ersten Zeile der Abbildung sind die Zeitpunkte dargestellt, zu denen der Funktionsaufruf stattfand

¹⁹Ein Vertreter dieser Programme ist der „Acrylic DNS Proxy“, der unter <http://mayakron.net.au/support/browse.php?path=Acrylic&name=Home> bereitgestellt wird.

(„Anfragen“). In den verbleibenden Zeilen sind die Zeitpunkte abgebildet, an denen der Netzwerk-Sniffer DNS-Anfragen an den rekursiven Nameserver aufgezeichnet hat. Da in den ersten drei Zeilen über lange Zeiträume keine Veränderungen auftreten, zeigt die Abbildung jeweils nur die interessanten Ausschnitte aus der Zeitachse auf Sekundenebene. Die Versuchsreihe für Mac OS X ist hingegen maßstabsgetreu dargestellt.

Wie in der Abbildung angedeutet senden die Betriebssysteme beim allerersten Funktionsaufruf eine DNS-Anfrage an den rekursiven Nameserver, da kein Eintrag im Cache vorliegt. Die weiteren Anfragen werden von Windows gemäß des gesetzten TTL-Werts T innerhalb der nächsten 5 Minuten aus dem Cache bedient. Das untersuchte Linux sendet mangels Zwischenspeicher jede Anfrage an den rekursiven Nameserver.

Windows erzeugte im Versuch reproduzierbar unnötige DNS-Anfragen: Der Stub-Resolver sendet nach Ablauf der Gültigkeitsdauer des Cache-Eintrags erwartungsgemäß eine neue DNS-Anfrage an den rekursiven Nameserver. Obwohl dessen Antwort unmittelbar beim Client eintrifft, sendet der Stub-Resolver beim nächsten Aufruf von *gethostbyname* (eine Sekunde später) eine weitere Anfrage an den Nameserver, obwohl diese eigentlich bereits aus dem Cache beantwortet hätte werden können.

Im Unterschied zu Windows bzw. Linux wies der Stub-Resolver von MacOS X im Versuch ein vergleichsweise indeterministisches Verhalten auf. Der Resolver kontaktierte bei wiederkehrenden Anfragen für den Domainnamen teilweise deutlich vor bzw. nach Ablauf dessen Gültigkeit eine neue Anfrage an den rekursiven Nameserver. Die beobachtete Abweichung betrug dabei bis zu einer Minute – offenbar orientiert sich der Stub-Resolver also nur grob an der vom Nameserver übermittelten TTL.

Weiterhin deutet das bei MacOS X beobachtbare Verhalten darauf hin, dass bei diesem System nicht alle Funktionen, die zur Namensauflösung verwendet werden können, auf den bzw. auf denselben Cache zugreifen. Der Firefox-Browsers, der die Funktion *getaddrinfo* verwendet, profitierte bei der Namensauflösung eines Domainnamens nicht davon, dass dieser bereits im Cache vorhanden war, wenn der zugehörige Eintrag von einem anderen Prozess, der eine andere Namensauflösungsfunktion verwendet hatte (*gethostbyname*), zum Cache hinzugefügt worden war: Stattdessen stellte der Stub-Resolver eine neue Anfrage an den rekursiven Nameserver, um den *getaddrinfo*-Aufruf von Firefox zu bearbeiten.

Die Untersuchungen verdeutlichen, dass man nicht davon ausgehen kann, dass Endgeräte die vom rekursiven Nameserver übermittelte TTL in vollem Maße berücksichtigen.

2.7.3.3 Caching in Anwendungen

Üblicherweise wird der Stub-Resolver des Betriebssystems von den Anwendungen über die POSIX-konformen Systemfunktionen *gethostbyname* bzw. ihren Nachfolger *getaddrinfo* (vgl. Abschnitt 2.4.2.2) angesprochen. Beide Funktionen weisen eine wesentliche Einschränkung auf: Sie übermitteln als Ergebnis lediglich die IP-Adresse, jedoch nicht den

Wert der TTL aus der Antwort des rekursiven Nameservers bzw. des internen Zwischenspeichers. Bei der Spezifikation der Funktionsschnittstellen wurde eine Zwischenspeicherung der aufgelösten IP-Adressen in den Anwendungen offenbar nicht vorgesehen. Stattdessen wurde unterstellt, dass eine Anwendung vor jedem Verbindungsaufbau erneut die Funktion zur Namensauflösung aufruft. Während die für das DNS relevanten RFCs diesbezüglich keinerlei Vorgaben enthalten, findet sich in manchen Implementierungshandbüchern die explizite Empfehlung, das Ergebnis der Namensauflösung nicht zwischenzuspeichern.²⁰

RFC 4192, welcher sich mit Migrationsprozessen zur Einführung von IPv6 befasst, schlägt als Alternative die Verwendung der Funktion *getrrsetbyname* vor [BLD05, S. 12], welche dem Aufrufer Zugriff auf die Felder einer DNS-Antwort – und damit auf die TTL – bietet. Solange diese Funktion allerdings nicht in den POSIX-Standard aufgenommen wird, ist nicht mit einer breiten Unterstützung in den gängigen Betriebssystemen zu rechnen. Unterstützung für *getrrsetbyname* findet sich u. a. in OpenBSD (seit Version 3.0) sowie in der Resolver-Bibliothek von BIND 9 (vgl. die Manual Page der Funktion unter [Sch07a]). Windows-Betriebssysteme bieten mit der DnsQuery-API eine vergleichbare, jedoch ebenfalls noch kaum genutzte Möglichkeit zum Zugriff auf die Felder einer DNS-Antwort [Mic12a].

Da der Aufruf der Namensauflösungsfunktionen in der Praxis zu teilweise erheblichen Wartezeiten führen kann bzw. die in den Stub-Resolvern implementierten Zwischenspeicher aus Sicht der Anwendungsentwickler offenbar unzureichend waren, verfügen einige Anwendungen bzw. Laufzeitumgebungen inzwischen über einen eigenen DNS-Cache. Mangels Zugriff auf die vom DNS übermittelten TTL-Werte müssen Anwendungsentwickler eine willkürliche Aufbewahrungszeit für Einträge im Zwischenspeicher wählen.

Versuch zum Caching in Anwendungen Um die gängige Praxis zu evaluieren, wurde eine Auswahl von Web-Browsern sowie plattformunabhängige Laufzeitumgebungen untersucht. Wie Tabelle 2.2 zeigt, verwenden die betrachteten Anwendungen äußerst unterschiedliche Strategien.

Für die Web-Browser Firefox und Chromium ist der Quellcode öffentlich verfügbar. Um das tatsächlich implementierte Verhalten zu bestimmen, wurde der Quellcode der in Tabelle 2.2 angegebenen Versionen inspiziert. Beim **Firefox-Browser** ergibt sich das Verhalten des DNS-Caches anhand der Implementierung in der Datei *netwerk/dns/nsDNS-Service2.cpp*. Die maximale Größe beträgt demnach 400 Einträge (Parameter: *maxCacheEntries*), wobei jeder Eintrag exakt drei Minuten aufbewahrt wird. Für Einträge, welche in der letzten Minute ihrer Lebenszeit (Parameter *lifetimeGracePeriod*) aus dem Cache abgerufen werden, ruft der Browser im Hintergrund die Namensauflösung erneut auf, um

²⁰Als Beispiel sei auf den Band „Computer Science and Perl Programming“ aus der Reihe „Best of the Perl Journal“ [Dom02, S. 175] verwiesen, in dem folgender Hinweis steht: „Memoizing *gethostbyname* is technically incorrect, because the address data might change between calls. But in practice, address data doesn't usually change very quickly, and memoizing *gethostbyname* doesn't lead to any real problems except in long-running programs.“

Tabelle 2.2: Caching-Verhalten ausgewählter Anwendungen

Anwendung	Eigenschaften des DNS-Caches
Mozilla Firefox	Die Lebensdauer der Einträge beträgt stets 180 Sekunden; Erneuerung innerhalb der letzten 60 Sekunden (analyisierte Version: 11.0).
Google Chrome	Der Stub-Resolver wird verwendet. Die Lebensdauer der Einträge beträgt stets 60 Sekunden (analyisierte Version: 18.0.1025.168).
Google Chromium	Die integrierte Stub-Resolver-Implementierung nutzt die TTL des rek. Nameservers (analyisierte Version: 20.0.1108.0).
MS Internet Explorer	<i>Vor Version 4.0</i> betrug die Lebensdauer der Einträge 24 Stunden, <i>bei späteren Versionen</i> stets 30 Minuten [Mic11b].
Oracle Java VM	<i>Bis Version 1.5</i> haben Einträge eine unbegrenzte Lebensdauer, <i>ab Version 1.6</i> beträgt die Lebensdauer 30 Sekunden (ohne Security-Manager) bzw. ist sie unbegrenzt (mit Security-Manager).
Perl, Python, Ruby	Diese Laufzeitumgebungen nutzen den Stub-Resolver und verfügen über keinen integrierten DNS-Cache.

sie proaktiv zu aktualisieren. Für fortlaufend angefragte Domainnamen steht dadurch das Ergebnis der Auflösung stets verzögerungsfrei zur Verfügung. **Google Chrome** bzw. und das zugehörige Open-Source-Projekt **Chromium** verfahren ähnlich, verwenden jedoch eine Lebensdauer von nur 60 Sekunden (vgl. den Parameter `kCacheEntryTTLSeconds` in der Datei `net/base/host_resolver_impl.cc`).

Während die bislang genannten Browser sehr konservative (niedrige) Werte für die Lebensdauer von Cache-Einträgen verwenden, beträgt diese beim **Internet Explorer** 30 Minuten [Mic11b]. Der Vorteil eines so hohen Wertes liegt in der Reduktion der Anzahl der DNS-Anfragen an den rekursiven Nameserver. Kurzfristige Änderungen von IP-Adressen, wie sie etwa bei Webseiten vorkommen können, die zur Lastverteilung bzw. Erhöhung der Ausfallsicherheit Domainnamen mit sehr kurzen TTLs verwenden (s. Abschnitt 2.8.1), bemerkt der Internet Explorer im Vergleich zu den anderen Browsern dadurch allerdings deutlich später. Daraus können kurzfristige Verbindungsprobleme resultieren.

Dass die Namensauflösung beim Internet-Surfen eine wichtige Rolle spielt, lässt sich auch daran erkennen, dass sich die Browser-Hersteller nicht mehr mit den Möglichkeiten der Stub-Resolver-Implementierungen zufrieden geben, sondern direkte Kontrolle über den Prozess der Namensauflösung anstreben. Der Open-Source-Browser **Chromium** und der kommerzielle Ableger **Chrome** können so konfiguriert werden, dass sie statt des Stub-Resolvers des Betriebssystems eine eigenständige Stub-Resolver-Implementierung

verwenden (mit dem „Built-in Asynchronous DNS“-Flag, vgl. [Bea12; Gri12]). Dadurch erhält der Browser Zugriff auf die TTL-Werte der DNS-Antworten, wodurch eine effektivere Cache-Verwaltung realisiert werden kann, die sich zudem an die Vorgaben der autoritativen Nameserver hält. Auch das Mozilla-Team arbeitet nach eigenen Angaben daran, einen eigenständigen Resolver in die Gecko-Engine von Firefox zu integrieren [Moz13a].

Viele Hochsprachen bzw. Skriptsprachen greifen während der Ausführung auf eine Laufzeitumgebung zurück. Die Laufzeitumgebungen der Sprachen **Perl**, **Python** und **Ruby** bieten hingegen sowohl Komfort-Funktionen, die es z. B. ermöglichen, mit einem einzigen Funktionsaufruf eine Datei von einer URL (Uniform Resource Locator, [BLMM94]) mittels HTTP herunterzuladen, als auch direkten Zugriff auf die Systemfunktionen *gethostbyname* bzw. *getaddrinfo*. Ein DNS-Cache ist bei diesen drei Sprachen nicht vorgesehen.

Bei **Java** erlaubt die Laufzeitumgebung hingegen keinen direkten Zugriff auf die grundlegenden Systemfunktionen zur Namensauflösung. Stattdessen gibt es hier eine eigenständige, plattformunabhängige Funktionsbibliothek, welche zwar intern auf die Systemfunktionen zurückgreift, ihre Existenz bzw. die genaue Implementierung jedoch vollständig vor den Anwendungsentwicklern verbirgt. Im Unterschied zu den erstgenannten Laufzeitumgebungen implementiert die Java Virtual Machine (Java VM) für jeden Java-Prozess einen eigenständigen DNS-Cache, von dem Java-Anwendungen automatisch profitieren. Bis einschließlich Version 1.5 der Java VM wurden einmal im Cache abgelegte Einträge nicht mehr entfernt. Als Grund für dieses Verhalten nennt die API-Dokumentation der Funktion *InetAddress.getByName(String hostname)* den Schutz vor DNS-Spoofing-Angriffen [Ora10]. Mit der Einführung von Version 1.6 wurde dieses Verhalten geändert [Ora11]: Fordert eine Java-Anwendung einen sog. Security-Manager von der Java VM an, wird das bisherige Verhalten beibehalten. Andernfalls werden Einträge nach Ablauf einer festgelegten Dauer aus dem Cache entfernt. Der genaue Wert kann gemäß API-Dokumentation von der Implementierung der Java VM eigenständig festgelegt werden. Anhand einer Quellcode-Analyse lässt sich nachvollziehen, dass die Implementierung von Sun bzw. Oracle einen Wert von 30 Sekunden verwendet (vgl. den Quellcode der Klasse *sun.net.InetAddressCachePolicy*). Die Standard-Werte können Anwendungen durch Ändern einer Security-Property (*networkaddress.cache.ttl*) beeinflussen. Die Java-Umgebung der Android-Plattform für Smartphones führt seit Version 4.0 („Ice Cream Sandwich“) hingegen kein eigenes Caching mehr durch, da der Stub-Resolver des Betriebssystems die tatsächlichen TTL-Werte berücksichtigt [Goo14].

2.8 Veränderungen seit der Standardisierung

Die Nutzung des DNS hat sich im Laufe der Zeit durch äußere Einflüsse verändert. In diesem Abschnitt werden die fünf wesentlichen Veränderungen erläutert, welche im Zusammenhang mit der Betrachtung der Beobachtungsmöglichkeiten und der Möglichkeiten zum Schutz vor Beobachtung von Bedeutung sind.

Dieser Abschnitt ist wie folgt aufgebaut: Zunächst wird in Abschnitt 2.8.1 auf Content-Delivery-Netze eingegangen, bei denen autoritative Nameserver zu Lastverteilern werden. Anschließend werden zwei Bestrebungen angesprochen, in das Antwortverhalten der rekursiven Nameserver einzugreifen: DNS-basierte Internetsperren (Abschnitt 2.8.2) und die Manipulation von NXDOMAIN-Antworten (Abschnitt 2.8.3). Diese Entwicklungen sind mit dafür verantwortlich, dass Nutzer zunehmend auf rekursive Nameserver von Drittanbietern ausweichen, wie in Abschnitt 2.8.4 ausgeführt wird. Abschließend wird in Abschnitt 2.8.5 die Passive-DNS-Replication-Technik beschrieben, welche die Protokollierung und Analyse von DNS-Anfragen erleichtert.

Insbesondere die Verwendung der rekursiven Nameserver von Drittanbietern ist im Hinblick auf die Beobachtungsmöglichkeiten im DNS kritisch zu sehen.

2.8.1 Content-Delivery-Netze auf Basis des DNS

Content-Delivery-Netze geben Anbietern im Internet die Möglichkeit, Inhalte hochverfügbar und performant zur Verfügung zu stellen. Zur Realisierung greifen einige Lösungen auf das DNS zurück und nutzen es auf eine Art und Weise, die ursprünglich nicht vorgesehen war. Daraus resultiert ein verändertes Nutzungsverhalten des DNS, das bei der Betrachtung von Beobachtungsmöglichkeiten und Mechanismen zum Schutz vor Beobachtung berücksichtigt werden muss.

Dieser Abschnitt ist wie folgt aufgebaut: Zunächst wird in Abschnitt 2.8.1.1 die Problemstellung aufgezeigt, welche mittels des in Abschnitt 2.8.1.2 beschriebenen Lösungsansatzes von Content-Delivery-Netzen adressiert wird. Danach werden in Abschnitt 2.8.1.3 die Komponenten beschrieben, aus denen Content-Delivery-Netze typischerweise bestehen, bevor in Abschnitt 2.8.1.4 darauf eingegangen wird, welche Rolle das DNS beim Routing der Anfragen spielt. In Abschnitt 2.8.1.5 wird schließlich auf die Kritik eingegangen, der Content-Delivery-Netze ausgesetzt sind.

2.8.1.1 Problemstellung

Ende der 1990er Jahre kam es vermehrt zu Engpässen bei der Bereitstellung von Inhalten im WWW. Einige Inhaltsanbieter sahen sich mit unvorhersehbaren Lastspitzen mit spontan extrem stark ansteigenden Besucherzahlen (sog. „Flash-Crowds“ [Ari+03]), die weit über den Normalbetrieb hinausgingen, konfrontiert. Diese Engpässe waren zum einen auf zu gering dimensionierte Webserver zurückzuführen, was einen **Verlust der Verfügbarkeit** der Inhalte zur Folge hatte. Als Flaschenhals erwiesen sich auch die Kommunikationsverbindungen zwischen den Inhaltsanbietern und den Internetzugangsanbietern, woraus ein **Verlust der Erreichbarkeit** der Angebote resultierte.

Häufig genannt werden in der Literatur in diesem Zusammenhang die damals äußerst populäre Webseite der Firma „Victoria's Secret“, die Veröffentlichung des „Starr-Reports“ (mit den Untersuchungsergebnissen bzgl. der Affäre um US-Präsident Clinton) sowie

Sportereignisse wie die Olympischen Spiele [DK01; Ari+03]. Bei solchen *planbaren Ereignissen* könnten die Inhaltsanbieter die *Verfügbarkeit* prinzipiell durch eine temporäre Erweiterung der Webserver-Ressourcen aufrechterhalten. Bei *nicht-vorhersehbaren Ereignissen* gibt es allerdings häufig keine Vorlaufzeit. Als Beispiel kann hier der Angriff auf das World Trade Center am 11. September 2001 angeführt werden, der zu einer Überlastung bzw. dem Ausfall der Webseiten zahlreicher großer Nachrichtendienste führte [JKR02]. Zur Sicherstellung der Verfügbarkeit müssten die Inhaltsanbieter also *andauernd* eine großzügig bemessene Kapazitätsreserve vorhalten, was unwirtschaftlich wäre. Und: Auch bei entsprechender Vorsorge durch den Anbieter können die Inhalte für eine Vielzahl von Nutzern *unerreichbar* sein, wenn durch eine Flash-Crowd der Zugriffe die *Kommunikationsverbindungen* überlastet sind.

2.8.1.2 Lösungsansatz

Sogenannte „Content Delivery Networks“ (CDNs; teilweise auch als „Content Distribution Networks“ bezeichnet [PBV08, S. 9]) versprechen, auch bei großer Nachfrage die Verfügbarkeit und Erreichbarkeit von Inhalten zu gewährleisten. Dieses Ziel soll dadurch erreicht werden, dass die Inhalte nicht nur an einer Stelle im Netz vorgehalten werden, sondern auf möglichst vielen Servern, die zudem möglichst „nah“ bei den Nutzern platziert werden. Durch die Verteilung können Flaschenhälse vermieden werden.

Es gibt zahlreiche Architekturen und Implementierungsvarianten von CDNs, die ausführlich in der Literatur beschrieben werden. Einen guten Überblick gibt der Sammelband [BPV08]. Die folgende Darstellung beschränkt sich auf die zwei zentralen Aufgabenbereiche, die sich bei den in der Praxis verbreiteten CDNs identifizieren lassen: die **Verteilung der Inhalte** und das **Routing der Anfragen**.

2.8.1.3 Verteilung der Inhalte

Zur Verteilung der Inhalte kommen in einem CDN zwei Arten von Servern zum Einsatz [PBV08; PB08; VP03]: Die Inhaltsanbieter betreiben **Quell-Server** (*origin servers*), auf denen sie die zu verteilenden Daten bereitstellen. Neue bzw. aktualisierte Inhalte werden automatisch entweder von den Quell-Servern auf die **Replika-Server** (*surrogate servers* oder *edge servers*) kopiert („push“) oder von diesen heruntergeladen („pull“). Im Unterschied zu den Quell-Servern werden die Replika-Server in den Rechenzentren von Internetzugangsanbietern platziert, um eine möglichst geringe Distanz zu den Nutzern zu erreichen.

Die Replikation von Inhalten war im Internet schon vor der Entwicklung von CDNs bekannt [Bae+97; Dil+02]: So wurden unter anderem die umfangreichen Archive der Linux-Distributionen auf einer Vielzahl von *Mirror-Servern* zur Verfügung gestellt. Die Auswahl des für einen Download zu verwendenden Mirrors war jedoch *nicht transparent*; mitunter musste sie vom Nutzer manuell getätigt werden oder wurde mittels Heuristiken,

etwa anhand des Herkunftslandes oder der im Browser eingestellten Sprache vorgenommen. Bei CDNs ist das Routing der Anfragen zu einem Replika-Server hingegen völlig **transparent**, also für den Nutzer nicht erkennbar [PBV08, S. 9]. Dabei wird nicht nur die Distanz zwischen dem Nutzer und dem jeweiligen Replika-Server berücksichtigt, sondern auch die aktuelle Verkehrssituation sowie die Verfügbarkeit und Auslastung der einzelnen Replika-Server im CDN [PB08, S. 51 f.]. Dadurch können lokal begrenzte Ausfälle toleriert bzw. Engpässe bei der Bereitstellung der Inhalte vermieden werden.

Die Verbreitung von CDNs hat in den letzten Jahren erheblich zugenommen. Im Jahr 2004 griffen mehr als 3000 Unternehmen weltweit darauf zurück [PV06]. Einer der wichtigsten CDN-Anbieter ist die im Jahr 1998 gegründete Firma Akamai [Dil+02], die im Jahr 2012 nach eigenen Angaben zwischen 15 % und 30 % des gesamten globalen Datenverkehrsvolumens abwickelte [KFH12]. Einer Studie des Telekommunikationsausrüsters Calix zufolge, in der das Internetverhalten in ländlichen Gegenden der Vereinigten Staaten von Amerika untersucht wurde, wurden im ersten Quartal des Jahres 2012 83 % des heruntergeladenen Datenvolumens über CDNs abgewickelt [Cal12]. In der Studie von Callahan et al. war bei 24 % der beobachteten DNS-Anfragen ein Nameserver von Akamai an der Namensauflösung beteiligt [CAR13].

2.8.1.4 Routing der Anfragen mittels des DNS

Bei vielen in der Praxis eingesetzten CDNs erfolgt das Routing der Nutzeranfragen durch eine spezielle Behandlung von DNS-Anfragen in autoritativen Nameservern, die vom CDN-Anbieter betrieben werden (vgl. etwa [VP03; NSS10]). Die anzubietenden Inhalte werden dazu unter einem vom CDN-Anbieter festgelegten, sich im Zeitverlauf nicht ändernden Domainnamen N zur Verfügung gestellt. Die Routing-Entscheidung, also die Auswahl des Replika-Servers $s \in \{s_1, \dots, s_i\}$, der einen Nutzer bedienen soll, wird während der Namensauflösung in den autoritativen Nameservern des CDNs getroffen, indem Anfragen für N mit der jeweiligen IP-Adresse des ermittelten Replika-Servers s beantwortet werden.

Für den Inhaltsanbieter hat diese Realisierung den Vorteil, dass er beim Gestalten seiner Webseiten keine dynamisch generierten URL (Uniform Resource Locator, [BLMM94]) verwenden muss, da der tatsächlich dem Nutzer zugewiesene Replika-Server s immer unter demselben Domain-Namen N angesprochen wird. Dieses Prinzip wird auch als **Late Binding** bezeichnet [Pan+04].

Die Verwendung der Namensauflösung für das Routing der Anfragen erscheint angesichts der im DNS bereits vorgesehenen Round-Robin-Funktionalität (s. Abschnitt 2.5) naheliegend. Bei einem Round-Robin-Eintrag werden für einen Namen mehrere IP-Adressen in der Zone abgespeichert. Die Einträge werden bei jeder eingehenden Anfrage zyklisch vertauscht. Da Clients die Adressen üblicherweise in der angebotenen Reihenfolge verwenden, lässt sich somit innerhalb einer Zone eine pseudozufällige, zustandslose Lastverteilung für einen Domainnamen realisieren. CDNs erweitern dieses Grundkonzept und bieten zudem die Möglichkeit, die Lastverteilung situationsabhängig zu beeinflussen.

Akamais Infrastruktur, die ausführlich in [NSS10] beschrieben wird, besteht aus autoritativen DNS-Servern, welche das Anfrage-Routing übernehmen, sowie den bei Akamai als *Edge-Server* bezeichneten Replika-Servern, welche die Inhalte am „Rand“ des Internets vorhalten. Akamai platziert stets eine Gruppe von Edge-Servern, welche für dieselben Inhalte zuständig sind, bei einem Internetzugangsprovider. Eine solche Anordnung wird in [NSS10] als *Cluster* bezeichnet. Bei den DNS-Servern unterscheidet Akamai zwischen **Top-Level-Nameservern (TLNS)**, welche eine Zuordnung der Anfrage auf der Cluster-Ebene vornehmen, und **Low-Level-Nameservern (LLNS)**, welche für das Routing zu einzelnen Edge-Servern zuständig sind.

Das Anfrage-Routing von Akamai ist Gegenstand umfangreicher Untersuchungen [Dil+02; PHL03; Su+09; NSS10; KFH12]. Auf eine ausführliche Beschreibung soll an dieser Stelle zu Gunsten eines illustrativen Beispiels verzichtet werden:

Beispiel 2.5. Ein Nutzer ruft im Browser die Webseite *www.audi.de* auf, welche zur Lastverteilung auf das Akamai-CDN zurückgreift. Die nachfolgende Beschreibung geht davon aus, dass keine für die Namensauflösung relevanten Daten in den Zwischenspeichern der beteiligten Komponenten vorliegen. Im einzelnen werden folgende Schritte durchlaufen (Stand: Januar 2014):

1. Der Browser sendet über den Stub-Resolver eine rekursive DNS-Anfrage für den Namen *www.audi.de* an den rekursiven Nameserver (**RNS**).
2. Der RNS leitet die Anfrage an einen Root-Server weiter. Dieser liefert die NS-Records für die TLD *de* zurück, in der die zuständigen autoritativen Nameserver aufgelistet sind.
3. Der RNS wählt einen TLD-Nameserver aus, im Beispiel *f.nic.de*, und übermittelt ihm die Anfrage. Der TLD-Nameserver antwortet mit NS-Records, in denen die autoritativen Nameserver der SLD *audi.de* genannt werden.
4. Der RNS wählt einen SLD-Nameserver aus, im Beispiel *ns2.audi.de*, und übermittelt ihm die Anfrage. Der Nameserver antwortet mit einem CNAME-Record, der besagt, dass *www.audi.de* ein Alias für ***www.audi.de.edgesuite.net*** ist.
5. Nun muss der Alias *www.audi.de.edgesuite.net* aufgelöst werden. Dazu werden die Schritte 1–4 in analoger Weise durchlaufen. Am Ende erfährt der RNS von einem autoritativen Nameserver der Domain *edgesuite.net*, die von Akamai verwaltet wird, dass es sich bei *www.audi.de.edgesuite.net* wiederum um einen Alias handelt, dessen kanonischer Name ***a1845.ga.akamai.net*** ist.
6. Bei der Auflösung der Domain *a1845.ga.akamai.net* findet das eigentliche Anfrage-Routing statt. Der RNS übermittelt die Anfrage an den TLD-Nameserver der *net*-Domain und erfährt von ihm die NS-Einträge der Domain *akamai.net*. Dabei handelt es sich um die TLNS von Akamai. Da Akamai keine Kontrolle darüber hat, in welcher Reihenfolge die NS-Einträge vom TLD-Nameserver angeordnet werden, kann in

diesem Schritt noch kein Anfrage-Routing durchgeführt werden. Um eine gute Erreichbarkeit der TLNS für alle Clients zu gewährleisten, werden diese – wie die DNS-Root-Server – durch IP-Anycast-Routing global im Netz verteilt (s. Abschnitt 3.5.3).

7. Der RNS wählt einen TLNS aus, im Beispiel *zc.akamaitech.net*, und übermittelt die Anfrage für *a1845.ga.akamai.net* dorthin. Der TLNS antwortet mit den NS-Einträgen der Domain *ga.akamai.net*. Dabei handelt es sich um die LLNS von Akamai. Unabhängig vom Standort des Clients liefern die TLNS immer dieselben Namen in den NS-Einträgen zurück (im Beispiel: *n0ga.akamai.net* bis *n7ga.akamai.net*); die in Glue-Records (s. Beispiel 2.2) mitgelieferten IP-Adressen der LLNS unterscheiden sich jedoch je nachdem von welchem Standort (bzw. von welcher IP-Adresse aus) die Anfrage gestellt wurde [PHL03]. Dadurch wird die Anfrage einem (bzw. mehreren) Edge-Clustern zugeordnet.
8. Schließlich wählt der RNS einen LLNS aus, im Beispiel *n5ga.akamai.net*, und übermittelt ihm die Anfrage für *a1845.ga.akamai.net*. Der LLNS antwortet mit einer Liste von (bei Akamai meist zwei) A-Records, in denen die IP-Adressen der Edge-Server enthalten sind, die für diesen Client ausgesucht wurden. □

Durch das zweistufige Routing der Anfragen und die redundante Auslegung der Infrastruktur auf allen Ebenen kann Akamai flexibel auf Ausfälle und Engpässe reagieren, welche einzelne Edge-Server, ganze Server-Cluster oder Kommunikationsverbindungen in Teilen des Internets betreffen. Das Beispiel macht deutlich, dass das Anfrage-Routing im Vergleich zur regulären Namensauflösung ohne CDN (entspricht den Schritten 1–3 im Beispiel) wesentlich mehr Schritte benötigt. Dadurch kann die Wartezeit für den Nutzer erheblich steigen [KWZ01]. Um die Wartezeit zu senken, werden in den NS-Einträgen der TLNS und LLNS große TTL-Werte verwendet. Beim letzten Schritt, der Auswahl des Edge-Servers, ist das DNS-Caching hingegen hinderlich: Um auf plötzliche Änderungen schnell reagieren zu können, werden hier sehr kleine TTL-Werte verwendet: Bei Akamai beträgt die TTL hier typischerweise 20 Sekunden.

2.8.1.5 Kritik

Wie [FMA11] feststellen führen CDNs durch die dynamische, ortsabhängige Beantwortung von DNS-Anfragen dazu, dass das DNS seine globale Konsistenz (vgl. „eventual consistency“ in Abschnitt 2.7.1) verliert. Das DNS-basierte Routing basiert auf zwei Annahmen, die in der Praxis nicht immer zutreffen. Die Technik steht daher in der Kritik.

Lokalisierungsfehler Zum einen wird unterstellt, dass sich der rekursive Nameserver, der die Anfragen eines Clients an die autoritativen Server von Akamai übermittelt, in der Nähe des Clients befindet. Die DNS-Server von Akamai erfahren nämlich nicht die IP-Adresse des Clients, sondern lediglich die Adresse seines rekursiven Nameservers – sie wählen also einen Replika-Server aus, der sich netztopologisch in der Nähe des

rekursiven Nameservers befindet, jedoch nicht unbedingt in der Nähe des Clients. Bei einigen Internetzugangsanbietern befindet sich der rekursive Nameserver jedoch nicht im selben Netzsegment, über das die Kunden an das Internet angebunden sind. In der Folge kommunizieren die Kunden mit einem weiter entfernten Replika-Server, sodass die Verwendung eines CDNs mitunter sogar zu einer Verschlechterung der Performanz führt [KWZ01; Vix09]. In einer Studie von Mao et al. waren nur 15 % der Nutzer im selben Netzsegment wie ihr rekursiver Nameserver [Mao+02] und in [STA01] betrug die Distanz in 30 % der Fälle mehr als acht Hops. Insbesondere bei Nutzern, die einen rekursiven Nameserver eines Drittanbieters verwenden (s. Abschnitt 2.8.4), kann es zu Lokalisierungsfehlern kommen [KFH12].

Derzeit werden verschiedene Anpassungen diskutiert, um über die DNS-Anfragen Informationen zu den autoritativen Nameservern zu transportieren, welche für die Routing-Entscheidung relevant sind [Con+12; FMA11]. Am weitesten entwickelt ist ein Konzept, mit dem die IP-Adresse des Clients an die autoritativen Nameserver übermittelt werden kann. Die Protokoll-Spezifikation („draft-vandergaast-edns-client-subnet-01“) befindet sich derzeit allerdings noch im Entwurfsstadium [Con+12]. Sie sieht vor, dass rekursive Nameserver in die Anfragen, welche sie an die autoritativen Server senden, eine EDNS0-Option mit der IP-Adresse des jeweiligen Clients einfügen. Zur Wahrung der Privatsphäre soll statt der vollständigen Adresse jedoch lediglich eine auf 24 Bit verkürzte Subnetz-Adresse an den autoritativen Server übermittelt werden. Weiterhin sollen Clients gegenüber dem rekursiven Nameserver ausdrücken können, dass ihre Adresse nicht eingebettet werden soll. Die Implementierung des Verfahrens gestaltet sich aufwendig, da Anpassungen an allen an der Namensauflösung beteiligten Komponenten erforderlich sind.

Vernachlässigung der TTLs Die zweite Annahme betrifft die Reaktionsgeschwindigkeit der Clients bei Änderungen im Routing. Fällt ein Edge-Server oder ein Cluster aus, werden die betroffenen Adressen von den TLNS bzw. LLNS nicht mehr herausgegeben und durch andere Server ersetzt. Clients, welche den Domainnamen vor der Änderung aufgelöst haben, verwenden jedoch nach wie vor den ihnen zugewiesenen Replika-Server. Durch die Verwendung besonders kurzer TTLs sollen die Clients zum häufigen Aktualisieren gezwungen werden, um sicherzustellen, dass die von der Störung betroffenen Clients die Routing-Anpassung innerhalb weniger Sekunden erfahren. In der Praxis werden die TTLs jedoch häufig ignoriert. Pang et al. stellten bei der Analyse von Log-Dateien fest [Pan+04], dass bei einem TTL-Wert von 10 Minuten nach einer Änderung der Adresszuordnung noch 47 % der Clients auf die vorher eingetragenen Adressen zugriffen. Ein Großteil der betroffenen Clients hatte auch zwei Stunden später noch nichts von der Umstellung bemerkt. Diese Beobachtung lässt sich durch die vielschichtige DNS-Caching-Infrastruktur, bestehend aus rekursivem Nameserver, Betriebssystem und Browser, erklären, auf die in Abschnitt 2.7 bereits eingegangen wurde. Überraschend ist hingegen das Ergebnis einer weiteren Untersuchung, in der Pang et al. das Verhalten von rekursiven Nameservern untersuchten: Immerhin 14 % der Nameserver hielten sich nicht an den in der Zone definierten TTL-Wert und übermittelten veraltete Daten an ihre Clients.

Weitere Kritikpunkte führt RFC 3568 an [Bar+03], u.a. folgende:

- Durch kurze TTL-Werte steigt die Anzahl der DNS-Anfragen, was zu einer höheren Belastung der rekursiven Nameserver führen kann. Studien zur Untersuchung des Einflusses kurzer TTLS zeigten jedoch, dass es in der Regel nur zu einer moderaten Mehrbelastung kommt [Jun+02; BA11].
- Verwenden mehrere Benutzer einen rekursiven Nameserver, werden sie innerhalb der TTL alle auf denselben Replika-Server (bzw. denselben Cluster) verwiesen. Flash-Crowds können dadurch immer noch zu kurzfristigen Überlastungen führen.

Einfluss auf die Beobachtungsmöglichkeiten Durch die Verwendung kleiner TTL-Werte bei CDNs verbleiben Informationen für angefragte Domainnamen nur noch kurze Zeit in den Zwischenspeichern der Stub-Resolver (bzw. der rekursiven Nameserver). Bei großen TTL-Werten erhält der rekursive Nameserver nach der ersten DNS-Anfrage keine weiteren DNS-Anfragen, wenn der Nutzer auf einer Webseite navigiert und fortlaufend neue Inhalte abrufen. Bei kleinen TTL-Werten muss der Stub-Resolver hingegen in relativ kurzen Zeitabständen erneut eine DNS-Anfrage stellen, wodurch der rekursive Nameserver darauf schließen kann, wie lange ein Nutzer auf einer Webseite verweilt.

Bei der von Akamai gewählten Strukturierung des Namensraums, bei der lediglich die A-Einträge der Edge-Server eine kurze TTL besitzen, lassen sich anhand der wiederkehrenden Anfragen für den generischen Domainnamen (*a1845.ga.akamai.net* im Beispiel) jedoch keine Rückschlüsse auf die abgerufene Webseite ziehen.

Bei anderen CDNs sind anhand der verwendeten Namen ggf. jedoch Rückschlüsse möglich: In der Webseite des Anbieters Amazon werden eingebettete Bilder mittels eines CDNs u. a. von Edge-Servern abgerufen, die über den Domainnamen *ecx.images-amazon.com* mit einer TTL von 60 Sekunden angesprochen werden. Zudem setzen inzwischen auch Webseiten, die kein CDN nutzen, kleine TTL-Werte ein, um über das DNS eine Lastverteilung zu realisieren (vgl. die Erläuterungen zu niedrigen TTLS in Abschnitt 2.7.2).

Unabhängig davon, ob die wiederkehrend angefragten Domainnamen Hinweise auf die besuchte Webseite geben oder nicht, führen kleine TTL-Werte zu einem Wachstum der Anzahl der DNS-Anfragen, die ein Nutzer in einer Sitzung stellt. Die wiederkehrenden Anfragen für einen Domainnamen können die verhaltensbasierte Verkettung von Sitzungen, auf die in Kapitel 6 eingegangen wird, begünstigen: In den in Abschnitt 7.3 durchgeführten Untersuchungen sinkt die erzielbare Verkettungsgenauigkeit erheblich, wenn die Einträge länger im *Cache* aufbewahrt werden.

2.8.2 DNS-basierte Internetsperren

In mehreren Ländern gibt es Bestrebungen, durch gesetzliche Regelungen den Zugang zu bestimmten Internetseiten zu unterbinden. Angesichts der zentralen Rolle, welche

die rekursiven Nameserver bei der Namensauflösung spielen, eignen sie sich prinzipbedingt dazu, die Online-Aktivitäten ihrer Nutzer einzuschränken: Verweigert der rekursive Nameserver die Namensauflösung für einen Domainnamen, kann der Browser dessen IP-Adresse nicht ermitteln und folglich keine Verbindung zu der Seite herstellen.

Die Motivation für die Nutzung des DNS zur Etablierung von Internetsperren besteht darin, dass die Sperre anhand des Domainnamens im Vergleich zu alternativen Formen, insbesondere der Sperre auf Basis der IP-Adresse des Webservers, präziser ist [Var10]: Bei **Internetsperren auf Basis von IP-Adressen** besteht zum einen das Problem des sog. „Underblockings“ (vgl. u. a. [RHR04]), d. h. es werden nicht alle IP-Adressen gesperrt, von denen der zu sperrende Inhalt abgerufen werden kann. Diese Situation kann zum Beispiel entstehen, wenn der zu sperrende Inhalt mittels eines Content-Delivery-Netzes auf zahlreichen Webservern vorgehalten wird (s. Abschnitt 2.8.1). Andererseits besteht das Problem des sog. „Overblockings“ [RHR04], d. h. die Sperre betrifft auch erwünschte Inhalte, wenn diese über dieselben IP-Adressen abgerufen werden können wie die zu sperrenden Inhalte. Diese Situation tritt insbesondere beim Virtual-Hosting auf, das im Internet weitverbreitet ist [SKG07].

Zu den Staaten, in denen solche DNS-basierte Internetsperren implementiert oder diskutiert wurden, gehören u. a. China [ZE03], Pakistan [Nab13], Iran [AAH13] und die Türkei [VG12]. In den USA wurde die Implementierung von DNS-Sperren im Zuge der Einführung des „Stop Online Piracy Acts“ (SOPA) bzw. des „Protect Intellectual Property Acts“ (PIPA) vorgeschlagen [LLP11]. In Deutschland wurden DNS-Sperren u. a. in Zusammenhang mit dem Zugangserschwerungsgesetz („Gesetz zur Bekämpfung der Kinderpornografie in Kommunikationsnetzen“) diskutiert [Kle10].

Anhand der DNS-basierten Internetsperren wird deutlich, dass die rekursiven Nameserver ein erhebliches Kontroll- und Beobachtungsinstrument darstellen. Wie im Folgenden erläutert wird, eignen sie sich jedoch nicht zur zuverlässigen Kontrolle.

Umsetzung von DNS-basierten Internetsperren und Umgehungsmöglichkeiten

Die Vorschläge zur Implementierung von Zugangsbeschränkungen mittels des DNS sehen üblicherweise vor, die Internetzugangsanbieter des jeweiligen Landes zu verpflichten, ihre rekursiven Nameserver so zu konfigurieren, dass diese bestimmte Domainnamen nicht auflösen (durch REFUSED- oder NXDOMAIN-Antworten, s. Abschnitt 2.6.3) oder Anfragen für diese Domainnamen auf eine entsprechende Hinweisseite umleiten (durch Manipulation der im A-Record übertragenen IP-Adresse).

Eine solche Internetsperre können die Nutzer leicht umgehen, da sie nicht zwangsläufig den rekursiven Nameserver des Internetzugangsanbieters verwenden müssen; sie können ihre Systeme so konfigurieren, dass diese die DNS-Anfragen an einen anderen Nameserver senden, der z. B. in einer anderen Jurisdiktion betrieben wird, die keine DNS-Sperren implementiert. Zahlreiche Anbieter stellen rekursive Nameserver – üblicherweise kostenfrei – zur Verfügung (s. Abschnitt 2.8.4). Eine wirkungsvolle Zugangsbeschränkung kann

allein durch eine auf den rekursiven Nameservern implementierte DNS-Sperre daher nicht erreicht werden.

Eine zusätzliche Maßnahme zur Implementierung von Internetsperren, die etwa in China eingesetzt wird [ZE03], besteht darin, den DNS-Datenverkehr zu fremden rekursiven Nameservern zu unterbinden, etwa durch Filterung des Datenverkehrs auf dem UDP- bzw. TCP-Port 53 (vgl. Abschnitt 2.6.1). Durch diesen Angriff auf die Verfügbarkeit (s. Abschnitt 3.5) können die Nutzer nicht mehr auf einen anderen rekursiven Nameserver ausweichen. Eine solche rein portbasierte Filterung lässt sich jedoch überwinden, indem die DNS-Anfragen auf einem anderen Port an einen entsprechend konfigurierten rekursiven Nameserver gesendet werden.

Die Verwendung eines alternativen Ports zum Nachrichtentransport scheitert, wenn mittels Deep-Packet-Inspection- [Dha+03] oder Verkehrsanalyse-Techniken [ZNA05; Fin+10] DNS-Anfragen auf beliebigen Ports anhand des Nachrichtenaufbaus erkannt und unterdrückt werden. Ein Zugriff auf die gesperrten Internetseiten ist selbst in diesem Fall allerdings noch möglich, etwa indem die Nutzer einen Anonymitätsdienst wie Tor [DMS04] oder AN.ON [BFK00] einsetzen. Generische Anonymitätsdienste können grundsätzlich auch dazu eingesetzt werden, lediglich den DNS-Datenverkehr zu einem nicht-zensierenden rekursiven Nameserver zu transportieren, da die generischen Dienste jedoch zum Transport von größeren Nachrichten vorgesehen sind, ist die solchermaßen abgesicherte Namensauflösung relativ ineffizient und langsam (s. Abschnitt 5.2.2).

Die obigen Betrachtungen zeigen, dass eine wirksame Zugriffsbeschränkung mit DNS-Sperren kaum durchzusetzen ist. Eine wesentlich wirkungsvollere Maßnahme setzt hingegen bei den autoritativen Nameservern an, welche die Resource-Records vorhalten. Werden die NS-Einträge für diese Nameserver in der nächsthöheren Ebene des Namensraums, üblicherweise in den TLD-Servern, entfernt, laufen die oben angesprochenen Umgehungsmaßnahmen ins Leere, da der gewünschte Domainname von überhaupt keinem Punkt im Netz mehr aufgelöst werden kann.

2.8.3 Manipulation von NXDOMAIN-Antworten

Erreicht einen autoritativen Nameserver eine Anfrage für eine Domain, für die er zuständig ist, antwortet er entweder mit dem zugehörigen RR aus seiner Zonendatei (bei dem es sich ggf. um ein Referral zu einem anderen Nameserver handelt, s. S. 19) oder mit einem NXDOMAIN-Fehler, falls es keinen Eintrag für den angefragten Namen gibt (s. Abschnitt 2.6.3). Die meisten interaktiven Client-Anwendungen, etwa Web-Browser, präsentieren dem Nutzer üblicherweise eine Fehlermeldung, in der sie auf den nicht-existierenden Namen hinweisen. Eine häufige Ursache sind Tippfehler oder falsch geschriebene Links in Webseiten.

Motiviert durch potenzielle zusätzliche Einnahmequellen sind einige Betreiber von Nameservern dazu übergegangen, bei nicht-existent Domains statt eines NXDOMAIN-Fehlers eine normale DNS-Antwort zurückzuliefern. Diese Antwort enthält einen A-

Record, der auf einen Webserver des Nameserver-Betreibers verweist, der den Nutzer über den Fehler informiert. Üblicherweise werden dort zusätzlich Werbefbanner oder Verweise auf Partnerangebote eingeblendet, von denen der Nameserver-Betreiber eine Vergütung erhält. Auf der Fehlerseite werden mitunter auch Vorschläge zur korrekten Schreibweise des Namens oder dazu passende Suchergebnisse einer Suchmaschine aufgeführt, was nach Aussagen einiger Anbieter zu einer besseren Benutzbarkeit des World Wide Webs führe und mitunter vollmundig als „error-path correction“ [Dag+08a] bezeichnet wird. Streng genommen stellen solche Aktivitäten jedoch einen **Angriff auf die Integrität** dar (s. auch Abschnitt 3.6).

2.8.3.1 NXDOMAIN-Manipulation durch autoritative Nameserver

Die erste Realisierung einer solchen „NXDOMAIN-Substitution“ *durch autoritative Server* wurde im Jahr 2004 durch die ICANN (Internet Corporation for Assigned Names and Numbers) in einem ausführlichen Bericht dokumentiert [Sec04]: Die Firma Verisign, die seit dem Jahr 2000²¹ für die Verwaltung der TLDs „com“ und „net“ zuständig ist, aktivierte am 15. Oktober 2003 ihren Dienst „SiteFinder“. Dazu wurde in jede der beiden Zonen ein Wildcard-Eintrag (s. Beispiel 2.3) eingefügt. Dies hatte zur Folge, dass bei Anfragen für unregistrierte Domains kein NXDOMAIN-Fehler mehr ausgeliefert wurde, sondern die IP-Adresse eines SiteFinder-Webservers.

Unmittelbar nach der Einführung wurde die Lösung von DNS-Experten und der ICANN vehement kritisiert [Sec04]: Zum einen wurde Verisign vorgeworfen, seine herausgehobene Marktposition als Registrar unangemessen auszunutzen und die Nutzer zu bevormunden. Der SiteFinder-Dienst war nicht nur ungefragt für alle Nutzer eingeführt worden; die Nutzer hatten auch keine Möglichkeit, den Dienst wieder zu deaktivieren, d. h. es gab keine „Opt-Out“-Möglichkeit. Zum anderen machte SiteFinder bereits existierende konkurrierende Lösungen, etwa eine entsprechende Browser-Funktion zum automatische Aufruf einer Suchmaschine im Falle einer NXDOMAIN-Antwort, unbrauchbar.

Im ICANN-Bericht [Sec04] wird auch der Umgang des Systems mit E-Mails kritisiert, die einen Tippfehler in der Domain von Empfänger-E-Mail-Adressen enthalten. Durch den Wildcard-Eintrag erhielten nicht nur Web-Browser und Client-Anwendungen bei Anfragen für unregistrierte Domains eine IP-Adresse, sondern auch Mailserver – schließlich kann der autoritative Nameserver bei einer eingehenden Typ-A-Anfrage nicht unterscheiden, für welches Anwendungsprotokoll der Name aufgelöst werden soll.²² Um den Absender über den Tippfehler zu informieren, akzeptierten die SiteFinder-Server eingehende SMTP-Verbindungen zunächst, um eingehende E-Mails dann im Laufe des Einlieferungsprozesses mit einer Fehlermeldung („bounce“) abzuweisen. Durch diese Vorgehensweise

²¹Im Jahr 2000 hat Verisign die Firma Network Solutions, Inc. übernommen, die zu diesem Zeitpunkt als Registrar für die beiden TLDs zuständig war [Net].

²²Mailserver stellen üblicherweise zunächst eine Anfrage für einen MX-Record, fallen jedoch automatisch auf eine A-Anfrage zurück, falls für eine MX-Anfrage keine Daten vorliegen.

erlangte Verisign jedoch Kenntnis von Absender- und Empfängeradressen, was von den Kritikern als Verstoß gegen einschlägige Datenschutzrichtlinien eingestuft wurde. Da die von SiteFinder ausgelieferten Werbebanner ein Tracking-Cookie setzten, das eine Nachverfolgung der Nutzeraktivitäten ermöglichte, waren Verisigns Beteuerungen, die Privatsphäre der Nutzer zu respektieren, wenig glaubhaft [Sec04].

Nach der Intervention der ICANN schaltete Verisign den SiteFinder-Dienst am 1. November 2003 wieder ab. Angesichts der schlechten Erfahrungen mit Verisign veröffentlichte die ICANN im Jahr 2009 ein weiteres Dokument mit Richtlinien-Charakter, welches auf die Gefahren der Verwendung von Wildcard-Einträgen auf Ebene der TLDs hinweist und erläutert, warum die ICANN diese Praktik in den gTLDs nicht duldet [ICA09].

2.8.3.2 NXDOMAIN-Manipulation durch rekursive Nameserver

Neben den autoritativen Servern sind auch *rekursive Nameserver* in der Lage, NXDOMAIN-Antworten zu manipulieren. Auswertungen auf Basis der Daten, die mit dem am ICSI (*International Computer Science Institute* der University of California in Berkeley) entwickelten DNS-Analyse-Werkzeug *Netalyzer*²³ erhoben wurden, deuten auf eine hohe Verbreitung hin: Von 198 000 Netalyzer-Sitzungen, die zwischen den Jahren 2009 und 2011 aufgezeichnet wurden, gab es bei 27 % der Sitzungen Anzeichen dafür, dass ein rekursiver Nameserver verwendet wurde, welcher NXDOMAIN-Antworten manipuliert [Wea+11]. In einer Untersuchung von Dagon et al., in der etwa 600 000 öffentlich erreichbare rekursive Nameserver überprüft wurden, konnten bei 2 % der Server Hinweise auf die Implementierung der NXDOMAIN-Manipulation nachgewiesen werden [Dag+08a]. Der Anteil der tatsächlich betroffenen Nutzer ist in vielen Ländern jedoch deutlich höher, da vor allem große Internetzugangsanbieter wie T-Online und Alice, aber auch populäre Drittanbieter wie OpenDNS (s. Abschnitt 2.8.4) die Technik anwenden [WKP11].

Zwar erlauben die meisten Anbieter ihren Kunden inzwischen, die Funktion abzuschalten, davon macht jedoch laut Weaver et al. nur eine Minderheit Gebrauch: Nur bei 30 % der Netalyzer-Sitzungen, die Kunden der Deutschen Telekom beigesteuert hatten, gab es *keine* Anzeichen für die NXDOMAIN-Manipulation [WKP11]. Abbildung 2.10 zeigt ein Beispiel einer Fehlerseite, wie sie durch die als „Navigationshilfe“ bezeichnete Funktion von T-Online im Jahr 2009 angezeigt wurde.

Derzeit wird die NXDOMAIN-Manipulation meist direkt im rekursiven Nameserver des Internetzugangsanbieters implementiert: Weaver et al. führen in [WKP11] zahlreiche darauf spezialisierte Dienstleister an, welche den Internetzugangsanbietern die benötigte Infrastruktur zur Verfügung stellen. Die Ersetzung im rekursiven Nameserver des Internetzugangsanbieters können die Kunden durch Verwendung eines rekursiven Nameservers eines Drittanbieters (s. Abschnitt 2.8.4) umgehen. Erkenntnissen von Weaver et al. und Vixie zufolge nehmen einige Anbieter die Ersetzung daher durch direkte Manipulation

²³Netalyzer ist unter <http://netalyzer.icsi.berkeley.edu> erreichbar.

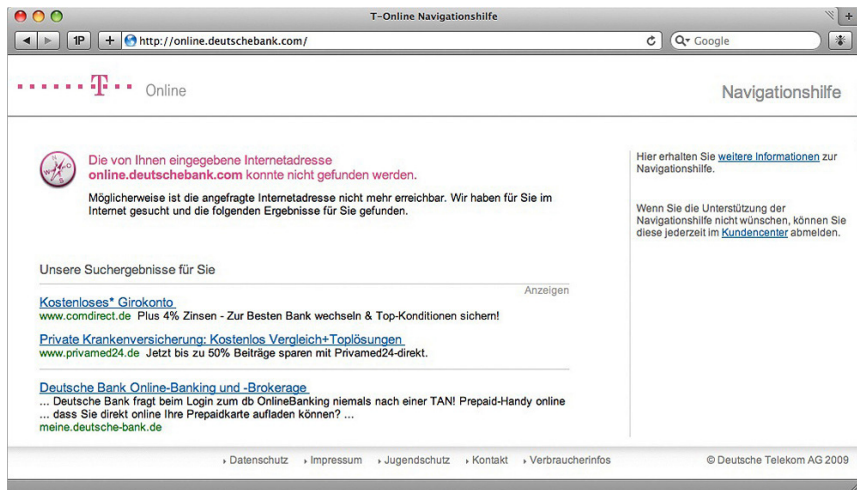


Abbildung 2.10: NXDOMAIN-Manipulation am Beispiel der „Navigationshilfe“ des Anbieters T-Online (Abb. nach [Lup09]; nachbearbeitet)

der übertragenen DNS-Nachrichten vor [WKP11; Vix09]; die Ersetzung kann dann nicht mehr durch die Verwendung eines Drittanbieter-Nameservers umgangen werden.

Die NXDOMAIN-Manipulation durch rekursive Nameserver wird vom Security and Stability Advisory Committee (SSAC) der ICANN abgelehnt. Als sie im Jahr 2008 dazu einen gesonderten Bericht mit Empfehlungs-Charakter [ICA08] veröffentlicht hatte, wurde von den Befürwortern der NXDOMAIN-Manipulation noch an einer Standardisierung entsprechender Maßnahmen gearbeitet: Im Jahr 2009 erschien der erste Entwurf des Internet-Standards „Recommended Configuration and Use of DNS Redirect by Service Providers“ bei der IETF [Cre+09]. Das Standardisierungsvorhaben, das u. a. von Vixie massiv kritisiert wurde [Vix09], ist im April 2011 allerdings ergebnislos ausgelaufen.

2.8.4 Verwendung rekursiver Nameserver von Drittanbietern

Im Normalfall verwenden Internetnutzer den rekursiven Nameserver ihres Internetzugangsanbieters. Bei gängigen Breitbandzugängen übermittelt der Anbieter die IP-Adressen von zwei rekursiven Nameservern bei der Einwahl mittels DHCP an den Internet-Router des Kunden (s. Abschnitt 2.4.2.3). Es besteht jedoch auch die Möglichkeit, im Router oder direkt in den Endgeräten einen anderen rekursiven Nameserver einzutragen.

Seitdem Google im Jahr 2009 kostenlose rekursive Nameserver eingeführt hat, haben sich zahlreiche professionelle Angebote am Markt positioniert, die häufig unter den Begriffen

„Third-party DNS“ [Age+10], „Public DNS“ [Hua+11] oder „Cloud DNS“ [KFH12] zusammengefasst werden. Während vor der Einführung von Googles Dienst wohl nur versierte Anwender von der Möglichkeit den rekursiven Nameserver eines Drittanbieters zu nutzen Gebrauch machten, wird das dazu nötige Wissen zunehmend verbreitet. Inzwischen raten auch Tageszeitungen wie die *Washington Post* und die *New York Times* sowie Zeitschriften wie die *c't*, die sich an eine breite Öffentlichkeit richten, dazu, bei Performance-Problemen auf einen Drittanbieter auszuweichen [Peg10; Pog10; Kal13].

Dieser Abschnitt ist wie folgt aufgebaut: Zunächst werden in Abschnitt 2.8.4.1 die Ursachen die zunehmende Inanspruchnahme der Dienste von Drittanbietern identifiziert. Im Anschluss daran werden in Abschnitt 2.8.4.2 die wichtigsten Anbieter vorgestellt. Abschließend werden in Abschnitt 2.8.4.3 die Konsequenzen beschrieben, welche sich aus der Inanspruchnahme eines Drittanbieters ergeben.

2.8.4.1 Motivation für den Wechsel des rekursiven Nameservers

Eine häufige Ursache für den Wechsel zu einem Drittanbieter ist Unzufriedenheit mit dem rekursiven Nameserver des eigenen Internetzugangsanbieters. Diese Unzufriedenheit erwächst zum einen aus einer **geringen Zuverlässigkeit der Namensauflösung**. Sind die rekursiven Nameserver eines Providers durch eine netzseitige Störung nicht verfügbar, führt dies für unbedarfte Nutzer auch zum Ausfall ihres Internetzugangs. Das Ausmaß von DNS-bezogenen Störungen im deutschen Markt lässt sich anhand der Datenbank des „iMonitor“-Dienstes von *heise.de* abschätzen. Bei iMonitor handelt es sich um ein Webangebot des Heise-Verlages, auf dem Nutzer Störungen ihres Internetzugangs melden können.²⁴ Für jede Störungsmeldung werden Zeitraum, das betroffene Gebiet (anhand der Vorwahl), der Internetzugangsanbieter sowie der Grund der Störung erfasst. Weiterhin gibt es bei iMonitor die Möglichkeit, statistische Auswertungen über verschiedene Zeiträume und Filterkriterien zu erstellen. In Tabelle 2.3 sind die Ergebnisse einer solchen Auswertung, für welche die Daten aus dem Zeitraum 01.08.2003 bis 31.07.2012 herangezogen wurden, dargestellt. Jede Spalte enthält die Anzahl der Störungsmeldungen, die zwischen dem 1. August und 31. Juli des Folgejahres erfasst wurden, sowie den Anteil der DNS-bezogenen Störungen. Bis zum Jahr 2008 betrug der Anteil der DNS-bezogenen Störungen stets über 10 %; in den letzten Jahren ist er auf unter 5 % gefallen. Die Ursache für den Rückgang lässt sich auf Basis der vorhandenen Daten nicht ermitteln. Auffällig ist jedoch, dass der Rückgang mit der Ankündigung des rekursiven Nameservers von Google zusammenfällt, über den „heise online“ am 4. Dezember 2009 erstmals berichtet hat [Kir09]. Möglicherweise sind die von DNS-Störungen betroffenen Nutzer zu Googles Nameserver gewechselt.

Eine weitere Ursache für die Unzufriedenheit mit dem rekursiven Nameserver des eigenen Internetzugangsanbieters ist eine **geringe Performanz**, wodurch es zu langen Wartezeiten bei der Namensauflösung und in der Folge zu einem langsamen Aufbau von Webseiten

²⁴Homepage: <http://www.heise.de/netze/netzwerk-tools/imonitor-internet-stoerungen/>

Tabelle 2.3: Anteil der jährlichen DNS-Störungsmeldungen im heise iMonitor

Jahr	2003	2004	2005	2006	2007	2008	2009	2010	2011
alle	3970	4372	5731	4511	3487	2929	5899	5320	5144
DNS	639	604	588	624	545	326	315	158	199
Anteil [%]	16,1	13,8	10,3	13,8	15,6	11,1	5,3	3,0	3,9

kommt. Auf Basis von 130 000 Sitzungen, die im Netalyzr-Projekt in den Jahren 2009 und 2010 erhoben wurden, stellten Kreibich et al. fest, dass einige Nameserver einen Flaschenhals darstellen: bei 9 % der Sitzungen dauerte die Namensauflösung über 300 ms länger als die bloße Paketumlaufzeit (Round-Trip-Time, abgekürzt: RTT), bei 4,6 % sogar über 600 ms länger [Kre+10]. Kreibich et al. führen dies auf eine unzureichende Dimensionierung der rekursiven Nameserver zurück. Als weitere Ursache identifizieren sie bei einem Teil der Sitzungen eine hohe Distanz zwischen Client und rekursivem Nameserver: Die Auflösung von Domainnamen, deren Eintrag bereits im Cache enthalten war, dauerte bei 11 % der Sitzungen länger als 200 ms.

Neben einer hohen Performanz und Zuverlässigkeit der Namensauflösung sind auch **Korrektheit bzw. Unverfälschtheit der Antworten** entscheidend. Weaver et al. konnten nachweisen, dass die rekursiven Nameserver einiger Internetzugangsanbieter hier erhebliche Defizite aufweisen: Neben den in Abschnitt 2.8.3 beschriebenen NXDOMAIN-Manipulationen werden mitunter auch die Adressen von existierenden Domains wie *www.google.com* oder *www.bing.com* manipuliert, um die Kunden zu werbefinanzierten eigenen Varianten dieser Seiten umzuleiten [WKP11]. Abgesehen von diesen wirtschaftlich motivierten Manipulationen durch die Anbieter selbst dürfen einige Anbieter mitunter wegen der in Abschnitt 2.8.2 erwähnten politischen Einflussnahme („DNS-Sperren“) nicht für alle Domainnamen korrekte Antworten liefern. Beschränken sich entsprechende Eingriffe lediglich auf die rekursiven Nameserver der ortsansässigen Internetzugangsanbieter, können die Nutzer durch einen Wechsel zu einem Drittanbieter diese Einschränkungen umgehen (vgl. [MA08, S. 67]).

2.8.4.2 Angebote und Verbreitung

Im Folgenden werden zunächst die größten Anbieter rekursiver Nameserver vorgestellt, OpenDNS und Google. Im Anschluss wird auf weniger bekannte Angebote eingegangen.

OpenDNS Die Firma OpenDNS wurde im Jahr 2006 von David Ulevitch gegründet [Ope].²⁵ Neben einem kostenlosen Angebot, das sich an Privatkunden richtet, werden kommerziell ausgerichtete Lösungen für Unternehmen, öffentliche Einrichtungen und Schulen

²⁵Homepage: <http://www.opendns.com/>

offert. Die Nutzung der rekursiven Nameserver von OpenDNS ist ohne vorherige Anmeldung möglich. Weitere Zusatzdienste werden teilweise gegen Entgelt angeboten. Als Vorzüge seiner rekursiven Nameserver stellt OpenDNS auf seiner Webseite u. a. folgende Aspekte heraus:

Schutz vor Infektionen mit Schadsoftware Durch die Analyse des Anfrageverhaltens aller OpenDNS-Nutzer werden Zugriffe auf die Kontrollserver von Botnetzen erkannt und verhindert, um eine Ausbreitung einzudämmen. Als Beispiel ist die Flashback-Schadsoftware zu nennen, welche sich im April 2012 auf zahlreichen Apple-Rechnern verbreitet hat [Rho12].

Filterung von Web-Inhalten Anhand einer von OpenDNS gepflegten Filter-Liste mit verschiedenen Kategorien können z. B. DNS-Anfragen, welche zu Webseiten mit jugendgefährdenden Inhalten führen, auf eine Fehlerseite umgeleitet werden.

Schutz vor DNS-Rebinding-Angriffen Durch Analyse aufeinanderfolgender Anfragen versuchen die OpenDNS-Nameserver DNS-Rebinding-Angriffe zu verhindern.

Hohe Performanz Durch die Abwicklung einer großen Menge von Anfragen liegen zu jedem Zeitpunkt die Resource-Records einer Vielzahl von Domainnamen im Cache der rekursiven Nameserver vor, wodurch kurze Antwortzeiten möglich sind.

Die rekursiven Nameserver sind durch die Verwendung der IP-Anycast-Technik unter den weltweit einheitlichen Adressen 208.67.222.222 und 208.67.220.220 ansprechbar. Huang et al. zufolge unterhielt OpenDNS im Jahr 2011 Nameserver an acht verschiedenen Standorten in Nordamerika bzw. an zwei Standorten in Europa [Hua+11]. Als nachteilig wird hingegen die Tatsache eingeschätzt, dass die Nameserver von OpenDNS NXDOMAIN-Antworten (s. Abschnitt 2.8.3) manipulieren [WKPI1].

Google Public DNS Im Dezember 2009 kündigte Google seinen Dienst „Public DNS“ an [Ram09].²⁶ Dabei handelt es sich um allgemein verfügbare rekursive Nameserver, welche unter den Adressen 8.8.8.8 und 8.8.4.4 erreichbar sind. Auch Google setzt bei seinen rekursiven Servern die IP-Anycast-Technik ein. Mit fünf Standorten in Nordamerika, vier in Europa, zwei in Asien sowie einem in Südamerika wiesen die rekursiven Nameserver von Google im Jahr 2011 eine höhere geographische Verbreitung auf als die Server von OpenDNS [Hua+11]. Google verfolgt eigenen Angaben zufolge drei Ziele mit dem Public-DNS-Dienst [Ram09; Goo12a]: eine höhere Geschwindigkeit, besserer Schutz vor DNS-Spoofing-Angriffen sowie wahrheitsgemäße Antworten. Im Folgenden werden die von Google identifizierten Ursachen und die implementierten Lösungsansätze kurz erläutert.

Das aus Googles Sicht wichtigste Ziel betrifft die **unzureichende Geschwindigkeit der Namensauflösung**. Die Paketumlaufzeit zwischen Client und rekursivem Nameserver ist laut Google nicht der ausschlaggebende Grund für lange Antwortzeiten bei der Namensauflösung. Stattdessen sieht Google „Cache-Misses“, also Anfragen, die der rekursive

²⁶Homepage: <https://developers.google.com/speed/public-dns/>

Nameserver nicht direkt aus seinem Cache beantworten kann, als Hauptursache für eine hohe Antwortzeiten. Neben möglichst kurzen Distanzen zwischen Nameservern und Clients sowie der ausreichenden Dimensionierung der verfügbaren Ressourcen weist Google daher besonders auf seine mehrstufige, von den Nameservern gemeinsam genutzte Cache-Hierarchie hin, um unnötige Cache-Misses zu vermeiden. Weiterhin werden populäre Namen vor Ablauf der TTL automatisch erneut aufgelöst („Prefetching“), so dass Client-Anfragen für diese Namen stets völlig verzögerungsfrei beantwortet werden können.²⁷

Google argumentiert weiterhin, dass auch das zweite Ziel – der **Schutz der Integrität der übermittelten Daten** – bei den Nameservern vieler Internetzugangsanbieter nicht gewährleistet ist: Diese weisen Google zufolge mitunter erhebliche Sicherheitsmängel auf, weshalb Außenstehende DNS-Spoofing-Angriffe (s. S. 116) durchführen könnten, solange der Sicherheitsstandard DNSSEC (s. S. 120) noch nicht flächendeckend verfügbar ist. Die Nameserver von Google implementieren nach eigenen Angaben daher zahlreiche Sicherheitsmechanismen, um solche Angriffe zu erschweren. Zu beachten ist allerdings, dass diese Mechanismen lediglich Schutz gegenüber außenstehenden Angreifern bieten, die Nutzer jedoch nicht vor Manipulationen durch Google, den Betreiber der Nameserver, schützen können.

Als dritten Vorzug führt Google an, dass der Public-DNS-Dienst stets **unmodifizierte und wahrheitsgemäße DNS-Antworten** liefert – im Gegensatz zu den rekursiven Nameservern von Internetzugangsanbietern und Anbietern wie OpenDNS.

Weitere Angebote Neben OpenDNS und Google gibt es inzwischen zahlreiche weitere Drittanbieter, die rekursive Nameserver für Privat- und Geschäftskunden anbieten. Die Angebote adressieren verschiedene Zielsetzungen, u. a. den Schutz der Privatsphäre sowie den unbeschränkten Zugang zu allen Domainnamen sowie den Schutz vor Schadsoftware und jugendgefährdenden Inhalten. Neben den kommerziellen Betreibern gibt es auch eine Reihe von Nameservern, die von gemeinnützigen Organisationen betrieben werden, etwa dem Digitalcourage e. V. (vormals: Verein zur Förderung des öffentlichen bewegten und unbewegten Datenverkehrs e. V., abgek. „FoeBuD“).

Tabelle 2.4 gibt einen Überblick über eine Auswahl von Drittanbietern.²⁸ Die ausgewählten Anbieter waren im Januar 2014 am Markt aktiv und offerierten einen öffentlichen rekursiven Nameserver. Zusätzlich ist in der Tabelle der amerikanische Internetzugangsanbieter Level 3 aufgeführt, welcher dafür bekannt ist, dass er die Nutzung seiner rekursiven Nameserver nicht auf seinen Kundenkreis beschränkt und fremde Nutzer duldet. Level 3 bewirbt seine Nameserver jedoch im Gegensatz zu den anderen Anbietern nicht als eigenständiges Produkt. Die Tabelle nennt jeweils beispielhaft die IP-Adresse eines

²⁷Die Prefetching-Funktionalität wurde lediglich in der ursprünglichen Ankündigung erwähnt [Ram09]. In der derzeit verfügbaren Erläuterung auf der Projekt-Webseite [Goo12a] findet sich hingegen kein Hinweis mehr darauf (Stand: Januar 2014).

²⁸Eine aktuelle Aufstellung öffentlich verfügbarer Nameserver findet sich etwa unter <http://public-dns.tk/>.

Tabelle 2.4: Drittanbieter für rekursive Nameserver

Anbieter und Homepage		Nameserver	Z	S	F	PP	AC
1	Digitalcourage	85.214.20.141	ja				
2	German Privacy Foundation	87.118.100.175	ja				
3	Censurfridns	89.233.43.71	ja				
4	OpenDNS	208.67.222.222		ja	ja	ja	ja
5	Google Public DNS	8.8.8.8		ja		ja	ja
6	Symantec Norton DNS	199.85.126.10		ja	ja	ja	ja
7	Comodo Secure DNS	8.26.56.26		ja			ja
8	Neustar DNS Advantage	156.154.70.1		ja			ja
9	Securly	184.169.143.224			ja		
10	Level 3	4.2.2.2					ja

Beworbene Eigenschaften: [Z] keine Implementierung von Zensurmechanismen; [S] Schutz vor Schadsoftware; [F] Möglichkeit zur Filterung von Seiten mit unerwünschten Inhalten; **weitere Eigenschaften:** [PP] öffentlich verfügbare Privacy-Policy für Nameserver; [AC] Nameserver auf mehrere Standorte verteilt (IP-Anycast-Technik)

Internetauftritte der Anbieter: ¹<http://digitalcourage.de>, ²<http://server.privacyfoundation.de>, ³<http://blog.censurfridns.dk>, ⁴<http://opendns.com>, ⁵<https://developers.google.com/speed/public-dns>, ⁶<https://dns.norton.com>, ⁷<http://comodo.com>, ⁸<http://dnsadvantage.com>, ⁹<http://securly.com>, ¹⁰<http://level3.com>

rekursiven Nameservers pro Anbieter. Weiterhin sind die auf den jeweiligen Internetseiten der Anbieter beworbenen Vorzüge dargestellt: Die Mehrzahl der Anbieter gibt an, ihre Kunden durch die Hinterlegung von entsprechenden Domain-Listen vor dem Besuch von Webseiten, die Schadsoftware verbreiten, zu schützen. Einige Betreiber bieten darüber hinausgehende Filter-Möglichkeiten an. Die nichtkommerziellen Dienste versprechen hingegen, sich staatlichen Zensurbestrebungen zu widersetzen und eine völlig ungefilterte Namensauflösung zu ermöglichen. Auffällig ist, dass nur vier der elf Angebote auf ihren Webseiten eine Datenschutz-Erklärung vorhalten, in der explizit darauf eingegangen wird, welche Daten auf den Nameservern erhoben werden und wie diese verarbeitet werden. Die Mehrzahl der kommerziellen Angebote betreibt ihre Nameserver mittels der IP-Anycast-Technik an mehreren Standorten.

Zunehmende Popularität Es gibt Hinweise darauf, dass inzwischen ein substanzieller Anteil der Nutzer die rekursiven Nameserver von Drittanbietern verwendet. Nach eigenen Angaben stieg die Anzahl der DNS-Anfragen beim Dienst OpenDNS von durchschnittlich drei Milliarden pro Tag im Jahr 2007 auf 30 Milliarden pro Tag im Jahr 2010 [Ope07; Owe11]. Während das Unternehmen im Jahr 2010 angab, dass etwa 1 % aller Internetnutzer den Dienst nutzten [Owe11], wurde diese Schätzung im Jahr 2012 bereits auf 2 % angehoben. Googles Public-DNS-Dienst, der erst seit 2009 angeboten wird, beantwortete nach eigenen

Angaben im Februar 2012 bereits durchschnittlich 70 Milliarden Requests pro Tag [Che12]. Die übrigen Drittanbieter machen keine Angaben zum Anfragevolumen.

Unabhängig von den eigenen Angaben der Anbieter ist die Nutzung von Drittanbieter-Nameservern auch in wissenschaftlichen Untersuchungen feststellbar. Huang et al. veröffentlichten im Jahr 2011 die Ergebnisse einer Studie von Microsoft Research, in der sie bei einem Teil der Besucher einer populären Webseite den verwendeten Nameserver ermittelten [Hua+11]. Der Datensatz umfasst einen Zeitraum von acht Wochen und enthält Zugriffe von etwa fünf Millionen unterschiedlichen IP-Adressen. Etwa 2,5 % der Internetnutzer verwendeten einen rekursiven Nameserver eines Drittanbieters. Level 3 und OpenDNS, auf die jeweils 1 % der Anfragen entfielen, waren dabei deutlich populärer als Googles Public-DNS-Dienst, der zum Zeitpunkt der Untersuchung erst seit sechs Monaten am Markt war. Zu vergleichbaren Ergebnissen kommt die im Jahr 2012 veröffentlichte Studie von Gehlen et al., in welcher der Datenverkehr von etwa 30 000 Kunden eines italienischen Internetzugangsanbieters analysiert wurde [Geh+12]. In diesem Datensatz nutzten mehr als 3 % der Kunden den Nameserver eines Drittanbieters, wobei auf OpenDNS etwa 1 % zurückgriffen und auf Google etwa 0,5 %. Die Nameserver-Drittanbieter werden demnach auch von europäischen Nutzern verwendet. In der Studie von Callahan et al. [CAR13] wurden 3 % der beobachteten Anfragen zu Nameservern von Drittanbietern gesendet, wobei 1 % auf die Nameserver von Google entfiel.

Es ist zu erwarten, dass der Anteil der Nutzer, die zu einem rekursiven Nameserver eines Drittanbieters wechseln, weiter steigt. Diese Entwicklung wird zum einen durch die allgemein steigende Sensibilisierung von Nutzern bzw. Unternehmen für Schadsoftware und Phishing begünstigt. OpenDNS bezeichnet sich inzwischen selbst als „Cloud-Internet-Security“-Dienst [Ope13]. Die Einstufung als Cloud-Dienst ist durchaus gerechtfertigt, da die kommerziell betriebenen rekursiven Nameserver eine gute Skalierbarkeit und Ressourcen-Elastizität aufweise – zwei Kriterien, die für den allgemeinen Erfolg von Cloud-Computing-Diensten verantwortlich sind (vgl. die Überblicksartikel von Armbrust et al. [Arm+09; Arm+10]). Auch die einfache Einrichtung, der Verzicht auf Hard- oder Software vor Ort beim Kunden sowie die gut aufbereiteten Auswertungen und Anpassungsmöglichkeiten erleichtern den Wechsel. Ob die in den Nameservern hinterlegten Filterlisten einen tatsächlichen Sicherheitsgewinn bringen, ist für die meisten Nutzer allerdings nur schwer überprüfbar. Sie sind auf die Einschätzung von Sicherheitsexperten angewiesen, etwa dem Bundesamt für Sicherheit in der Informationstechnik (BSI), das in seinen „Empfehlungen zur Cybersicherheit“ explizit die Verwendung der Nameserver von OpenDNS [Bun12b; Bun12a] empfiehlt, um Windows-PCs abzusichern. Dass das BSI dabei nicht auf die Risiken hinweist, ist allerdings problematisch.

2.8.4.3 Konsequenzen der Nutzung von Drittanbietern

Der Wechsel zu einem rekursiven Nameserver eines Drittanbieters hat weitreichende Konsequenzen für den Nutzer. Im Folgenden wird dies anhand der vier Aspekte Zuverlässigkeit, Performanz, Sicherheit und Beobachtbarkeit herausgearbeitet.

Zuverlässigkeit Da es sich um ihr Kerngeschäft handelt, haben die Drittanbieter ein großes Interesse an einer hohen Zuverlässigkeit ihrer rekursiven Nameserver. Um Ausfälle zu vermeiden, setzen viele Anbieter Lastverteilungssysteme ein, welche die eingehenden Anfragen auf mehrere Server aufteilen. Die Anbieter können dadurch flexibel auf Lastspitzen reagieren, und Hardware-Ausfälle wirken sich nicht mehr unmittelbar auf die Verfügbarkeit des Dienstes aus. Die höhere Zuverlässigkeit ist auch in der Praxis messbar: Kreibich et al. stellten bei der Untersuchung des Netalyzr-Datensatzes fest, dass die Fehlerraten bei den OpenDNS-Sitzungen deutlich niedriger waren als im Mittel [Kre+10]. Während die Domain *www.microsoft.com* im Mittel in 0,8 % der Sitzungen nicht aufgelöst werden konnte, trat nur bei 0,4 % der OpenDNS-Sitzungen ein Fehler auf.

Performanz der Namensauflösung Überraschender ist hingegen die Tatsache, dass die Drittanbieter hinsichtlich der Performanz der Namensauflösung nicht eindeutig besser abschneiden als die Nameserver der Internetzugangsanbieter. Zu diesem Ergebnis kommt die o. g. Netalyzr-Studie: Während die minimale Antwortzeit bei den Nameservern der Internetzugangsanbieter nur bei 9 % der Sitzungen über 200 ms lag, wurde dieser Wert bei 16 % der OpenDNS-Sitzungen überschritten [Kre+10]. Zu einem vergleichbaren Ergebnis kommt die Studie von Ager et al., in der 50 Internetzugangsanbieter betrachtet wurden [Age+10]. Bei jedem der Anbieter wurde über den Internetzugang eines Kunden die Paketumlaufzeit zum Nameserver des Internetzugangsanbieters gemessen und mit den Umlaufzeiten zu den Servern der Drittanbieter Google und OpenDNS verglichen. Bei 21 Anbietern war die Paketumlaufzeit zu den Servern der Drittanbieter um mindestens 25 ms langsamer. Diese Ergebnisse weisen darauf hin, dass die Nameserver der Drittanbieter deutlich weiter von den Clients entfernt sind als die Nameserver der Internetzugangsanbieter. Diese Vermutung wird u. a. in [Hua+11; KFH12] empirisch bestätigt. Da CDNs zur Ermittlung des nächstgelegenen Replika-Servers den rekursiven Nameserver als Stellvertreter für den Client verwenden (vgl. Abschnitt 2.8.1), wirkt sich die Verwendung eines Drittanbieters bei solchen Angeboten sogar in doppelter Hinsicht negativ aus: Zum einen ist die Namensauflösung selbst langsamer, zum anderen ist die Antwortzeit des ausgewählten Webservers höher.

Sicherheit Weiterhin stellt sich die Frage, inwiefern die Nameserver der Drittanbieter ein höheres Sicherheitsniveau bieten können als die Nameserver der Internetzugangsanbieter. In diesem Zusammenhang geht es weniger um anbieterspezifische Funktionen, mit denen potenziell gefährliche Domainnamen gesperrt werden, sondern um den **Schutz der Integrität der DNS-Antworten** (s. Definition 3.2 in Abschnitt 3.6). Wegen der noch geringen Verbreitung von DNSSEC müssen die rekursiven Nameserver eine Reihe von Sicherheitsmechanismen implementieren, um sicherzustellen, dass Außenstehende die Antworten nicht manipulieren können.

Von den Drittanbietern nennt lediglich Google in der Dokumentation seines Public-DNS-Dienstes unter der Überschrift „Security Benefits“ [Goo12b] die implementierten Mechanismen zum Schutz gegen Cache-Poisoning-Angriffe (s. Abschnitt 3.7.3.2):

- Die rekursiven Nameserver führen eine grundlegende Validitätsprüfung der von den autoritativen Nameservern erhaltenen DNS-Antworten durch.
- Die rekursiven Nameserver erhöhen in den Anfragen an die autoritativen Nameserver die Entropie, etwa durch eine zufällige Wahl des UDP-Quell-Ports, die zufällige Variation des kontaktierten autoritativen Nameservers sowie durch die Verwendung des sog. 0x20-Encodings, bei dem Domainnamen in den DNS-Anfragen in zufälliger Groß-/Kleinschreibung übertragen werden.
- Die rekursiven Nameserver entfernen Duplikate aus der Anfrage-Warteschlange, um Geburtstagsangriffe zu verhindern.

In Abschnitt 3.6.3 wird auf diese Techniken zum Schutz der Integrität noch näher eingegangen. Die Ergebnisse der Netalyzr-Studie belegen, dass auch die Internetzugangsanbieter damit begonnen haben, Sicherheitsmechanismen zu implementieren [Wea+11]: Bei allen größeren Internetzugangsanbietern verwendeten die rekursiven Nameserver zumindest zufällige UDP-Quell-Ports.

Beobachtbarkeit Schließlich ist festzustellen, dass die Verwendung des rekursiven Nameservers eines Drittanbieters Auswirkungen auf die Beobachtbarkeit der Aktivitäten von Internetnutzern hat. Der Drittanbieter erfährt bei der Erbringung seiner Dienstleistung zwangsläufig die von den Nutzern angefragten Domainnamen, aus denen sich Interessen und Verhaltensweisen ableiten lassen. Natürlich sieht auch der Nameserver des eigenen Internetzugangsanbieters die aufgelösten Namen – allerdings ist dies im direkten Vergleich als weniger problematisch einzustufen, wie die folgenden Überlegungen zeigen.

Zum einen kann der Internetzugangsanbieter diejenigen Informationen, welche ein rekursiver Nameserver eines Drittanbieters sammeln kann, d. h. welche Domainnamen von einem Kunden zu welchem Zeitpunkt aufgelöst wurden, auch auf andere Weise ermitteln, etwa durch die **Analyse der übermittelten UDP-Pakete**.

Darüber hinaus hat der Internetzugangsanbieter noch **weitreichendere Möglichkeiten**, um die Aktivitäten seiner Kunden zu beobachten, da er Zugriff auf den gesamten Datenverkehr hat. Statt lediglich die DNS-Nachrichten heranzuziehen, kann er z. B. auch die Empfänger-IP-Adressen der TCP-Verbindungen auswerten, an die der Nutzer beim Besuch einer Webseite seine HTTP-Anfragen sendet. Durch Verwendung eines rekursiven Nameservers eines Drittanbieters kann das beim Internetzugangsanbieter vorhandene Beobachtungspotential daher nicht reduziert werden. Vielmehr entstehen durch den Wechsel zusätzliche Beobachtungsmöglichkeiten beim Drittanbieter.

Schließlich besteht mit dem Internetzugangsprovider üblicherweise ein **Vertragsverhältnis**, in dem geregelt ist, welche Informationen erhoben und gespeichert werden. Ein belastbares Vertragsverhältnis besteht bei Drittanbietern, welche die Namensauflösung mitunter kostenlos und ohne Anmeldung erbringen, hingegen häufig nicht. Weiterhin unterliegen die in Deutschland am Markt tätigen Internetzugangsanbieter – im Gegensatz zu den teilweise im Ausland ansässigen Drittanbietern – gesetzlichen Regelungen

wie dem Telemediengesetz und Telekommunikationsgesetz. Darin werden u. a. explizite Anforderungen an den Datenschutz formuliert.

2.8.5 Auswertung des DNS-Datenverkehrs

Das Interesse an der Analyse des DNS-Datenverkehrs ist in den letzten Jahren gestiegen. In diesem Abschnitt wird aufgezeigt, dass Aufzeichnung, Auswertung und Analyse inzwischen mit geringem Aufwand möglich sind.

Query-Log-Protokollierung Die gängigen Nameserver-Implementierungen bieten die Möglichkeit, die eingehenden DNS-Anfragen zu protokollieren und für spätere Auswertungen in einer Datei abzuspeichern. Bei BIND wird diese Funktion als „querylog“ bezeichnet [Int13b, S. 169]. Wird diese Funktion auf einem rekursiven Nameserver aktiviert, werden Zeitstempel, IP-Adresse des anfragenden DNS-Clients, die angefragte Domain und der angeforderte RR-Typ erfasst. Das Aktivieren der Anfrageprotokollierung führt in der Praxis zu einer erheblichen Mehrbelastung für den Nameserver. Diese Form der Aufzeichnung des DNS-Datenverkehrs ist daher nur eingeschränkt praxistauglich.

Passive-DNS-Replication Die Passive-DNS-Replication-Technik, die von Weimer vorgestellt wurde [Wei05], war ursprünglich nicht zur Beobachtung des Nutzerverhaltens vorgesehen, sie stellt jedoch einen wesentlichen Entwicklungsschritt dar, der die Beobachtung von Internetnutzern zukünftig erleichtern könnte.

Die Problemstellung, welche die Entwicklung der Passive-DNS-Technik motivierte, besteht darin, dass die auf den autoritativen Nameservern hinterlegten Daten flüchtig sind und nur so lange abgerufen werden können, wie der autoritative Nameserver sie vorhält. Dieser Umstand wird von sog. Botnetzen ausgenutzt, bei denen Schadprogramme durch einen sog. „Command & Control Server“ (abgek. „C&C-Server“) ferngesteuert werden. Die zum Auffinden des C&C-Servers verwendeten Domainnamen existieren allerdings nur sehr kurze Zeit (sog. „Fast-Flux-Netze“ [Hol+08]). Da die C&C-Server bei einer späteren forensischen Analyse des Schadprogramms nicht mehr auffindbar sind, ist die Verfolgung der Betreiber eines Botnetzes häufig unmöglich. Eine fortlaufende Abfrage der autoritativen Nameserver („polling“) ist nicht praktikabel, da die autoritativen Nameserver üblicherweise keine Zonentransfers zulassen (s. Abschnitt 2.6.4) und dementsprechend alle abzufragenden Domainnamen bekannt sein müssten. Dies ist bei Fast-Flux-Netzen jedoch gerade nicht der Fall. Zudem würde die kontinuierliche Abfrage einer großen Menge von Domainnamen ein erhebliches Datenvolumen erzeugen.

Die Idee der Passive-DNS-Replication-Technik besteht darin, anstelle des Pollings den ohnehin anfallenden DNS-Datenverkehr zu analysieren und dadurch eine Replika der verteilten DNS-Datenbank anzufertigen. Dazu betreibt eine Organisation, etwa ein Unternehmen oder ein Internetzugangsanbieter, passive Sensoren, welche die DNS-Anfragen

und -Antworten protokollieren und in komprimierter Form in einer proprietären Datenstruktur abspeichern (s. u. a. [ME12]). Dadurch lassen sich „Snapshots“ der DNS-Datenbank anfertigen, die alle Daten enthalten, die zu einem früheren Zeitpunkt auf den autoritativen Nameservern hinterlegt waren bzw. von diesen abgerufen worden waren. Weiterhin lässt sich für einen gegebenen Domainnamen nachvollziehen, wann dieser erstmals angefragt wurde, wann sich die hinterlegten RRs geändert haben und über welchen Zeitraum sich die Anfragen verteilt haben. Durch statistische Auswertung der Passive-DNS-Datenbank lassen sich Anomalien aufdecken [ZBW07; Der+12] und Fast-Flux-Netze entdecken [Kan11; PCG12]. Die Passive-DNS-Replication-Technik wird inzwischen an zahlreichen Standorten eingesetzt. Für die Protokollierung und effiziente Abspeicherung existieren stabile Sensor-Implementierungen für alle gängigen Betriebsumgebungen. Im Februar 2014 existieren zumindest zwei öffentlich abfragbare Datenbanken: http://www.bfk.de/bfk_dnslogger_de.html und <https://www.dnsdb.info>.

Konsequenzen für die Beobachtbarkeit von Nutzern Ursprünglich war bei der Passive-DNS-Replication-Technik vorgesehen, die DNS-Anfragen zwischen rekursiven und autoritativen Nameservern zu protokollieren, da für die forensische Analyse von Schadsoftware lediglich die Domainnamen und die hinterlegten RRs benötigt wurden. Weimer weist explizit darauf hin, dass die Privatsphäre der Nutzer durch die Speicherung und Analyse der erhobenen Daten nicht beeinträchtigt werde [Wei05]. Die von ihm entwickelte Software *dnslogger* entfernt dazu die Sender-IP-Adresse vor der Abspeicherung.

Allerdings kann eine Organisation ihre Passive-DNS-Replication-Infrastruktur grundsätzlich auch dazu einsetzen, um die DNS-Anfragen auf einem Verbindungsabschnitt zu protokollieren, der zwischen den Nutzern und dem rekursiven Nameserver liegt. Durch geringfügige Anpassungen an der Sensor-Implementierung kann neben den übrigen Daten auch die IP-Adresse der Anfrager erfasst werden. Es gibt bereits erste Vorschläge, von dieser Möglichkeit Gebrauch zu machen: So stellen Ishibashi et al. ein System vor, mit dem ein Internetzugangsanbieter durch passive Analyse der DNS-Anfragen die Kunden identifizieren kann, deren Endgeräte durch Schadsoftware befallen sind [Ish+05]. An der Universität Amsterdam wurde bereits zuvor ein Intrusion-Detection-System entwickelt, das die DNS-Anfragen von Nutzern und die IP-Adressen der Anfragenden heranzieht [SH06].

Die beschriebenen Entwicklungen deuten darauf hin, dass die Protokollierung und Auswertung weiter zunehmen werden. Durch die Verfügbarkeit effizienter Aufzeichnungs-, Speicher- und Auswertungstechniken, die aus der Passive-DNS-Replication-Technik hervorgegangen sind, ist der Aufwand deutlich geringer als bei der Verwendung der zuvor erläuterten Query-Log-Funktionalität.

2.8.6 Fazit

In diesem Abschnitt wurden fünf Veränderungen identifiziert, welche im Zusammenhang mit der Betrachtung der Beobachtungsmöglichkeiten und der Möglichkeiten zum Schutz vor Beobachtung von Bedeutung sind. Im einzelnen sind dies:

Die Verbreitung von **Content-Delivery-Netzen** nimmt zu. Diese implementieren Anfrage-Routing und Last-Verteilung, indem sie RRs mit kurzen TTLs nutzen. Dadurch sinkt die Zeitspanne, in der ein Nutzer wiederkehrende Anfragen für denselben Domainnamen für Außenstehende unbeobachtbar stellen kann, da der angefragte RR nur noch wenige Sekunden im Zwischenspeicher des Stub-Resolvers vorgehalten wird. Verletzungen der Integrität der Namensauflösung nehmen zu, wie die Bestrebungen zur Einführung **DNS-basierter Internetsperren** und die von einigen Anbietern praktizierte **NXDOMAIN-Manipulation** zeigen. Die o. g. bewusste Einschränkung der Dienstqualität bei den rekursiven Nameservern führt bei Nutzern zunehmend zum Wunsch, **rekursive Nameserver von Drittanbietern** zu verwenden. Dadurch entstehen zusätzliche Beobachtungsmöglichkeiten. Durch die Verfügbarkeit der **Passive-DNS-Replication-Techniken** sinkt der Aufwand für die Protokollierung, Speicherung und Analyse von DNS-Anfragen.

2.9 DNS-Anwendungen

Wie in Abschnitt 2.2 ausgeführt wurde das DNS ursprünglich als verteilter Namensdienst für das Internet entworfen, welcher die für Menschen leicht zu merkenden Domainnamen in die für das Routing von Datenpaketen benötigten IP-Adressen übersetzen sollte. Diese ursprüngliche Aufgabe wird im Folgenden als **Adressauflösung** bezeichnet, um die Abgrenzung vom bisher verwendeten generischen Begriff „Namensauflösung“ zu verdeutlichen.

Der äußerst generische Entwurf des DNS führte zur Entwicklung neuer Anwendungen, welche die geschaffenen Infrastrukturen, Schnittstellen und Protokolle für ihre eigenen Zwecke nutzen, also anstelle von IP-Adressen anwendungsspezifische Daten auf den autoritativen Nameservern hinterlegen und über einen Domainnamen abrufbar machen. Diese Anwendungen werden im Folgenden unter dem Begriff **DNS-Anwendungen** subsumiert. Die Nutzung des DNS für anwendungsspezifische Zwecke wird von der IETF toleriert und durch einen Leitfaden [Pet+12] unterstützt, in dem erörtert wird, welche Konsequenzen sich aus der Nutzung des DNS für neuartige Anwendungen ergeben und worauf Anwendungsentwickler achten müssen.

In diesem Abschnitt wird eine Auswahl der in der Praxis verbreiteten bzw. noch in der Entwicklung befindlichen DNS-Anwendungen vorgestellt. Die Auseinandersetzung mit existierenden und geplanten DNS-Anwendungen ist im Rahmen der Dissertation aus zwei Gründen erforderlich:

- Zum einen existieren bei einigen DNS-Anwendungen spezifische Beobachtungsmöglichkeiten, die sich von den Beobachtungsmöglichkeiten unterscheiden, die sich aus der Adressauflösung ergeben. Dies trifft insbesondere auf den E-Mail-Versand und -Empfang (s. Abschnitte 2.9.1 bis 2.9.3) sowie auf die ENUM- und ONS-Dienste (s. Abschnitte 2.9.4 und 2.9.5) zu.
- Zum anderen gibt es für einige DNS-Anwendungen, etwa den ENUM- und den ONS-Dienst (s. Abschnitte 2.9.4 und 2.9.5), bereits Vorschläge zum Schutz vor Beobachtung; auf diese wird in Abschnitt 5.2.2 und Abschnitt 5.2.3 eingegangen.

Die spezifischen Beobachtungsmöglichkeiten, die sich bei den einzelnen DNS-Anwendungen ergeben, werden in den jeweiligen Abschnitten erörtert. Auf die grundsätzlichen Beobachtungsmöglichkeiten, die sich durch die *Adressauflösung im Kontext von Arbeitsplatz-Rechnern beim Web-Surfen* ergeben, wird in Kapitel 4 eingegangen.

Dieser Abschnitt ist wie folgt aufgebaut: Zunächst werden in Abschnitt 2.9.1, der historischen Entwicklung folgend, die zur Zustellung von E-Mails verwendeten MX-Records sowie die generischen SRV- und TXT-Records beschrieben. In Abschnitt 2.9.2 werden DNS-Blocklisten vorgestellt, die zur Abwehr von unerwünschten Spam-Mails eingesetzt werden. Dieselbe Zielsetzung wird mit den in Abschnitt 2.9.3 behandelten Techniken SPF, SenderID und DKIM verfolgt. Anschließend werden in Abschnitt 2.9.4 der ENUM-Dienst sowie in Abschnitt 2.9.5 der ONS-Dienst vorgestellt. Abschließend folgt eine Beschreibung der Techniken DANE und CAA sowie des ICSI-Notary-Dienstes, mit denen die Gegenstellen-Authentifizierung im TLS-Protokoll verbessert werden soll (s. Abschnitt 2.9.6).

2.9.1 Generische Record-Typen

Eine der ersten Funktionen, die das DNS über die reine Adressauflösung hinaus erweitert, wird bereits in RFC 974 [Par86] definiert: Dabei handelt es sich um die Ermittlung des Mailservers, welcher für eine bestimmte Domain zuständig ist. Dessen IP-Adresse wird im **MX-Record** (abgeleitet von *Mail Exchanger*) einer Domain hinterlegt. Liefert ein Nutzer eine E-Mail bei einem Mailserver ein, stellt dieser eine MX-Anfrage an seinen rekursiven Nameserver, der diese an den autoritativen Nameserver der Domain des in der E-Mail angegebenen Empfängers weiterleitet. Ist ein MX-Record hinterlegt, enthält er den Domainnamen des Mailservers, der Mails für diese Domain entgegennimmt. Ist kein MX-Record hinterlegt, wird davon ausgegangen, dass der Mailserver die IP-Adresse hat, welche für den A-Record der Domain hinterlegt ist [Kle08b, S. 69 f.].

Aus Anfragen für MX-Records resultieren folgende **Beobachtungsmöglichkeiten**: Der vom Mailserver des Absenders verwendete **rekursive Nameserver** erfährt die Domainnamen der E-Mail-Adressen der Empfänger, die u. U. Rückschlüsse auf das Kommunikationsverhalten ermöglichen. Der **autoritative Nameserver** erfährt hingegen lediglich die IP-Adresse des rekursiven Nameservers (vgl. Abschnitt 2.6.2.3). Eine ausführlichere Diskussion der Beobachtungsmöglichkeiten der Nameserver folgt bei der Betrachtung

der *Beobachtungsmöglichkeiten beim Empfang von E-Mails* in den Abschnitten 2.9.2.4 und 2.9.3.4.

Neben dem MX-Record-Typ, der bereits zum Zeitpunkt der Standardisierung Teil des DNS war, wurden später weitere RR-Typen hinzugefügt, um Einträge mit einer Semantik zu versehen. Die Verwendung von dedizierten Record-Typen für neuartige Anwendungen entspricht der vom Internet Architecture Board präferierten Methode zur Erweiterung des DNS [Int+09]. Um unkontrollierte Änderungen am DNS-Standard zu vermeiden, muss zur Registrierung neuer Record-Typen ein Standardisierungsprozess durchlaufen werden.

Eine Generalisierung des MX-Record-Typs stellt der in RFC 2782 eingeführte **Record-Typ SRV** („Service Locator“) dar [GVE00]. Mit einem SRV-Record kann ein Zonenadministrator auf einen Host verweisen, der innerhalb einer Domain für einen bestimmten, wohldefinierten Service zuständig ist. So verweist der SRV-Record der Domain *_ldap._tcp.example.com* einen DNS-Client auf einen Host, auf dem ein TCP-fähiger LDAP-Server betrieben wird, der für die Domain *example.com* zuständig ist. Durch die Verwendung der generischen SRV-Records kann also auf die Definition protokollspezifischer Record-Typen (wie etwa MX) verzichtet werden. Am MX-Record-Typ wird aus Kompatibilitätsgründen jedoch festgehalten.

Hinsichtlich der **Beobachtungsmöglichkeiten** unterscheiden sich SRV-Records von A-Records: Während bei der normalen Adressauflösung der **rekursive Nameserver** lediglich darauf schließen kann, mit welchem Host ein Client in Verbindung treten möchte, erfährt er bei SRV-Records zusätzlich den in Anspruch genommenen Dienst. In Bezug auf die **autoritativen Nameserver** gilt das bereits Gesagte (vgl. Abschnitt 2.6.2.3).

In der Vergangenheit haben neuartige Anwendungen stattdessen häufig auf den **TXT-Record-Typ** zurückgegriffen, der bereits in RFC 1035 aufgeführt wird [Moc87b, S. 20]. Dieser Record-Typ war ursprünglich zur Speicherung von Zeichenfolgen vorgesehen, die Anwendern zusätzliche Hinweise zu einem Domainnamen geben sollten. TXT-Records lassen sich jedoch auch dazu verwenden, um beliebige Daten im DNS abzulegen. Diese Zweckentfremdung wird in RFC 1464 legitimiert [Ros93]. Die Daten müssen dazu nicht unbedingt im ASCII-Format vorliegen, da es keine Vorgaben bezüglich des Zeichensatzes für Labels („any binary string whatever can be used as the label of any resource record“ [EB97, S. 13 f.]) bzw. für die im TXT-RR hinterlegten Werte („<character-string> is a single length octet followed by that number of characters. <character-string> is treated as binary information, and can be up to 256 characters in length (including the length octet)“ [Moc87b, S. 13,20]) gibt. In der Praxis werden Binärdaten mitunter jedoch, etwa durch Base64-Kodierung, in den ASCII-Zeichenraum überführt, bevor sie im DNS als Label oder Wert verwendet werden.

Über die **Beobachtungsmöglichkeiten**, die sich bei TXT-Anfragen ergeben, lassen sich keine allgemeinen Aussagen treffen, da sie von den verwendeten Domainnamen und den Inhalten abhängen, die eine Anwendung dort hinterlegt.

2.9.2 DNS-Blocklisten

DNS-Blocklisten werden dazu verwendet, um Informationen über die Reputation von IP-Adressen oder Domains zu hinterlegen und über das DNS zur Verfügung zu stellen. Die Blocklisten werden von Systemadministratoren dazu verwendet, Datenverkehr und Nachrichten von Systemen, die in der Vergangenheit negativ aufgefallen sind, abzulehnen.

Die Entwicklung der DNS-Blocklisten geht auf Dave Rand und Paul Vixie zurück, die ab 1997 eine Liste von IP-Adressen führten, von denen unerwünschter Datenverkehr ausging bzw. die für den Versand von unerwünschten E-Mails (Spam) missbraucht wurden [Lev10, S. 1]. Da Vixie an der Entwicklung des DNS beteiligt war, lag es für ihn nahe, die Liste über das DNS der Öffentlichkeit zur Verfügung zu stellen. Diese erste DNS-Blockliste wurde als „Real-time Blackhole List“ bzw. RBL bezeichnet. Da die Abkürzung „RBL“ inzwischen ein Markenname der Firma Trend Micro ist [Lev10, S. 1], wird inzwischen üblicherweise die Abkürzung DNSBL verwendet.

Dieser Abschnitt ist wie folgt aufgebaut: Zunächst werden zwei gängige Varianten von Blocklisten beschrieben: Reputationsbasierte DNSBLs (s. Abschnitt 2.9.2.1) enthalten Angaben bezüglich der Vertrauenswürdigkeit von IP-Adressen, wohingegen inhaltsbasierte DNSBLs (s. Abschnitt 2.9.2.2) Aussagen über den Inhalt von E-Mails machen. In Abschnitt 2.9.2.3 wird auf die Schwächen der Verfahren eingegangen. Abschließend werden in Abschnitt 2.9.2.4 die Beobachtungsmöglichkeiten betrachtet, die sich aus der Nutzung von DNSBLs ergeben.

2.9.2.1 Reputationsbasierte DNS-Blocklisten

Es gibt zahlreiche Anbieter von reputationsbasierten DNSBLs, die sich hinsichtlich Ausrichtung (kostenlose oder kommerzielle Dienstleistung) und Aufnahmekriterien unterscheiden. Typische Aufnahmekriterien sind [Coo+06]:

- In der Vergangenheit wurden von einer IP-Adresse aus Spam-Nachrichten versendet. Einige Anbieter betreiben dedizierte, ungenutzte E-Mail-Server (sog. „spam traps“ oder „honeypots“), um eingehende Spam-Nachrichten zu analysieren; andere Anbieter bieten ihren Nutzern die Möglichkeit, auffällige Adressen zu melden bzw. selbst in die Liste einzutragen.
- Unter einer IP-Adresse ist ein Mailserver erreichbar, welcher den Versand von E-Mails ohne Authentifizierung ermöglicht (sog. „open relay“ [PHM08])
- Eine IP-Adresse gehört zu einem Adressbereich, welcher für die Einwahl von privaten Endkunden verwendet wird („dial-up addresses“ oder „dynamic addresses“). Dieses Aufnahmekriterium ist dadurch motiviert, dass reguläre Mailserver nicht von den Nutzern zu Hause betrieben werden, sondern von Internetzugangsanbietern oder dedizierten E-Mail-Anbietern. Eine Sperre solcher Adressbereiche verhindert die Annahme von Spam-Nachrichten, welche über Botnetze von den Endgeräten der Endkunden versendet werden.

Die konkreten Prozesse zur Aufnahme von IP-Adressen in die Blocklisten bzw. deren Entfernung sind anbieterspezifisch. RFC 6471 gibt diesbezüglich einige organisatorische Empfehlungen [LS12]. Die technische Umsetzung ist hingegen weitgehend einheitlich gestaltet. Sie orientiert sich an RFC 5782 [Lev10], in dem der Aufbau der Einträge und deren Semantik beschrieben werden. Für eine DNSBL wird demnach eine dedizierte Zone eingerichtet, zum Beispiel *sbl.spamhaus.org*. Für jede in der Blockliste enthaltene IP-Adresse existiert ein A-Record. Die Verwendung von DNSBLs wird an einem Beispiel deutlich.

Beispiel 2.6. Bei einer eingehenden Verbindung durchläuft ein Mailserver folgende Schritte (vgl. [Ait11, S. 201] und [Sch07b, S. 63]):

1. Der Mailserver ermittelt die IP-Adresse, von der aus die SMTP-Verbindung aufgebaut wurde, zum Beispiel: 13.0.1.77.
2. Anhand der IP-Adresse muss nun der Domainname in der DNSBL gebildet werden. Die Oktette der Adresse werden dazu wie bei der Rückwärtsauflösung von IP-Adressen in Domainnamen (s. Abschnitt 2.5) in umgekehrter Reihenfolge angeordnet. Im Beispiel ergibt sich so der Name *77.1.0.13.sbl.spamhaus.org*.
3. Der Mailserver stellt eine A-Anfrage für diesen Domainnamen an seinen rekursiven Nameserver. Dieser ermittelt die Antwort beim zugehörigen autoritativen Nameserver des DNSBL-Anbieters (oder entnimmt sie ggf. seinem Cache).
4. Der Mailserver nimmt die Antwort des rekursiven Nameservers entgegen. Existiert ein Eintrag für die IP-Adresse in der DNSBL, erhält er einen A-Record mit einer IP-Adresse aus dem Bereich von 127.0.0.1 bis 127.0.0.255. Üblicherweise wird dabei die Adresse 127.0.0.2 verwendet. Mit dem letzten Oktett der IP-Adresse können die DNSBL-Anbieter den Grund für die Eintragung kodieren. Ausführlichere Informationen werden mitunter in einem zusätzlichen TXT-Record hinterlegt. Wird hingegen ein NXDOMAIN-Fehler (s. Abschnitt 2.6.3) zurückgeliefert, ist die angefragte Adresse nicht in der DNSBL enthalten.
5. In Abhängigkeit von der Antwort weist der Mailserver den Verbindungswunsch ab oder fährt mit dem Empfang der E-Mail fort. □

Neben DNS-Blacklists, welche IP-Adressen mit schlechter Reputation enthalten, bieten einige DNSBL-Anbieter auch **DNS-Whitelists** an, welche die IP-Adressen von legitimen E-Mail-Servern enthalten. Da eine vollständige Datenbank aller legitimen E-Mail-Server im Internet kaum zu pflegen ist, beschränken sich DNS-Whitelists üblicherweise auf die Mailserver von großen E-Mail-Anbietern. Sie werden in Kombination mit einer DNS-Blacklist eingesetzt und sollen verhindern, dass durch einen fehlerhaften Eintrag in einer Blacklist versehentlich große Mengen von legitimen Nachrichten abgelehnt werden (sog. falsch-positive Fehler, s. Abschnitt 2.9.2.3).

Für den Betreiber eines Mailservers haben DNSBLs zwei Vorteile [Coo+06]: eine potenziell **hohe Erkennungsrate** bei gleichzeitig **geringem Ressourcenverbrauch**. Ein einzelner

Mailserver, der nur die bei ihm eingehenden Nachrichten sieht, kann mitunter nicht erkennen, dass von einer IP-Adresse gerade eine große Anzahl von Nachrichten an eine Vielzahl von Empfängern verschickt wird. Erst durch den Zugriff auf die DNSBL, die durch Aggregation vieler Einzelereignisse eine zentralisierte Datenbasis bereitstellt, kann er dieses Fehlverhalten erkennen und entsprechende E-Mails ablehnen. Die zentralisierte Bereitstellung reduziert zudem den Ressourcenverbrauch der Spam-Erkennung. Mit DNSBLs können eingehende Spam-Nachrichten bereits in einer sehr frühen Phase des Einlieferungsprozesses abgelehnt werden. Die wesentlich aufwendigere Analyse des Nachrichteninhalts ist dann nicht mehr erforderlich.

2.9.2.2 Inhaltsbasierte Blocklisten

Die bisher behandelten „klassischen“ DNSBLs enthalten Informationen bezüglich der Reputation von IP-Adressen und erlauben die Ablehnung von E-Mails unmittelbar nach dem Aufbau der SMTP-Verbindung. Zur zuverlässigen Erkennung von Spam-Nachrichten reicht es allerdings nicht aus, die IP-Adresse des einliefernden Mailservers zu überprüfen. Inhaltsbasierte Blocklisten bieten die Möglichkeit, als zusätzliches Kriterium den Nachrichteninhalt zu berücksichtigen. Im Unterschied zu den reputationsbasierten DNSBLs kommen inhaltsbasierte Blocklisten allerdings erst nach der Übermittlung des E-Mail-Inhalts zum Einsatz, woraus ein höherer Ressourcenverbrauch auf dem Mailserver resultiert.

Dazu können sog. **URIBLs** („URI Blacklists“) eingesetzt werden, die auf der Beobachtung basieren, dass Spam-Versender ihre Nachrichten zwar über eine Vielzahl von IP-Adressen bei den Mailservern einliefern können, sie jedoch nur eine begrenzte Menge von Webseiten betreiben, auf denen sie die beworbenen Produkte zum Kauf anbieten. URIBLs enthalten Einträge für Domainnamen, die im Text von Spam-Nachrichten vorkommen. Der Mailserver durchsucht beim Empfang einer E-Mail den Text nach Links und extrahiert daraus die Domainnamen. Die einzelnen Domainnamen werden dann mit dem Domainnamen der URIBL konkateniert, um sie in der URIBL nachzuschlagen. Ein Vertreter dieser Art ist der Dienst *uribl.com* [Uri].

Eine weitere Variante von DNSBLs, mit denen Spam-Nachrichten anhand ihres Inhalts erkannt werden können, setzt auf **Hashwerte des Nachrichteninhalts**. Um dabei kleinere Abweichungen im Nachrichtentext tolerieren zu können und auf denselben Hashwert abzubilden, kommt dabei üblicherweise ein unscharfes Hashverfahren (engl. „fuzzy hashing“) zum Einsatz. Im Rahmen des Projekts „NiX-Spam“ (<http://www.nixspam.org/>) des Heise-Verlags wird u. a. eine solche Hash-DNSBL angeboten, die in Verbindung mit dem populären Anti-Spam-Programm SpamAssassin eingesetzt werden kann.²⁹ Bei NiX-Spam werden die Fuzzy-Hashwerte ausschließlich aus der Struktur des E-Mail-Inhalts ermittelt: Dazu wird die Anzahl der Leerzeichen je Zeile sowie die Abfolge der Satzzeichen bestimmt und mit dem MD5-Verfahren in einen Hashwert überführt [Ung04].

²⁹Detail-Informationen sind unter <http://www.ixhash.net/> bzw. <http://spamassassin.apache.org/> verfügbar.

2.9.2.3 Kritik

Schryen benennt drei wesentliche Nachteile von DNS-basierten Blocklisten [Sch07b, S. 63 f.]: falsch-negative Fehler, falsch-positive Fehler, den steigenden Datenverkehr sowie die Abhängigkeit vom DNS.

Die Blocklisten können nie vollständig sein, da Spam-Versender IP-Adressen nur kurzzeitig verwenden. Dadurch veralten die Einträge in den Listen, und Spam wird nicht erkannt, was **falsch-negative Fehler** verursacht. Zum anderen kommt es in der Praxis vor, dass die IP-Adressen von legitimen Mailservern in die Blocklisten aufgenommen werden. In der Folge werden auch die E-Mails der berechtigten Nutzer dieser Mailserver abgewiesen (**falsch-positive Fehler**). Die Genauigkeit von DNSBLs wurde bereits eingehend untersucht [SBJ08]. Sinha et al. stellten im Jahr 2008 fest, dass etwa 20 % der Spam-Nachrichten von keiner der vier untersuchten DNSBLs erkannt wurden und bis zu 10 % der legitimen Nachrichten zu unrecht abgewiesen wurden [SBJ08]. Zudem weisen einige DNSBLs eine hohe Reaktionszeit auf, bis sie eine IP-Adresse in die DNSBLs aufnehmen [RFV07]. Da Spam-Versender inzwischen zunehmend auf Botnetze setzen, können solche DNSBLs mit den dort häufig wechselnden IP-Adressen nicht Schritt halten, wie eine Untersuchung am Beispiel des Conficker-Wurms zeigte [SG10].

Schryen weist ferner darauf hin, dass die Verwendung von DNSBLs zu einer **Steigerung des übertragenen Datenvolumens** führt. In der Tat sind Blocklisten inzwischen für einen erheblichen Anteil aller DNS-Anfragen verantwortlich: Jung und Sit gaben im Jahr 2004 den Anteil der DNSBL-Anfragen mit 14 % des gesamten Nachrichtenaufkommens an [JS04].

Ferner ist zu bedenken, dass die Verwendung von DNSBLs eine zusätzliche **Abhängigkeit vom DNS** bedingt. Zum einen muss sich ein Mailserver auf die Integrität der erhaltenen DNSBL-Antworten verlassen; zum anderen gelingt die Spam-Abwehr mit DNSBLs nur dann, wenn Verfügbarkeit bzw. Erreichbarkeit der beteiligten rekursiven bzw. autoritativen DNS-Server gewährleistet sind.

Neben der von Schryen vorgetragenen Kritik besteht bei DNSBLs das Risiko, dass der DNSBL-Anbieter seine Vertrauensstellung missbraucht. Da ein Mailserver die Entscheidungshoheit über den Empfang von E-Mails durch die Konsultation von DNSBLs teilweise aufgibt und an einen Drittanbieter auslagert, muss dessen Betreiber darauf vertrauen, dass der DNSBL-Anbieter alle Anfragen korrekt beantwortet und nicht unabsichtlich oder absichtlich falsche Antworten liefert.

2.9.2.4 Beobachtungsmöglichkeiten

Die DNS-Anfragen, die der Mailserver zur Abfrage der DNSBLs stellt, können Aufschluss über die Kommunikationsbeziehungen der Nutzer geben, die auf dem Mailserver ein E-Mail-Postfach haben. Dabei ist zwischen den Beobachtungsmöglichkeiten auf dem

rekursiven Nameserver, an den der Mailserver die Anfragen sendet, und den autoritativen Nameservern, auf denen die DNSBL vorgehalten wird, zu unterscheiden.

Der **rekursive Nameserver** kann in den DNS-Anfragen zum einen die IP-Adresse des Mailserver beobachten, da der Mailserver, der die E-Mails entgegennimmt, die DNS-Anfragen stellt. Darüber hinaus kann der rekursive Nameserver die aufgelösten Domainnamen beobachten. Die aufgelösten Domainnamen enthalten bei *reputationsbasierten DNSBLs* die IP-Adressen derjenigen Mailserver, welche E-Mails an den betrachteten Mailserver senden. Anhand der IP-Adressen der einliefernden Mailserver lässt sich mitunter auf die Organisation schließen, welcher ein Absender angehört. Dies gelingt allerdings nicht, falls ein Mailserver von einem externen Dienstleister oder dem Internetzugangsanbieter betrieben wird; in diesem Fall ist anhand der IP-Adresse des Mailservers kein Rückschluss auf die Organisation oder das Unternehmen des Absenders möglich. Bei *inhaltsbasierten DNSBLs* enthalten die DNS-Anfragen die Domainnamen von URLs (Uniform Resource Locator, [BLMM94]), die in den empfangenen E-Mails enthalten sind. Diese können Rückschluss auf den Nachrichteninhalte und die Organisation des Absenders geben. Nutzt ein Unternehmen einen rekursiven Nameserver eines Drittanbieters (s. Abschnitt 2.8.4), kann dieser also bei der Verwendung von DNSBLs u. U. auf Beziehungen zu anderen Unternehmen schließen.

Der **autoritative Nameserver**, auf dem die DNSBL vorgehalten wird, wird vom rekursiven Nameserver kontaktiert, den der Mailserver zur Namensauflösung einsetzt. Betreibt eine Organisation ihren rekursiven Nameserver selbst, kann dessen IP-Adresse Rückschlüsse auf die Identität der Organisation geben (s. auch Abschnitt 2.6.2.3). Der DNSBL-Betreiber kann in diesem Fall nachvollziehen, welche Mailserver E-Mails an die Organisation senden und – je nach Caching-Verhalten mehr oder weniger genau – die Zeitpunkte und das Volumen der empfangenen E-Mails bestimmen. Einblick in das Sendeverhalten der Organisation hat er hingegen nicht – es sei denn, die Mailserver der E-Mail-Empfänger verwenden dieselben DNSBLs wie der Absenders.

2.9.3 Authentifizierung von E-Mail-Absendern

Die in Abschnitt 2.9.2 beschriebenen DNS-Blocklisten ermöglichen die Ablehnung von unerwünschten Nachrichten anhand der Reputation der einliefernden Mailserver und anhand der Nachrichteninhalte. Ein komplementärer Ansatz besteht darin, legitime E-Mails als solche zu erkennen. Hierzu existieren drei Konzepte, welche auf das DNS zurückgreifen: das Sender Policy Framework (SPF), das Sender Identification Framework (SenderID) sowie DomainKeys Identified Mail (DKIM). Aus technischer Sicht liegt es nahe, das DNS für diesen Zweck zu verwenden, da das Routing von E-Mails zu ihrem Ziel ohnehin bereits durch das DNS kontrolliert wird und daher eine inhärente Abhängigkeit zum DNS besteht. Informationen bezüglich der empfangenden Mailserver liegen im Namensdienst bereits vor. Durch die vorgeschlagenen Anpassungen können nun auch Daten über die sendenden Mailserver erfasst werden – und zwar individuell durch die Betreiber der Mailserver, also genauso dezentral wie auch die Mailserver organisiert sind.

Alle drei Ansätze verfolgen das Ziel, die Identität des Senders einer E-Mail festzustellen und zu überprüfen, ob die angebliche Identität des Senders auch berechtigterweise verwendet wird, also ob die Identität *authentisch* ist. Dadurch kann verhindert werden, dass E-Mails unter Angabe eines falschen Namens bzw. einer falschen Absenderadresse versandt werden – eine Schwäche des E-Mail-Systems, die in der Vergangenheit für Phishing-Angriffe ausgenutzt wurde [Her09]. Das übergeordnete Ziel wird auf unterschiedlichen Wegen erreicht: SPF und SenderID erlauben es, den Pfad, auf dem eine E-Mail zum Ziel transportiert wird, etappenweise zu authentifizieren; bei DKIM wird hingegen unmittelbar der Inhalt der Nachricht authentifiziert [Mes11].

Die Argumentation der Befürworter der Techniken ist wie folgt [Cro08]: Nachrichten von einem vertrauenswürdigen Absender, dessen Identität durch eines der Verfahren bestätigt wurde, können dem Empfänger direkt zugestellt werden – falsche Ablehnungsentscheidungen (falsch-positive Fehler bei Blacklists) sind dabei ausgeschlossen. Für die verbleibenden E-Mails kann zur Entscheidungsfindung weiterhin auf Blacklists bzw. die Analyse des Nachrichten-Inhalts zurückgegriffen werden.

Dieser Abschnitt ist wie folgt aufgebaut: In Abschnitt 2.9.3.1 wird das SPF- und SenderID-Verfahren beschrieben, in Abschnitt 2.9.3.2 das DomainKeys-Verfahren. Im Anschluss daran wird in Abschnitt 2.9.3.3 die in der Literatur geäußerte Kritik an den Verfahren vorgetragen, bevor abschließend in Abschnitt 2.9.3.4 die Beobachtungsmöglichkeiten erörtert werden. Für eine detaillierte Darstellung wird auf die Veröffentlichungen der Messaging Anti-Abuse Working Group (MAAWG) [Cro08; Mes11] sowie den Artikel von Herzberg [Her09] verwiesen, auf denen die nachfolgenden Ausführungen basieren.

2.9.3.1 Funktionsweise von SPF und SenderID

Das Sender Policy Framework (SPF) ist in RFC 4408 spezifiziert. Es versetzt den Inhaber eines Domainnamens in die Lage, in seiner Zone Angaben zu den Mailservern zu machen, welche er für den Versand von E-Mails verwendet, d. h. in denen sein Domainname im Absenderfeld enthalten ist. Ein Mailserver, der eine Nachricht empfängt, kann dadurch während der Nachrichtenübermittlung überprüfen, ob er die E-Mail von einem von der Domain berechtigten Mailserver erhält. Dadurch soll verhindert werden, dass E-Mails mit gefälschten Absender-Adressen übermittelt werden.

Um SPF zu implementieren, muss der Inhaber einer Domain einen TXT-Record für seinen Domainnamen anlegen, in dem ein sog. Policy-Record hinterlegt wird. Langfristig soll statt des generischen TXT-Record-Typen hierfür der dedizierte SPF-Record-Type verwendet werden. Der SPF-Record enthält eine Liste von Termen, mit denen die berechtigten bzw. nicht berechtigten IP-Adressen definiert werden.

Die SPF-Implementierung auf dem empfangenden Mailserver basiert auf der E-Mail-Adresse, welche der sendende Mailserver mit dem SMTP-Kommando „MAIL FROM“ übermittelt. Der empfangende Mailserver ruft den Policy-Record für die Domain dieser E-Mail-Adresse ab und überprüft, ob die IP-Adresse des sendenden Mailservers darin als

berechtigt eingestuft wird. Die Funktionsweise wird anhand des nachfolgenden Beispiels deutlich.

Beispiel 2.7. In der Zone der Domain *example-bank.de* sei folgender TXT-Record eingetragen: „v=spf1 +IP4:123.1.1.10 +IP4:123.1.1.20 –all“. Ein Mailserver empfängt von der IP-Adresse 123.1.1.20 eine E-Mail. Dabei werden folgende Daten über die SMTP-Verbindung gesendet:

```
HELO mx1.example-bank.de
MAIL FROM: <info@example-bank.de>
RCPT TO: <max@mustermann.name>
DATA
From: Example Bank <info@example-bank.de>
To: Max Mustermann <max@mustermann.name>
Subject: Hinweis zu Ihrem Example-Bank-Depot
```

```
Hallo Herr Mustermann,
[ ... ]
.
QUIT
```

Der empfangende Mailserver ruft vor der endgültigen Zustellung den TXT-Record für den Domainnamen *example-bank.de* ab, da dieser im MAIL-FROM-Kommando übermittelt wurde. Am Präfix „v=spf1“ erkennt der Mailserver, dass es sich um einen SPF-Policy-Record handelt. Danach werden die einzelnen Terme von links nach rechts abgearbeitet, um die Identität des sendenden Servers zu überprüfen. Ein „+“ weist den Mailserver darauf hin, dass die darauffolgende IP-Adresse für den Mailversand berechtigt ist, ein „–“ hingegen deutet auf einen nicht berechtigten Mailserver hin. Im Beispiel sind zwei Adressen berechtigt; der letzte Term stuft alle verbleibenden Adressen als nicht berechtigt ein. Im Ergebnis wird der Mailserver die E-Mail also akzeptieren. □

Eine wesentliche Einschränkung von SPF besteht darin, dass lediglich die MAIL-FROM-Adresse zur Überprüfung herangezogen wird. Diese Adresse muss jedoch nicht unbedingt mit der E-Mail-Adresse im „From“-Header, welche dem Nutzer im E-Mail-Programm angezeigt wird, übereinstimmen. Dadurch könnte dem Nutzer eine gefälschte Absenderadresse präsentiert werden, obwohl die E-Mail die SPF-Authentifizierung erfolgreich durchlaufen hat. Dieser Mangel soll durch das **SenderID-Framework** [LW06b] behoben werden. Dabei kommen ebenfalls Policy-Records im DNS zum Einsatz, die jedoch eine leicht von SPF abweichende Syntax verwenden. Der wesentliche Unterschied zu SPF besteht darin, dass der empfangende Mailserver bei der SenderID-Authentifizierung die Policy-Records auch für die E-Mail-Adresse nachschlägt, die im „From“-Header übermittelt wird.

2.9.3.2 Funktionsweise von DKIM

Bei DomainKeys Identified Mail (DKIM), das in RFC [All+07] spezifiziert ist, wird die Authentizität des Senders sichergestellt, indem E-Mails vom Sender mit einer digitalen

Signatur versehen werden. DKIM unterscheidet sich von anderen Ansätzen zur Signatur (und Verschlüsselung) von elektronischen Nachrichten, etwa S/MIME [RS03] und PGP bzw. OpenPGP [Zim95; Cal+07], hinsichtlich folgender Aspekte:

- DKIM adressiert nicht das Ziel, den Nachrichteninhalt vertraulich zu übermitteln. Dementsprechend sieht es keine Möglichkeit zur Verschlüsselung der Nachricht vor.
- Bei S/MIME bzw. PGP/OpenPGP werden die Nachrichten üblicherweise vom sendenden Benutzer in dessen E-Mail-Programm signiert bzw. verschlüsselt. Die Prüfung der Signatur erfolgt im E-Mail-Programm des Empfängers. Bei DKIM wird die Signatur hingegen erst auf dem Transportweg ermittelt und der Nachricht hinzugefügt, etwa von dem Mailserver, bei dem die Nachricht vom Sender eingeliefert wird.
- Bei DKIM kann der Sender öffentlich bekannt geben, dass er ausschließlich signierte E-Mails versendet und demnach unsignierte E-Mails, die unter Verwendung seiner Identität verschickt werden, nicht von ihm stammen. Diese Information wird in einem speziellen DNS-Eintrag, der *Author Domain Signing Practice* (ADSP), hinterlegt [All+09]. Bei S/MIME bzw. PGP/OpenPGP gibt es diese Möglichkeit nicht.
- Bei S/MIME bzw. PGP/OpenPGP werden öffentliche Schlüssel üblicherweise von Zertifizierungsinstanzen oder anderen Nutzern beglaubigt und in Verzeichnissen zur Verfügung gestellt. Bei DKIM werden sie hingegen dezentral über das DNS zur Verfügung gestellt und lediglich vom jeweiligen Inhaber unterschrieben.

Die Funktionsweise von DKIM wird anhand des folgenden Beispiels deutlich.

Beispiel 2.8. Im Beispiel sendet Bob, der bei *firma.de* arbeitet, eine E-Mail an Alice, die bei *organisation.de* arbeitet. Bobs Firma hat DKIM implementiert und möchte alle Empfänger, die ebenfalls DKIM implementieren, darauf hinweisen, dass sämtliche E-Mails, die ihre Nutzer versenden, eine gültige DKIM-Signatur enthalten müssen. Andernfalls sollen die Empfänger davon ausgehen, dass die E-Mail einen gefälschten Absender aufweist und nicht zugestellt werden soll. Dazu veröffentlicht das Unternehmen einen TXT-Record für *_adsp._domainkey.firma.de* mit ihrer Author Domain Signing Practice. Im Beispiel ist dies „dkim=discardable“.

Beim Versand einer Nachricht fügt der Mailserver *mx2.firma.de* jeder E-Mail eine digitale Signatur hinzu, die über die E-Mail-Header gebildet wird, welche den Sender identifizieren und dem Nutzer angezeigt werden (u. a. der „From“-Header). Optional kann die Signatur auch über weitere Header-Felder sowie den eigentlichen Nachrichteninhalt gebildet werden. Die Signatur wird zusammen mit Angaben zur Bildungsvorschrift, zum verwendeten Algorithmus sowie zum Ort, an dem der öffentliche Schlüssel heruntergeladen werden kann, im Header „DKIM-Signature“ hinterlegt, wie der folgende Ausschnitt aus einer Nachricht zeigt (angelehnt an [Cro08, S. 15]):

```
Received: from mx2.firma.de (HELO mx2.firma.de) (212.3.0.4)
  by mx1.organisation.de with SMTP; 31 Aug 2012 19:53:28 -0000
```

```
DKIM-Signature: v=1; a=rsa-sha1; c=relaxed/relaxed; s=hamburg;
d=firma.de;
h=From:To:Subject:Mime-Version:Message-ID:Content-Type:Date;
i=author@firma.de; bh=EMR7D1qC7yKz41K8ArLct++IWxM=;
b=TGkNEq7fW40Ino/5DlX2qHDQeRmzhY+uiTzEcXu2KIKC+4B7+i2o1IWGZP9
JBnOR4Ck6iAiIdnRjDLuc2QJh3iFDNPWJ6xYjiuE73i1CZfbtN0r2MVke9p
RU4aydBQ5DSCFS7YhUFB22CT70MutZkaDFSZZpqI5vTlSWm9MI8PM=
Date: Fri, 31 Aug 2012 19:53:27 -0000
From: "Bob Smith" <bob@firma.de>
To: alice@organisation.de
Subject: Unser Meeting
Sender: bob@firma.de
Return-Path: bounce-4101674@firma.de Mime-Version: 1.0
Message-ID: <20080324040103985572.328428@firma.de
...
```

Der Mailserver, der E-Mails für *organisation.de* empfängt, implementiert ebenfalls DKIM. Für jede eingegangene E-Mail überprüft er, ob der Sender eine ADSP hinterlegt hat, indem er eine entsprechende TXT-Anfrage stellt. Im Beispiel erhält er eine positive Antwort, da der oben dargestellte ADSP-Eintrag existiert. Um die Signatur und damit die Identität von Bob zu überprüfen, muss der empfangende Mailserver *mx1.organisation.de* diese mit dem zugehörigen öffentlichen Schlüssel validieren. Den Schlüssel bezieht er ebenfalls mit einer TXT-Anfrage für den Domainnamen *hamburg._domainkey.firma.de*, den er aus den Angaben im „DKIM-Signature“-Header bildet. Im Beispiel sieht der Eintrag wie folgt aus:

```
g=*; k=rsa;
p=QIdfMAOGCSqGSib3DQEBAQUAZ4GNADCBiQKBgQC61RrUNTICNbf/+f5Co2V37GM
vPQdbUVyJgvLXrUKAXeJDwYVumAtE9BovuDZNYxcgG2oy7mkcZX/3rBF5SJX9Cp5y
w0axuMpzkuZPQq26h+2+MLuvTJtfdIzaHgNeEJOjMeq7s9RFQHRR9g261kZQTRAob
8YevaA1KHINnyIaZuQIDAQAB;
```

□

Ein wesentlicher Unterschied zu SPF und SenderID besteht darin, dass bei DKIM eine unmittelbare Authentifizierung der Senderidentität bzw. des Nachrichteninhalts stattfindet. Alle dazu erforderlichen Informationen und Modifikationen stehen unter der Kontrolle des Senders. Bei SPF und SenderID erfolgt die Authentifizierung hingegen anhand der IP-Adressen der am Transport beteiligten Mailserver, die mitunter nicht vom Sender kontrolliert werden. DKIM ist daher wesentlich flexibler und robuster als die anderen beiden Verfahren (s. auch Abschnitt 2.9.3.3).

2.9.3.3 Kritik

Die drei vorgestellten Verfahren weisen konzeptionelle Schwächen auf, die ihren Einsatz in der Praxis erschweren [Her09]. Wie Herzberg erläutert, führt die Verwendung von

Mail-Weiterleitungsdiensten oder Mailinglisten dazu, dass der Sender bei Verwendung von SPF und SenderID fälschlicherweise nicht authentifiziert werden kann, da es sich beim einliefernden Server nun nicht mehr um einen vom Sender autorisierten Server handelt, sondern um den Server des Weiterleitungsdienstes. Die vorgeschlagenen Abhilfen haben sich nach Herzbergs Ausführungen in der Praxis noch nicht durchgesetzt. Auch bei DKIM kann es zu fehlerhaften Zurückweisungen kommen, wenn die signierten Header von Intermediären unberechtigt modifiziert werden.

Weiterhin wird kritisiert, dass Phishing-Angriffe trotz Sender-Authentifizierung weiterhin möglich sind: Anstelle einen bereits registrierten Namen beim Mailversand zu nutzen und eine gefälschte Identität zu verwenden, können die Angreifer auch selbst Domains registrieren, die einen plausiblen, ähnlich klingenden Namen aufweisen. In ihrem eigenen Domainnamen können sie SPF-, SenderID- und DKIM-Records eintragen und dadurch authentifizierte E-Mails übermitteln. Herzbergs Untersuchungen deuten darauf hin, dass nur wenige Nutzer solche Phishing-Angriffe erkennen [Her09].

Herzberg kritisiert, dass die genannten Techniken darauf angewiesen sind, dass die übermittelten DNS-Nachrichten nicht manipuliert werden. Dies ist erst mit der Einführung von DNSSEC (s. Abschnitt 3.6.3) gewährleistet.

2.9.3.4 Beobachtungsmöglichkeiten

Die Sender-Authentifizierung mittels DNS-Anfragen kann auch Auswirkungen auf die Privatsphäre von Nutzern haben. Aus den gestellten DNS-Anfragen können sowohl der vom Mailserver des Empfängers verwendete rekursive Nameserver als auch die autoritativen Nameserver Informationen über die Kommunikationsbeziehungen der Nutzer gewinnen.

Der **rekursive Nameserver** erfährt bei allen drei Varianten der Sender-Authentifizierung die Domainnamen der einliefernden Mailserver bzw. die Domainnamen, welche in den E-Mailadressen der Absender enthalten sind; bei DNS-Blocklisten (s. Abschnitt 2.9.2) erfährt er hingegen lediglich die IP-Adressen der einliefernden E-Mail-Server. Insbesondere die Absender-Domainnamen sind tendenziell aussagekräftiger als die IP-Adressen der einliefernden Mailserver (vgl. Abschnitt 2.9.2.4). Wie beim Einsatz von DNS-Blocklisten liegen dem rekursiven Nameserver auch Informationen über den Absender einer E-Mail vor, so dass die übrigen Ausführungen in Abschnitt 2.9.2.4 auch bei der Verwendung der Techniken zur Sender-Authentifizierung zutreffen.

Ein Unterschied zu DNSBLs besteht allerdings in Bezug auf die Beobachtungsmöglichkeiten der **autoritativen Nameserver**. Die autoritativen Nameserver, auf denen die SPF-, SenderID- und DKIM-Records vorgehalten werden, stehen üblicherweise unter der Kontrolle der E-Mail-Absender, d. h. im Gegensatz zu DNSBLs werden sie nicht bei *allen* eingehenden E-Mails kontaktiert. Durch eine Beobachtung der dort eingehenden DNS-Anfragen kann der Absender einer E-Mail u. U. jedoch den Empfang der Nachricht auf dem Empfängerserver nachvollziehen: Der autoritative Nameserver wird vom rekursiven Nameserver des Empfängers kontaktiert, wenn die Nachricht dort eingeht und dabei die

Sender-Authentifizierung durchgeführt wird. Der Absender kann mit dieser Methode allerdings nicht feststellen, ob bzw. wann ein Nutzer eine E-Mail tatsächlich *gelesen* hat.

2.9.4 Der ENUM-Verzeichnisdienst für VoIP-Telephonie

Ein wesentlicher Aspekt des technischen Fortschritts ist die Konvergenz von Telekommunikations- und Informationstechnologien [Eur97; LHD05, S. 15 ff.]. Die ENUM-Technologie stellt einen Mechanismus zur Integration des Telefonnetzes mit dem Internet bereit [LHD05, S. 8]. Durch die Konvergenz der Netze werden die anfangs üblichen geschlossenen Kommunikationsdienste (etwa „Microsoft Netmeeting“ [HHS01, S. 181 f.], „MSN Messenger“ [SS05, S. 263 f.] sowie frühe Versionen von Skype³⁰), bei denen lediglich Nutzer des jeweiligen Dienstes miteinander sprechen konnten, von offenen Voice-over-IP-Diensten (VoIP-Dienste) abgelöst. Eine wesentliche Innovation der VoIP-Dienste besteht darin, dass die Kommunikationspartner nicht mehr zwingend denselben Dienst verwenden müssen. Zudem ist ein transparenter Netzübergang möglich, also eine Kommunikation zwischen Teilnehmern, die über das Telefonnetz angebunden sind, mit Teilnehmern, die über das Internet angebunden sind. Eine ausführliche Darstellung der Techniken, die bei VoIP-Diensten eine Rolle spielen, findet sich etwa bei Badach [Bad09].

Dieser Abschnitt ist wie folgt aufgebaut: Zunächst wird in Abschnitt 2.9.4.1 die Problemstellung erläutert, welche ENUM adressiert. In Abschnitt 2.9.4.2 wird dargelegt, auf welche Weise ENUM das DNS nutzt. Abschließend werden in Abschnitt 2.9.4.3 die Beobachtungsmöglichkeiten beschrieben, die sich durch die Nutzung von ENUM ergeben.

2.9.4.1 Problemstellung

ENUM adressiert das Teilproblem der Adressierung von Teilnehmern, welche über das Internet angebunden sind. Das im Folgenden geschilderte Einsatzszenario verdeutlicht eine von mehreren Herausforderungen, welche mit ENUM gelöst werden können. Weitere denkbare Einsatzszenarien sind in [Bad09, S. 112 f.] und [LHD05, S. 69 ff.] aufgeführt.

Damit die Interoperabilität mit dem existierenden Telefonnetz gewährleistet ist, ist angedacht, auch die über das Internet vermittelten Gespräche mit den bereits im Telefonnetz gebräuchlichen Rufnummern zu adressieren [Bad09, S. 108]. Endgeräte, welche über das Internet angebunden sind, können jedoch nicht unmittelbar mit einer Rufnummer adressiert werden; sie verfügen lediglich über eine (u. U. dynamische) IP-Adresse. Um einen Anruf aus dem Telefonnetz an einen über das Internet angebundenen Teilnehmer zu vermitteln, muss folglich anhand der gewählten Rufnummer des Angerufenen die IP-Adresse

³⁰Mit frühen Versionen der Software Skype konnten keine Verbindungen zu Rufnummern im Telefonnetz aufgebaut werden (vgl. hierzu etwa einen kritischen Beitrag in der Zeitschrift *NetworkWorld* vom 24.11.2003 [Bra03]). Später wurde entsprechende Funktionalität unter der Bezeichnung „SkypeOut“ eingeführt.

ermittelt werden, unter welcher sein VoIP-Endgerät gerade erreichbar ist. Hierzu sind zwei Adressumsetzungen erforderlich.

Die erste Umsetzung betrifft die **Ermittlung der aktuellen IP-Adresse** des Angerufenen. Diese Umsetzung ist bei jedem VoIP-Telefonat erforderlich, also auch dann, wenn beide Teilnehmer VoIP-Endgeräte verwenden. Bei den gängigen VoIP-Lösungen erfolgt die Rufsignalisierung über das *Session Initiation Protocol* (SIP; spezifiziert in [Ros+02]) unter Zuhilfenahme von SIP-Proxys [Bad09, S. 280 ff.], welche den Nachrichtentransport unterstützen. Zur Adressierung der Teilnehmer werden dabei sog. SIP-Adressen verwendet, welche als URIs (Uniform Ressource Identifier; [BLFM05]) dargestellt werden (z. B. *sip:user2152@voip-provider.de*). Damit eingehende Anrufe möglich sind, registriert sich ein VoIP-Endgerät mit seiner aktuellen IP-Adresse und dem SIP-URI seines Nutzers beim VoIP-Anbieter. Dieser hinterlegt die Zuordnung zwischen SIP-URIs und IP-Adressen in seinem *Location- oder Registrar-Dienst* [Bad09, S. 283, 330]. Geht ein Anruf für einen registrierten Teilnehmer ein, schlägt der Anbieter bei diesem Dienst anhand des übermittelten SIP-URIs die aktuelle IP-Adresse des Angerufenen nach, um den Anruf dorthin durchzustellen. Der Anrufer muss also nicht die aktuelle IP-Adresse, sondern lediglich den SIP-URI des Angerufenen kennen.

Da die alphanumerischen SIP-URIs auf herkömmlichen Telefonen nicht eingegeben werden können, erhalten Kunden von ihrem VoIP-Anbieter üblicherweise zusätzlich eine reguläre Telefonnummer. Bereits existierende Rufnummern aus dem Telefonnetz können (ggf. nach einer Portierung zum VoIP-Anbieter) weiterverwendet werden (vgl. hierzu auch [LHD05, S. 63]). Beim VoIP-Anbieter nimmt ein *Gateway* eingehende Anrufe aus dem Telefonnetz entgegen und leitet diese an das VoIP-Endgerät des Kunden weiter. Dabei findet die zweite Adressumsetzung (**Rufnummer-URI-Zuordnung**) statt, welche anhand der Rufnummer den SIP-URI des angerufenen Teilnehmers ermittelt. Hierzu greift das Gateway des VoIP-Anbieters auf eine Datenbank zurück, in der die Zuordnung zwischen den Rufnummern und den SIP-URIs seiner Kunden hinterlegt ist.

Bei dieser – in der Praxis derzeit üblichen – Realisierung ist lediglich der eigene VoIP-Anbieter dazu in der Lage, die Adressumsetzung zwischen Rufnummern und SIP-URIs vorzunehmen. Hierfür gibt es zwei Gründe: Zum einen ist die Zuordnung zwischen Rufnummern und SIP-URIs nicht öffentlich verfügbar. Zum anderen kann der Anrufer bzw. sein SIP-Proxy anhand der Rufnummer nicht ermitteln, ob der angerufene Teilnehmer einen VoIP-Dienst nutzt bzw. bei welchem VoIP-Anbieter die SIP-URI des Angerufenen in Erfahrung gebracht werden kann. Diese Einschränkung führt dazu, dass derzeit in der Praxis sämtliche Telefonate, welche anhand einer Rufnummer adressiert werden, stets über das herkömmliche Telefonnetz vermittelt werden – auch dann, wenn sowohl der anrufende als auch der angerufene Teilnehmer über VoIP-Telefonanschlüsse verfügen und ein Anruf somit grundsätzlich vollständig über das Internet abgewickelt werden könnte.³¹

³¹Der Umweg über das Telefonnetz kann lediglich dann vermieden werden, wenn beide Kommunikationspartner bei demselben VoIP-Anbieter registriert sind bzw. falls die VoIP-Anbieter untereinander die erforderlichen Routing-Informationen ausgetauscht haben.

ENUM erlaubt es hingegen zukünftig die Rufnummern-URI-Zuordnung auf einheitliche Weise und an einer festgelegten Stelle zu veröffentlichen, sodass Anrufe zwischen VoIP-fähigen Endgeräten ohne Umwege vermittelt werden können.

2.9.4.2 Infrastruktur und Einsatz

Die Abkürzung ENUM steht für „Telephone Number URI Mapping“.³² Bei ENUM handelt es sich um einen Übersetzungsdienst, mit dem herkömmliche Telefonnummern in URIs – insbesondere SIP-URIs – umgesetzt werden können. Die Internet Engineering Task Force (IETF) hat im Jahr 1999 eine ENUM-Arbeitsgruppe eingerichtet, welche mit der Standardisierung beauftragt wurde [LHD05, S. 10]. Der ursprüngliche ENUM-Standard wurde in den RFCs 2915 und 2916 [MD00a; Fal00] festgeschrieben; die aktuell gültige Fassung findet sich in RFC 3761 [FM04].³³

Für den ENUM-Dienst wird im DNS-Namensraum die Domain *el64.arpa* bereitgestellt.³⁴ Unterhalb dieser Domain können Resource-Records für die einzelnen Rufnummern abgelegt werden, mit denen die Adressumsetzung vorgenommen werden kann. Ursprünglich war angedacht, den Endkunden vollständigen Zugriff auf ihre ENUM-Einträge anzubieten, um die Adressumsetzung unmittelbar zu beeinflussen. Dieser in der Praxis bislang kaum anzutreffende Anwendungsfall, der auch als „End User ENUM“ bezeichnet wird, stößt inzwischen zunehmend auf Kritik, u. a. wegen des erforderlichen technischen Sachverstands [LHD05, S. 84]. Als potenzielle Alternative gilt ein Szenario, in dem die Telefonanbieter bzw. die Anbieter von VoIP-Diensten entsprechende ENUM-Einträge vornehmen, was auch als „Carrier ENUM“ bezeichnet wird [LHD05, S. 76].

Der ENUM-Domainname eines Eintrags lässt sich wie folgt aus einer Rufnummer ableiten (nach [Bad09, S. 110]):

1. Alle nichtnumerischen Zeichen werden aus der Rufnummer entfernt.
2. Die Reihenfolge der Ziffern wird umgekehrt.
3. Die Ziffern werden jeweils durch einen Punkt voneinander getrennt.
4. Am Ende dieser Zeichenfolge wird das Suffix *.el64.arpa* angehängt.

Der Raum der ENUM-Domains wird wie der übrige DNS-Namensraum von einzelnen Registraren verwaltet. Für die deutsche Subdomain *9.4.el64.arpa* ist derzeit die DeNIC zuständig [Bad09, S. 109]. Für jede ENUM-Domain können ein oder mehrere NAPTR-Resource-Records (vgl. Abschnitt 2.5) abgelegt werden, um eine priorisierte Adressumsetzung für verschiedene Dienste vorzusehen. Neben der Umsetzung in SIP-URIs sehen die

³²Auch die alternativen Bezeichnungen „Telephone Number Mapping“, „Electronic Numbering“ sowie „Enhancement of Numbering and Naming“ sind gebräuchlich [Bad09, S. 92].

³³Die ENUM-Anwendung wurde auch in das generischen „Dynamic Delegation Discovery System“ (DDDS) eingebettet, das in den RFCs 3401 bis 3405 spezifiziert wird [Mea02a; Mea02b; Mea02c; Mea02d; Mea02e].

³⁴Der Name leitet sich aus dem E.164-Standard der ITU-T (Telecommunication Standardization Sector der International Telecommunication Union) ab [Bad09, S. 109].

Standards auch Schemata zur Umwandlung von Rufnummern in E-Mail-Adressen bzw. URLs von Webseiten vor (vgl. [Fal00, S. 8] und [FM04, S. 11]). Spezifische Hinweise zur Integration von ENUM und SIP finden sich in RFC 3842 [Pet+04]. Das folgende Beispiel veranschaulicht die Umsetzung von Rufnummern in Domains und die hinterlegten Daten.

Beispiel 2.9. Die Rufnummer +49-40-42883-2092 wird durch Anwendung der obigen Regeln in die ENUM-Domain `2.9.0.2.3.8.8.2.4.0.4.9.4.e164.arpa` umgeformt. Für diese Domain seien folgende NAPTR-RRs hinterlegt (vgl. [FM04, S. 11]):

```
NAPTR 10 100 "u" "E2U+sip" "!^.*$!sip:sr3191@voiper.com!" .  
NAPTR 10 101 "u" "E2U+msg" "!^.*$!mailto:sabine.ross@mailor.de!" .
```

Diese Einträge besagen, dass der Nutzer bevorzugt per SIP kontaktiert werden möchte, ersatzweise per E-Mail. Der reguläre Ausdruck im vorletzten Feld der ersten Zeile selektiert die gesamte ENUM-Domain und ersetzt diese durch den angegebenen SIP-URI. Anhand der SIP-URI kann über SIP dann die IP-Adresse des VoIP-Endgeräts ermittelt werden. □

2.9.4.3 Beobachtungsmöglichkeiten

Durch ENUM ergeben sich zwei Beobachtungsmöglichkeiten: In [Kam+05] wird zum einen darauf hingewiesen, dass durch die Veröffentlichung der NAPTR-RRs auf **autoritativen Nameservern** eine öffentliche und leicht durchsuchbare Datenbank entsteht, anhand der sich unmittelbar ermitteln lässt, ob eine Rufnummer einem Teilnehmer zugewiesen ist. Enthalten die Angaben in den SIP-URIs zusätzliche Hinweise auf die Identität des Nutzers, kann im Prinzip jeder Internetnutzer ermitteln, zu welcher Person eine Rufnummer gehört (Rückwärtssuche). Durch den systematischen Aufbau der ENUM-Domains und die Beschränkung auf Ziffern lässt sich zudem die vollständige Datenbank mit überschaubarem Aufwand enumerieren.

Die zweite Beobachtungsmöglichkeit ergibt sich auf den **rekursiven Nameservern**. Da beim Herstellen einer Telefonverbindung eine ENUM-Anfrage für die gewählte Rufnummer übermittelt wird, kann der rekursive Nameserver unmittelbar die ausgehenden Anrufe eines Nutzers (der ggf. anhand seiner IP-Adresse identifizierbar ist) beobachten. Ist eine Beobachtung über einen längeren Zeitraum möglich kann der rekursive Nameserver anhand der beobachteten Rufnummern auf das Kommunikationsverhalten seiner Nutzer schließen. Auf diese Beobachtungsmöglichkeit wird bereits in [CPGA08] hingewiesen; auf den dort beschriebenen Schutzmechanismus wird in Abschnitt 5.2.3.2 noch näher eingegangen.

2.9.5 Der Object Name Service (ONS)

Der Object Name Service (ONS) ist ein auf dem DNS basierender Namensdienst, der in Zukunft als Teil des EPCglobal-Netzes den Zugriff auf Daten erlauben soll, die zu RFID-Tags hinterlegt sind [EPC08; EPC13].

Dieser Abschnitt ist wie folgt aufgebaut: Zunächst wird in Abschnitt 2.9.5.1 der Kontext erläutert, in dem der ONS eingesetzt werden soll. In Abschnitt 2.9.5.2 wird die geplante Implementierung des ONS erläutert. Abschließend werden in Abschnitt 2.9.5.3 die Beobachtungsmöglichkeiten beschrieben, die sich durch ONS-Anfragen möglicherweise ergeben werden.

2.9.5.1 Hintergrund

RFID-Tags sind Mikrochips, deren Inhalt mittels des Radio-Frequency Identification-Verfahrens drahtlos über eine Funkschnittstelle ausgelesen werden kann.³⁵ RFID-Tags werden vor allem im Bereich der Fertigung und der Logistik eingesetzt; langfristig sollen sie die aufgedruckten Barcodes im Einzelhandel ablösen. Da RFID-Tags über sehr wenig Speicherplatz verfügen, können sie nicht dazu verwendet werden, um darin detaillierte Informationen über das zugehörige Objekt abzulegen. Stattdessen dienen sie lediglich zur Speicherung bzw. zum Auslesen einer eindeutigen Kennung. Der Aufbau dieser Kennungen ist standardisiert: In Anlehnung an den Universal Product Code, der im amerikanischen Raum bei Barcodes zum Einsatz kommt, wird die Kennung bei RFID-Tags als Electronic Product Code (EPC) bezeichnet.

Zur Hinterlegung von Informationen zu einem EPC kommt ein verteilter Verbund von Informationssystemen zum Einsatz, das EPCglobal-Netz [EPC13]. Im EPCglobal-Netz können für jeden EPC Informationen über das zugehörige Produkt hinterlegt und abgefragt werden. Neben Produkteigenschaften zählen hierzu Angaben zum Hersteller und zur Lieferkette. Die Datenhaltung erfolgt im EPCglobal-Netz verteilt: Jeder Hersteller betreibt eine eigene Datenbank für seine Produkte, die über das Internet zur Verfügung gestellt wird. Um zu ermitteln, welcher Server für einen EPC zuständig ist, wird der ONS konsultiert.

2.9.5.2 Implementierung des ONS mit dem DNS

Ein EPC wird durch eine URI (Uniform Resource Identifier; [BLFM05]) dargestellt, eine mehrgliedrige Zeichenkette, mit der sich ein hierarchischer Namensraum definieren lässt. Eine EPC URI besteht aus fünf Segmenten, die durch Doppelpunkte getrennt werden. Die Segmente sind *Namespace* (derzeit stets „urn“), *Subspace* (derzeit stets „epc“), *Typ* (derzeit stets „id“), *Sector* und *ID*. Der Sector gibt den Industriebereich an, welchem das Objekt zuzuordnen ist, z. B. „sgtin“ („Serialized Global Trade Item Number“). Der Aufbau der ID hängt vom Sector ab; beim Industriebereich „sgtin“ wird das Schema „CompanyPrefix.ItemReference.SerialNumber“ verwendet.

Der Informationsabruf im EPCglobal-Netz lässt sich grob in drei Phasen untergliedern: Zunächst wird der EPC in einen DNS-Domainnamen umgewandelt. Dann wird dieser

³⁵Die Darstellung in Abschnitt 2.9.5 folgt dem Buch „RFID Essentials“ von Glover und Bhatt [GB06].

Name im DNS nachgeschlagen. Anhand der Antwort wird schließlich eine URL gebildet, über welche die eigentlichen Informationen zu dem EPC erreichbar sind. Ein EPC wird wie folgt in einen Domainnamen umgewandelt [EPC13, S. 10 f.]:

1. Die Felder Namespace und Subspace am Anfang der URI werden entfernt.
2. Das Feld SerialNumber am Ende der URI wird entfernt.
3. Die Felder werden in umgekehrter Reihenfolge angeordnet.
4. Die Doppelpunkte werden durch einfache Punkte ersetzt.
5. An die Zeichenkette wird der reservierte Domainname *onsepc.com* angehängt.³⁶

Für den daraus resultierenden Domainnamen wird dann der Inhalt des NAPTR-Records (vgl. Abschnitt 2.5) abgerufen, um aus dem EPC den URI des EPC-Dienstes zu ermitteln, welcher für den betreffenden EPC zuständig ist [EPC13].

Beispiel 2.10. Es sei der EPC *urn:epc:id:sgtin:0614141.000024.400* gegeben. Der zum EPC korrespondierende Domainnamen lautet demnach *000024.0614141.sgtin.id.onsepc.com*. Der NAPTR-Record dieses Namens enthält im Beispiel den regulären Ausdruck

`^.*$!http://example.com/cgi-bin/epcis!`

Dieser wird auf den EPC angewendet. Es ergibt sich die URL *http://example.com/cgi-bin/epcis?urn:epc:id:sgtin:0614141.000024.400*, unter welcher die Informationen zum EPC vorgehalten werden (Beispiel nach [EPC08, S. 13 ff.]). □

2.9.5.3 Beobachtungsmöglichkeiten

Im Gegensatz zum DNS, dessen Root-Server von mehreren Unternehmen betrieben werden, war ursprünglich vorgesehen, dass die Firma Verisign die Nameserver, die für die ONS-Root-Domain *onsepc.com* autoritativ sind, betreiben sollte [Vio04]. Kritiker haben auf die Nachteile dieses Ansatzes, die hohe Machtkonzentration sowie die geringe Ausfallsicherheit, hingewiesen [EFG08]. Bei diesem Betriebsmodell haben die **ONS-Root-Server umfassende Beobachtungsmöglichkeiten**. Die im Jahr 2013 verabschiedete Version 2 des ONS-Standards reduziert die Machtkonzentration, indem den länderspezifischen GSI-Mitgliedsorganisationen die Möglichkeit eingeräumt wird, eigene Root-Domains zu verwenden [EPC13, S. 13]. Weiterhin wurden Vorschläge für einen föderierten Betrieb des ONS veröffentlicht [BKFS11; Li+13]; ob und wann diese in der Praxis umgesetzt werden, ist derzeit allerdings noch nicht absehbar.

Unabhängig davon existieren voraussichtlich erhebliche Beobachtungsmöglichkeiten auf den **rekursiven Nameservern**: Fabian et al. weisen in [FGS05] darauf hin, dass durch die Übermittlung der Datenfelder *CompanyPrefix* und *ItemReference* in den ONS-Anfragen jedem Beobachter auf dem Transportweg die Möglichkeit eröffnet wird, Hersteller und Artikel zu ermitteln, auf die sich die Anfrage bezieht. Die Autoren geben zu bedenken, dass

³⁶Jede Mitgliedsorganisation verwendet einen eigenen Domainnamen [EPC13, S. 13 f.]

Unternehmen und Privatanutzer dadurch u. U. unbewusst und auf unerwünschte Weise Auskunft über ihre Besitzverhältnisse geben. Anfragen für seltene EPCs oder Kombinationen aus mehreren EPCs können zudem zu einem eindeutigen Erkennungsmerkmal werden, anhand dessen sich Nutzer wiedererkennen lassen. Als Abhilfe wird in [GABK09] die Verwendung des Tor-Anonymitätsdienstes (s. Abschnitt 5.2.2) vorgeschlagen.

2.9.6 Unterstützung der Gegenstellen-Authentifizierung in TLS

Zahlreiche populäre Anwendungsprotokolle greifen zur Sicherung der übertragenen Daten auf das *Transport-Layer-Security-Protokoll* (TLS; RFC 5246 [DR08]) zurück. Insbesondere wird es von zahlreichen Webservern verwendet, um Daten vertraulich mit einem Browser auszutauschen („HTTPS“-Schema; RFC 2818 [Res00]). Das TLS-Protokoll wird jedoch nicht nur zur Verschlüsselung von Verbindungen eingesetzt. Die Kommunikationspartner haben damit auch die Möglichkeit, sich davon zu überzeugen, dass es sich bei der Gegenstelle um die tatsächlich gewünschte handelt – und nicht um einen sog. Man-in-the-Middle-Angreifer (MitM-Angreifer) [PP09, S. 342 ff.], der die Kommunikation (etwa durch einen DNS-Spoofing-Angriff, s. S. 116) zu sich umgeleitet hat. Für diese **Gegenstellen-Authentifizierung** kommen X.509-Zertifikate zum Einsatz, die in RFC 5280 im Kontext von TLS unter der Bezeichnung „PKIX-Zertifikate“ definiert werden [Coo+08]. RFC 6125 beschreibt das Authentifizierungsprotokoll im Detail [SAH11].

Die Gegenstellen-Authentifizierung weist allerdings konzeptionelle Schwachstellen auf, woraus sich Sicherheitsrisiken ergeben. Es existieren drei Vorschläge, mit denen die Ausnutzbarkeit der Schwachstellen durch Nutzung des DNS behoben werden sollen: Zum einen gibt es die zwei **Domain-Inhaber-Konzepte** DANE und CAA, welche die Sicherheit erhöhen, indem der Domain-Inhaber zusätzliche RRs in seiner Zone definiert. Zum anderen gibt es **Drittanbieter-Konzepte** (sog. Notary-Dienste), die zur Zertifikatsvalidierung herangezogen werden. Ein DNS-basierter Vertreter ist der „ICSI Certificate Notary“. Insbesondere die Drittanbieter-Konzepte können zu zusätzlichen Beobachtungsmöglichkeiten führen.

Dieser Abschnitt ist wie folgt aufgebaut: Im Folgenden wird zunächst in Abschnitt 2.9.6.1 die Problemstellung erläutert, d. h. welche Schwächen die TLS-Gegenstellen-Authentifizierung aufweist. Anschließend werden die Lösungsansätze betrachtet, die auf dem DNS basieren. Hierzu werden in Abschnitt 2.9.6.2 DANE und CAA vorgestellt, bevor in Abschnitt 2.9.6.3 der „ICSI Certificate Notary“ beschrieben wird.

2.9.6.1 Schwächen der aktuellen Gegenstellen-Authentifizierung bei TLS

Die Schwächen der derzeit implementierten Form der Gegenstellen-Authentifizierung lassen sich anhand des Abrufs einer Webseite von einer HTTPS-URL illustrieren. In der Praxis wird bei HTTPS-Anfragen häufig auf eine gegenseitige Authentifizierung verzichtet; nur der Server authentifiziert sich gegenüber dem Browser, jedoch nicht anders herum.

Stellt der Browser eine HTTPS-Verbindung zu einem Webserver her, ermittelt er zunächst anhand des Domainnamens in der HTTPS-URL die IP-Adresse des Webserver. Zu diesem Webserver baut der Browser eine TCP-Verbindung auf und stößt den sog. TLS-Handshake an [Res00, S. 2]. Während des TLS-Handshakes überprüft der Client, ob der Server dazu berechtigt ist, Webseiten für die betreffende Domain auszuliefern. Als Beweis der Authentizität fordert der Browser vom Webserver ein PKIX-Zertifikat an [Bar11a, S. 1]. Der Server kann sich nur dann gegenüber dem Browser erfolgreich authentifizieren, wenn drei Bedingungen erfüllt sind [Bar11a, S. 4]:

1. Der im Server-Zertifikat enthaltene Domainname stimmt mit dem Domainnamen überein, der in der URL der abzurufenden Webseite enthalten ist [SAH11].
2. Das Server-Zertifikat wurde von einer vertrauenswürdigen Zertifizierungsstelle (Certification Authority; CA) ausgestellt, was in RFC 5280 als „Validierung des Zertifizierungspaths“ bezeichnet wird [Coo+08, S. 77].
3. Der Server erbringt in einem Challenge-Response-Protokoll den Nachweis, dass er im Besitz des privaten Schlüssels ist, der zum Server-Zertifikat gehört.

Wurde die Gegenstellen-Authentifizierung erfolgreich durchlaufen, wird ein Sitzungsschlüssel etabliert und die HTTP-Anfrage übermittelt; andernfalls zeigt der Browser dem Nutzer eine Fehlermeldung an. Zur Validierung des Zertifizierungspaths greifen die Browser auf einen eingebauten Zertifikatsspeicher zurück, in dem die Browser-Hersteller die Wurzelzertifikate von Zertifizierungsstellen hinterlegen, die sie als vertrauenswürdig einstufen [Bar11a, S. 4]. Die Sicherheit der Server-Authentifizierung ist gewährleistet, solange die beiden folgenden Annahmen erfüllt sind:

1. Der private Schlüssel ist dem Angreifer nicht zugänglich.
2. Alle im Browser hinterlegten CAs arbeiten korrekt, d. h. lediglich der rechtmäßige Betreiber eines bestimmten Online-Angebots bzw. der tatsächliche Inhaber einer Domain kann ein Zertifikat für diese Domain erhalten.

Solange beide Annahmen erfüllt sind, ist es einem Angreifer nicht möglich, die Identität des rechtmäßigen Webserver anzunehmen, da er nicht dazu in der Lage ist, die drei Bedingungen zu erfüllen, die während der Gegenstellen-Authentifizierung durchlaufen werden. Die erste Annahme kann durch mangelhafte Sicherheitsvorkehrungen auf dem Webserver des Dienstbetreibers verletzt werden. Eine Verletzung der zweiten Annahme ist hingegen schwerwiegender. Zum einen sind die Auswirkungen nicht auf einen einzelnen Webserver beschränkt, zum anderen werden die betroffenen Angebote (bzw. die Webserver) gar nicht unmittelbar angegriffen. Der Betreiber eines Webserver kann daher keine Maßnahmen zu seinem Schutz ergreifen, sondern muss darauf vertrauen, dass die CAs ihre Aufgabe korrekt erfüllen.

Die wesentliche Schwäche des derzeitigen PKIX-Konzepts – die durch DANE adressiert wird – besteht darin, dass alle vertrauenswürdigen CAs von den Browsern gleich behandelt werden. Jede CA, deren Wurzelzertifikat im Zertifikatsspeicher hinterlegt ist, ist prinzipiell berechtigt und auch dazu in der Lage, ein Zertifikat für einen beliebigen Domainnamen

auszustellen, das von allen Browsern akzeptiert wird (s. [Bar11a, S. 4] und [SS11]). Darüber hinaus können Zertifizierungsstellen durch Ausstellung von sog. Intermediate-Zertifikaten andere Zertifizierungsstellen (sog. Sub-CAs) dazu ermächtigen, Zertifikate auszustellen – ebenfalls für beliebige Domainnamen. Den Ergebnissen einer Studie der Electronic Frontier Foundation (EFF) zufolge vertrauten im Jahr 2010 die Web-Browser Firefox und Internet Explorer indirekt somit den Zertifikaten von mehr als 650 Zertifizierungsstellen.³⁷ Diese konzeptionelle Schwäche wurde in der Vergangenheit mehrmals ausgenutzt, um MitM-Angriffe durchzuführen: Im Jahr 2011 gelang es Angreifern durch Kompromittierung einer Sub-CA von Comodo sowie der CA DigiNotar valide Zertifikate für populäre Webseiten wie Google und Paypal auszustellen [Lea11].

2.9.6.2 DANE und CAA

Aus den DNS-Anfragen, die beim Einsatz der Techniken DANE und CAA gestellt werden, ergeben sich auf den rekursiven und autoritativen Nameservern **keine zusätzlichen Beobachtungsmöglichkeiten**, die über die bereits durch die Namensauflösung existierenden Möglichkeiten hinausgehen. Daher werden diese Techniken nur kurz beschrieben.

DANE Die Abkürzung DANE steht für „DNS-based Authentication of Named Entities“. DANE ist in den RFCs 6394 und 6698 standardisiert [Bar11b; HS12e]. Das DANE-Konzept ermöglicht es, die globale Autorität, die jeder CA im PKIX-Konzept eingeräumt wird, einzuschränken. Dabei wird die Tatsache ausgenutzt, dass der Domain-Inhaber üblicherweise genau weiß, welche Zertifikate er besitzt und von welchen CAs diese ausgestellt wurden. Dadurch können unberechtigt ausgestellte Zertifikate von den Clients erkannt werden.

DANE führt den neuen DNS-Record-Typ „TLSA“ ein, der in RFC 6698 spezifiziert wird [HS12e]. Darin kann der Domain-Inhaber Details bezüglich der Server-Zertifikate bekanntgeben, die er in seiner Domain einsetzt. DANE-fähige Clients rufen die TLSA-Records während des TLS-Handshakes ab und überprüfen die während des Handshakes präsentierten Server-Zertifikate auf Konformität mit den Angaben in den TLSA-Records. Um zu verhindern, dass ein MitM-Angreifer die Angaben in den TLSA-Records manipulieren kann, setzt DANE die Verwendung von DNSSEC zur Sicherung der Integrität voraus.

TLSA-Records können gemäß RFC 6394 in den folgenden Anwendungsfällen eingesetzt werden [Bar11b; Bar11a, S. 5]:

- **Bekanntgabe von CA-Einschränkungen:** Der Client soll lediglich Server-Zertifikate akzeptieren, die von einer bestimmten CA ausgestellt wurden.
- **Bekanntgabe von Zertifikateinschränkungen:** Der Client soll lediglich ein bestimmtes Server-Zertifikat akzeptieren.

³⁷Datensatz und Präsentationsfolien sind unter <https://www.eff.org/observatory/> verfügbar.

- **Definition eines eigenen Vertrauensankers:** Der Client soll ein bestimmtes, im TLSA-Record zur Verfügung gestelltes, Wurzelzertifikat zur Überprüfung von Zertifikaten innerhalb der Domain verwenden.

TLSA-Records können von allen Anwendungen eingesetzt werden, die auf TLS zurückgreifen. Es existieren bereits die ersten praktischen Implementierungen in Web-Browsern; die Integration in E-Mail-Clients zur Überprüfung von S/MIME-Zertifikaten ist geplant [HS12f]. Der genaue Aufbau von TLSA-Records ist in RFC 6698 beschrieben.

CAA Ein zu DANE komplementäres Konzept ist „DNS Certification Authority Authorization“ (CAA) [HBS13]. Auch beim CAA-Ansatz werden in einer Zone zusätzliche RRs eingefügt, mit denen der Domain-Inhaber spezifizieren kann, welche CA Server-Zertifikate für die jeweilige Domain ausstellen darf. Im Gegensatz zu DANE, bei dem ein TLS-Client diese RRs abrufen und mit dem präsentierten Zertifikat vergleicht, richten sich die CAA-Records jedoch an CAs. Eine CAA-konforme CA muss bei der Ausstellung eines Server-Zertifikats für eine Domain überprüfen, ob der Domain-Inhaber für diese einen CAA-Record hinterlegt hat. Nur wenn in diesem CAA-Record die CA als berechnigte CA aufgeführt ist, darf sie das Zertifikat ausstellen.

Problematisch an diesem Ansatz ist, dass er die Sicherheit nur erhöht, wenn alle CAs CAA-konform handeln. Zudem schützt das Konzept nicht vor Angreifern, die in den Besitz des Zertifizierungsschlüssels einer CA gelangen (wie im Fall von DigiNotar) und selbst entscheiden, für welche Domainnamen sie Server-Zertifikate ausstellen.

2.9.6.3 ICSI Certificate Notary

Der vom ICSI betriebene DNS-Dienst „Certificate Notary“ (<http://notary.icsi.berkeley.edu/>) verfolgt einen anderen Ansatz, um die Schwachstellen der Gegenstellen-Authentifizierung zu überwinden. Clients erhalten dadurch die Möglichkeit, zu erkennen, ob ein von einem Webserver präsentiertes Server-Zertifikat authentisch ist oder ob ein MitM-Angreifer dem Client ein nicht-autorisiertes Zertifikat präsentiert.

Dabei wird ausgenutzt, dass MitM-Angreifer typischerweise nur einen lokal abgegrenzten Teil des Internets kontrollieren und den Angriff zu einem bestimmten Zeitpunkt starten. Der ICSI-Notary-Dienst ermöglicht es dem Client zu überprüfen, wie das Server-Zertifikat des Webservers aus einer anderen Perspektive aussieht. Die Betreiber des Notary-Dienstes sammeln Server-Zertifikate, indem sie beobachten, welche Zertifikate im Datenverkehr von großen Internetanbietern übertragen werden (genaue Angaben zu den Quellen machen sie allerdings nicht). Wenn das dem Client präsentierte Zertifikat mit dem Zertifikat übereinstimmt, das dem Notary-Dienst für den jeweiligen Domainnamen bekannt ist, kann der Client davon ausgehen, dass gerade kein MitM-Angriff durchgeführt wird. Andernfalls kann der Nutzer entsprechend gewarnt werden.

Das Prinzip eines Notary-Dienstes wurde bereits zuvor von „Perspectives“ [WAP08] und „Convergence“ umgesetzt, zwei Add-Ons für den Firefox-Browser.³⁸ Der ICSI-Notary-Dienst unterscheidet sich von diesen Angeboten in Bezug auf den Zugriff: Bei Perspectives und Convergence erfolgt die Kommunikation mit den Notary-Diensten per HTTPS, d. h. die Betreiber der Notare können anhand der IP-Adresse des Clients und der angefragten Server-Zertifikate Rückschlüsse auf das Benutzerverhalten ziehen.

Diese Beobachtungsmöglichkeit gibt es beim ICSI-Notary-Dienst hingegen nicht, da er mittels DNS-Anfragen abgefragt wird, die über einen rekursiven Nameserver an den Dienst übermittelt werden. Die Informationen zu den Server-Zertifikaten werden in einem autoritativen Nameserver, der für die Domain *notary.icsi.berkeley.edu* zuständig ist, vorgehalten. Die Abfrage erfolgt ähnlich wie bei den Hash-basierten Blocklisten: Der SHA1-Hashwert (Fingerprint) des Zertifikats wird als Label dem Domainnamen des Dienstes vorangestellt und mittels einer DNS-Anfrage über den rekursiven Nameserver an den autoritativen Nameserver übermittelt. Die Antwort gibt Auskunft darüber, seit wann das entsprechende Zertifikat dem Notary-Dienst bekannt ist und ob sich eine Vertrauenskette zu einem der Root-Zertifikate herstellen lässt, die im Zertifikatsspeicher des Firefox-Browsers enthalten sind. Das nachfolgende Beispiel (angelehnt an das Beispiel unter <http://notary.icsi.berkeley.edu/>) verdeutlicht den Ablauf und die bei der Abfrage übertragenen Inhalte.

Beispiel 2.11. Ein Client möchte mittels des ICSI-Notary-Dienstes das Server-Zertifikat für den Domainnamen *svs.informatik.uni-hamburg.de* authentifizieren. Er stellt daher mit einem Browser eine HTTPS-Verbindung zu diesem Webserver her, um das Server-Zertifikat abzurufen und den Fingerprint zu ermitteln. Alternativ kann dazu das Programm *openssl* verwendet werden:

```
$ openssl s_client -connect sv.s.informatik.uni-hamburg.de:443 \
    < /dev/null 2> /dev/null openssl x509 -outform DER | openssl sha1
```

Der Client erhält dadurch den Fingerprint *4fc6a7cdecf5b64224f7765704a36c41d27e750c* (Stand: Januar 2014). Der Hashwert wird mit dem Domainnamen des Notary-Dienstes konkateniert und somit der Notary-Dienst wie folgt angefragt:

```
$ dig +short txt \
    4fc6a7cdecf5b64224f7765704a36c41d27e750c.notary.icsi.berkeley.edu
"version=1 first_seen=15650 last_seen=15650 times_seen=1 validated=1"
```

In der Antwort sind die Anzahl der Sichtungen („times_seen“) sowie die Zeitpunkte der ersten („first_seen“) und letzten Sichtung („last_seen“) dieses Zertifikats enthalten. Die Zeitpunkte werden durch die Anzahl der Tage seit dem 1.1.1970 angegeben; der Wert 15650 entspricht dem 6. November 2012. Die Angabe von „validated=1“ besagt, dass sich bei der letzten Sichtung eine Vertrauenskette zu einem Root-Zertifikat herstellen ließ. □

³⁸Homepages: <http://perspectives-project.org/> bzw. <http://convergence.io/>

Beobachtungsmöglichkeiten Im Gegensatz zu Perspectives und Collusion ist es den Betreibern des ICSI-Notary-Dienstes nicht möglich, das Verhalten einzelner Nutzer zu beobachten. Die Betreiber der **autoritativen Nameserver** des ICSI-Notary-Dienstes wissen zwar, zu welcher Domain der angefragte Fingerprint gehört, sie können jedoch – zumindest falls ein rekursiver Nameserver zur Abfrage verwendet wird – nicht die IP-Adresse des anfragenden Clients feststellen.

Der **rekursive Nameserver** kennt zwar die IP-Adresse des Clients, er kann jedoch anhand des Fingerprints nicht ohne weiteres den Domainnamen des zugehörigen Zertifikats ermitteln. Um den Domainnamen zu einem vorliegenden Fingerprint zu ermitteln, müsste sich der Betreiber probeweise mit allen Webservern verbinden, die ein Server-Zertifikat einsetzen, das Zertifikat herunterladen und den Fingerprint auf Übereinstimmung mit dem vorliegenden Fingerprint prüfen.

Der Aufwand, zu einem Fingerprint den Domainnamen zu ermitteln, sinkt allerdings erheblich, wenn der Betreiber des rekursiven Nameservers auf einen Datensatz zurückgreift, in dem alle im Internet verwendeten Server-Zertifikate enthalten sind. Einen solchen Datensatz stellt etwa das Unternehmen Rapid7 (u. a. Entwickler der Software „Metasploit“) unter <https://scans.io> zu Forschungszwecken zur Verfügung [Moo13].

2.10 Fazit

In diesem Kapitel wurden Aufbau und Funktionsweise des DNS erläutert sowie die in der Dissertation verwendete Terminologie eingeführt. Dazu wurde zunächst der hierarchisch organisierte Namensraum vorgestellt. Anschließend wurden die wesentlichen Komponenten (Stub-Resolver, rekursive und autoritative Nameserver) charakterisiert. Weiterhin wurden das DNS-Anfrage-Protokoll und das Caching-Konzept beschrieben. Zum Abschluss wurden aktuelle Entwicklungen, welche die Nutzungsweise des DNS beeinflussen, identifiziert und neue DNS-Anwendungen vorgestellt.

Anhand der vorstehenden Darstellungen wird die Motivation für die fokussierte Formulierung von Forschungsfrage 1, *„Welche Beobachtungsmöglichkeiten existieren grundsätzlich im DNS und auf den rekursiven Nameservern im Besonderen? Welche Informationen lassen sich durch die Beobachtung der zur Adressauflösung dienenden DNS-Anfragen eines Nutzers gewinnen (Monitoring)?“*, deutlich:

- In der Dissertation werden vordergründig die Beobachtungsmöglichkeiten untersucht, die sich aus der ursprünglichen DNS-Anwendung, der **Auflösung von Domainnamen in IP-Adressen (Adressauflösung)**, ergeben. Eine Beurteilung von Ausmaß und Auswirkung der Beobachtungsmöglichkeiten, die sich durch die geplante Einführung der in diesem Kapitel beschriebenen DNS-Anwendungen (u. a. der ONS-Dienst und der ENUM-Verzeichnisdienst) ergeben, ist derzeit noch nicht möglich und bleibt somit zukünftigen Arbeiten vorbehalten.

- In der Dissertation werden vordergründig die Beobachtungsmöglichkeiten untersucht, die sich durch die Auswertung der DNS-Anfragen einzelner Nutzer ergeben. Über diese Möglichkeit verfügen insbesondere die rekursiven Nameserver, die bei jeder DNS-Anfrage sowohl vom angefragten Domainnamen als auch von der Absender-IP-Adresse des Nutzers Kenntnis erlangen. Die autoritativen Nameserver sehen bei eingehenden Anfragen hingegen lediglich die IP-Adresse eines rekursiven Nameservers. Da typischerweise mehrere Nutzer denselben rekursiven Nameserver verwenden, ist eine Beobachtung des Verhaltens einzelner Nutzer auf den autoritativen Nameservern nicht möglich. In der Dissertation liegt der Fokus daher auf den **Beobachtungsmöglichkeiten auf den rekursiven Nameservern**. Über dieselben Beobachtungsmöglichkeiten verfügen Beobachter auf der Kommunikationsstrecke zwischen dem Nutzer und dem rekursiven Nameserver; diese werden aus Gründen der Übersichtlichkeit im Folgenden jedoch nicht mehr explizit erwähnt, wenn von den Beobachtungsmöglichkeiten der rekursiven Nameserver gesprochen wird.

Die in diesem Kapitel identifizierten Trends deuten darauf hin, dass die Beobachtungsmöglichkeiten auf den rekursiven Nameservern zukünftig an Bedeutung gewinnen werden: Inzwischen stehen geeignete Techniken und Software-Komponenten zur Aufzeichnung und Auswertung des DNS-Datenverkehrs zur Verfügung. Erfassung, Speicherung und Auswertung des DNS-Datenverkehrs sind daher für den Betreiber eines rekursiven Nameservers **mit geringem Aufwand möglich**.

Zum anderen ist der Trend erkennbar, dass Nutzer zur Namensauflösung zunehmend auf **Drittanbieter** zurückgreifen. Anstelle des rekursiven Nameservers des Internetzugangsanbieters verwenden sie **rekursive Nameserver von Drittanbietern**, etwa Google oder OpenDNS. Der Wechsel zu einem Drittanbieter ist einerseits durch die geringe Dienstqualität (lange Wartezeiten und häufige Server-Ausfälle) der rekursiven Nameserver der Internetzugangsanbieter motiviert. Darüber hinaus wird auf den rekursiven Nameservern der Internetzugangsanbieter zunehmend bewusst in die Namensauflösung eingegriffen, etwa um sog. Internet-Sperren zu realisieren oder um den Nutzer bei Eingabe eines ungültigen Domainnamens auf werbefinanzierte Suchseiten umzuleiten.

Im Hinblick auf den **Schutz der Privatsphäre** der Nutzer ist die Inanspruchnahme eines Drittanbieters kritisch zu bewerten. Bei der Untersuchung der Monitoring-Möglichkeiten in Kapitel 4 wird sich zeigen, dass der Anbieter anhand der DNS-Anfragen u. a. das Web-Nutzungsverhalten seiner Nutzer rekonstruieren kann, und die Ergebnisse der Untersuchung der Tracking-Möglichkeiten deuten darauf hin, dass eine Beobachtung auch über längere Zeiträume möglich ist (s. Kapitel 6). Auf Anwendungsmöglichkeiten im Bereich der **IT-Forensik** wird in Abschnitt 4.1 eingegangen. Im nächsten Kapitel folgt ein Überblick über die Bedrohungen und die existierenden Sicherheitsmechanismen im DNS.

Beobachtungsmöglichkeiten im Domain Name System
Angriffe auf die Privatsphäre und Techniken zum
Selbstdatenschutz

Herrmann, D.

2016, XIV, 490 S. 67 Abb. in Farbe., Softcover

ISBN: 978-3-658-13262-0