- Der DNS-Server befragt den Cache (bzw. Datenbank im Falle autoritativer Server) anderer fest definierter DNS-Server.
- Der DNS-Server führt eine sukzessive Auflösung beginnend bei DNS-Rootservern durch

Dabei wird jede erhaltene Antwort im lokalen Cache für einige Tage gespeichert, um die Anfragen möglichst direkt beantworten zu können.

b) iterative Anfragen: Der Client befragt der Reihe nach weitere Server, wenn er keine Antwort erhält.

Effizienz. Aus Effizienzgründen sollte ein Endsystem meist den topologisch nahegelegensten DNS-Server zur Adressauflösung verwenden.

Transport. Der Transport von DNS-Anfragen wird über UDP abgewickelt. Der mit TCP verbundene Overhead kann vermieden werden, da die Sicherungsmechanismen von TCP nicht benötigt werden.

Hierarchie. Das DNS ist in einer Baumstruktur organisiert.

4 Transportschicht (Transport Layer)

4.1 User Datagram Protocol

4.2 Transmission Control Protocol

Charakteristika. Eine TCP-Verbindung stellt einen Duplex-Kanal zwischen Sender und Empfänger bereit. Sie stellt einen bytestromorientierten Dienst zur Verfügung. TCP nutzt verschiedene Mechanismen um eine zuverlässige Verbindung herzustellen:

- Quittungen zur Entdeckung verlorener Pakete
- Sequenznummern um Reihenfolge zu gewährleisten
- Prüfsumme um Integrität der Daten zu garantieren

Bezeichnung	Länge [bit]	Beschreibung
Quellport	16	
Zielport	16	
Sequenznummer	32	gemessen in Byte
Quittung	32	nächste vom Empfänger erwartete Sequenznummer
Offset		Anzahl der 32-Bit-Wörter im TCP-Header
reserviert		
URG	1	gesetzt falls Urgent-Pointer verwendet
ACK	1	unterscheidet bei SYN=1 TConReq von TConCnf, zeigt Quittung an
PSH	1	übergebene Daten sofort weiterleiten
RST	1	Zurücksetzen der Verbindung
SYN	1	zeigt TConReq oder TConCnf an
FIN	1	Sender möchte keine Daten mehr senden
Empfangsfenster	16	Fenstergröße für Sender in Byte zur Flusskontrolle
Prüfsumme	16	über TCP-Kopf und Daten
Urgent Pointer	16	Zeiger auf wichtige Daten
Optionen	$k \cdot 32$	
Daten		

Table 4: Felder einer TCP-Dateneinheit

TCP-Dateneinheit.

Verbindungsaufbau (3-Way Handshake). Der Verbindungsaufbau wird vom Client initiiert.

1. TConReq(SYN=1, seq=client-isn) →
2. ← TConCnf(SYN=1, ACK=1, seq=server-isn, ack=client-isn + 1)
3. ACK(SYN=0, ACK=1, seq=client-isn + 1, ack=server-isn + 1) →

Verbindungsabbau. Bei TCP wird wenn möglicher ein expliziter Verbindungsabbau durchgeführt, statt die Verbindung per Reset zurückzusetzen, da zum Zeitpunkt des Verbindungsabbaus noch Daten unterwegs sein können. Bei einem Reset wäre es nicht unbedingt erkennbar, welche Daten noch korrekt empfangen wurden. Der Verbindungsabbau kann sowohl vom Client als auch vom Server initiiert werden. Nach dem letzten ACK wird noch gewartet, bevor der lokaler Kontext gelöscht wird, da noch Pakete verlorengehen könnten.

1. FIN →
2. ← ACK