IT-Themenzusammenfassung

Dieses Markdown-Dokument soll verschiedene IT-Themen zusammenfassen, die ich hin und wieder angeschaut habe.

Disclaimer

Dieses Markdown-Dokument wurde in VScode mit diveresen Plugins verfasst. Das bedeutet, dass hier nicht 100% auf Markdown gesetzt wird. **Es kann vorkommen, dass Elemente anders oder gar nicht angezeigt werden, wird VScode mit den entsprechenden Plugins nicht benutzt.** Ich versuche mich jedoch daran zu halten, ausschließlich Markdown zu benutzen. Es kann ebenfalls dazu kommen, dass **HTML-Elemente** wie `<sup>` oder auch `<sub>` eingebaut werden, um Fußnoten darzustellen! Außerdem werden für Diagramme **mermaid** benutzt! Eine aktuelle PDF-Kopie ist immer beigelegt!

Benutze VSCode Plugins

- Markdown All in One
- Markdown PDF
- Markdown Preview Mermaid Support
- Markdownlint

Inhaltsverzeichnis

APN - Access Point Name

"*APN*" steht für "**A**ccess **P**oint **N**ame" - nicht zu verwechseln mit einem Access-Point als Netzwerkgerät. Ein APN ist eine Art "*Gateway*" eines Mobilfunkbetreibers, welches ein Netzwerk für das jeweilige mobile Endgerät zur Verfügung stellt. Mithilfe dieses Netzwerks kann ein mobiles Endgerät das **öffentliche Netzwerk** (Internet) erreichen. Ein APN dient jedoch nicht nur als Gateway, sondern setzt weitere verschiedene Eigenschaften für das jeweilige mobile Endgerät wie zum Beispiel wie es sich am **Netzwerk authentifiziert**, welche **Kommunikationswege** es benutzt als auch welche **Sicherheitsmaßnahmen** für das mobile Endgerät greifen.

Mithilfe eines APNs ist es möglich verschiedene Endgeräte als auch IoT-Geräte zu administrieren und **voneinander zu trennen**. Um diese Geräte jedoch in ein APN-Netzwerk aufnehmen zu können, muss als aller erstes ein APN angefordert / gekauft werden. Ein solcher APN wird i.d.R. bei einem **Mobilfunkanbieter** gemietet. Dabei gibt es jedoch ein paar Dinge wie folgend zu beachten:

- Für jedes neue APN-Netzwerk muss dieses beim Mobilfunkanbieter gemietet werden.
- Die APNs werden (meist) monatlich abgerechnet.
- Jedes mobile Endgerät oder IoT-Gerät erhält meist eine SIM-Karte, auf welcher der APN bereits fest eingetragen ist.
    - Oft kommt es aber auch vor, dass der APN in den LTE-Einstellungen des Geräts eingegeben werden muss, damit sich das Gerät mit dem Mobilfunkanbieter verbindet (siehe Telekom).
- Jedes mobile Endgerät oder IoT-Gerät innerhalb eines APN-Netzwerks muss den gleichen Mobilfunkanbieter besitzen. Es ist nur unter Umständen möglich, dass mobile Endgeräte zu verschiedenen APN-Netzen eine Verbindung aufbauen können.
    - Ist der Mobilfunkanbieter im Ausland nicht unter gleichem Namen gelistet, wird der entsprechende, zum Unternehmen gehörende APN-Übergangspunkt für die Einwahl benutzt. Existiert der Mobilfunkanbieter gar nicht, so versucht sich die SIM-Karte in das nächst beste Netzwerk / Parnternetzwerk einzuwählen.

Wie bereits erwähnt ist es möglich, dass ein Gerät zwar mehrere verschiedene APNs von verschiedenen Mobilfunkanbietern besitzen kann, dies aber nur mit erhöhten Kosten möglich ist. Dafür gibt es zwei Möglichkeiten, um dieses Vorhaben umzusetzen:

1. Die **preisgünstigere** Variante ist es, dass der Mobilfunkanbieter mehrere APN-Profile auf seiner SIM-Karte hinterlegt (oder entsprechend in der Software). Somit können jedoch nur die vorgefertigten APN-

Profile für andere Mobilfunkanbieter benutzt werden.
2. Die **teurere** Variante ist es eine E-SIM zu benutzen, welche ohne physische SIM-Karte mehrere APN-Profile innerhalb eines Speicherchips abspeichern kann.

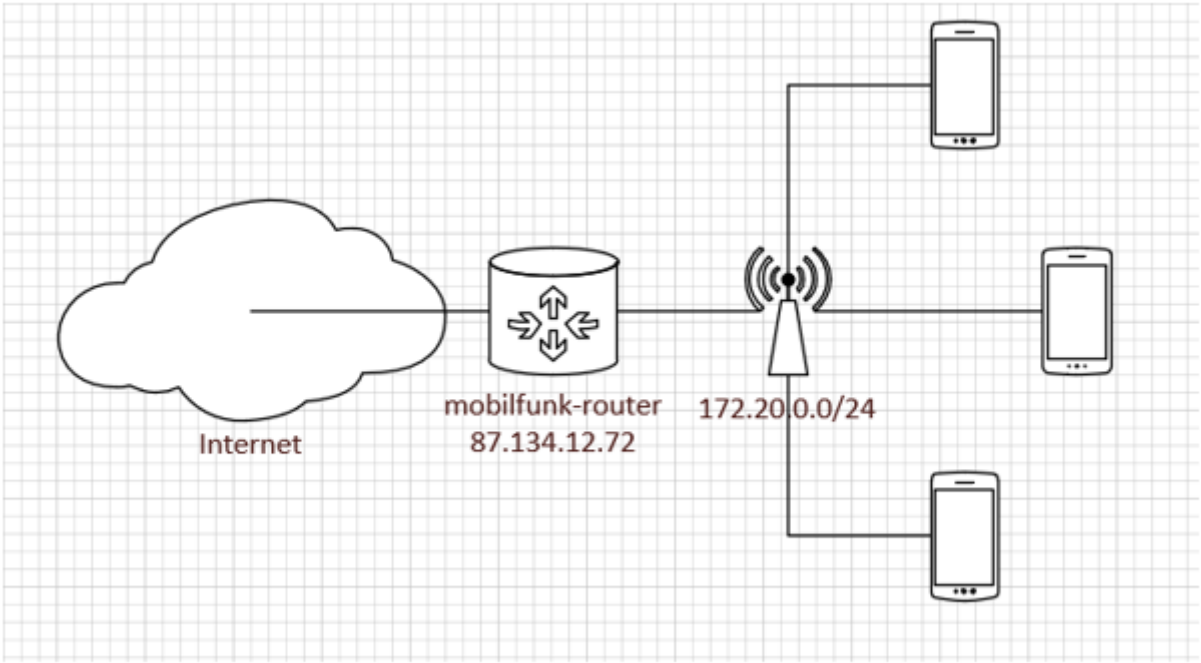
Kommunikationswege der APN-Typen

Vor der Mietung eines APNs sollte man im Klaren sein, welche Art von APN man benutzen möchte. Dabei gibt es zwei Arten mit jeweils statischer und dynamischer IP-Adressvergabe.

Öffentlicher APN

Wird ein öffentliches APN-Netzwerk benutzt, dann können mobile End- und IoT-Geräte **direkt mit dem Internet** mit Umweg über den Mobilfunkanbieter kommunizieren. Ist die IP-Adressvergabe dynamisch konfiguriert, so holt sich das jeweilige zu verbindende Endgerät entsprechend nach DHCP eine freie IP-Adresse aus einem verfügbaren Pool und benutzt diese so lange, bis das Endgerät die Verbindung abbaut und somit die Session schließt (oftmals auch im entsprechend Management-Portal konfigurierbar).

Dabei gibt das mobile End- oder IoT-Gerät die IP-Adresse wieder zurück an den Adresspool. Bei einem neuen Verbindungsaufbau wird eine neue freie IP-Adresse aus dem vorhandenen Adresspool benutzt. Ist das APN-Netzwerk statisch konfiguriert, so besitzt jedes mobile End- oder IoT-Gerät seine eigene statische Adresse innerhalb des Netzwerk. Diese wird für jede Verbindung mit dem Internet benutzt.

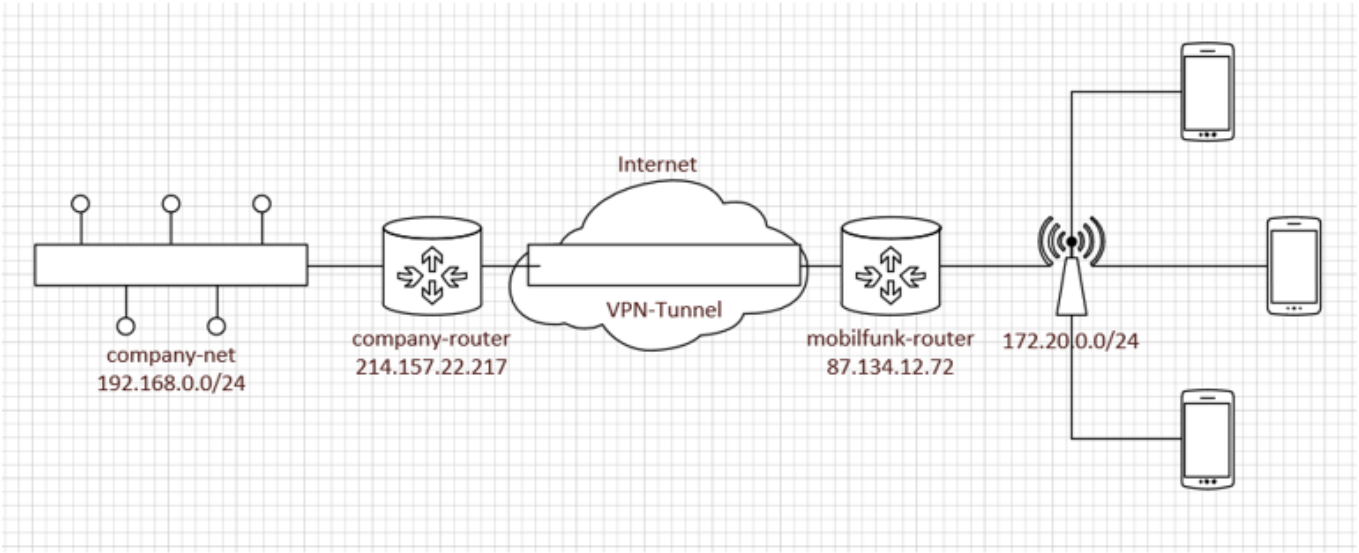


Privater APN

Im Vergleich zu einem öffentlichen APN-Netzwerk, welches die Daten der mobilen End- und IoT-Geräte **in das Internet routen**, macht das private APN-Netzwerk **dies nicht unbedingt**. Grundsätzlich ist es möglich auch diesen Datenverkehr in das Internet zu routen, jedoch kann man bei einem privaten APN mehrere **Sicherheitseinstellungen** einstellen als bei einem öffentlichen APN.

Meist werden private APNs jedoch dafür benutzt, um den **Datenverkehr mittels VPN** zum eigenen (Unternehmens-) Netzwerk zu routen. So ist es möglich, dass Dienste innerhalb eines privaten APN-Netzwerks mit Diensten innerhalb des eigenen (Unternehmens-) Netzwerk kommunizieren können. Damit diese Dienste

miteinander kommunizieren können, wird klassischerweise ein VPN-Tunnel vom Mobilfunkanbieter zum eigenen (Unternehmens-) Netzwerk erstellt. Die IP-Adressen innerhalb eines privaten APN-Netzwerks können ebenfalls **statisch** oder **dynamisch** verteilt werden.



APN vs. VPN

Es kommt hin und wieder vor, ass ein APN mit einem VPN verwechselt oder verglichen wird. Dies ist nicht möglich, da die Aufgaben von einem APN und eines VPNs **sehr unterschiedlich** sind. ***Während der APN die Verbindung für ein mobiles End- und IoT-Gerät in das Netzwerk des Mobilfunkanbieters herstellt, stellt ein VPN einen Tunnel zwischen zwei Standorten her.*** Zudem verschlüsselt ein VPN den Datenverkehr innerhalb eines VPN-Tunnels, während ein APN keine Verschlüsselung verwendet,

Cisco - Passwort vs. Secret

Standardmäßig sind Cisco-Geräte wie Router oder Switches ***nicht*** passwortgeschützt. Das bedeutet, dass ein Zugriff auf das Gerät als auch in den erhöhten EXEC-Modus **ohne Authentifizierung** stattfinden kann. Da dies jedoch ein **enormes Sicherheitsrisiko** darstellt, werden Passwörter benutzt, um diese Bereiche abzusichern. Bei Cisco-Geräten gibt es zwei Methoden, um Passwörter zu setzen:

1. Mittels dem `enable password`-Kommando
2. Mittels dem `enable secret` -Kommando

Doch es gibt einen großen Unterschied, welcher im Nachfolgenden erklärt wird.

Das Password-Kommando

Zu Beginn wurden Cisco-Geräte **ausschließlich** mit dem Passwort-Kommando hergestellt. Dieses Kommando setzt ein Passwort für die verschiedenen zu sichernden Bereiche (1-15). Das Problem ist jedoch, dass dieses im **Klartext** in die Running / Startup-Config geschrieben wird. Somit kann es einfach ausgelesen werden.

Damit dies verhindert wird, hat Cisco **Passwortsicherheitsypen** als auch den Befehl `service password-encryption` eingeführt. Mit dem Kommando `service password-encryption` werden alle im Klartext gespeicherten Kennwörter **verschlüsselt**. Dies ist zwar im Vergleich zum vorherigen Verfahren eine gute Verbesserung, da dieses mit einer einfachen **Vigenère-Verschlüsselung** verschlüsselt wurde. Heutzutage existieren jedoch einige Programme, die eine solche Verschlüsselung **brechen** können. Damit das Risiko für

das Auslesen oder der Entschlüsselung des Passworts gemindert werden kann, wurde das `secret`-Kommando eingeführt.

Das Secret-Kommando

Das **Secret**-Kommando löst das ursprüngliche **Password**-Kommando ab. Aus historischen Gründen und für Entwicklungsumgebungen ist das Passwort-Kommando immer noch im Cisco IOS enthalten. Es wird jedoch ausdrücklich geraten dieses ***nicht mehr einzusetzen***. Stattdessen soll das Secret-Kommando als Passwort benutzt werden. Das Secret-Kommando ist ein zweites Passwort, welches auf dem System hinterlegt wird. **Existiert** bereits ein Passwort, welches durch das **Passwort-Kommando** gesetzt wurde, wird dieses vom **Secret-Kommando abgelöst**. Wie beim Passwort-Kommando gibt es auch beim Secret-Kommando unterschiedliche **Passwortsicherheitstypen.** Zu benutzende Sicherheitstypen bei Cisco sind:

- **Typ 5** → MD5-Hash (***nicht empfohlen***)
- **Typ 8** → PBKDF2 ()
- **Typ 9** → SCRYPT (**empfohlen**)

Secret-Verschlüsselungstypen erklärt

Generell wird **abgeraten** andere Verschlüsselungstypen, die **nicht** den oben genannten entsprechen, für Cisco-Geräte **weiter einzusetzen**. **MD5** wird zwar mit einem **zusätzlichem Salt**[1] versehen, ist dennoch weiterhin ein schnell zu berechenbarer Hash-Algorithmus. Auch hier gibt es bereits ein paar Programme, welche es schaffen MD5-Hashes umzukehren.

Deswegen und daher, dass MD5 bereits im Jahr **2008** von der IETF (Internet Engineering Task Force) als **unsicher** begutachtet wurde, wird auch davon **abgeraten**, MD5 weiter als Passwort-Hash einzusetzen. MD5 sollte **nur dann** eingesetzt werden, wenn es **keinen weiteren verfügbaren Hashalgorithmus** zur Verfügung steht. Auch beim sichereren PBKDF2-Hashalgorithmus scheiden sich in der Sicherheit der Verwendung die Gemüter, da es, je nach **Stärke der Verschlüsselung** schneller oder langsamer mittels eines **Bruteforce-Angriffs** entschlüsselt werden kann. Daher wird auch davon abgeraten PBKDF2 zu benutzen.

Als letzte Option bleibt daher **SCRYPT**. SCRYPT ist die empfohlene Variante, um Passwörter **sicher** zu verschlüsseln. Zwar ist kein Algorithmus vor einer Bruteforce-Attacke sicher, jedoch macht SCRYPT es einem Angreifer besonders schwer an das Passwort zu gelangen. Denn SCRYPT sorgt mit seiner Berechnung dafür, dass eine hohe Menge an Hardware-Ressourcen benötigt werden, um überhaupt eine Chance zu besitzen die Verschlüsselung brechen zu können.

DAHER WIRD DAZU GERATEN ALLE ALTEN UND NEUEN PASSWÖRTER MITTELS SCRYPT ZU VERSCHLÜSSELN.

- [1] Ein bestimmter oder unbestimmter Wert, welcher als zusätzlicher Sicherheitsfaktor mit in die Passwortverschlüsselung genommen wird.

Quellen

- https://community.cisco.com/t5/networking-documents/understanding-the-differences-between-the-cisco-password-secret/ta-p/3163238
- https://learningnetwork.cisco.com/s/article/cisco-routers-password-types
- https://de.wikipedia.org/wiki/Scrypt

- https://security.stackexchange.com/questions/19906/is-md5-considered-insecure
- https://www.section.io/engineering-education/what-is-md5/
- https://www.oreilly.com/library/view/hardening-cisco-routers/0596001665/ch04.html

DNS - Domain Name System

DNS - Quick Facts

- **Osi-Schicht:** 7
- **Ports:**
  - 53 (TCP / UDP)
  - 853 TLS - TCP
  - 853 DTLS - UDP
- **Standards:**
  - RFC 1034
  - RFC 1035

Einleitung - Was ist DNS?

Das **D**omain **N**ame **S**ystem (DNS) ist ein integraler Bestandteil des Internets. Ohne diesen würde das Internet so in der heutigen Form nicht funktionieren. DNS beschreibt die Übersetzung eines Namens wie z.B. `www.wikipedia.org` in die zugehörige IPv4 / IPv6 Adresse `91.198.174.192` / `2620:0:862:ed1a::1`. Um *DNS* auszuführen werden entsprechende DNS-Server eingesetzt. Von diesen gibt es weltweit **mehrere tausende** und sie besitzen unterschiedlichste Funktionen und Aufgaben.

Die Zeit vor dem DNS

Bevor es das DNS gab, zu Zeiten des ARPANETs (Vorgänger des Internets) wurden, wurden Namensauflösungen von **Webseiten** oder einfachen **Computernamen** anhand der sogenannten **Host**-Datei aufgelöst. Die Host-Datei ist eine Datei, welche -auch heute noch- auf dem Betriebssystem des Computers bzw. eines zentralen Computers abgelegt ist. Sie enthält alle Einträge, welche ein Systemadministrator für wichtig hielt einzutragen.

Und genau hier liegt der Knackpunkt. Denn mit der Host-Datei kommen folgende Probleme auf:

- Ein **Einzelner** konnte bestimmen, welche Einträge in der Host-Datei existieren und welche nicht → **Kein Schutz vor Manipulation**.
- Es gab keine zentrale Verwaltung der unterschiedlichen Namens-Zuordnungen.
- Bei **steigender** Hostanzahl wurde die **manuelle** Administration deutlich schwieriger.
  - Dadurch wurde die Aktualität der Host-Datei gefährdet.
- Die Host-Datei wurde **manuell** gepflegt. Es gab keine automatisch anpassbaren Lösungen.

Diese Probleme zeigten schnell Grenzen der Benutzung der lokalen Host-Datei auf, weshalb ein neues, (de-) zentrales System mit einer besseren Namensordnung entwickelt werden musste. Dies ist als das heutige **Domain Name System** bekannt.

Die Domain

Ein integraler Bestandteil des Domain Name System ist natürlich die **Domain**. Eine Domain ist ein Bereich, welcher Computernamen verwaltet und diese entsprechend hierarchisch gliedert. Ein wichtiger Bestandsteil

einer Domain ist der sogenannte **Domain-Name**.

Der Domain-Name wird dazu verwendet, um bestimmte Computer innerhalb des Domain-Namen zugeordneten Bereichs zu identifizieren. Sie wird ebenfalls im sogenannten Uniform Resource Locator (URL) verwendet, welcher eine **einheitliche Angabeform für Ressourcen** innerhalb eines Netzwerks darstellt. Ein URL beginnt meistens mit dem Dienst, gefolgt von den unterschiedlichen Domain-Hierarchien und endet symbolisch mit einem Punkt, welcher jedoch weggelassen werden kann. Ein URL teilt sich dementsprechend wie folgt auf: `computername.sub-level-domain.second-level-domain.top-level-domain.` bzw. `dienst.sub-level-domain.second-level-domain.top-level-domain.`. Die `sub-level-domain` ist meist optional und muss nicht angegeben werden, wenn diese nicht existiert. Folglich sind folgende URLs meist anzutreffen:

- `ftp.data.de`
- `https://wikipedia.org`
- `www.wikipedia.org`
- `de.wikipedia.org`

Die hier aufgelisteten Beispiele werden auch als **Fully Qualified Domain Name** (FQDN) bezeichnet.

Bemerke: Bei einer Namensauflösung wird der FQDN von **rechts nach links** gelesen. Für das Beispiel `www.wikipedia.org` würde dies wie folgt aussehen: `.org.wikipedia.www` bzw. `org.wikipedia.www`.

Wie bereits angedeutet gibt es unterschiedliche Domain-Arten. Dabei sind die **Second-Level-Domain** (SLD) und **Top-Level-Domain** (TLD) am bekanntesten und immer verwendet, wenn ein Computername aufgerufen wird.

Top-Level-Domain

Die TLD ist in der DNS-Baum-Hierarchie an zweiter Stelle. Hier eingesetzte Server verwalten Namensbereiche, welche wie "*.de*", "*.org*" oder auch "*.uk*" verwalten. Dabei muss man zwischen zwei Arten von TLDs unterschieden:

1. **ccTLD:** Ausgeschrieben bedeutet das "*Country Code Top-Level-Domain*" und beschreibt Ländercodes, die nach ISO 31166-1 spezifiziert sind.
2. **gTLD:** Ausgeschrieben bedeutet das "*Generic Top-Level-Domain*" und beschreibt generische bzw. geografische TLDs.

Auflistung an ccTLDs:

Domain	Land
at	Österreich
au	Australien
cc	Kokos-Inseln
ch	Schweiz
de	Deutschland
fr	Frankreich

Domain	Land
gb	Groß-Britannien
ie	Irland
it	Italien
li	Lichtenstein
...	...

Auflistung an gTLDs:

Domain	Organisationsform
aero	Lufttransportindustrie
arpa	Alte ARPANET Domäne
biz	Buiseness
com	Kommerzielle Domain
gov	Regierungsstelle der vereinigten Staaten von Amerika
net	Nutzspezifische Angebote und Dienste
org	Nicht kommerzielle Unternehmen, Projekte
...	...

Second-Level-Domain

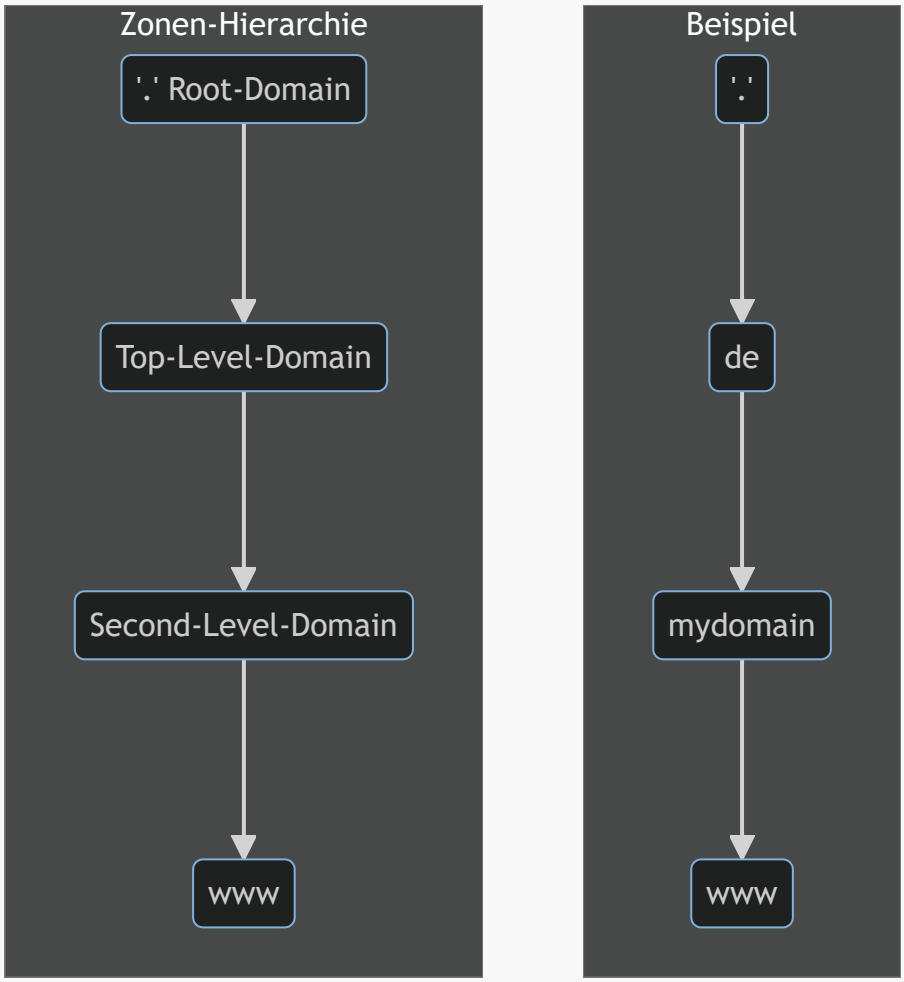
Die SLD ist in der DNS-Baum-Hierarchie an dritter Stelle. Hier existierende Namensbereiche können von jedem frei erfunden und registriert werden. Wichtig ist jedoch, dass eine SLD unterhalb einer TLD **eindeutig** sein muss. Eine Vergabe von doppelten Namen kann aufzulösende Namen durcheinanderbringen.

DNS-Zone

Eine DNS-Zone ist ein **Verwaltungs-** bzw. **Verantwortungsbereich** eines DNS-Servers. In einer Zone befindet sich mindestens ein DNS-Server. Ein DNS-Server kann zudem auch mehrere Zonen besitzen. Innerhalb jeder Zone existiert die **komplette** Datenbasis für einen bestimmten Bereich des Domänen-Namensraums.

Die Daten der DNS-Zone wird in einer sogenannte **(DNS-) Zonendatei** lokal auf dem DNS-Server gespeichert. Dabei handelt es sich um eine Textdatei, welche die Einträge zeilenweise pflegt und nur wenige Megabytes groß ist. Pro Zeile können entweder Kommandos, gekennzeichnet an einem `$`-Symbol, vorkommen, welche Aktionen wie `INCLUDE`-Statements ausführen oder einfache DNS-Einträge (Resource Records).

Die Zonen der Baum-Hierarchie sehen dabei wie folgt aus:



Zoneneinträge

Jeder einfache DNS-Eintrag innerhalb einer DNS-Zone wird als **Resource Record** bezeichnet. Jeder Resource Record bezieht sich auf einen bestimmten **Record-Type**, welcher bestimmte Informationen enthält.

Die Record-Types sind wie folgt:

Record-Type	Eintrag
A	IPv4-Adresse
AAAA	IPv6-Adresse
CNAME	Verweis; Weiterleitung; Alias; Verweis von einem Namen auf einen anderen Namen
MX	Zuständiger Mailserver für die Zone (Mail Exchange)
NS	Zuständiger Nameserver für die Zone
SRV	Server für einen Dienst im Windows-AD
PTR	Weist einer IP-Adresse einen Namen zu. (IPv4 → **IN-ADDR-ARPA**, IPv6 → **IPv6.ARPA**)
TXT	Liefert einen Text zurück
SOA	Ansprechpartner und Parameter zur abgefragten Zone (Start of Authority)

DNS-Cluster

Damit die Zonen-Dateien auf unterschiedlichen Servern in den unterschiedlichen Domain-Level-Bereichen gleich sind und redundant gespeichert werden, werden Server, die einer gleichen Domain zugeordnet sind, in Clustern zusammengeschlossen. Dies erhöht die Redundanz und vermindert die Ausfallswahrscheinlichkeit. Zudem werden die Zonendateien über die jeweiligen Server entsprechend abgeglichen und aktuell gehalten.

DNS-Server-Cluster werden dabei in zwei Gruppen eingeteilt:

1. **Primary:** Hier werden die Zonendateien gespeichert, angepasst und modifiziert. Die Primary-DNS-Server sind zudem jene, welche bei einer Namensauflösung zuerst angesprochen werden. Sie teilen ihre Zonendatei mit anderen, in der gleichen Domain befindlichen Primary-Servern.
2. **Secondary:** Die Secondary-DNS-Server sind dafür da, um die entsprechenden Zonendateien abzuspeichern. Sie dienen ausschließlich als Zonendatei-Backup.

DNS-Server Arten

Den einen DNS-Server gibt es so nicht, denn es gibt unterschiedliche DNS-Server, die unterschiedliche Aufgaben übernehmen. Jedoch ist zwischen zwei Haupt-DNS-Arten zu unterscheiden:

1. Autoritative DNS-Server
2. Nicht-autoritative DNS-Server

Autoritativer DNS-Server

Ein autoritativer DNS-Server speichert alle DNS-Informationen für seine befugte Zone. Das bedeutet, dass dieser die **Endstation** einer DNS-Anfrage ist. **Er besitzt die Informationen, um den jeweiligen DNS-Namen in eine IP-Adresse aufzulösen**. Daher gilt die Kommunikation bzw. die Antwort als **verbindlich** / **gesichert** (autoritativ).

Nicht-autoritativer DNS-Server

Ein nicht-autoritativer DNS-Server ist **nicht selbst** für die DNS-Zone verantwortlich und bedient sich für die Namensauflösung daher an Dritte (DNS-Server) derer DNR-Einträge (Resource Records). DNS-Einträge, welche er aufgelöst hat, werden dabei für eine bestimmte Zeit (**TTL** → Time to Live) im Cache (RAM) gespeichert des DNS-Servers gespeichert. Da sich die Einträge der ursprünglichen Zonendatei in der Zwischenzeit jedoch ändern kann, gelten die abgefragten Informationen als **nicht gesichert** und daher auch als **nicht-autoritativ**.

Bei dieser Vorgehensweise nimmt der DNS-Server die Rolle eines **Resolvers** ein, weshalb dieser auch als **DNS-Resolver** bezeichnet wird.

Weitere DNS-Server Arten

Root-DNS-Server: Der Root-DNS-Server ist ein **autoritativer** Server, welcher sich um die Root-Zone kümmert. Er besitzt alle Einträge für die TLD-Server weltweit. Ohne diesen wäre das DNS-Netzwerk nicht möglich.

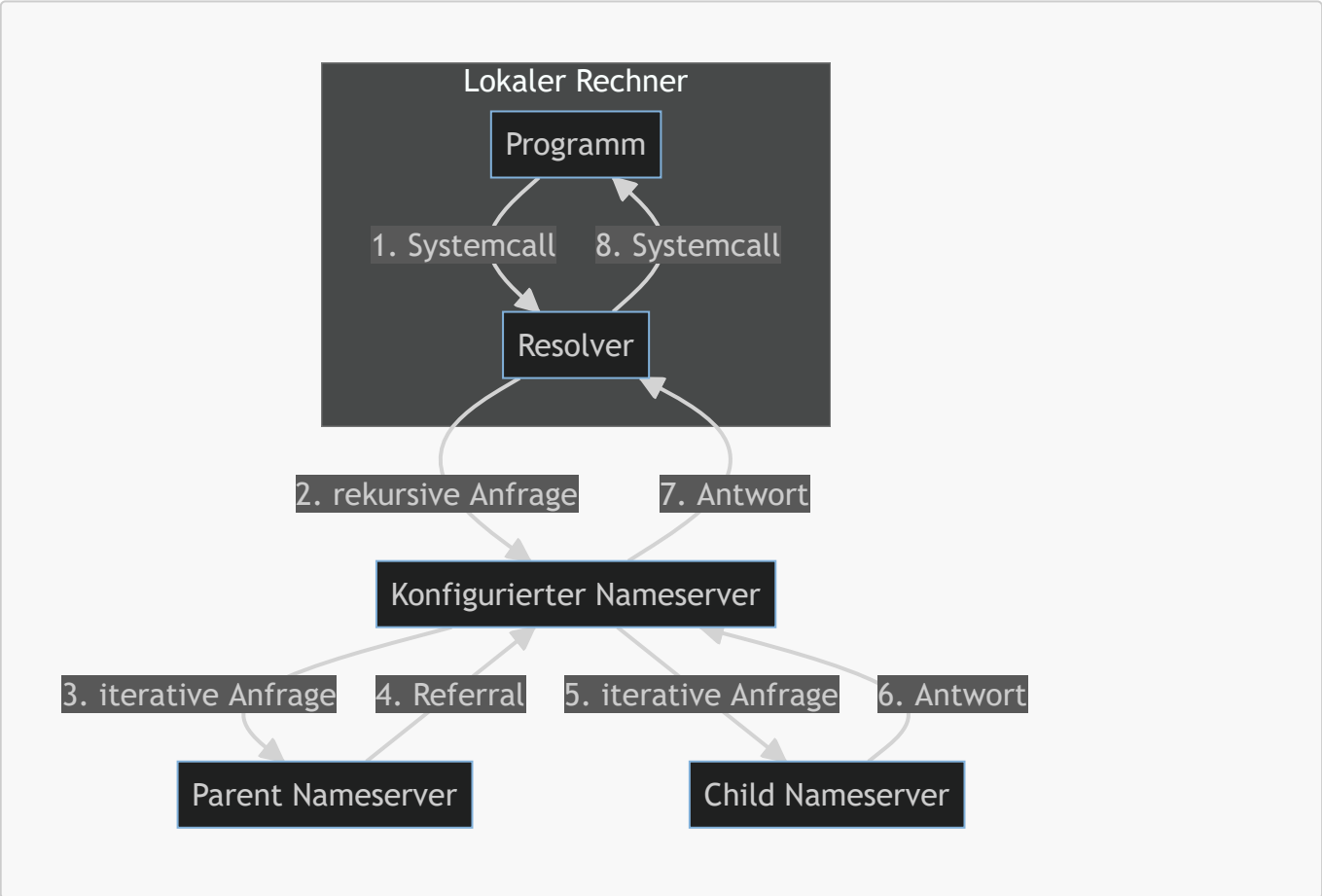
TLD-DNS-Server: Der TLD-DNS-Server ist ebenfalls ein **autoritativer** Server, welcher alle Second-Level-Domain-Server / Namen für seine entsprechende Zone besitzt.

DNS-Resolver

Ein **Resolver** ist per se kein DNS-Server, sondern vielmehr eine **ogrammschnittstelle** (Vermittlungsstelle) für Anwendung und DNS. Die Aufgabe des Resolvers ist es, sich mit den entsprechenden autoritativen DNS-Servern auseinanderzusetzen, um die jeweilige IP-Adresse zu bekommen. Dabei besitzt dieser auch zweit Methoden:

1. **Rekursives Resolving**
    1. Kontaktiert den zugeordneten Nameserver
    2. Wenn dieser die Adresse nicht im Datenbestand hat, fragt der Nameserver weitere Nameserver ab.
    3. Dies geht solange bis entweder der Adressname aufgelöst wird oder eine negative Antwort eines autoritativen DNS-Servers zurückkommt.
2. **Iteratives Resolving**
    1. Entweder bekommt der Resolver den entsprechenden Resource Record direkt vom DNS-Cache mitgeteilt oder er bekommt eine Adresse eines weiteren Nameservers zurück.
    2. Im zweiten Fall würde der Resolver so lange Namerserver nach Nameserver abfragen, bis dieser eine verbindliche Antwort enthält.
    3. Dabei werden die Anfragen meist eine Instanz höher (eine Zone nach oben) weitergeleitet, um die Adresse aufzulösen.

Bemerke: Das Iterative-Verfahren wird kaum für Resolver oder Clients eingesetzt, da diese damit nicht umgehen können. Deshalb wird das Verfahren nur unter DNS-Servern eingesetzt.

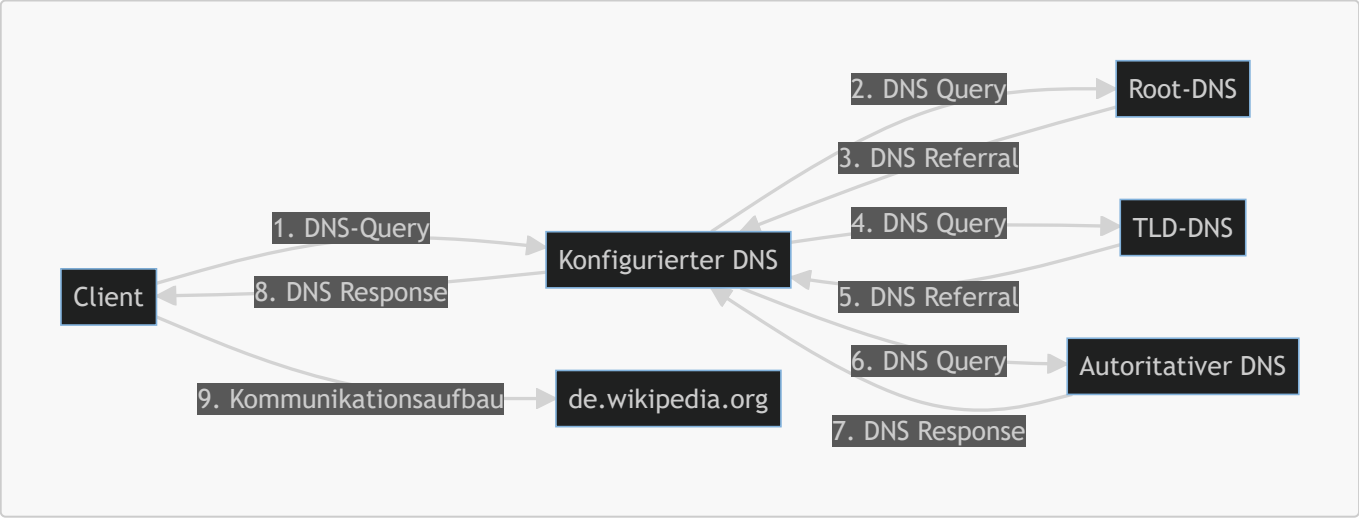


Funktionsweise eines DNS-Lookups

Bemerke: Nachfolgender Ausschnitt wurde größtenteils von Wikipedia übernommen!

Angenommen, ein Rechner X (Zur Einfachheit: Alice) möchte eine Verbindung zu `de.wikipedia.org` (Rechner Y, zur Einfachheit: Bob) aufbauen. Dazu braucht er dessen IP-Adresse. Falls Alice IPv6-fähig ist, läuft der Vorgang zunächst für IPv6 (AAAA-Record) und **sofort** danach für IPv4 (A-Record) ab. Falls am Ende eine IPv6 **und** eine IPv4 Adresse für Rechner Y ermittelt wurde, wird i.d.R. laut der Default Policy RFC 6724 die Kommunikation zwischen Alice und Bob über IPv6 bevorzugt, es sei denn im Betriebssystem oder in den benutzten Anwendungen, wie z.B. ein Webbrowser, wurde dieses Verhalten anders eingestellt.

1. Alice sucht in ihrer lokalen *host*-Datei nach dem Eintrag `de.wikipedia.org`. Falls dieser nicht existiert, fragt Alice bei ihrem konfigurierten DNS-Server (konfigurierter DNS-Server) nach.
2. Hat der DNS-Server die IP-Adresse von Bob zwischengespeichert, antwortet er mit dieser und beendet die Kommunikation. Andernfalls fragt er bei einem Root-Server nach `de.wikipedia.org`.
3. Der Root-Server findet heraus, dass `de.wikipedia.org` zur TLD `.org` gehört und schickt die Adresse des zuständigen Nameservers an den Alices konfigurierten DNS-Servers zurück (NS Resource Record und AAAA oder A Resource Record)
4. Nun fragt der konfigurierte DNS-Server von Alice den zurückbekommenen Nameserver für die `.org`-Zone nach `wikipedia.org` an.
5. Der ".org"-Nameserver findet den Eintrag von "wikipedia.org" und sendet den autoritativen Nameserver mit IP-Adresse zurück an den DNS von Rechner X.
6. Anschließend fragt der konfigurierte DNS-Server von Alice den autoritativen Nameserver von `wikipedia.org` nach `de.wikipedia.org`.
7. Der autoritative Nameserver von `wikipedia.org` schickt die IP-Adresse für `de.wikipedia.org` zurück an den konfigurierten DNS-Server von Alice.
8. Der konfigurierte DNS-Server von Alice sendet anschließend an Alice.
9. Alice kann anschließend mit Bob (`de.wikipedia.org`) eine Kommunikation aufbauen.



Sichtbarkeit

DNS-Server können einerseits **privat** andererseits **öffentlich** betrieben werden. Im folgenden werden die Unterschiede gezeigt:

Public DNS	Private DNS
Öffentlich zugänglich	Nur für private Nutzer eines (Unternehmens-) Netzwerks

Public DNS	Private DNS
Wird durch keine Firewall blockiert	Ist in einem internen (Unternehmens-) Netzwerk. Befindet sich hinter einer Firewall
Wird mittels öffentlichen IP-Adressen angesprochen	Besitzt private IP-Adresse
Lösen öffentliche Adressen auf	Lösen private und öffentliche Adressen auf

Angriffsvektoren

Nachfolgend werden zwei Angriffszenarien gezeigt. Es gibt natürlich noch weitere Angriffsmöglichkeiten, als hier dargestellt.

Distributed Denial of Service (DDoS)

Hier werden meist Botnetzwerke benutzt, um einen DNS-Server mit vielen Anfragen zu fluten und die **Systemressourcen stark zu belasten**, bis dieser, im (aus Angreifersicht) Best-Case abstürzt. Zwar sind vor allem Root-DNS-Server sehr performante DNS-Server, können jedoch bei einem großflächigem Angriff temporär ausfallen. Daher ist es wichtig DNS-Server in **Cluster** zu stecken, um die **Ausfallsicherheit** zu erhöhen.

Ein DDoS-Angriff kann ebenfalls in Verbindung mit anderen Angriffen erfolgen.

DNS-Spoofing - Cache Poisoning

Cache Poisoning beschreibt ein Vorgehen, bei welchem der Angreifer versucht dem DNS-Server eine **gefälschte IP-Adresse** für einen DNS-Namen unterzubringen. Dadurch können DNS-Anfragen von unwissenden Clients auf die entsprechende Adresse auf das **Ziel des Hackers** umgeleitet werden. Dieser kann z.B. eine entsprechende Webseite so präparieren, sodass diese täuschend echt zur Originalseite ist. Hier geben Unwissentliche ihre **geheimen Informationen** an, die der Hacker anschließend missbrauchen kann.

Ein möglicher Ablauf von Cache-Poisoning:

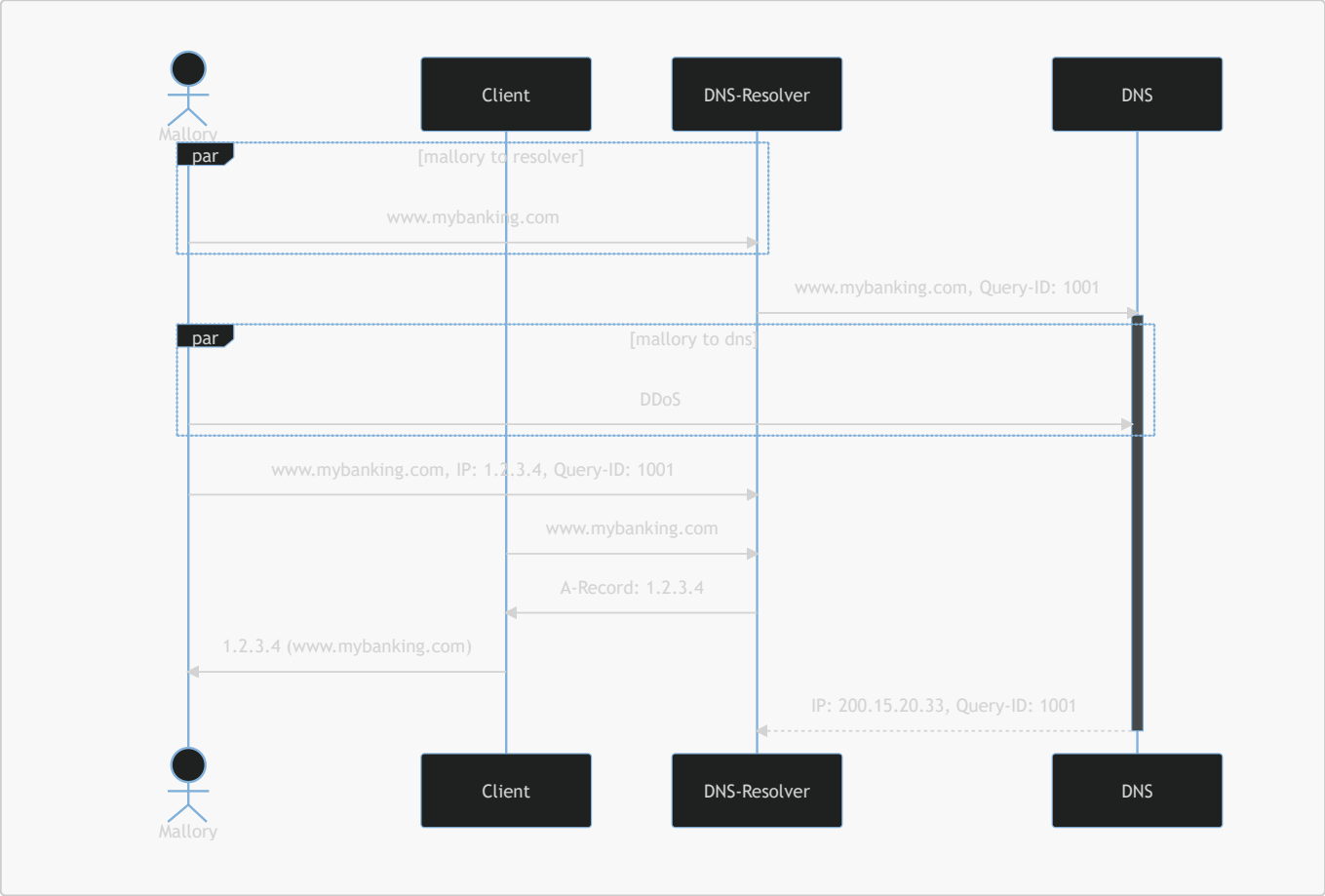
Der Angreifer sendet eine DNS-Anfrage an den entsprechenden lokalen DNS-Server. Dieser DNS-Name kann entweder eine existierende Seite oder eine ausgedachte sein. In beiden Fällen **muss** es jedoch ein DNS-Name sein, die der DNS-Server **auflösen muss**.

Sobald der Hacker seine DNS-Anfrage versendet hat und den entsprechenden Upstream-DNS-Server kennt, versucht er diesen z.B. mittels DDoS-Attacken zu beschäftigen, **sodass die Antwortzeit stark verlängert wird**. Während dieser Zeit versucht der Angreifer die präparierte IP-Adresse dem DNS-Server für den entsprechend angefragten DNS-Namen unterzubringen.

Problem ist jedoch, dass der DNS-Server eine **Query-ID** zu seinem Upstream-DNS-Server schickt, welche bei einer Antwort des Upstream-DNS als Authentitäts-Code benutzt wird. Ohne die entsprechende Query-ID kann der Angreifer seine präparierte IP-Adresse nicht als Antwort zum DNS-Server senden. Da DNS jedoch grundsätzlich **unverschlüsselt** ist, kann der Angreifer die Query-ID entweder mit einem **Sniffer** auslesen oder er **errät** diese. Das Erraten der Query-ID ist nicht umfangreich, da die Query-ID *nur* zwischen 1-65535 groß sein kann. Im Durchschnitt braucht man meist jedoch nur $2^{16} / 2 =$ **32.768** Versuche, um die richtige Query-ID

zu erraten. Zudem wird das Erraten dadurch vereinfacht, dass mehrere Query-Antworten parallel geschickt werden können.

Hat der Angreifer die Query-ID des DNS-Servers erraten, schickt dieser auf die Anfrage des DNS-Servers seine präparierte IP-Adresse. Wenn nun ein User den entsprechenden DNS-Namen eingibt, wird dieser anstatt auf die Originalwebseite auf die gefälschte Webseite des Angreifer weitergeleitet. Hier kann der Angreifer nun die geheimen Daten abgreifen und entsprechend missbrauchen.



Sicherheitserweiterungen

Notiz Stand 07.11.2022: Aufgrund der Komplexität und Länge der nachfolgenden Themen, wurden diese nur kurz beschrieben. Eine ausführliche Ausarbeitung ist noch geplant.

Transaction Signature (TSIG)

TSIG beschreibt ein Verfahren, bei welchem die **Authentizität** von kommunizierenden DNS-Server bewahrt werden soll. TSIG wird hauptsächlich zwischen **DNS-Servern** benutzt. Es funktioniert mittels symmetrischer Verschlüsselung. Um TSIG konfigurieren zu können, muss auf den entsprechenden DNS-Server mindestens ein Administrator-Zugriff bereitgestellt sein.

Auf den jeweiligen DNS-Servern muss ein **geteiltes Geheimnis** konfiguriert werden. Mit diesem berechnen die DNS-Server den **MD5-Hash** des zu verschickenden DNS-Pakets und hängen diesen an besagtes. Der jeweilige empfangenden Server berechnet ebenfalls mit dem geheimen Schlüssel den MD5-Hash des empfangenen Pakets und überprüft beide Hashes. Sind sie **gleich**, ist die Authentizität **gewährleistet**. Unterscheiden sich die Hashes, wird die DNS-Anfrage nicht zurückgewiesen.

Grundsätzlich ist dies ein deutlich einfacheres Verfahren als **DNSSEC**, welches auf **Public-Key-Infrastructure** setzt. Jedoch ist der **Konfigurationsaufwand** bei vielen Servern **sehr hoch**, weshalb der Einsatz von PKI hier ein Vorteil besitzt. TSIG ist in RFC 2845 beschrieben.

DNS over https (DoH)

Beschreibt die Funktion, wie DNS über das **https-Protokoll** verwendet werden kann, um **Sicherheit und Authentizität** der DNS-Anfragen zu stärken. DoH ist ein RFC 8484 beschrieben.

DNS over TLS (DoT)

Beschreibt die Funktion, wie DNS über TLS verwendet werden kann, um **Sicherheit und Authentizität** der DNS-Anfragen zu stärken. Es ist ähnlich zu DNS over https (DoH) und ist in RFC 8310 und in RFC 7858 beschrieben.

DNSSEC

Beschreibt eine Reihe von Internetstandards zur Stärkung der DNS-Abfragen in **Authentizität** und **Integrität** der Daten, nicht aber der Server. Es wurde entwickelt, um **DNS-Poisoning-Angriffe** entgegenzuwirken. Für **Vertraulichkeit** ist DNSSEC **nicht** vorgesehen! DNS-Daten werden zudem **nicht verschlüsselt**!

DNS-Zensur

In verschiedenen Ländern der Welt -so auch in Deutschland- werden ebenfalls DNS-Sperren für bestimmte Websites verhängt. Auch hier könnte man von einem *DNS-Poisoning* reden, da die Einträge entweder auf **andere Webseiten verwiesen** werden oder einfach **gesperrt** werden. Im letzteren Fall wird der abzufragende DNS-Name gegen eine **Sperrliste** (Blacklist) verglichen. Ist der DNS-Name dort enthalten, wird der Zugriff entsprechend **blockiert**. Es existieren jedoch DNS-Service-Provider, die darauf schören **zensurfrei** zu sein.

Quellen

- https://de.wikipedia.org/wiki/Domain_Name_System
- https://www.ionos.de/digitalguide/server/knowhow/namensaufloesung-im-netz-was-ist-ein-dns-server/
- https://www.varonis.com/de/blog/was-dns-ist-wie-es-funktioniert-und-schwachstellen
- https://www.elektronik-kompendium.de/sites/net/0901141.htm
- https://aws.amazon.com/de/route53/what-is-dns/
- https://de.wikipedia.org/wiki/TSIG
- https://www.cira.ca/resources/anycast/guide-how/using-transaction-signatures-tsig-secure-dns-server-communication
- https://de.wikipedia.org/wiki/Cache_Poisoning
- https://en.wikipedia.org/wiki/DNS_over_HTTPS
- https://en.wikipedia.org/wiki/DNS_over_TLS
- https://en.wikipedia.org/wiki/Domain_Name_System_Security_Extensions
- https://www.ionos.de/digitalguide/server/knowhow/dns-zone/
- https://www.elektronik-kompendium.de/sites/net/1910181.htm

Domain und Domaincontroller

Hinweis: Der nachfolgende Text entspricht fast 1:1 dem aus der Quelle https://www.ip-insider.de/was-ist-ein-domaenencontroller-a-626094/ stammenden Text.

Was ist eine Domain?

Innerhalb eines Unternehmensnetzwerks gibt es viele Netzwerkressourcen wie Drucker, Server, Anwendungen u.v.m. Um diese Geräte und dessen Benutzern bzw. die Benutzerzugriffe zu verwalten, werden diese zu einem (oder mehreren Netzwerk-) Domains zusammengefasst. Diese Netzwerkdomains kommen in Unternehmensnetzwerken zum Einsatz, um die **Struktur des Unternehmens nachzubilden**. Es handelt sich hierbei um voneinander administrativ **klar abgegrenzte** Netzwerkbereiche, in denen User unterschiedliche Rechte- und Sicherheitsrichtlinien erhalten. Zur Unterscheidung der Domains besitzen diese **eindeutige Namen**.

Dadurch entstehen **zentral verwaltete Sicherheitsbereiche** mit administrierten Ressourcen. Die Netzwerkdomains sind **hierarchisch** strukturiert und verwenden sogenannte "*Domaincontroller (DC)*" für die Zuteilung der Benutzerrechte. Ein Vorteil der Netzwerkdomain ist, dass die verschiedenen Informationen und Richtlinien zu den unterschiedlichen Objekten **nicht mehr lokal** auf den jeweiligen Rechner selbst, sondern **zentral verwaltet werden**. Die Domain sorgt für die Authentisierung und ermöglicht die Umsetzung des Berechtigungskonzepts für Endgeräte und Netzlaufwerke sowie Ressourcen wie Netzwerkdrucker.

Die Struktur einer Domain

Die Namenskonvention und Struktur von Domains basieren auf dem "*Domain Name System (DNS)*" (siehe DNS - Domain Name System). Unterhalb einer Stammdomain lassen sich mehrere untergeordnete Domains in einer **Baumstruktur** anlegen. Die Domainstruktur bezeichnet man daher auch als *Domainbaum*. Alle, einer Stammdomain untergeordneten Domains besitzen in ihren Namen den Namensteil der Stammdomäne.

Was ist ein Domaincontroller?

Ein Domaincontroller (DC) ist ein Server, der eine Domain uns seiner verschiedenen Ressourcen / Objekte **zentral verwaltet und kontrolliert**. Anwender, die sich an einer Netzwerkdomain anmelden möchten, wenden sich zuerst an den für ihre Domain zuständigen DC.

Ursprünglich wurde der DC bereits 1970 von IBM eingeführt. Microsoft hat den Begriff übernommen und nutzt den Domaincontroller, abgekürzt "*DC*", für Windows-Netzwerke. Dabei handelt es sich um eine zentrale Instanz innerhalb einer Domain, die für die **Authentifizierung und Rechtesteuerung** der Nutzer zuständig ist. Ein Vorteil von solch einer zentralen Instanz ist, dass Benutzer und ihre zugehörigen Rechte **nicht mehr lokal anzulegen und zu verwalten sind**. Ebenso ist es möglich, sich an weiteren, in der Domain befindlichen Geräten anzumelden, ohne dass es den Benutzer bereits lokal am jeweiligen System geben muss. Änderungen über den Domaincontroller gelten für **alle Benutzer und Objekte der Domain**. Die Rolle des Domaincontrollers kann **jeder Server übernehmen, welcher Mitglied in der Domain ist**.

Je nach Größe und Komplexität des Netzwerk kommen pro Domain ein oder mehrere DCs zum Einsatz. Da sich **ohne einen funktionierenden DC kein User an der Domain anmelden kann**, sind die DCs grundsätzlich **redundant** realisiert. Um eine höhere Verfügbarkeit und bessere Lastverteilung sicherzustellen, sind meist **mindestens zwei** oder mehr DCs pro Domain vorhanden. Diese replizieren ihre Informationen regelmäßig und können die Masterrolle beim Ausfall des Domain-Masters **ohne Funktionseinschränkungen** übernehmen (siehe DNS Cluster).

Anwender müssen sich gegenüber dem Controller authentifizieren. Können sie nachweisen, dass sie Mitglied der Domain sind, erhalten sie die entsprechenden Benutzerrechte für beispielsweise bestimmte Verzeichnisse oder Druckerressourcen. Es ist anzumerken, dass es mittlerweile auch Lösungen gibt, die es gestatten, Server als DC zu benutzen, auf denen **Nicht-Windows-Betriebssysteme** wie Linux betrieben werden können. Die Kompatibilität ist jedoch in einigen Bereichen eingeschränkt.

Die Redundanz des Domaincontrollers

Da der DC eine zentrale Rolle für die User zur Nutzung der Netzwerkressourcen darstellt, gab es von Microsoft frühzeitig entsprechende Redundanzkonzepte. Früher, in den **NT4-Domains**, existierte ein "*Primary Domain Controller (PDC)*" und ein "*Backup Domain Controller (BDC)*". Änderungen waren nur auf dem PDC möglich. Der BDC hielte eine regelmäßig aktualisierte Sicherheitskopie der Daten und ließ sich **bei Bedarf zum Primary-System ernennen**.

Seit Windows 2000 bietet Microsoft das "*Active Directory (AD)*" (bei Linux Samba oder eDirectory) mit der sogenannten "*Multimaster-Replikation*" an. Alle DCs besitzen nun eine beschreibbare Kopie der AD-Datenbank. **Jede Änderung wird automatische an alle anderen DCs repliziert.** Dieser Mechanismus sorgt dafür, dass alle DCs sich steht auf dem **gleichen Informationsstand** befinden. Fällt ein DC aus, hat dies **keinen Informationsverlust** zur Folge und ein anderer DC übernimmt dessen Funktion. Seit 2008 existiert zusätzlich das Konzept eines "*Read Only*"-Domain Controllers.

Rollen des Domaincontrollers innerhalb des Active-Directories

Insgesamt kann ein Domaincontroller innerhalb des Microsoft ADs bis zu **fünf verschiedene Rollen** annehmen. Das dahinterstehende Konzept nennt sich "*Flexible Single Master Operations*" (FMSOs). Abhängig von der jeweiligen Rolle, existiert diese einmalig pro Domain oder einmalig pro Gesamtstruktur.

- **PDC-Emulator (Einmal pro Domain):** Er ist für die Verwaltung und die Anwendung der Gruppenrichtlinien zuständig. Darüber hinaus ist er verantwortlich für Kennwortänderungen bei den Benutzern und dient als Zeitserver.
- **RID-Master (Relative-ID, Einmal pro Domain):** Er gestattet es, neue Objekte in die Domain aufzunehmen. Hierfür weist er eindeutige relative Bezeichner (RIDs) zu.
- **Infrastrukturmaster (Einmalig pro Domain):** Er kümmert sich um das Auflösen der Gruppen über mehrere Domains hinweg und sorgt für die Steuerung von Benutzerrechten der User aus unterschiedlichen Domains.
- **Schemamaster (Einmal pro Gesamtstruktur):** Er erlaubt das Erweitern und Verändern von Schemas des ADs.
- **Domainnamenmaster:** Dieser wird immer dann benötigt, wenn eine **neue Domain in die Gesamtstruktur** aufgenommen werden soll.

Vor- und Nachteile eines Domain-Controllers

Vorteile	Nachteile
Zentrale Benutzerverwaltung	Beliebtes Ziel von Cyberangriffen
Ermöglicht gemeinsame Nutzung verschiedener Ressourcen	Muss immer auf dem neusten Stand sein / Sicherheitspatches müssen direkt eingespielt werden

Vorteile	Nachteile
Verknüpfte Konfiguration bei Redundanz	Netzwerk hängt von der Verfügbarkeit des DCs ab
Lässt sich gut skalieren und replizieren	Muss sicherheitstechnisch abgeschottet sein
Trägt zu einer verbesserten Netzwerksicherheit bei	

Quellen

- https://de.wikipedia.org/wiki/FSMO
- https://en.wikipedia.org/wiki/Domain_controller_(Windows)
- https://www.ip-insider.de/was-ist-ein-domaenencontroller-a-626094/
- https://www.varonis.com/de/blog/was-ist-ein-domanen-controller-wann-wird-er-gebraucht-und-eingerichtet
- https://www.ip-insider.de/was-ist-eine-domaene-netzwerkdomaene-a-626054/
- https://www.ip-insider.de/was-ist-ein-active-directory-a-626455/

X11

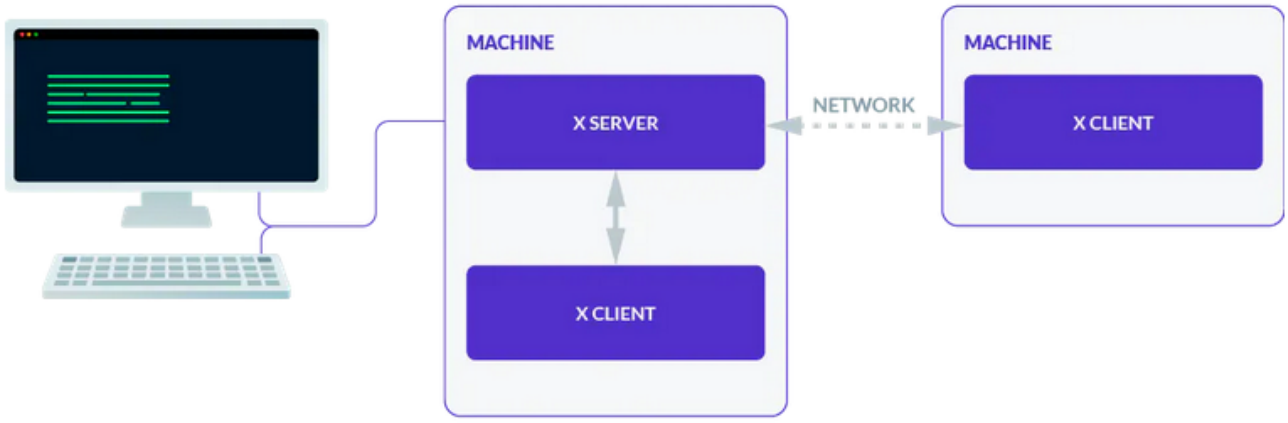
X11 ist ein Protokoll-Stack bestehend aus unterschiedlichen Komponenten. *X* bezieht sich hierbei auf das *X-Window-System*, welches in den meisten Linux-Distributionen vorkommt und dazu benutzt wird, primitive Grahpical User Interfaces (GUIs) zu erstellen. Die *11* bedeutet, dass sich das Protokoll in der elften Version befindet, welche seit dem Jahr **1987** besteht.

X Window System

Das *X Window System* ist ein Framework, um primitive GUIs zu erstellen, welche meist nicht mehr als aus einem Display-Device[1] und Canvas[2] besteht und man mit diesem per Maus und Tastatur interagieren kann. Jedoch ist es für das X Window System nicht möglich komplexere Inhalte wie Buttons zu generieren. Hierfür werden meist Toolkits wie GTK benutzt, um entsprechende komplexe GUI-Darstellung zu erstellen.

X arbeitet in einem transparenten **Client-Server-Modell**, wodurch es an Unabhängigkeit vom eingesetzten Betriebssystem und GUI-Toolkits ist. Der Client ist das **Frontend** des Systems. Es erstellt die primitiven GUI-Elemente, welche durch die entsprechend eingesetzten Toolkits verschönert werden können. Es arbeitet komplett isoliert vom Server, was bedeutet, dass der Stil sich pro eingesetzten Client / Toolkit unterscheiden kann. Der Server ist das **Backend** des Systems. Es kümmert sich um die Anfragen des Clients, der Bereitstellung der Ressourcen und der Kommunikation mit dem Linux-Kernel. Auch dieser kann isoliert vom Frontend benutzt werden.

Das X Window System ist **netzwerkfähig**, was bedeutet, dass es nicht nur lokal auf einem Rechner ausgeführt werden kann, sondern auch über das Netzwerk verteilt. Dabei agiert der PC als **XServer** (Backend, stellt Ressourcen zur Verfügung) und kommuniziert mit einem entfernten Server (XClient, Frontend) mitsamt GUI-Frontend.



(Sicherheits-) Probleme

- Der X11-Standard ist bereits veraltet und dennoch großer Beliebtheit. Dennoch gibt es bereits ähnliche Programme wie Wayland, welche deutlich effizienter laufen.
- Die netzwerkbasierte Kommunikation erfolgt **unverschlüsselt**. Dies macht die Benutzung von X11 über Netzwerk sehr anfällig für MiTM-Angriffe.
- X11 liefert viel Programmcode mit sich und bläst das OS ggf. unnötig auf.
- Viele Prozesse von X11 laufen mit **root-Berechtigungen**. Ein kleiner Fehler im Programmcode kann es erlauben sämtliche Kontrolle des Prozesses zu übernehmen und schadhaften Code als root auszuführen.

X11 Forwarding

Da X11 **unverschlüsselt** kommuniziert, wird SSH oftmals als Tunnel verwendet, um X11 benutzen zu können. Dieses Vorgehen ist auch bekannt als *X11 Forwarding*. Durch die verschlüsselte SSH-Kommunikation ist es Angreifern nicht möglich auf die Daten während der Kommunikation zuzugreifen.

X11 Forwarding könnte man auch als *Linuxartiges RDP* bezeichnen. Da auf den meisten unixbasierten Produktiv-Servern in einem Unternehmensnetzwerk sowieso keine GUI vorhanden ist, kann in der SSH-Konfiguration (sshd_config) das X11 Forwarding **deaktiviert** werden.

Fußnoten

- [1] Hiermit sind nicht unbedingt Displays als Hardware gemeint. Gemeint können auch virtuelle Desktops sein.
- [2] Canvas ist bei Computern ein Container, welcher eine Vielzahl an unterschiedlichen Elemente beinhalten kann.

Quellen

- https://goteleport.com/blog/x11-forwarding/
- https://security.stackexchange.com/questions/4641/why-are-people-saying-that-the-x-window-system-is-not-secure
- https://askubuntu.com/questions/101829/why-is-x11-a-security-risk-in-servers
- https://www.baeldung.com/linux/x11
- https://unix.stackexchange.com/questions/276168/what-is-x11-exactly
- https://en.wikipedia.org/wiki/X_Window_System