DNS Sicherheit

Vortrag zum Kryptologie-Projekt

Wilhelm Bartel (19INM-VZ), Justin Kromlinger (19INM-VZ), Tom Wegener (18INM-TZ)

January 20, 2021

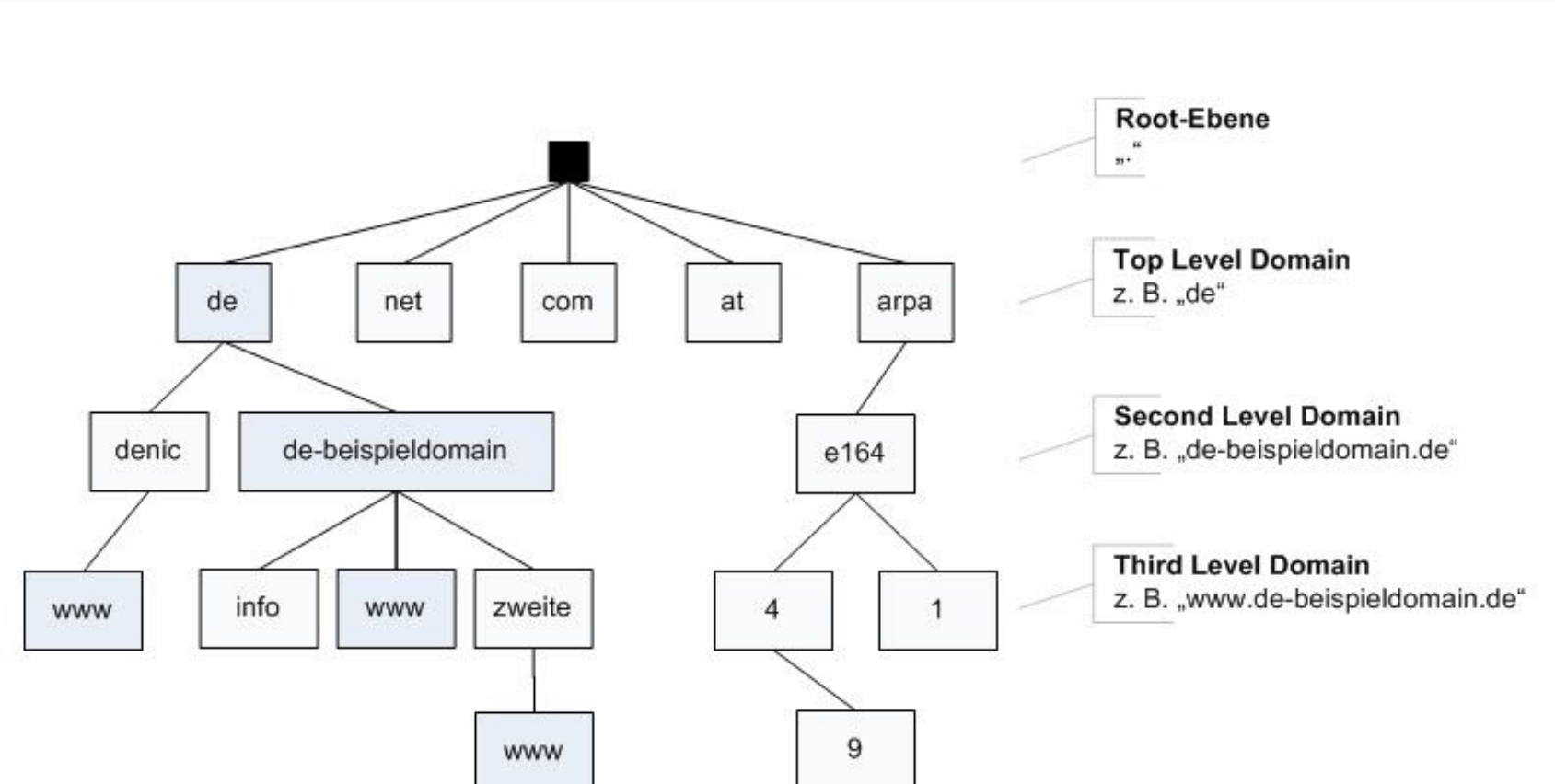
Grundlagen

Grundlagen

- Domain Name System
- Auflösung von Domain (`www.htwk-leipzig.de`) z.B. zu einer IP Addresse

Aufbau

- hierarchisch Aufgebaute, verteilte Datenbank
- arbeitet mit “Zonen” getrennt durch “.”: z.B.: `www.htwk-leipzig.de.`
- jeder Knoten kann ein Server sein
- Wurzel = Root-DNS-Resolver



Funktionsweise - Resource Records

- zuordnung des Domain Namens zu Daten
- Beispiele:
  - A: IPv4 Adresse
  - AAAA: IPv6 Adresse
  - MX: Mailserver
  - TXT: Freitext

Funktionsweise - Abfrage

- zu besuchende Webseite: `www.wikipedia.org`

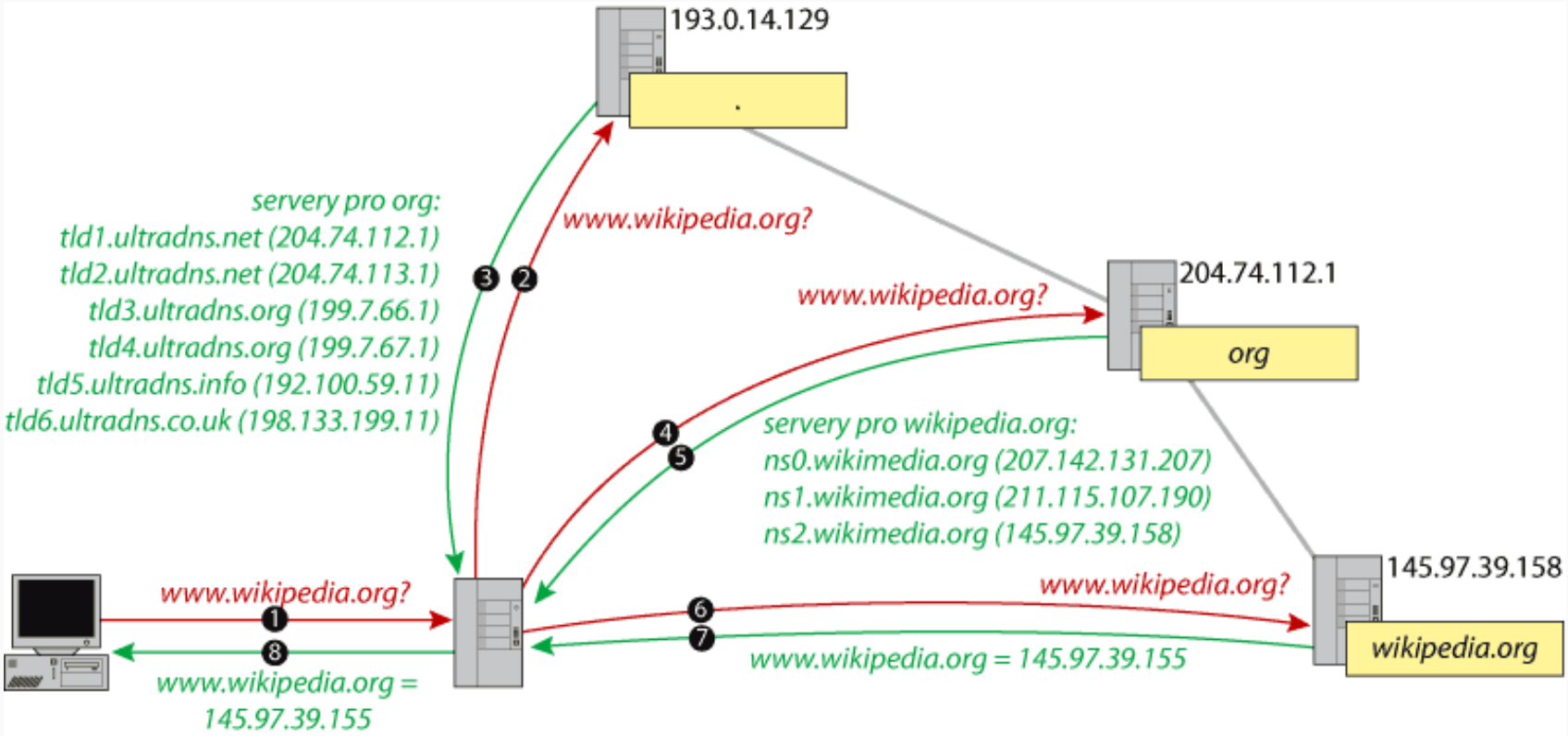


Figure 2: DNS Abfrage Reihenfolge (Wikimedia Commons 2006)

Funktionsweise - Caching

- Caching ermöglicht durch Time-To-Live (TTL) an jedem RR
- Jeder DNS Server kann mit RRs anderer Zonen Antworten
- spart Netzwerkauslastung und Latenz

Angriffsmöglichkeiten

Angriffsmöglichkeiten

- DNS vollständig über UDP
- nicht signiert, nicht verschlüsselt
- Angriffe auf:
  1. Integrität / Authentizität
  2. Anonymität

Integrität / Authentizität

- Antworten vom DNS Server nicht signiert -> können von Angreifer unbemerkt verändert werden
- DNS Spoofing = austauschen des DNS Servers

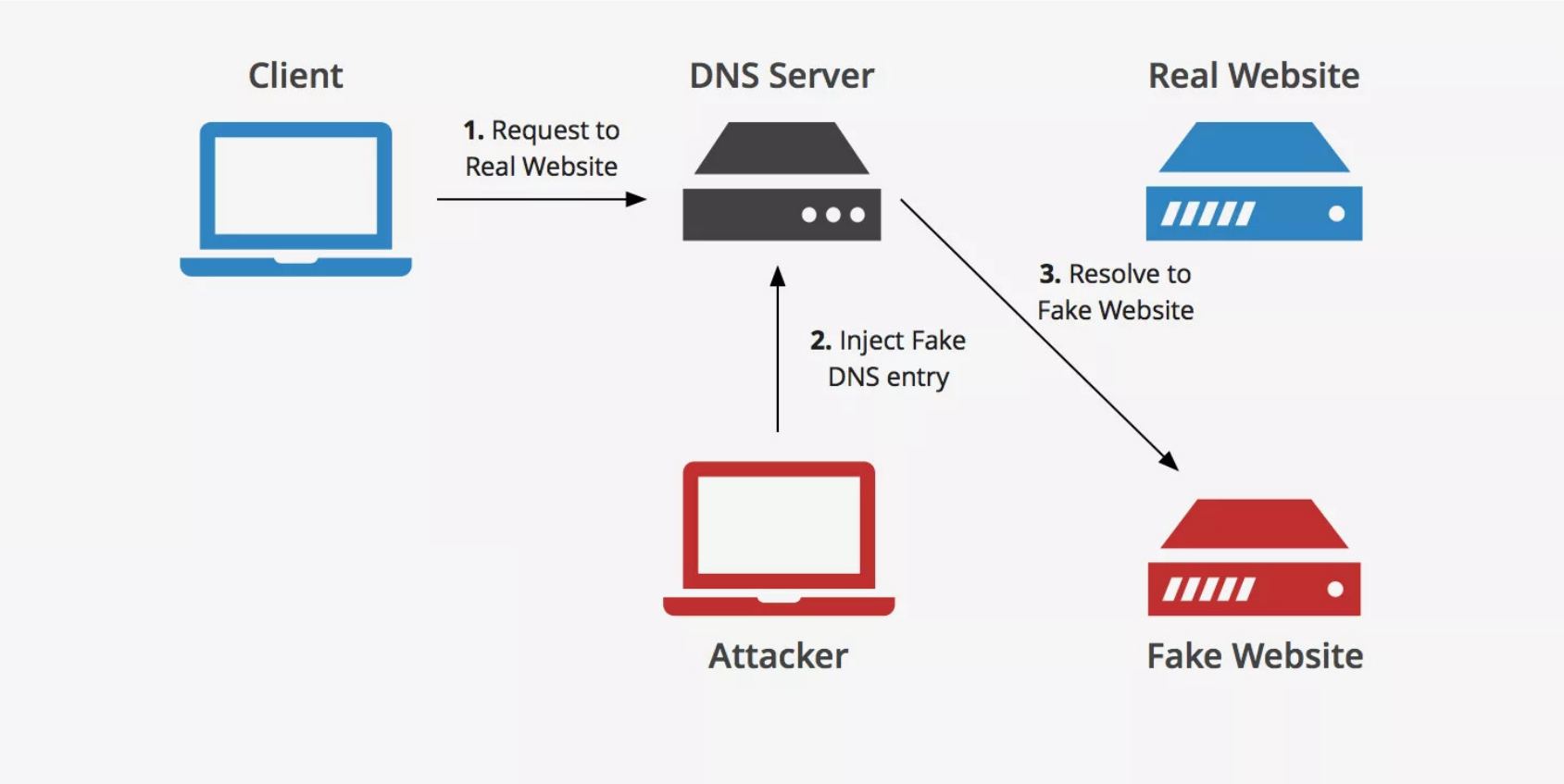


Figure 3: DNS spoofing (keycdn 2008)

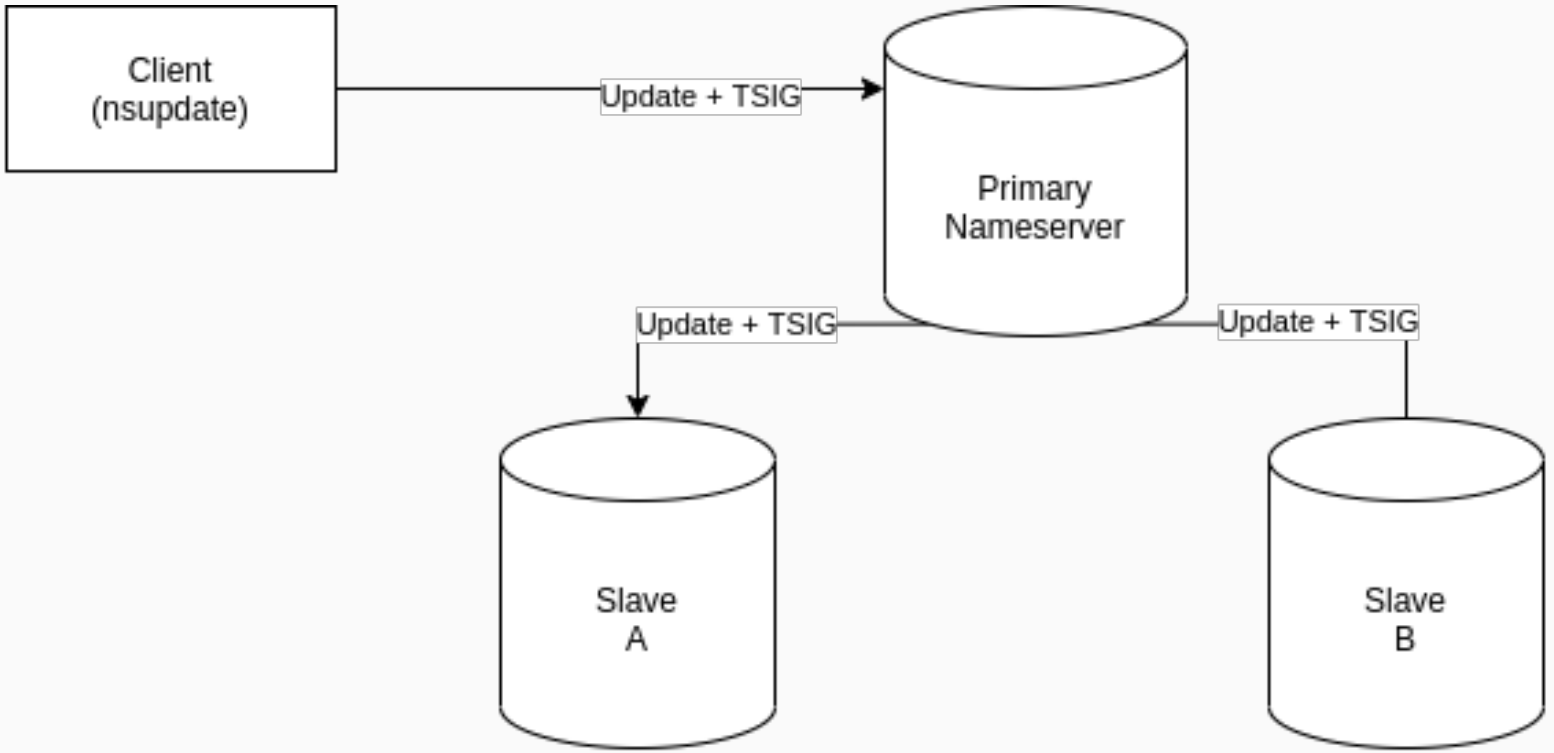
Anonymität

- Anfragen unverschlüsselt
- Anfragekette ist vom Provider und den DNS Servern ersichtlich.
- kann zur Deanonymisierung und zum Profiling benutzt werden

TSIG zum Update von DNS-Einträgen

TSIG zum Update von DNS-Einträgen

- Secret Key Transaction Authentication
- **Authentizität** von **DNS-Partnern**
- **Integrität** von Updates
- **Pre-Shared** Key
- TSIG-Record mit verschlüsseltem **Zeitstempel**, **Hashing** Algorithmus sowie Update-Hash
- Zeitstempel verhindert Wiederverwendung
- Antwort wird bei Erfolg ebenfalls mit TSIG-Record versehen
- Verteilung von Schlüsseln mit TKEY-Record



TSIG record fields

Field	Bytes	Value	Description
NAME	Max. 256	Varies	Key name; identifies key on both client and server
TYPE	2	TSIG (250)	
CLASS	2	ANY (255)	
TTL	4	0	TSIG records must not be cached
RDLENGTH	2	Varies	Length of RDATA field
RDATA	Variable	Varies	Structure containing the timestamp, algorithm and hash data

Figure 4: TSIG Beispiel

DNSSEC

DNSSEC

- **Signierte Antworten** von DNS-Resolvern
- Authentizität, Integrität, Existenzverweigerung
- **Vertrauenskette** und **Asymmetrische Verschlüsselung**
  - PubKeys von Rootservern wird vertraut
  - Rootserver signieren PubKeys von TLDs
  - TLDs signieren PubKeys von Sub-Nameservern
  - usw.
- Ressource Records
  - DNSSEC: öffentlicher Schlüssel
  - DS: Delegierte Zone für die Vertrauenskette
  - RRSIG: Record Signatur für Prüfung von Authentizität und Integrität
  - NSEC: Signierte Existenzverweigerung (NXDOMAIN)
- Nameserver halten **nur den Public Key** vor
- DNS-Antworten werden **vorab** mit Private Key signiert

DNSSEC - Signierte Existenzverweigerung

- Antworten auf Anfragen für **nicht existierende Zonen** müssen signiert werden
- Direkt nur über **Private Key** möglich – unsicher!
- Lösung: **NSEC**
  - **Sortierung** der Zonen und Aufteilen in **Paare**
  - **Lokale** Signierung der Paare mit Private Key
  - Liegt eine angefragte Zone in einem Paar, wird dieses und dessen Signatur ausgeliefert
- Problem: **Zone Walking**
- Lösung: **NSEC3**
  - Zonen werden vor der Sortierung gehashed
- Problem:
  - Lokale Wörterbuchattacke auf Hashes
- Lösung: **NSEC5**
  - Signed Key Hashes

```
% dig dnssec-failed.org

; <<>> DiG 9.16.10 <<>> dnssec-failed.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 14125
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;dnssec-failed.org.             IN      A

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed Jan 20 10:47:08 CET 2021
;; MSG SIZE  rcvd: 46

% dig dnssec-failed.org @1.1.1.1

; <<>> DiG 9.16.10 <<>> dnssec-failed.org @1.1.1.1
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: SERVFAIL, id: 41006
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 1
```

Figure 5: Abfrage mit falschem DNSSEC

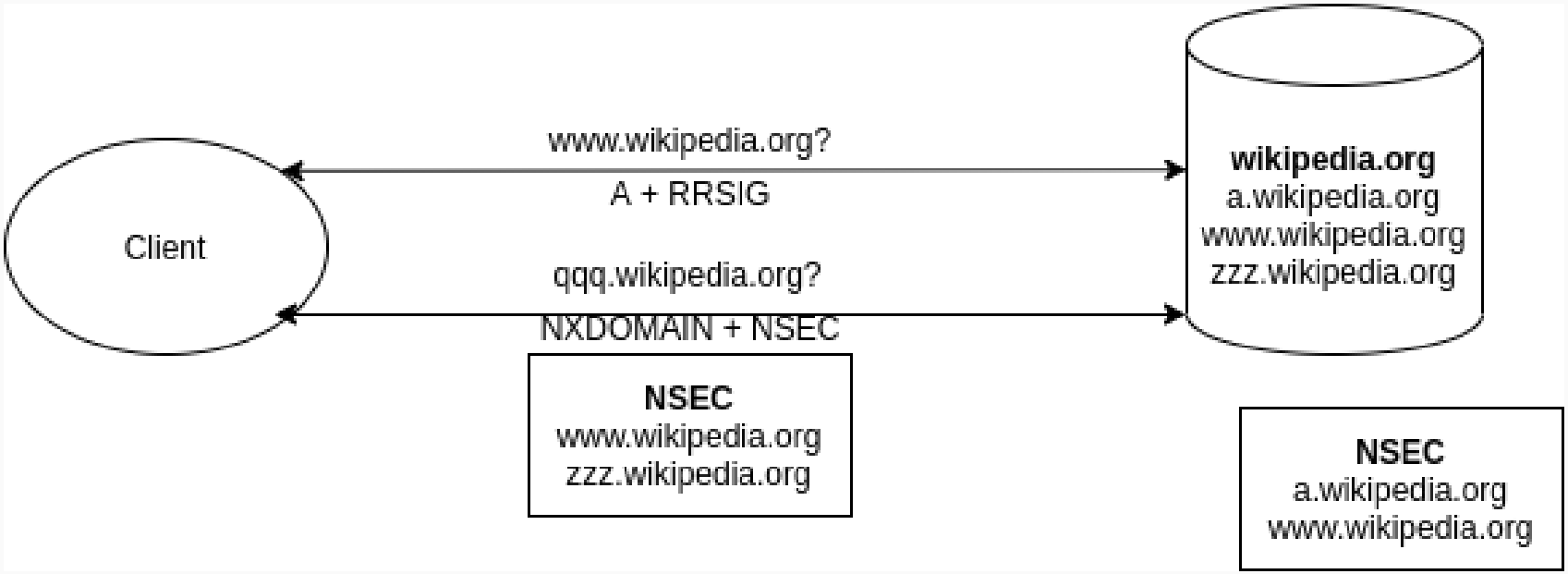


Figure 6: RRSIG + NSEC Beispiel

DNS over TLS (DoT)

DNS over TLS (DoT)

- **Vertraulichlichkeit** bis zum DNS-Resolver gesichert
- **TLS-Wrapper** für DNS-Protokoll
- **853/TCP** statt 53/UDP
- DNS-Resolver haben nun eine zum Zertifikat passende Domain
- Leichter Overhead (durch TLS)
- **Kein** E2E!

```
% ssh hera cat /etc/unbound/unbound.conf | grep tls
        tls-service-key: "/etc/letsencrypt/live/dns.hash.works/privkey.pem"
        tls-service-pem: "/etc/letsencrypt/live/dns.hash.works/fullchain.pem"
        tls-port: 853
```

Figure 7: DoT in unbound als eigener Resolver

DoT – Public Resolver

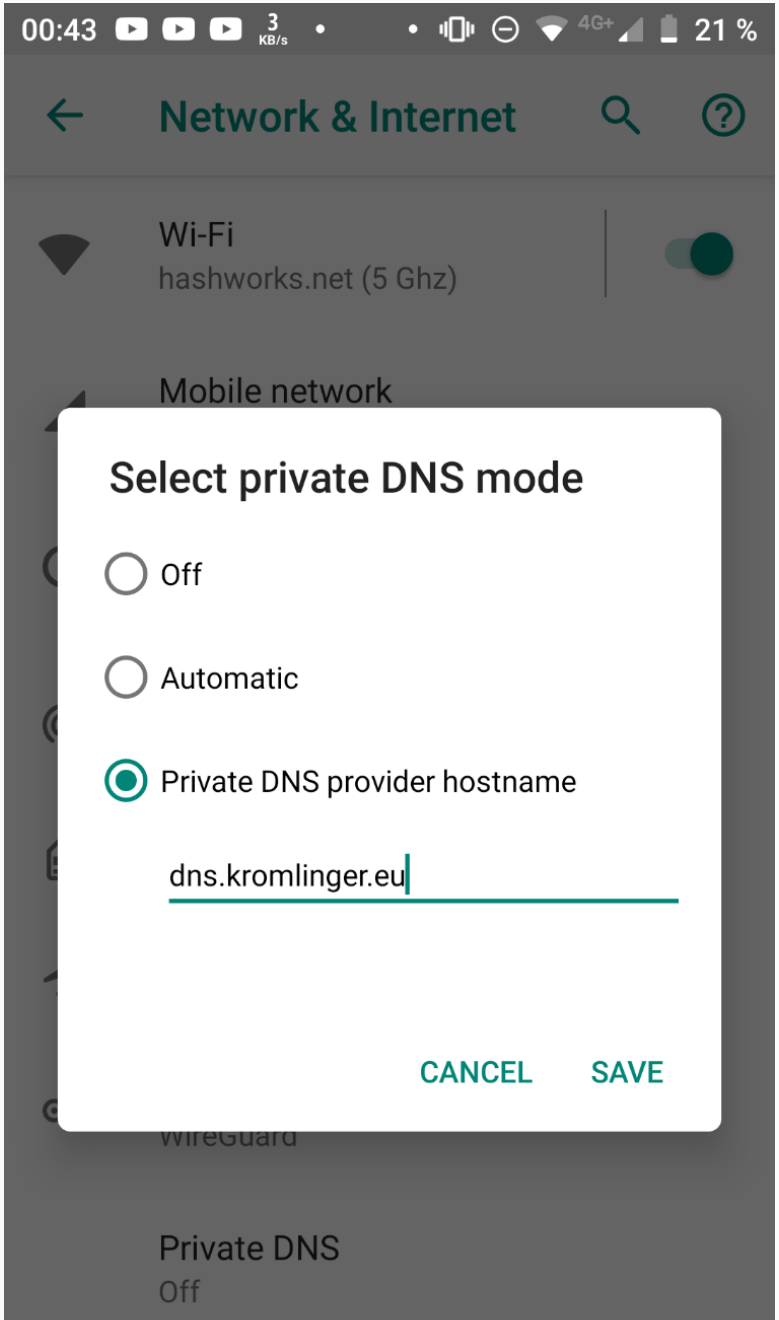
- *1.1.1.1@853#*.cloudflare-gateway.com*
- *8.8.8.8@853#dns.google.com*
- *9.9.9.9@853#dns.quad9.net*

/etc/systemd/resolved.conf.d/dns_over_tls.conf

```
[Resolve]
DNS=9.9.9.9#dns.quad9.net
DNSOverTLS=yes
```

Figure 8: DoT über systemd-resolved

DoT – Android >= Pie

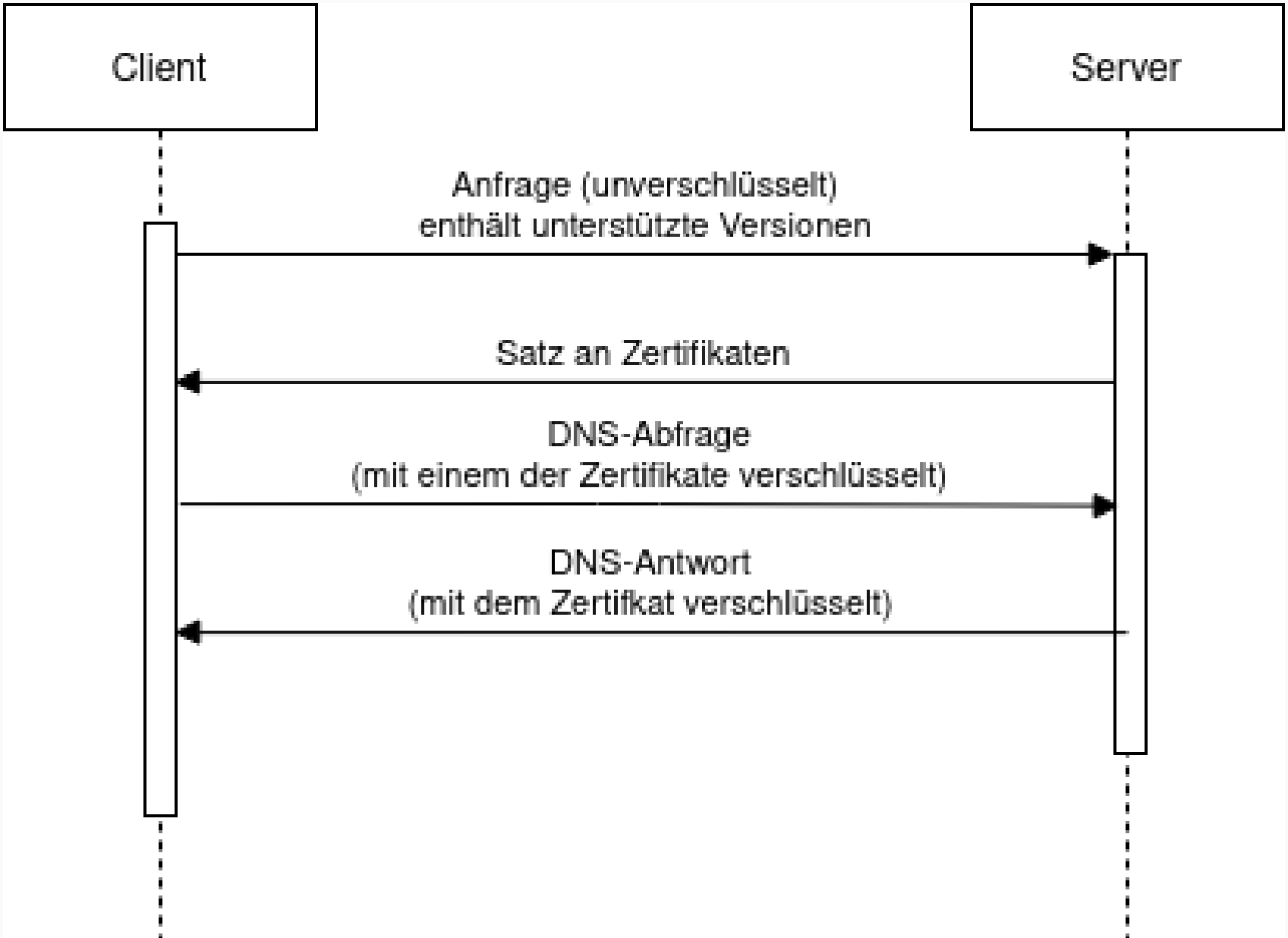


DNSCRYPT

Überblick

- Protokoll zur Authentizitäts-Prüfung und Verschlüsselung von DNS-Verkehr
- Schutz vor Manipulation von Antworten, kein Schutz vor Dritten
- Kommunikation über (präferiert) TCP oder UDP
- basierend auf einem Server-Client-Prinzip

Ablauf


Client
Server
Anfrage (unverschlüsselt)
enthält unterstützte Versionen
Satz an Zertifikaten
DNS-Abfrage
(mit einem der Zertifikate verschlüsselt)
DNS-Antwort
(mit dem Zertifkat verschlüsselt)

Verbreitung

- keine große Verbreitung, obwohl das Protokoll von OpenDNS und damit auch von Cisco unterstützt wird
- verschiedene Implementationen von z.B. von AdGuard oder Surfshark
- Server von z.B. AdGuard oder Cisco (über OpenDNS)

DNS over HTTPS (DoH)

Überblick

- RFC8484 von ICANN und Mozilla
- DNS-Verkehr wird über HTTPS (Port 443) geleitet
- nutzt damit standartmäßige HTTPS-Verschlüsselung (also TLS)
- kein Unterschied zu “normalem” HTTPS-Traffic, also schwer zu blockieren
- meist auf Anwendungsebene implementiert

Eigenschaften

Pro

- kann nicht so einfach geblockt werden
- verschlüsselte DNS-Abfragen

Contra

- gleiches Problem mit Resolvern, die nach dem DNS-Server kommen
- gewisse Unsicherheit von HTTPS

Verbreitung

- in den führenden Browsern (Chrome, Edge, Firefox) integriert
- weitere Applikationen in verschiedenen Programmier-Sprachen und auch als Pakete/Module/Crates...
- grundlegend von iOS 14 und MacOS unterstützt, in einem Insider-Preview-Build von Microsoft Windows implementiert
- Server z.B. von Google, Cloudflare und Quad9

Literatur

Literatur

Denic. 2020. “Das Domain Name System (DNS).”
https://www.denic.de/wissen/domain-name-system-dns/.

keycdn. 2008. “DNS Spoofing.”
https://www.keycdn.com/support/dns-spoofing.

Wikimedia Commons, Pavel.satrapa at. 2006. “DNS Abfrage Bild.”
https://commons.wikimedia.org/wiki/File:Dns-wikipedia.png.