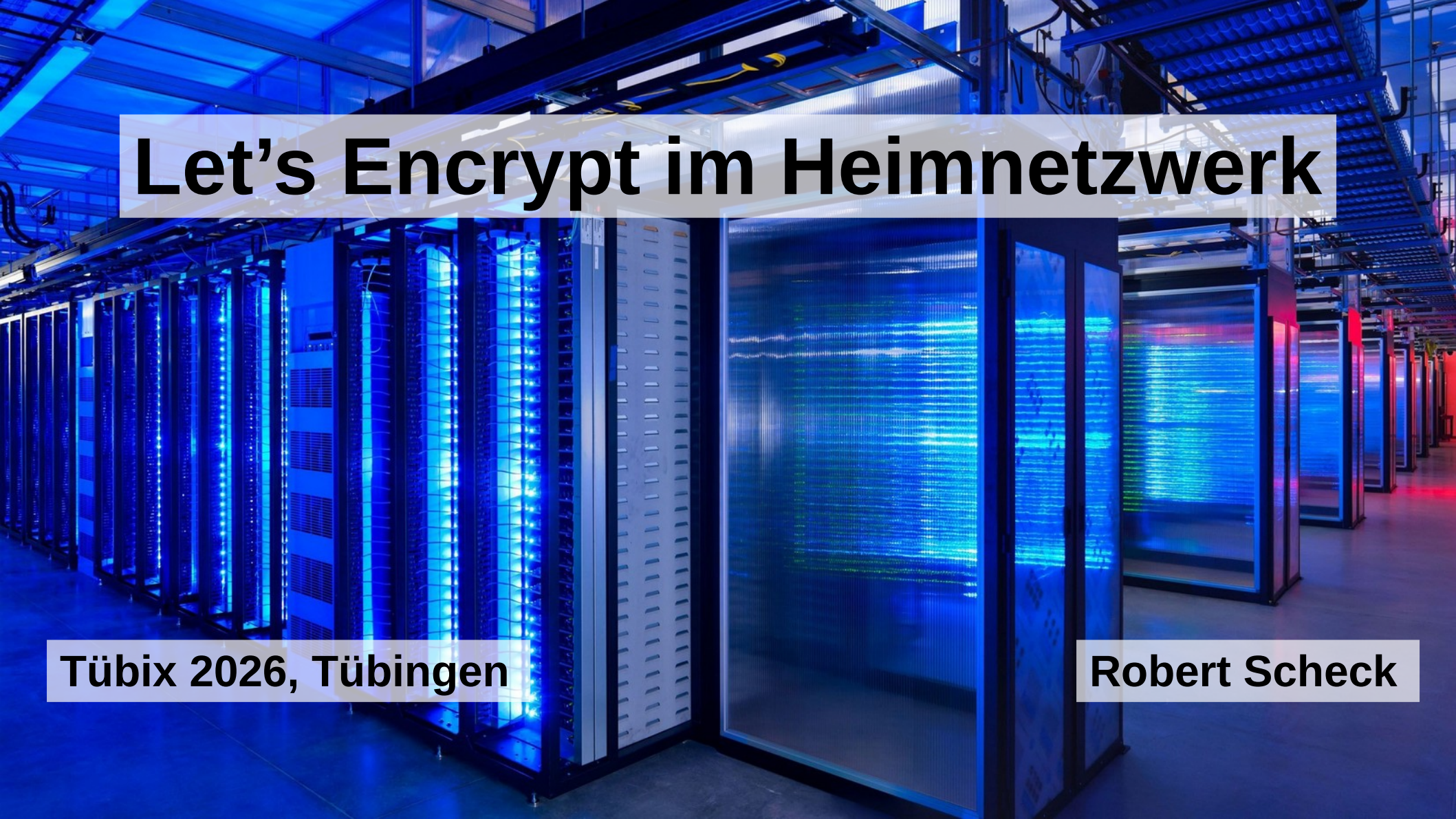


Let’s Encrypt im Heimnetzwerk

Tübix 2026, Tübingen

Robert Scheck

Robert Scheck

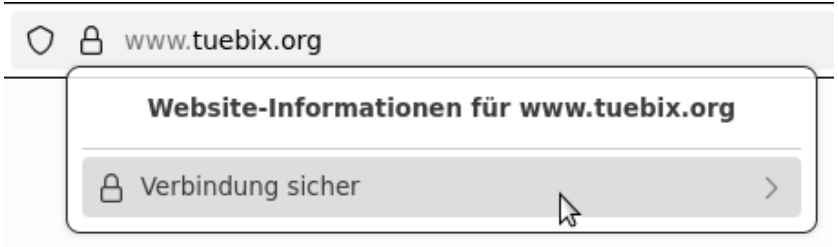
- Mitwirkender beim Fedora Project (seit 2005)
  - Package Maintainer (ca. 250 Pakete) mit Fokus auf EPEL
  - Provenpackager und Packager Sponsor
- Mitwirkender bei diversen anderen Open-Source-Projekten

  u.a. aggregate6, Baresip, bgpdump, bgpq4, Coturn, CUPS, iputils, jwhois, Linux-Kernel, MIMEDefang, Nextcloud, OpenSSL, ownCloud, Partclone, PlayOnLinux, phpMyAdmin, popt, rpki-client, systemd

- Mail: robert@fedoraproject.org
- Web: https://fedoraproject.org/wiki/RobertScheck

Was ist SSL/TLS?

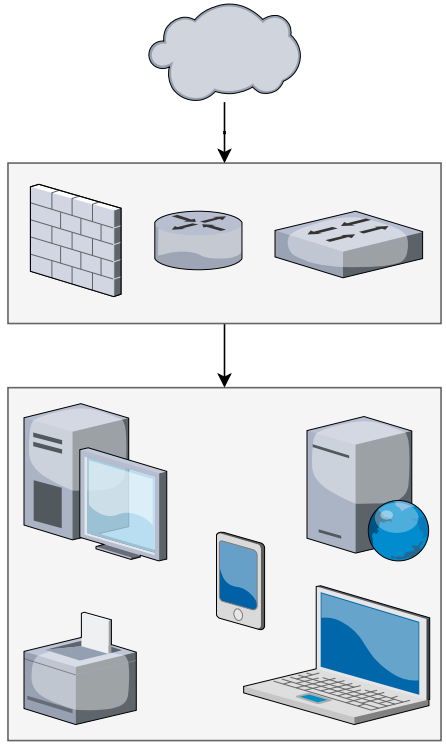
- Verschlüsselung zwischen Client und Server
  - SSL: Secure Sockets Layer
  - TLS: Transport Layer Security



- TLS bietet:
  - Vertraulichkeit: Dritte können nicht mitlesen
  - Integrität: Daten können nicht unbemerkt verändert werden
  - Authentizität: Zertifikat bestätigt die Identität des Servers
- Bekannte Verwendung: HTTPS = HTTP + TLS
  - SSL-Zertifikat auf bestimmte(n) (Sub-) Domainnamen ausgestellt

Was ist ein Heimnetz?

- Glasfaser/DSL/Kabel-Modem mit Router, Switch, Access-Point
  - z.B. FRITZ!Box, Turris Omnia
- Nicht von außen erreichbares Netzwerk
  - Private IPv4-Adressen mit (Provider-)NAT
  - Dynamischer IPv6-Präfix
- Clients
  - PC, Laptop, Smartphone, Tablet, Drucker
- Server/Geräte: Single-Board-Computer, NAS
  - z.B. Raspberry Pi, QNAP NAS, Synology NAS



HTTPS im Heimnetz?

- Verschlüsselung im lokalen Netzwerk sinnvoll
- Moderne Clients, insbesondere Browser, erwarten HTTPS
  - „HTTPS by default“ ab Google Chrome 154 im Oktober 2026
- Herausforderungen und Probleme bei SSL/TLS:
  - Clients, z.B. Browser, warnen bei selbst signierten SSL-Zertifikaten
  - Eigene Zertifizierungsstelle (CA): Komplex und Pflegeaufwand
  - Import bei Smartphones, Tablets, Smart-TV schwierig bis unmöglich
- Aber: Öffentlich vertrauenswürdige Zertifizierungsstelle (CA) setzt für Validierung öffentliche Erreichbarkeit voraus

Domainname und DNS-Server?

- (Familien- oder Bastel-)Domain meistens bereits vorhanden
  - Let’s Encrypt im Heimnetz setzt öffentliche (Sub-)Domain voraus
  - Verwendung einer Subdomain möglich, z.B. `*.home.example.net`
- Entweder: Lokale DNS-Zone, z.B. auf Pi-hole oder OPNsense
  - Split-Horizon: Auflösung direkt im Heimnetz auf private IP-Adressen
  - Stolperfalle: DNSSEC beim Provider kann zu Problemen führen
- Oder: Private IP-Adressen in öffentlicher DNS-Zone
  - Pflege der internen DNS-Einträge dann beim Webhosting-Provider
  - Auflösung über öffentliche DNS-Server des Webhosting-Providers

Warum Let’s Encrypt?

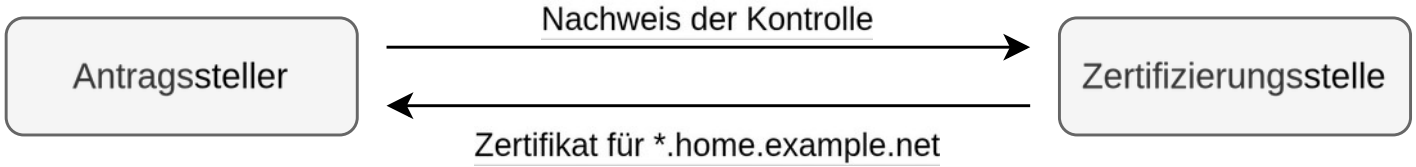
- CA/Browser Forum verkürzt Zertifikatslaufzeiten seit 01.07.2012
  - 2005: Zusammenschluss zur gemeinsamen Richtlinienerstellung
  - 2026: Über 50 Zertifizierungsstellen, über 10 (Browser-)Hersteller

≤ 60 Monate	≤ 39 Monate	≤ 825 Tage	≤ 398 Tage	≤ 200 Tage	≤ 100 Tage	≤ 47 Tage
01.07.2012	01.07.2016	01.03.2018	01.09.2020	15.03.2026	15.03.2027	15.03.2029

- Einstellung von Validierungen mit Medienbruch
  - Validierung per Telefon zum 15.03.2027, per E-Mail zum 15.03.2028
- Automatisierte Validierung und Zertifikatsaustausch erforderlich
  - Manuelle Zertifikatsverwaltung skaliert (bald) (gar) nicht mehr
- Es geht eigentlich um Automatisierung, nicht um Let’s Encrypt

Domainkontrolle

- Zertifizierungsstellen (CAs) müssen Kontrolle über Domain prüfen
  - Technische und organisatorische Vorgaben des CA Browser/Forums



  - Fehler können zu Vertrauensverlust der (Browser-)Hersteller führen
    - DigiNotar (2011), WoSign & StartCom (2017), Symantec (2018), Entrust (2024)
- Automation setzt automatisierte Domainkontrolle voraus:
  - RFC 8555: Automatic Certificate Management Environment (ACME)
  - Proprietäre APIs verschiedener kommerzieller Zertifizierungsstellen
  - Reseller haben teilweise eigene (andere) proprietäre APIs/Verfahren

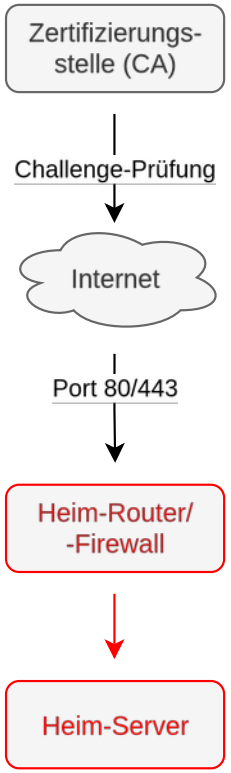
Zertifikatstransparenz

- Im Zertifikat enthaltene (Sub-)Domains sind öffentlich einsehbar
  - Betrifft alle Zertifikate aller öffentlich vertrauenswürdigen CAs
  - Sogenannte Certificate-Transparency-Logs (CT), z.B. https://crt.sh
  - Software-/Produktnamen wie `gitlab.home.example.net` vermeiden
  - Alternativ: Selbstsignierte Zertifikate bzw. eigene Zertifizierungsstelle
- Zertifizierungsstellen haben bereits falsche Zertifikate ausgestellt
  - Beispiel: Zertifikat für www.google.com durch Symantec (2015)
- Security by Obscurity: Wildcards können Offenlegung vermeiden
  - Nachteil: Gleicher privater Schlüssel evtl. auf verschiedenen Geräten
  - Abwägung zwischen Schlüsseltrennung und Informationsvermeidung

Challenges: HTTP-01 und TLS-ALPN-01

ACME setzt Domainkontrolle über sogenannte „Challenges“ um

- Challenge HTTP-01 (Port 80):
  - CA ruft HTTP-URL `/.well-known/acme-challenge/…` ab
  - ACME-Client stellt erwartete Antwort auf dieser URL bereit
- Challenge TLS-ALPN-01 (Port 443):
  - CA prüft Domain über speziellen TLS-ALPN-Handshake
  - ACME-Client präsentiert temporäres Challenge-Zertifikat



Aber: Keine direkte Erreichbarkeit für CA im Heimnetz

- OPNsense, Traefik, Caddy, NGINX Proxy Manager

Challenge: DNS-PERSIST-01

- Domainkontrolle über DNS-TXT-Eintrag statt Serverzugriff
  - `_validation-persist.home.example.net. IN TXT ("letsencrypt.org;" " accounturi=https://acme-v02.api.letsencrypt.org/acme/acct/…")`
- DNS-Server werden meist vom Webhosting-Provider betrieben
  - Keine Erreichbarkeit des Heimnetzes durch die CA notwendig
- Eher bequeme als sichere Lösung:
  - Aus Nachweis für Challenge wird eine dauerhafte Berechtigung
  - Fehlerhafte oder vergessene Einträge bleiben dauerhaft aktiv
  - Ablaufdatum kann mitangegeben werden – wer pflegt das manuell?
- Let’s Encrypt unterstützt Challenge nicht (Stand 04.07.2026)

Challenge: DNS-01

- Domainkontrolle über DNS-TXT-Eintrag statt Serverzugriff
  - `_acme-challenge.home.example.net. IN TXT "…"`
- DNS-Server werden meist vom Webhosting-Provider betrieben
  - Keine Erreichbarkeit des Heimnetzes durch die CA notwendig
- Unterschiede zur Challenge „DNS-PERSIST-01“:
  - Breite Unterstützung der Challenge durch CAs und ACME-Clients
  - ACME-Client setzt passenden DNS-TXT-Eintrag temporär
  - Dynamische Aktualisierungsmöglichkeit der DNS-Zone erforderlich

Anforderungen an DNS-Anbieter

Verschiedene technische Umsetzungsmöglichkeiten:

- API bei DNS/Webhosting-Provider zur Pflege von TXT-Records
  - z.B. Hetzner, netcup, deSEC, Scaleway DNS, DNScale, ClouDNS, OVHcloud, IONOS Cloud DNS, Gcore DNS, Bunny DNS, RcodeZero
- Provider mit Unterstützung von:
  - RFC 2136 (Dynamic DNS) im DNS-Server für `nsupdate`
  - Hidden Primary DNS: Primary muss für Secondaries erreichbar sein
- Auslagerung der ACME-Challenge in eine eigene DNS-Zone
- Betrieb eigener öffentlicher DNS-Server

ACME-Client

- Implementiert das ACME-Protokoll (RFC 8555)
- Fordert Zertifikate von CA an und erneuert sie automatisch
- Führt die erforderlichen Challenges durch
- Kann Skripte oder Hooks für die Zertifikatsverteilung aufrufen
- Verschiedene Implementationen:
  - Certbot: Bekanntester ACME-Client (in Python), sehr viele Plugins
  - dehydrated: Kompaktes und gut lesbares Bash-Skript
  - acme.sh: Maximale Portabilität durch POSIX-konformes Shell-Skript
- Wird im Heimnetzwerk betrieben, z.B. auf Linux-Systemen

Umsetzungsbeispiel mit ACME-Client (1)

- Beispiel mit Hetzner auf Fedora 44 und Certbot in 7 Schritten:
  - `dnf install certbot python3-pip`
  - `pip install certbot-dns-hetzner`
  - Bei Hetzner anmelden und API-Token generieren wie beschrieben unter: https://docs.hetzner.com/de/cloud/api/getting-started/generating-api-token/
  - `echo 'dns_hetzner_api_token = <…>' > /etc/letsencrypt/hetzner.ini`
  - `chmod 600 /etc/letsencrypt/hetzner.ini`
  -
```
certbot certonly --dns-hetzner \
  --dns-hetzner-credentials /etc/letsencrypt/hetzner.ini \
  --domain home.example.net --domain '*.home.example.net'
```
  - `systemctl enable --now certbot-renew.timer`
- Siehe: https://github.com/rolschewsky/certbot-dns-hetzner-cloud

Umsetzungsbeispiel mit ACME-Client (2)

- Beispiel für RFC 2136 (Dynamic DNS) via `nsupdate` mit Certbot
  - `certbot certonly --manual --manual-auth-hook '/path/to/hook auth' \ --manual-cleanup-hook '/path/to/hook cleanup' -d '*.home.example.net'`
  - Beispiel für benötigtes Skript `/path/to/hook`:

```
#!/bin/bash

case "$1" in
  "auth")
    printf 'server ns.example.net\nupdate add _acme-challenge.%s 30 IN TXT "%s"\nsend\n' \
      "${CERTBOT_DOMAIN}" "${CERTBOT_VALIDATION}" | nsupdate -k /etc/rndc.key
    sleep 30  # Wait for DNS propagation (DNS caching, zone transfers)
    ;;
  "cleanup")
    printf 'server ns.example.net\nupdate delete _acme-challenge.%s TXT "%s"\nsend\n' \
      "${CERTBOT_DOMAIN}" "${CERTBOT_VALIDATION}" | nsupdate -k /etc/rndc.key
    ;;
esac
```

Verteilungsmöglichkeiten

- Zertifikate anfordern und verlängern ist nur der erste Schritt
- Mechanismen zur Verteilung:
  - `scp` / `sftp` bzw. `rsync` über SSH, z.B. Linux-Systeme
  - APIs, z.B. OPNsense, QNAP NAS, Synology NAS
  - HTTP POST, z.B. FRITZ!Box
  - Konfigurationsmanagement: Ansible, OpenVox, Chef, Salt, …
- Dienste müssen meist neu geladen oder neu gestartet werden
  - z.B. Neuladen des Apache HTTP Servers, Neustart beim vsFTPd
- Verteilung und Rollout sind (meist) der eigentliche Aufwand

Umsetzungsbeispiel für Verteilung

- Beispiel zur Verteilung auf anderen Server mittels Deploy-Hook
  - `certbot certonly --deploy-hook /path/to/deploy \`
    `--domain home.example.net --domain '*.home.example.net'`
  - Beispiel für benötigtes Skript `/path/to/deploy`:

```
#!/bin/bash

chmod 600 "${RENEWED_LINEAGE}/privkey.pem"
scp -p "${RENEWED_LINEAGE}/privkey.pem" \
  root@web.home.example.net:/etc/pki/tls/private/home.example.net.pem
scp -p "${RENEWED_LINEAGE}/fullchain.pem" \
  root@web.home.example.net:/etc/pki/tls/certs/home.example.net.pem
ssh root@web.home.example.net 'systemctl reload httpd.service'
```

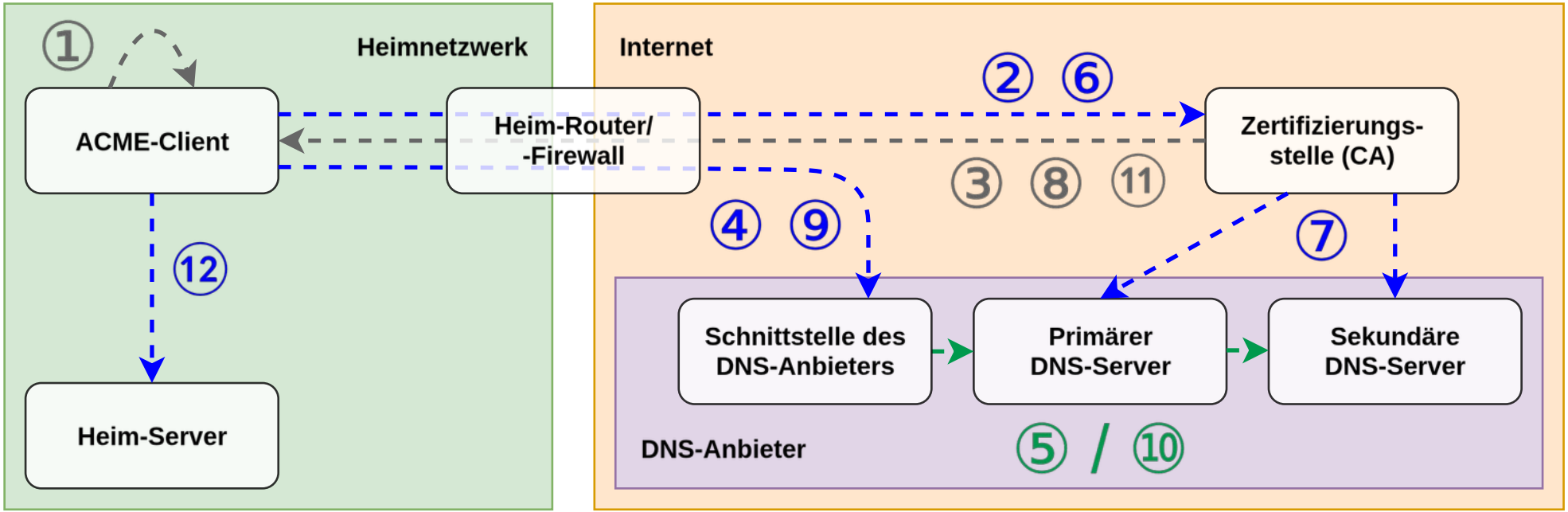
- Deploy-Hook muss ausführbar sein, kann in beliebiger Skript- oder Programmiersprache implementiert werden

Vollständiges Certbot-Beispiel mit Hooks

- Beispiel für Certbot mit Hooks für Challenge und Verteilung
- Erforderlich wenn es keine passenden Certbot-Plugins gibt
  - `dnf install certbot`
  - ```
    certbot certonly \
      --manual \
      --manual-auth-hook '/path/to/hook auth' \
      --manual-cleanup-hook '/path/to/hook cleanup' \
      --deploy-hook /path/to/deploy \
      --domain home.example.net \
      --domain '*.home.example.net'
    ```
  - `systemctl enable --now certbot-renew.timer`
- Benötigt eigene Skripte `/path/to/hook` und `/path/to/deploy`
- Plugins: https://eff-certbot.readthedocs.io/en/stable/using.html
    ```

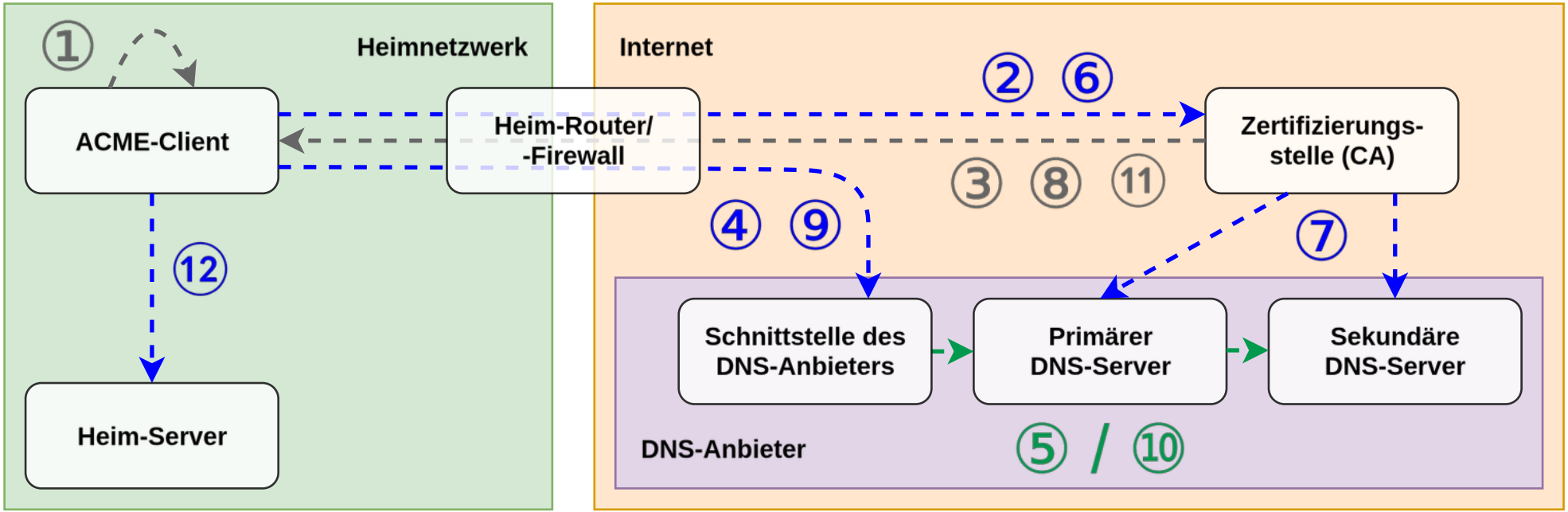
Vereinfachte Gesamtübersicht

1. ACME-Client erstellt privaten Schlüssel und Zertifikatsanfrage (CSR)



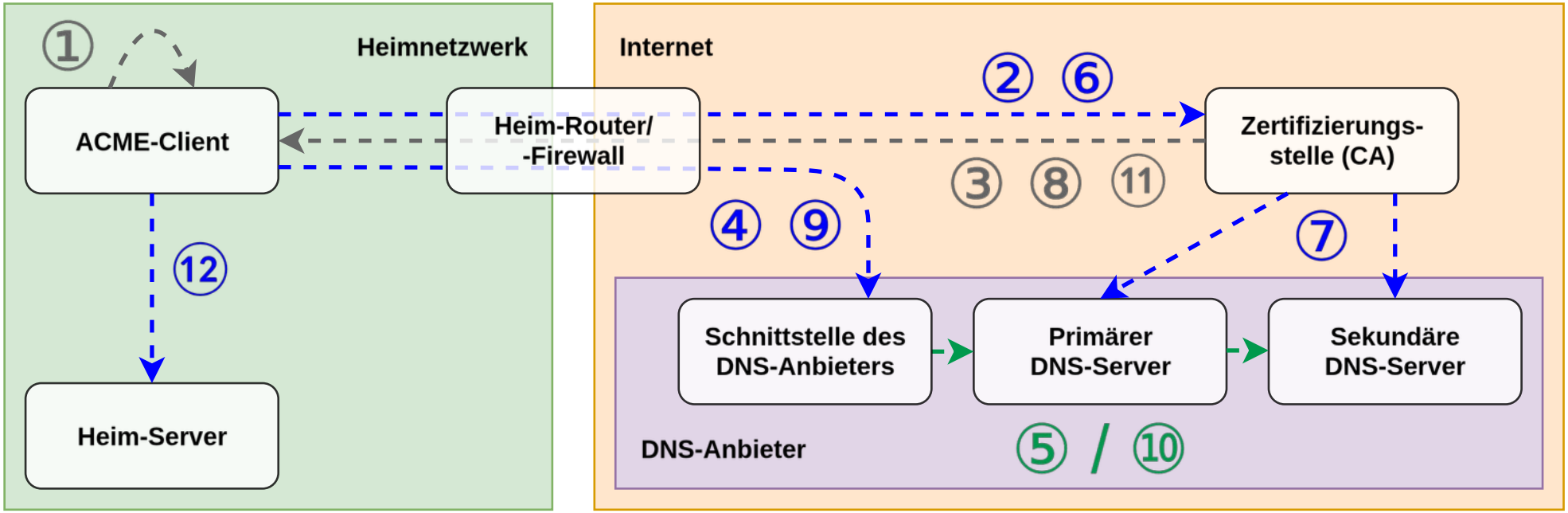
Vereinfachte Gesamtübersicht

2. ACME-Client stellt Zertifikatsanforderung an Zertifizierungsstelle (CA)



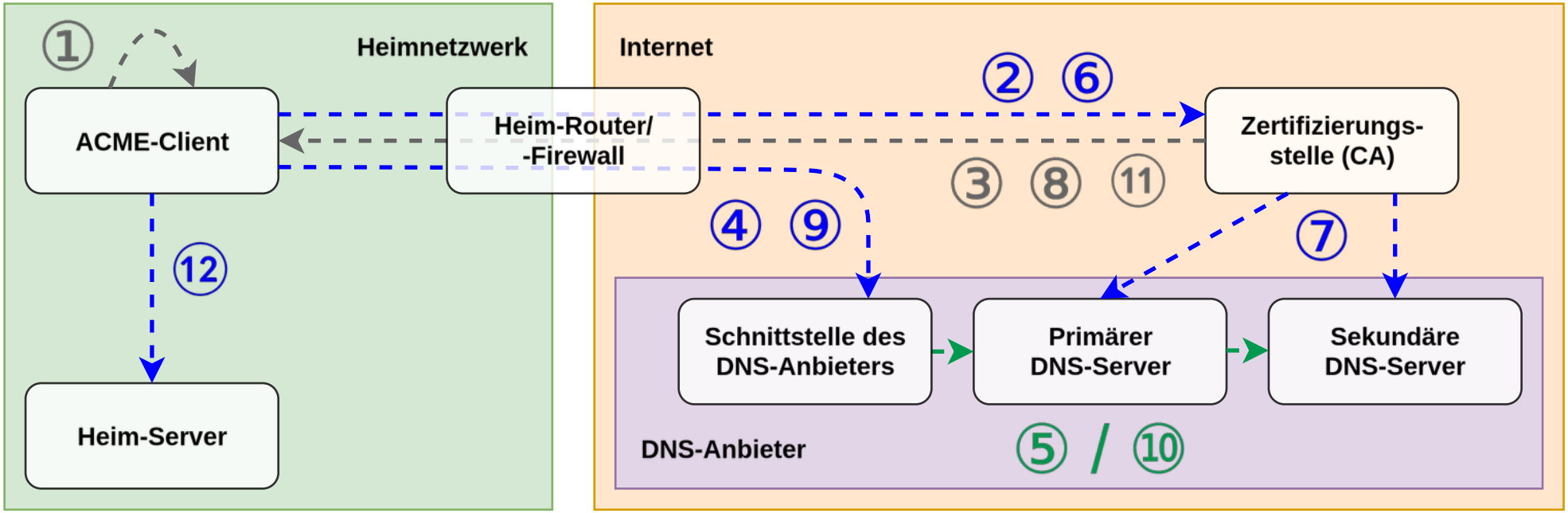
Vereinfachte Gesamtübersicht

3. Zertifizierungsstelle meldet zurück was im DNS zu hinterlegen ist



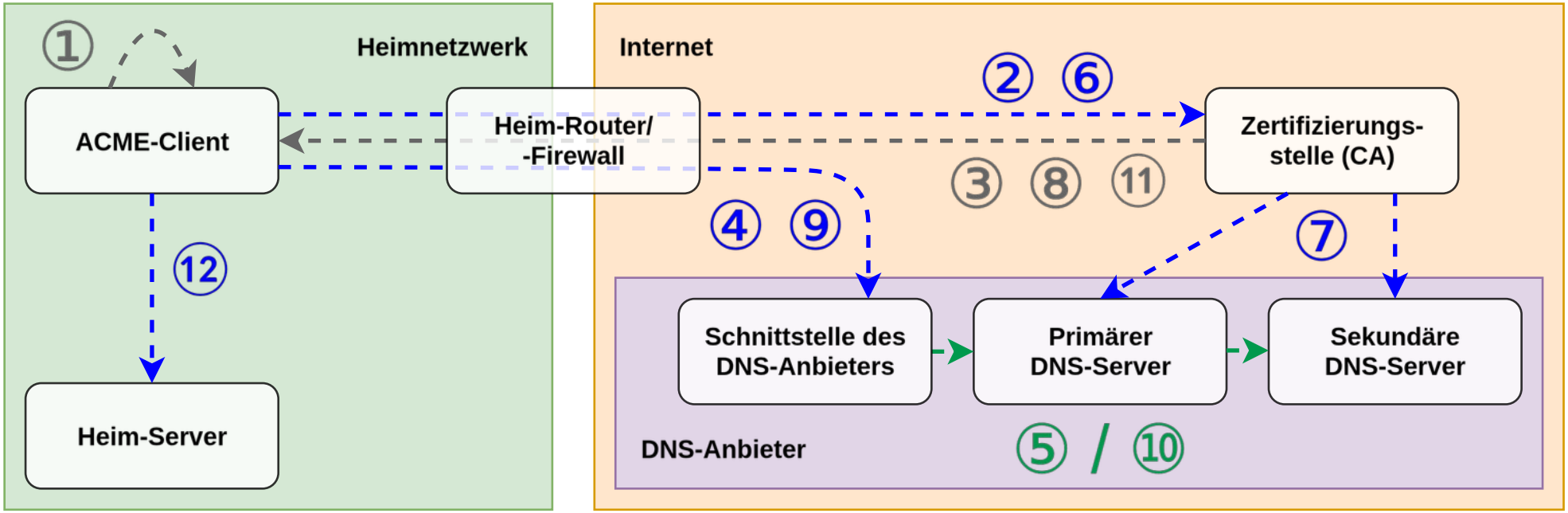
Vereinfachte Gesamtübersicht

4. ACME-Client veranlasst DNS-Eintrag über Hook/Skript beim DNS-Anbieter



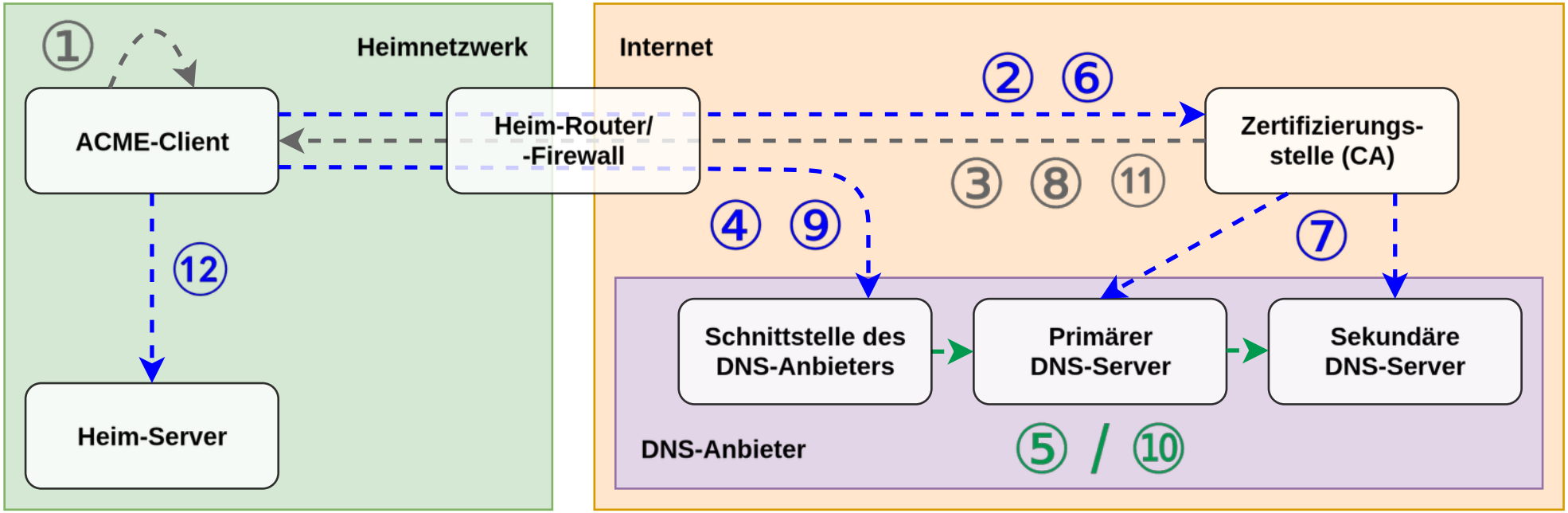
Vereinfachte Gesamtübersicht

5. Warten bis die DNS-Anpassung an alle DNS-Server repliziert wurde



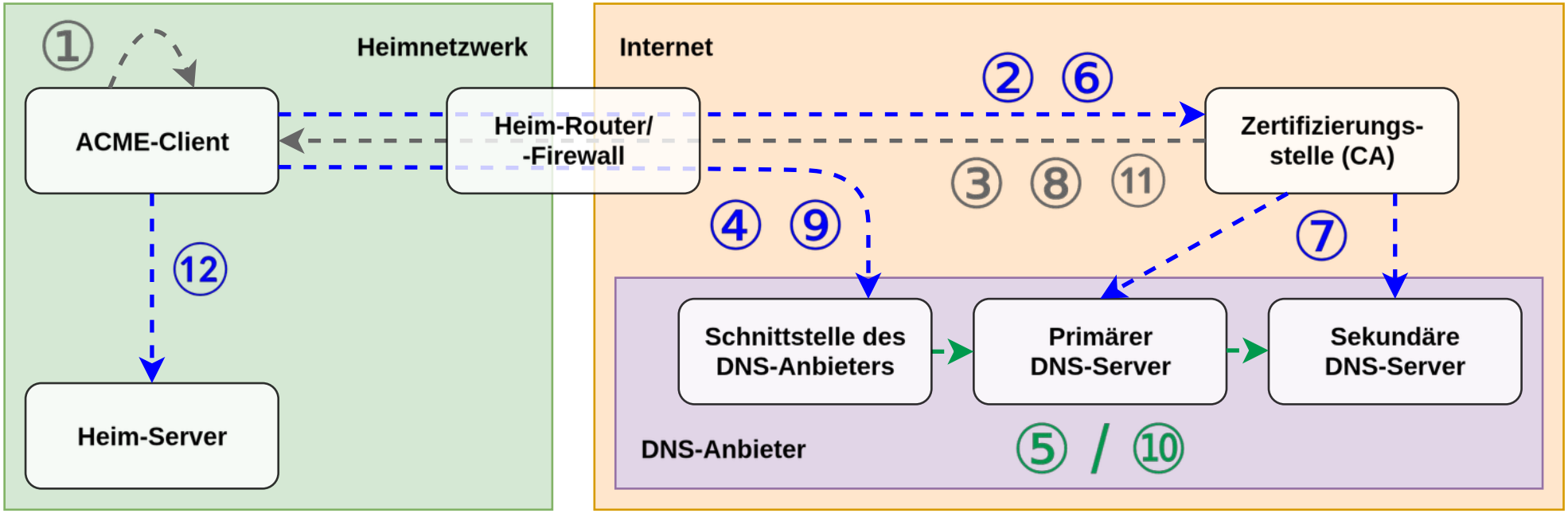
Vereinfachte Gesamtübersicht

6. ACME-Client teilt der Zertifizierungsstelle mit, dass DNS-Eintrag vorhanden ist



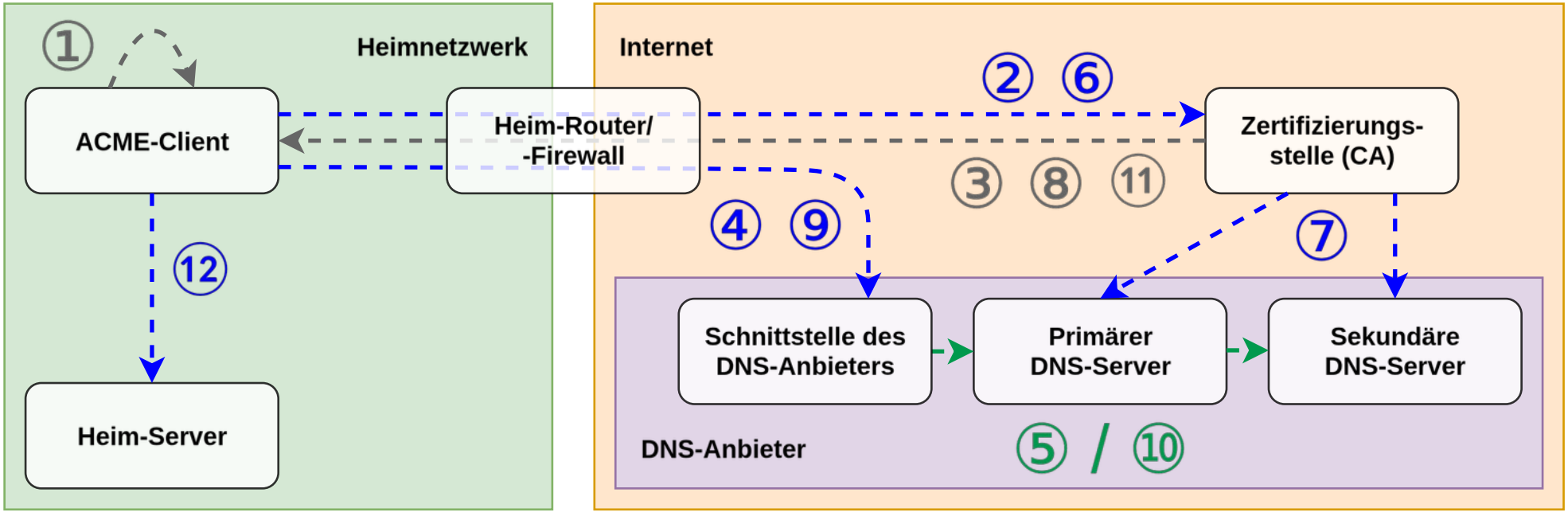
Vereinfachte Gesamtübersicht

7. Zertifizierungsstelle frägt DNS-Eintrag bei DNS-Servern ab und prüft diesen



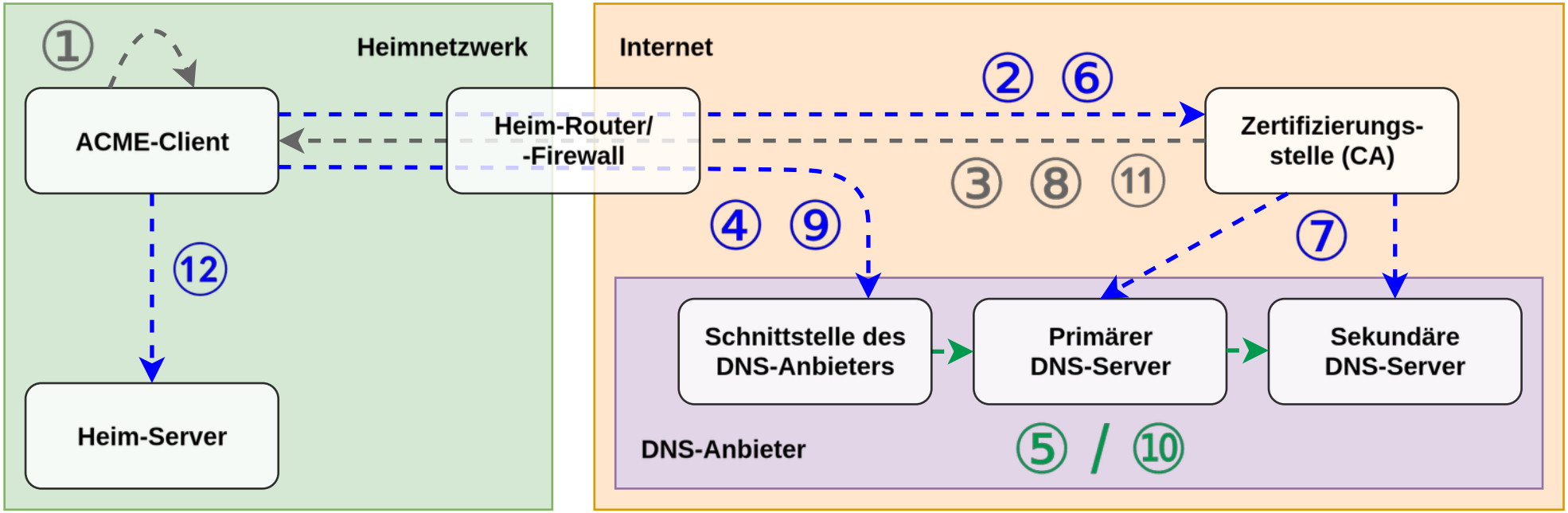
Vereinfachte Gesamtübersicht

8. ACME-Client erhält Rückmeldung über Validierung durch Zertifizierungsstelle



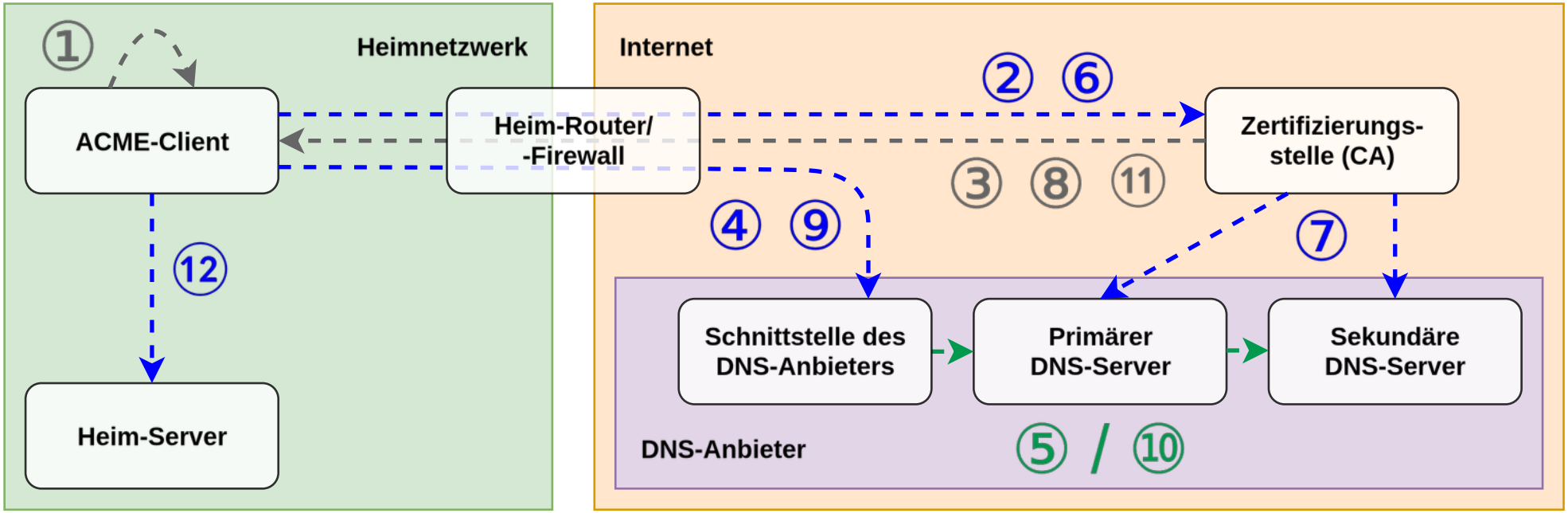
Vereinfachte Gesamtübersicht

9. ACME-Client entfernt DNS-Eintrag über Hook/Skript beim DNS-Anbieter



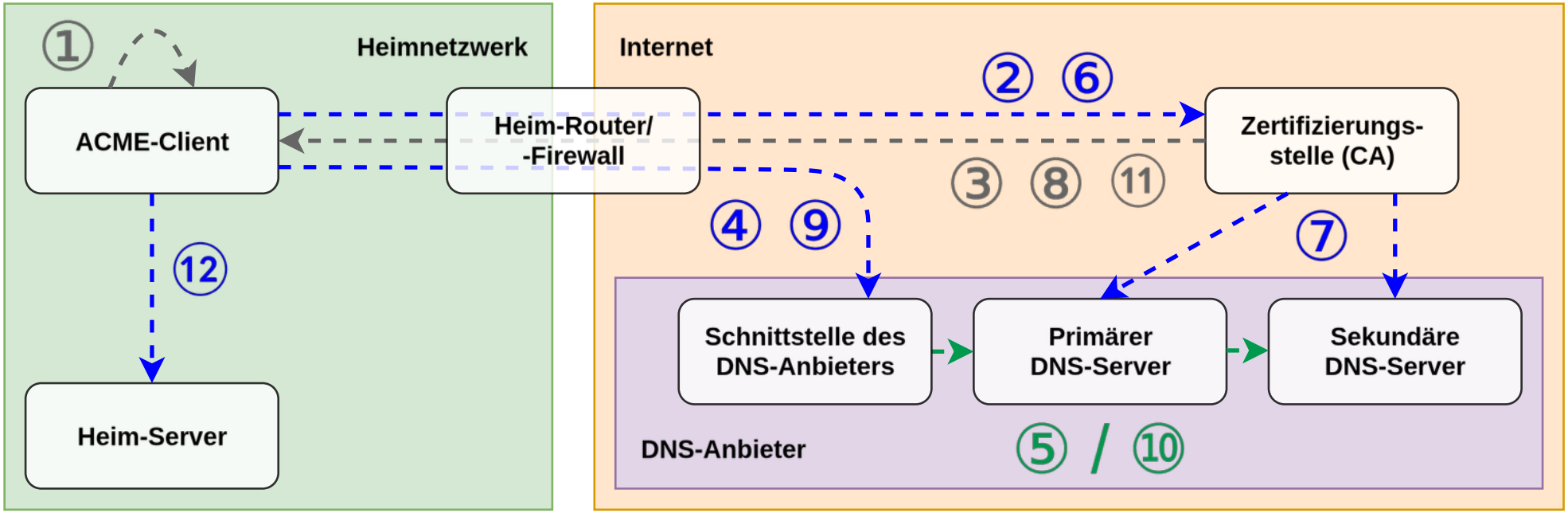
Vereinfachte Gesamtübersicht

10. Warten bis die DNS-Anpassung an alle DNS-Server repliziert wurde



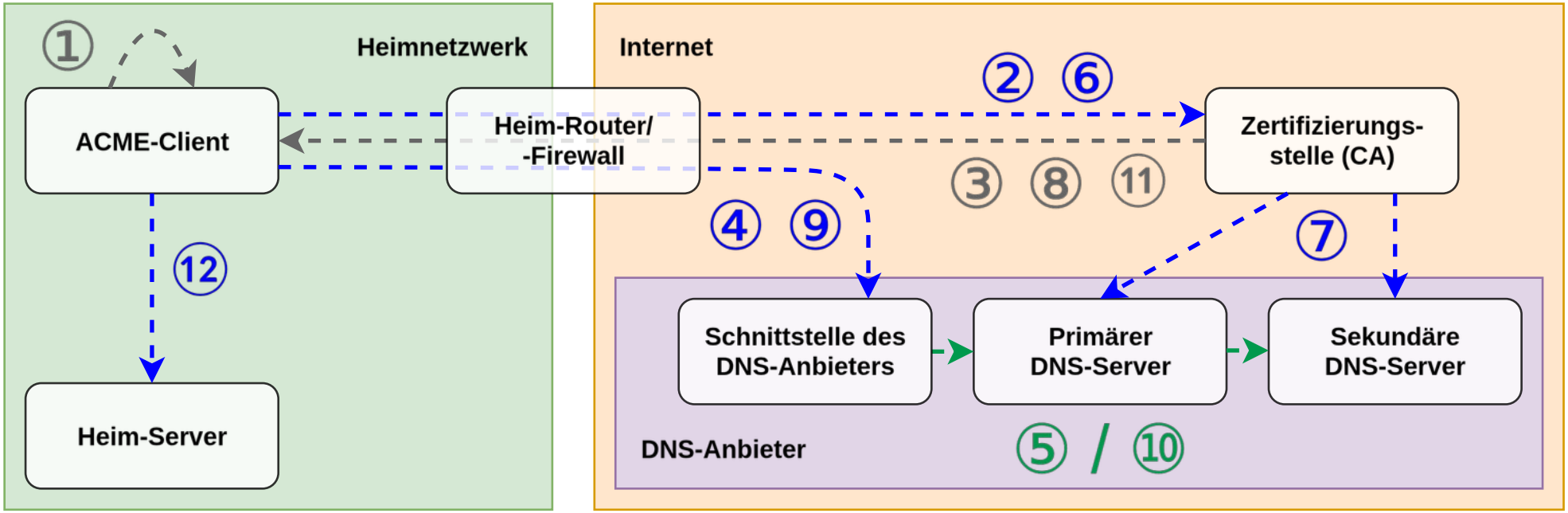
Vereinfachte Gesamtübersicht

11. ACME-Client erhält ausgestelltes SSL-Zertifikat von Zertifizierungsstelle



Vereinfachte Gesamtübersicht

12. ACME-Client veranlasst Verteilung des SSL-Zertifikats über Hook/Skript



Vom Heimnetz ins Firmennetz

- Zertifikatsaustausch bei schlechter/kaputter Software schwierig
- Überwachung:
  - Integration in bestehende Monitoring-Systeme, z.B. Nagios, Icinga2
  - Früherkennung fehlgeschlagener Zertifikatsverlängerungen
  - Benachrichtigungen bei ablaufenden oder ungültigen Zertifikaten
- Kommerzielle Zertifizierungsstellen:
  - OV (Organization Validated) und EV (Extended Validation) möglich
  - ACME-Anbindung über External Account Binding (EAB)
- Datensicherung von Konfigurationen und Rollout-Werkzeugen
- Redundanzen, z.B. mehrere Zertifizierungsstellen

Fazit

- Eigentlich gings nicht um „Let’s Encrypt im Heimnetzwerk“
  - Sondern um ACME, viel Automatisierung – und DNS
- Öffentliche SSL-Zertifikate sind auch im Heimnetzwerk möglich
  - DNS-01 meist die beste ACME-Challenge für Domainkontrolle
  - DNS muss sich automatisieren lassen (API oder Dynamic DNS)
  - Automatisierung der Zertifikatserneuerung und -verteilung
- Gleiche Konzepte funktionieren auch im Unternehmensumfeld


Fragen?


Vielen Dank!