# Routing über Consul

Router per Consul-Tags, wie der Edge den Katalog liest, TCP-Routen.

- Router per Consul-Tags definieren
- So liest der Edge den Consul-Katalog
- TCP-Routen für Datenbanken

# Router per Consul-Tags definieren

> Zielgruppe: Anwender und Betreuer · Stand: 06.11.2026

Der Edge-Traefik baut seine Router aus den Tags der Services im Consul-Katalog. Diese Seite zeigt, welche Tags Du an Deinen Service schreibst, damit er unter einem Hostnamen am Edge erreichbar ist. Wie Du den Service überhaupt registrierst, steht in Dienst im Consul-Katalog registrieren.

# Voraussetzungen

- Der Dienst ist im Consul-Katalog registriert und sein Health-Check ist grün.
- Er ist eine Oberfläche für Menschen und gehört damit an den Edge (siehe Edge-Grundsatz ).
- Der Hostname liegt unter der Wildcard der Umgebung. **Lab-PoC:** `*.lab.bre1.dexpro.services`.

# Vorgehen

1. **Freischalten:** Der Edge übernimmt keinen Service von selbst. Setz `traefik.enable=true` und den Stage-Tag des Edge. **Lab-PoC:** Das Lab ist die Stage `pre-development` und läuft im sogenannten DEV-Netz, der Stage-Tag heißt deshalb `dev`.
2. **Router anlegen:** Ein Router braucht einen eindeutigen Namen, eine Regel, einen Entrypoint und TLS:

```
tags = [
  "<stage-tag>",
  "traefik.enable=true",
  "traefik.http.routers.<router>.rule=Host(`<hostname>`)",
  "traefik.http.routers.<router>.entrypoints=https",
  "traefik.http.routers.<router>.tls=true",
]
```

- `<router>`: plattformweit eindeutiger Name, z. B. Dienstname plus Stage (`grafana-pilot`).

- `<hostname>`: voller Hostname unter der Wildcard, z. B. `grafana-pilot.lab.bre1.dexpro.services`.
- `https` ist der interne Entrypoint des Edge. Öffentlich aus dem Internet erreichbar ist ein Router nur mit `entrypoints=public`. Das ist die Ausnahme und wird mit den Betreuern abgestimmt.

3. **Dienst im Service Mesh:** Hängt Dein Dienst per Connect im Mesh (z. B. Nomad-Job mit `sidecar_service`), ergänze:

```
"traefik.consulcatalog.connect=true",
```

Der Edge spricht den Dienst dann per mTLS über Connect an. Dafür braucht es eine Intention `edge-traefik-<stage>` → `<dienst>`, die die Betreuer per Terraform anlegen.

4. **Optional, Zugriff beschränken:** Admin-Oberflächen hängst Du hinter die IP-Allowlist des Edge:

```
"traefik.http.routers.<router>.middlewares=admin-allowlist@file",
```

5. **Optional, Backend mit HTTPS:** Spricht Dein Dienst selbst HTTPS, prüft der Edge dessen Zertifikat über ein Backend-TLS-Profil, das die Betreuer am Edge hinterlegen:

```
"traefik.http.services.<router>.loadbalancer.server.scheme=https",
"traefik.http.services.<router>.loadbalancer.serverstransport=<profil>@file",
```

`<profil>` ist der Name des Backend-TLS-Profils. Frag die Betreuer, welches zu Deinem Dienst passt.

6. **Optional, anderer Port:** Ohne Angabe nutzt der Edge den Port aus der Service-Registrierung. Einen anderen Port setzt Du mit `traefik.http.services.<router>.loadbalancer.server.port=<port>`.

# Beispiel

Grafana als Nomad-Job im Mesh, erreichbar im Lab (Lab-PoC):

```
tags = [
  "dev",
  "traefik.enable=true",
  "traefik.consulcatalog.connect=true",
  "traefik.http.routers.grafana-pilot.rule=Host(`grafana-pilot.lab.bre1.dexpro.services`)",
  "traefik.http.routers.grafana-pilot.entrypoints=https",
```

```
  "traefik.http.routers.grafana-pilot.tls=true",
]
```

# Prüfen

- Im Traefik-Dashboard des Edge taucht der Router mit Status „Success“ auf.
- Prüf den Hostnamen im Browser oder per `curl`:

```
curl -sI https://<hostname>
```

Du bekommst die Antwort Deines Dienstes mit gültigem Zertifikat, kein `404` von Traefik.

# Stolperfallen

> Falscher Stage-Tag oder fehlendes `traefik.enable=true`: Der Dienst steht im Katalog, der Edge ignoriert ihn aber. Es gibt keine Fehlermeldung, der Hostname liefert `404`.

- **Doppelter Router-Name:** Zwei Services mit gleichem Router-Namen, aber anderer Konfiguration verwirft Traefik. Wähl eindeutige Namen.
- **Fehlende Intention:** Bei `traefik.consulcatalog.connect=true` ohne Intention erreicht der Edge den Dienst nicht, weil Default-Deny die Verbindung blockt.
- **Backticks:** Die Regel braucht Backticks um den Hostnamen (`Host(`…`)`), keine Anführungszeichen.
- **Keine Anmeldung:** Der Edge ersetzt keine Authentifizierung. Die Oberfläche braucht eine eigene Anmeldung oder mindestens `admin-allowlist@file`.

# Weiterführend

- So liest der Edge den Consul-Katalog
- TCP-Routen für Datenbanken
- Beispiele in GitLab: `container-platform-core-services/nomad-jobs/pilot-grafana`, `container-platform-core-services/nomad-jobs/traefik-ingress`
- Traefik-Doku: Consul Catalog Provider

# So liest der Edge den Consul-Katalog

> Zielgruppe: Betreuer · Stand: 06.11.2026

Diese Seite erklärt, wie der Edge-Traefik den Consul-Katalog liest: welche Services er übernimmt, mit welchem Token und wie er Dienste im Service Mesh erreicht. Du brauchst sie, wenn Du einen Edge aufbaust oder suchst, warum ein Router fehlt.

# Voraussetzungen

- Auf jedem Edge-Knoten läuft ein Consul-Client. Traefik spricht ihn lokal über `127.0.0.1:8500` an.
- Für den Knoten gibt es ein Token der Policy `traefik-catalog` (Muster `traefik-catalog-<stage>`):

```
service_prefix "" { policy = "read" }
node_prefix ""    { policy = "read" }
service "edge-traefik-<stage>" { policy = "write" }
```

Tokens und Policies entstehen per Terraform im Stack `cloud-core-infrastructure/bremen1/consul-servers`.

# Vorgehen

1. **Rolle `traefik` mit Consul-Provider ausrollen.** Der Provider wird nur gerendert, wenn die Rolle mit dem Ansible-Tag `consul` läuft. Die Rolle erzeugt dann in `config.toml`:

```
[providers.consulCatalog]
  exposedByDefault = false
  watch = true
  constraints = "Tag(`<stage>`)"
  connectAware = true
  serviceName = "edge-traefik-<kv_stage>"
  [providers.consulCatalog.endpoint]
```

```
    address = "127.0.0.1:8500"
    scheme = "http"
```

- `exposedByDefault = false`: Nur Services mit `traefik.enable=true` werden übernommen.
- `constraints`: Nur Services mit dem Tag der technischen Stage (`stage`) werden übernommen.
- `connectAware` und `serviceName` setzt die Rolle nur bei `traefik_consul_connect_aware: true`.
- Das Token steht in `config.toml`. Die Datei ist deshalb nur für `root` und die Gruppe `traefik` lesbar.

2. **Variablen setzen** (Rolle `traefik`):

| Variable | Bedeutung |
|---|---|
| `traefik_consul_token` | Token der Policy `traefik-catalog`. Kommt aus der Secret-Variable, nie im Repo. |
| `traefik_consul_kv_stage` | Stage für KV-Präfixe und den Connect-Namen `edge-traefik-<kv_stage>`. Default: `stage`. |
| `traefik_consul_connect_aware` | `true` schaltet Connect ein (Weg zu Diensten im Mesh). |
| `traefik_consul_address` / `traefik_consul_scheme` | Consul-Endpoint, Default `127.0.0.1:8500` / `http`. |

3. **Selbstregistrierung prüfen.** Die Rolle registriert den Edge selbst in Consul als `edge-traefik-<kv_stage>` (Tag `loadbalancer`, Port 443, Check gegen den lokalen Ping). Mit Connect ist es ein Connect-Native-Service. Registriert wird mit dem Traefik-Token, weil das Agent-Token bei Default-Deny keinen Service schreiben darf.
4. **Intentions anlegen.** Für jeden Dienst im Mesh, den der Edge erreichen soll, braucht es eine Intention `edge-traefik-<kv_stage>` → `<dienst>`. Sie wird nur per Terraform im Stack `bremen1/consul-servers` (`config_entries`) ergänzt.

> Lab-PoC: Im Lab ist die technische Stage `dev` (Lab = Stage `pre-development` im sogenannten DEV-Netz), die KV-Stage aber `lab`. Der Constraint filtert also auf den Tag `dev`, der Connect-Name ist `edge-traefik-lab`.

# Prüfen

```
consul catalog services -tags
consul catalog nodes -service=edge-traefik-<kv_stage>
```

Der Edge-Service ist registriert und sein Check ist grün. Im Traefik-Dashboard stehen die Router der Services mit Provider `consulcatalog`.

# Stolperfallen

- **Reihenfolge beim Einschalten der ACL:** Erst das Traefik-Token ausrollen, dann die ACL einschalten. Sonst verliert der Edge alle Catalog-Routen. Ein Default-Token wird nicht gesetzt.
- **Manuelle Intentions:** Ein Neuaufbau des ACL-Setups setzt manuell angelegte Intentions zurück. Danach sind Dienste im Mesh am Edge nicht mehr erreichbar.
- **Rollenlauf ohne Tag `consul`:** Dann fehlt der Consul-Provider in `config.toml`. Der Edge zeigt nur noch die Router aus dem File-Provider.
- **Restart bei Konfigurationsänderung:** Ein Rollenlauf mit geänderter Konfiguration startet Traefik neu. Abgelaufene Backend-Zertifikate fallen erst dabei auf.

# Weiterführend

- Router per Consul-Tags definieren
- Dienst im Consul-Katalog registrieren
- README der Rolle in GitLab: `cloud-core-infrastructure/ansible/roles/traefik`
- Traefik-Doku: Consul Catalog Provider

# TCP-Routen für Datenbanken

> Zielgruppe: Betreuer · Stand: 06.11.2026

Datenbanken erreichen Clients über eigene TCP-Entrypoints am Edge, z. B. Port `33xx` für einen MariaDB-Cluster hinter MaxScale. Der Entrypoint kommt aus Consul KV, der Router aus den Tags des Datenbank-Service. Diese Seite zeigt beide Teile. Lab-PoC: Umgesetzt ist das bisher nur im Lab.

# Voraussetzungen

- Der Datenbank-Dienst ist im Consul-Katalog registriert, z. B. über die Rolle `consul.maxscale`.
- Schreibrecht auf `traefik/<kv_stage>/tcp/` (Policy `db-stack-<stage>`) für den Stack, der die Datenbank baut.
- Der Edge-Stack liest den Präfix mit einem eigenen Token (Policy `traefik-pipeline-<stage>`, nur lesen).

# Vorgehen

1. **Entrypoint in KV eintragen.** Je Entrypoint ein Schlüssel `traefik/<kv_stage>/tcp/<entrypoint>` mit JSON-Wert:

```
{"port": <port>, "service": "<dienst>"}
```

- `<entrypoint>`: Name des Entrypoints, z. B. `<cluster>-sql`.
- `<port>`: Port am Edge, z. B. aus dem Bereich `33xx`.

2. **Edge-Stack neu ausrollen.** Entrypoints sind statische Konfiguration. Die Pipeline des Edge-Stacks liest den KV-Präfix beim Plan und gibt die Entrypoints an die Rolle `traefik` (`traefik_tcp_entrypoints`). Traefik lauscht dann auf der internen VIP (`traefik_tcp_bind_address`). Ein CIDR-Präfix an der Adresse schneidet die Rolle ab.
3. **Router per Tags am Service.** Die Rolle `consul.maxscale` setzt sie so:

```
tags = [
  "traefik.enable=true",
  "<stage-tag>",
  "traefik.tcp.routers.<name>.rule=HostSNI(`*`)",
```

```
  "traefik.tcp.routers.<name>.entrypoints=<entrypoint>",
  "traefik.tcp.routers.<name>.service=<name>",
  "traefik.tcp.services.<name>.loadbalancer.server.port=<db-port>",
]
```

`HostSNI(*)` leitet reines TCP weiter. Die TLS-Verbindung, die MariaDB im Protokoll aushandelt, läuft unverändert durch.

# Prüfen

- Im Traefik-Dashboard steht der TCP-Router auf dem neuen Entrypoint mit Status „Success“.
- Ein Client verbindet sich über `<edge-hostname>:<port>` mit erzwungenem TLS.

# Stolperfallen

> Die Datenbank sieht als Client-Adresse die Adresse des Edge, nicht die des echten Clients. Grants mit Host-Beschränkung müssen das berücksichtigen.

- **Neuer Schlüssel ohne Rollout:** Ein neuer KV-Eintrag wirkt erst nach einem Lauf des Edge-Stacks.
- **Ohne Consul kein Plan:** Ist Consul nicht erreichbar, kann der Edge-Stack seine TCP-Entrypoints nicht lesen und nicht planen.

# Weiterführend

- Router per Consul-Tags definieren
- So liest der Edge den Consul-Katalog
- Rolle in GitLab: `cloud-core-infrastructure/ansible/roles/consul.maxscale`
- Traefik-Doku: TCP-Router