

STATUS · Methoden-Protokoll · API- und Spurentrennung · v0.4 · publikationsbereit
ZAHLEN · keine empirischen Zahlen; Vorlagen ohne echte Schlüssel
PARTNER · Ein-Mensch-Werkstatt · unabhängig · privat finanziert
DATUM · 2026-09-23 · Kennung FORSCH-PROV0446 (PROVISORISCH)
AUTOR · Thomas Schüller · Syntagma Forschung · Wien · ORCID 0009-0003-9799-6747

API- und Spurentrennung — .env-Vorlagen und Prüfwerkzeug

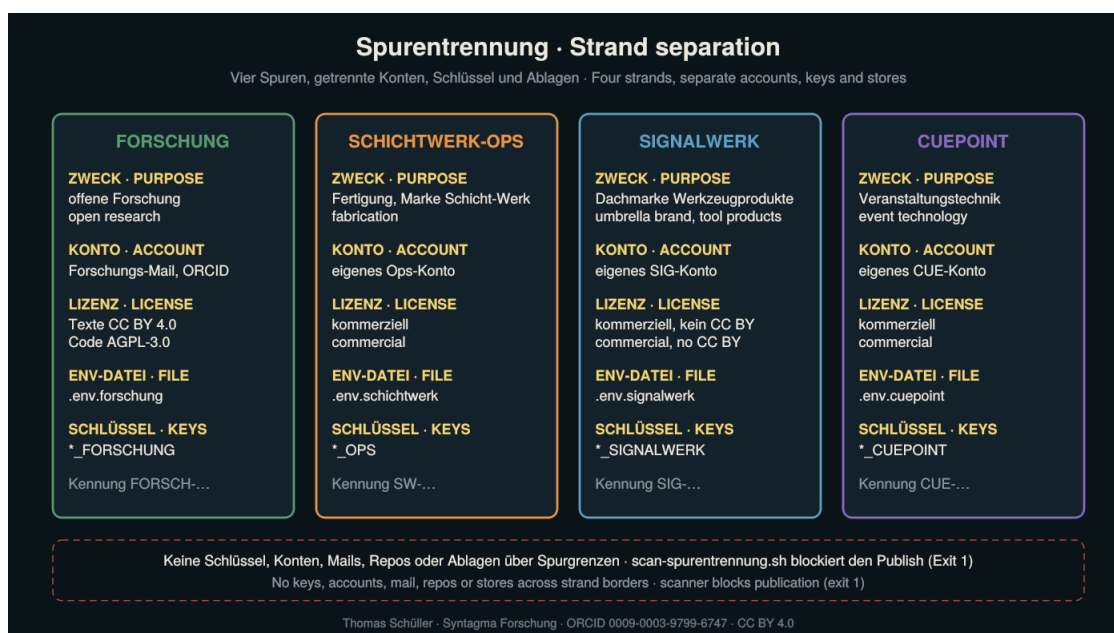


Abbildung 1: Vier Spuren mit getrennten Konten, Schlüsseln, Lizenzen und Ablagen. Das Prüfwerkzeug blockiert die Veröffentlichung, sobald Spuren einer fremden Spur im Bündel stehen.

Grundregel: je Spur eine eigene .env-Datei, ein eigenes Konto, eigene Schlüssel, eigene Repositorien. Nie ein Schlüssel einer Spur in der Datei einer anderen. Diese Fassung ist bewusst ohne Betriebsdaten der kommerziellen Spuren geschrieben: Adressen, Mail-Konten und Anbieter stehen als Platzhalter.

Die vier Spuren

Spur	Zweck	Lizenz der Ergebnisse	Kennungsreihe	Schlüssel-Suffix
FORSCHUNG	offene Forschung (Syntagma Forschung)	Texte CC BY 4.0, Code AGPL-3.0	FORSCH-	_FORSCHUNG
SCHICHTWERK-OPS	kommerzielle Fertigung, Marke Schicht-	kommerziell	SW-	_OPS

Spur	Zweck	Lizenz der Ergebnisse	Kennungsreihe	Schlüssel-Suffix
	Werk			
SIGNALWERK	Dachmarke der Werkzeugprodukte	kommerziell, kein CC BY auf Produktcode	SIG-	_SIGNALWERK
CUEPOINT	Veranstaltungstechnik	kommerziell	CUE-	_CUEPOINT

Verzeichnis

~/secrets/

```
├── .env.forschung
├── .env.schichtwerk
├── .env.signalwerk
├── .env.cuepoint
└── README.md
```

Alle vier Dateien: `chmod 600`, außerhalb jedes Repositoriums, nie eingcheckedt.

Vorlage .env.forschung

```
TRACK=forschung
AUTHOR_NAME="Thomas Schüller"
AUTHOR_ORCID="0009-0003-9799-6747"
AUTHOR_MAIL="forschung@tschueller.com"
AUTHOR_BRAND="Syntagma Forschung · Wien"
ZENODO_TOKEN_FORSCHUNG=
ORCID_CLIENT_ID_FORSCHUNG=
ORCID_CLIENT_SECRET_FORSCHUNG=
ANTHROPIC_API_KEY_FORSCHUNG=
HOSTING_KEY_FORSCHUNG=
GIT_TOKEN_FORSCHUNG=
```

Vorlage .env.schichtwerk

```
TRACK=schichtwerk
COMPANY_MAIL=<Ops-Konto>
COMPANY_DOMAIN=<Ops-Domain>
HOSTING_HOST_OPS=
HOSTING_USER_OPS=
HOSTING_PASS_OPS=
SHOP_API_TOKEN_OPS=
ANTHROPIC_API_KEY_OPS=
GIT_TOKEN_OPS=
```

Vorlage .env.signalwerk

```
TRACK=signalwerk
BRAND_MAIL=<SIG-Konto>
BRAND_DOMAIN=<SIG-Domain>
LICENSE_SERVER_KEY_SIGNALWERK=
```

```
STORE_API_TOKEN_SIGNALWERK=  
ANTHRÖPIC_API_KEY_SIGNALWERK=  
GIT_TOKEN_SIGNALWERK=
```

Vorlage .env.cuepoint

```
TRACK=cuepoint  
BRAND_MAIL=<CUE-Konto>  
BRAND_DOMAIN=<CUE-Domain>  
HOSTING_KEY_CUEPOINT=  
ANTHRÖPIC_API_KEY_CUEPOINT=  
GIT_TOKEN_CUEPOINT=
```

Lade-Mechanismus (Python)

```
from pathlib import Path
```

```
SPUREN = {"forschung": "_FORSCHUNG", "schichtwerk": "_OPS",  
          "signalwerk": "_SIGNALWERK", "cuepoint": "_CUEPOINT"}
```

```
def load_track_env(track: str) -> dict:  
    assert track in SPUREN, f"unbekannte Spur: {track}"  
    env_path = Path.home() / "secrets" / f".env.{track}"  
    mode = env_path.stat().st_mode & 0o777  
    assert mode == 0o600, f"{env_path} muss chmod 600 sein"  
    out = {}  
    for line in env_path.read_text().splitlines():  
        line = line.strip()  
        if not line or line.startswith("#") or "=" not in line:  
            continue  
        key, _, val = line.partition("=")  
        out[key.strip()] = val.strip().strip('\"')  
    suffix = SPUREN[track]  
    fremd = [k for k in out if k.endswith(tuple(s for s in  
SPUREN.values() if s != suffix))]  
    assert not fremd, f"fremde Schlüssel in .env.{track}: {fremd}"  
    return out
```

Der Lader prüft Dateirechte und lehnt Schlüssel mit dem Suffix einer anderen Spur ab.

Browser-Werkzeuge (BYOK)

Browser-Werkzeuge (STIMME, AKTE, RECHT) nutzen keine .env. Nutzer:innen tragen den eigenen Schlüssel lokal ein:

```
const loadBYOK = (tool, provider) => localStorage.getItem(`$  
{tool}_byok_${provider}`);  
const saveBYOK = (tool, provider, key) => localStorage.setItem(`$  
{tool}_byok_${provider}`, key);  
const clearBYOK = (tool, provider) => localStorage.removeItem(`$  
{tool}_byok_${provider}`);
```

Der Schlüssel geht an keinen eigenen Server, nur direkt an den gewählten Anbieter.

Prüfwerkzeug `scan-spurentrennung.sh` (neutrale Edition)

Die verbotenen Muster stehen nicht im Skript, sondern in Textdateien im Ordner `muster/`: je Spur eine Datei (`forschung.txt`, `ops.txt`, `signalwerk.txt`, `cuepoint.txt`) und `alle.txt` für Muster, die in jeder Spur verboten sind (gängige Schlüsselformate). Veröffentlicht wird die neutrale Edition mit Beispielwerten (`*.example.invalid`). Jede Organisation trägt dort die Kontakte, Domains und Konten ihrer eigenen Spuren ein; diese echten Muster bleiben intern.

```
./scan-spurentrennung.sh <verzeichnis>
# Spur forschung, Muster ./muster
./scan-spurentrennung.sh <verzeichnis> --track=cuepoint
./scan-spurentrennung.sh <verzeichnis>
--muster=/pfad/zu/internen/mustern
```

Exit	Bedeutung
0	sauber
1	Treffer, Veröffentlichung blockiert
2	Aufruffehler (Verzeichnis, Musterordner oder Musterdatei fehlt)

Gewollte, dokumentierte Nennungen tragen den Marker SPUREN-DOC. Formatkonvertierung (etwa MD → HTML mit pandoc) entfernt HTML-Kommentare; nach jeder Konvertierung wird der Scan deshalb erneut über alle Ausgabeformate gefahren. Die Testsuite `test-scan-spurentrennung.sh` prüft sieben Fälle mit eigenen Testmustern: sauber, Leck einer anderen Spur, Marker, Leck einer vierten Spur, Schlüssel, fehlendes Verzeichnis, fehlende Musterdatei.

Checkliste vor der Veröffentlichung

1. Scan über alle Formate sauber
2. keine `.env`-Datei im Bündel
3. keine Schlüssel im Code
4. keine Betriebsdaten kommerzieller Spuren im Forschungsmaterial
5. Spur im STATUS-Kopf und in der Übergabe benannt

Schlüsselwechsel

Spur	Intervall
FORSCHUNG	90 Tage
SCHICHTWERK-OPS	60 Tage
SIGNALWERK	60 Tage
CUEPOINT	60 Tage

Beim Spurwechsel im Browser ein anderes Browser-Profil verwenden, nicht denselben Tab.

Lizenz: Text CC BY 4.0 · Skripte AGPL-3.0-or-later. Closure-Typ methoden-protokoll.