Netzwerk-Dokumentation: IOT & Homelab

Allgemeine Netzwerkkonfiguration

Standard-LAN (Homelab & Management)

VLAN ID: Default/untagged
Subnetz: 192.168.1.0/24
Gateway: 192.168.1.1
DNS: 192.168.1.1, 8.8.8.8
DHCP-Bereich: 192.168.1.100 - 192.168.1.200 (für automatische Zuweisung)

IOT-VLAN

VLAN ID: 100 (IOT-VLAN)
Subnetz: 192.168.100.0/22
Gateway: 192.168.100.1
DNS: 192.168.100.1, 8.8.8.8
DHCP-Bereich: 192.168.102.1 - 192.168.102.254 (für automatische Zuweisung)

Gäste-VLAN

VLAN ID: 200 (Gäste-VLAN)
Subnetz: 192.168.200.0/24
Gateway: 192.168.200.1
DNS: 192.168.1.3 (Pi-hole für Ad-Blocking)
DHCP-Bereich: 192.168.200.10 - 192.168.200.250 (für automatische Zuweisung)

Netzwerkaufteilung und IP-Bereiche

Standard-LAN (192.168.1.0/24) - Homelab & Management

Bereich	IP-Bereich	Anzahl IPs	Verwendung
Gateway	192.168.1.1	1	UniFi Gateway
Core Infrastructure	192.168.1.2 - 192.168.1.20	19	UniFi Controller, Pi-hole+Unbound, Switches, APs
Homelab Core	192.168.1.21 - 192.168.1.40	20	Proxmox Hosts, Storage
Homelab Services	192.168.1.41 - 192.168.1.99	59	VMs, Docker Container, Services
DHCP Pool	192.168.1.100 - 192.168.1.200	101	Automatische Zuweisung
Client Devices	192.168.1.201 - 192.168.1.220	20	Desktop, Laptop (kabelgebunden + WiFi), Management
Reserve	192.168.1.221 - 192.168.1.254	34	Für zukünftige Erweiterungen

IOT-VLAN (192.168.100.0/22) - Smart Home Geräte + Mobile Clients

Raum	IP-Bereich	Anzahl IPs	Verwendung
Unterverteilung	192.168.100.1 - 192.168.100.62	62	Zentrale Steuergeräte, Homematic CCU
Flur	192.168.100.65 - 192.168.100.126	62	Shelly Schalter, Homematic Sensoren
Arbeitszimmer	192.168.100.129 - 192.168.100.190	62	Shelly Relais, Hue Arbeitsplatz
Schlafzimmer	192.168.100.193 - 192.168.100.254	62	Hue Lampen, Klimasensoren, Jalousien
Wohnzimmer	192.168.101.1 - 192.168.101.62	62	Hue Lampen, Sonos Lautsprecher, TV-Geräte
Küche	192.168.101.65 - 192.168.101.126	62	Küchengeräte, Sonos, Hue Unterschrank
Bad	192.168.101.129 - 192.168.101.190	62	Feuchtigkeitssensoren, Lüftungssteuerung
Mobile Clients	192.168.101.191 - 192.168.101.230	40	**Smartphones, Tablets, Smart-TVs**
Reserve	192.168.101.231 - 192.168.103.254	536	Für zukünftige Erweiterungen

Gäste-VLAN (192.168.200.0/24) - Gast-Zugang

Bereich	IP-Bereich	Anzahl IPs	Verwendung
Gateway	192.168.200.1	1	VLAN Gateway
Reserve	192.168.200.2 - 192.168.200.9	8	Für spezielle Konfiguration
DHCP Pool	192.168.200.10 - 192.168.200.250	241	Gäste-Geräte (automatische Zuweisung)
Reserve	192.168.200.251 - 192.168.200.254	4	Für zukünftige Erweiterungen

DNS-Naming-Konvention

Standard-LAN Schema: `[geraetetype]-[nummer].lab.enzmann.online`

IOT-VLAN Schema: `[geraetetype]-[raum]-[nummer].iot.enzmann.online`

Gerätetypen (Präfixe)

Homelab & Infrastructure (Standard-LAN)

- **pve-** : Proxmox VE Hosts
- **vm-** : Virtuelle Maschinen
- **docker-** : Docker Hosts/Swarm Nodes
- **ha-** : Home Assistant Instanzen
- **nas-** : NAS/Storage Systeme
- **unifi-** : UniFi Controller
- **switch-** : Managed Switches
- **ap-** : Access Points

Technische Geräte (IOT-VLAN - detailliert)

- **shelly-dimmer-** : Shelly Dimmer
- **shelly-pro1pm-** : Shelly Pro 1PM (mit Leistungsmessung)
- **shelly-1-** : Shelly 1 (Relais)
- **shelly-button1-** : Shelly Button1
- **shelly-flood-** : Shelly Flood Sensor
- **hm-window-** : Homematic Fensterkontakt
- **hm-motion-** : Homematic Bewegungsmelder
- **hm-thermo-** : Homematic Thermostat
- **hm-temp-** : Homematic Temperatursensor
- **hm-humid-** : Homematic Feuchtigkeitssensor
- **hm-smoke-** : Homematic Rauchmelder

Consumer-Geräte (IOT-VLAN - einfach)

- **hue-** : Philips Hue Lampen, Sensoren, Bridge
- **sonos-** : Sonos Lautsprecher

Raum-Abkürzungen

- **flur** : Flur
- **wz** : Wohnzimmer
- **sz** : Schlafzimmer

- **az** : Arbeitszimmer
- **bad** : Bad
- **kueche** : Küche
- **uv** : Unterverteilung

Beispiele

Standard-LAN (Homelab)

```
pve-01.lab.enzmann.online                    → Proxmox Host 1
vm-homeassistant-01.lab.enzmann.online       → Home Assistant VM
docker-01.lab.enzmann.online                 → Docker Swarm Manager
ha-prod-01.lab.enzmann.online                → Home Assistant Produktiv
unifi-controller-01.lab.enzmann.online       → UniFi Controller
```

IOT-VLAN (Smart Home)

```
shelly-dimmer-flur-01.iot.enzmann.online     → Shelly Dimmer im Flur
shelly-pro1pm-kueche-01.iot.enzmann.online   → Shelly Pro 1PM in der Küche
hue-wz-03.iot.enzmann.online                 → Hue Lampe im Wohnzimmer
sonos-kueche-01.iot.enzmann.online           → Sonos in der Küche
hm-temp-sz-01.iot.enzmann.online             → Homematic Temperatursensor Schlafzimmer
hm-window-sz-01.iot.enzmann.online           → Homematic Fensterkontakt Schlafzimmer
```

UniFi-spezifische Konfiguration

Standard-LAN Einstellungen

1. **Standard-Netzwerk (Default):**
   - Name: "Standard-LAN"
   - VLAN: Untagged/Default
   - Subnetz: 192.168.1.0/24
   - DHCP aktivieren: Ja

Standard-LAN Einstellungen

1. **Standard-Netzwerk (Default):**
   - Name: "Standard-LAN"
   - VLAN: Untagged/Default
   - Subnetz: 192.168.1.0/24
   - DHCP aktivieren: Ja

2. **WiFi-Netzwerk:**
   - Name: "Enzian"
   - Sicherheit: WPA2/WPA3
   - VLAN: Standard-LAN (Default)
   - Bandsteuerung: Dual-Band (2.4 + 5 GHz)

IOT-VLAN Einstellungen

1. **Netzwerk erstellen:**
   - Name: "IOT-VLAN"
   - VLAN ID: 100
   - Subnetz: 192.168.100.0/22
   - DHCP aktivieren: Ja (für Fallback)
2. **WiFi-Netzwerk:**
   - Name: "Enzian-IOT"
   - Sicherheit: WPA2/WPA3
   - VLAN: IOT-VLAN (100)
   - Gast-Isolation: Aktiviert

Gäste-VLAN Einstellungen

1. **Netzwerk erstellen:**
   - Name: "Gäste-VLAN"
   - VLAN ID: 200
   - Subnetz: 192.168.200.0/24
   - DHCP aktivieren: Ja
2. **WiFi-Netzwerk:**
   - Name: "Enzian-Gast"
   - Sicherheit: WPA2/WPA3 (einfaches Passwort)
   - VLAN: Gäste-VLAN (200)
   - Gast-Isolation: Aktiviert
   - Bandbreiten-Limit: Optional (z.B. 50 Mbit/s)

UniFi Zone Matrix Konfiguration

Übersicht Zone Matrix

Die Zone Matrix bietet eine **grafische Oberfläche** zur einfachen Konfiguration von VLAN-zu-VLAN Kommunikation ohne komplexe Firewall-Regeln.

Zone-Definitionen

1. Zonen konfigurieren (Settings → Security → Zones)

Zone 1: "Internal" (Built-in)

- **Netzwerke:** Standard-LAN (192.168.1.0/24)
- **Beschreibung:** Vorkonfigurierte UniFi Zone für interne Netzwerke
- **Farbe:** Blau (Standard)
- **Status:** Bereits vorhanden, nur Netzwerk zuweisen

Zone 2: "IOT" (Neu erstellen)

- **Netzwerke:** IOT-VLAN (192.168.100.0/22)
- **Beschreibung:** Smart Home Geräte + Mobile Clients
- **Farbe:** Grün
- **Status:** Neu erstellen

Zone 3: "Hotspot" (Built-in)

- **Netzwerke:** Gäste-VLAN (192.168.200.0/24)
- **Beschreibung:** Vorkonfigurierte UniFi Zone für Gäste-Zugang
- **Farbe:** Orange (Standard)
- **Status:** Bereits vorhanden, nur Netzwerk zuweisen

2. Zone Matrix konfigurieren (Settings → Security → Zone Matrix)

Zone Matrix Tabelle:

	Internal	IOT	Hotspot	Internet
Internal	✅	✅	❌	✅
IOT	🔸	✅	❌	✅
Hotspot	🔸	❌	✅	✅
Internet	✅	✅	✅	✅

Legende:
✅ = Erlaubt (Allow)
❌ = Blockiert (Block)
🔸 = Begrenzt (Limited) - nur spezifische Ports

Zone Matrix Einstellungen im Detail

Internal → IOT: ✅ Allow

- **Begründung:** Home Assistant muss auf IOT-Geräte zugreifen
- **Ports:** Alle (für Administration und Setup)

IOT → Internal: 🔸 Limited

- **Begründung:** Mobile Apps brauchen Zugriff auf Home Assistant
- **Erlaubte Ports:**
  - `53` (DNS zu Pi-hole: 192.168.1.3)
  - `123` (NTP für Zeitserver)
  - `8123` (Home Assistant Web-Interface)
  - `1883/8883` (MQTT Broker)
  - `5353` (mDNS für Device Discovery)

Internal → Hotspot: ❌ Block

- **Begründung:** Keine Verwaltung von Gäste-Geräten nötig

IOT → Hotspot: ❌ Block

- **Begründung:** Smart Home soll nicht mit Gäste-Netz kommunizieren

Hotspot → Internal: 🔸 Limited

- **Begründung:** Gäste brauchen nur DNS-Auflösung
- **Erlaubte Ports:**
  - `53` (DNS zu Pi-hole: 192.168.1.3)
  - `123` (NTP für Zeitserver)
- **Vorteil:** Hotspot-Zone hat bereits optimierte Gäste-Einstellungen

Hotspot → IOT: ❌ Block

- **Begründung:** Gäste sollen keinen Zugriff auf Smart Home haben

Alle → Internet: ✅ Allow

- **Begründung:** Internet-Zugang für alle Zonen erforderlich

Vorteile der Zone Matrix mit Built-in Zonen

Nutzt UniFi-Standards optimal

- **Internal-Zone:** Bereits für interne Netzwerke optimiert

- **Hotspot-Zone:** Bereits für Gäste-Zugang konfiguriert (Bandbreiten-Limits, Isolation)
- **Nur eine neue Zone:** "IOT" muss erstellt werden

Weniger Konfigurationsaufwand

- **Vorkonfigurierte Einstellungen** der Built-in Zonen nutzen
- **Bewährte UniFi-Praktiken** werden automatisch angewendet
- **Konsistent mit UniFi-Dokumentation**

Automatische Optimierungen

- **Hotspot-Zone** bringt bereits Gäste-spezifische Einstellungen mit
- **Internal-Zone** ist für Verwaltungsaufgaben optimiert
- **Standard-Firewall-Templates** werden angewendet

Einfache Konfiguration

- **Grafische Oberfläche** statt komplexer Firewall-Regeln
- **Matrix-Ansicht** macht Beziehungen sofort sichtbar
- **Ein Klick** zum Ändern von Allow/Block/Limited

Automatische Firewall-Regeln

- **UniFi generiert automatisch** die entsprechenden Firewall-Regeln
- **Bidirektionale Regeln** werden automatisch erstellt
- **Konsistente Regelanwendung** auf alle Geräte in einer Zone

Troubleshooting

- **Übersichtliche Darstellung** aller Zonen-Beziehungen
- **Einfache Änderungen** für Tests
- **Logging** zeigt blockierte Verbindungen pro Zone

Alternative Architektur: Dedizierte IOT-Services

Konzept-Übersicht

Anstatt alle Services im Standard-LAN zu betreiben und über Firewall-Regeln auf das IOT-VLAN zuzugreifen, werden **IOT-spezifische Services direkt in der IOT-Zone** bereitgestellt.

Service-Aufteilung nach Zonen

Internal Zone (192.168.1.x) - Core Infrastructure

bash

```
# Reine Infrastruktur-Services
192.168.1.21   → pve-01.lab.enzmann.online (Proxmox Host 1)
192.168.1.22   → pve-02.lab.enzmann.online (Proxmox Host 2)
192.168.1.25   → nas-01.lab.enzmann.online (Storage)
192.168.1.3    → pihole-01.lab.enzmann.online (DNS)
192.168.1.48   → traefik-01.lab.enzmann.online (Reverse Proxy)
192.168.1.50   → portainer-01.lab.enzmann.online (Docker Management)
192.168.1.51   → grafana-01.lab.enzmann.online (Monitoring)
192.168.1.52   → influx-01.lab.enzmann.online (Monitoring DB)
```

IOT Zone (192.168.100.x) - Smart Home Services

bash

```
# IOT-spezifische Services (dedizierte VMs/Container)
192.168.100.41 → ha-prod-01.iot.enzmann.online (Home Assistant)
192.168.100.42 → ha-test-01.iot.enzmann.online (Home Assistant Test)
192.168.100.45 → mqtt-01.iot.enzmann.online (MQTT Broker)
192.168.100.46 → nodered-01.iot.enzmann.online (Node-RED)
192.168.100.47 → zigbee2mqtt-01.iot.enzmann.online (Zigbee Bridge)
192.168.100.48 → esphome-01.iot.enzmann.online (ESP Home)
192.168.100.50 → influx-iot-01.iot.enzmann.online (IOT Metrics DB)

# Smart Home Hardware
192.168.100.10 → hm-ccu-uv-01.iot.enzmann.online (Homematic CCU)
192.168.101.1  → hue-wz-bridge01.iot.enzmann.online (Hue Bridge)
# ... weitere IOT-Geräte
```

Technische Umsetzung

Proxmox VM-Deployment

bash

```
# Home Assistant VM in IOT-Zone
VM-Name: HA-Prod-IOT
vCPUs: 4
RAM: 8GB
Storage: 100GB SSD
Network: IOT-VLAN (VLAN 100)
IP: 192.168.100.41

# MQTT Broker VM in IOT-Zone
VM-Name: MQTT-IOT
vCPUs: 2
RAM: 4GB
Storage: 50GB SSD
Network: IOT-VLAN (VLAN 100)
IP: 192.168.100.45
```

Docker Swarm in IOT-Zone

yaml

```yaml
# docker-compose-iot.yml (deployed on IOT network)
version: '3.8'
services:
  homeassistant:
    image: homeassistant/home-assistant:stable
    networks:
      - iot-services
    ports:
      - "192.168.100.41:8123:8123"

  mosquitto:
    image: eclipse-mosquitto:latest
    networks:
      - iot-services
    ports:
      - "192.168.100.45:1883:1883"

  nodered:
    image: nodered/node-red:latest
    networks:
      - iot-services
    ports:
      - "192.168.100.46:1880:1880"

networks:
  iot-services:
    driver: bridge
    ipam:
      config:
        - subnet: 192.168.100.0/22
```

Vereinfachte Firewall-Regeln

Zone Matrix (deutlich einfacher)

	Internal	IOT	Hotspot
Internal	✅	🔸	❌
IOT	🔸	✅	❌
Hotspot	🔸	❌	✅

Spezifische Regeln (weniger komplex)

bash

```
# Internal → IOT (Limited)
Port 22:   SSH für VM-Management
Port 443:  HTTPS für Web-Interfaces via Traefik
Port 3000: Grafana → InfluxDB-IOT für Monitoring

# IOT → Internal (Limited)
Port 53:   DNS zu Pi-hole
Port 123:  NTP für Zeitserver
Port 443:  Backup/Update Services
```

Vorteile der dedizierten IOT-Services

Sicherheit

- **Vollständige Isolation** - IOT-Services laufen komplett getrennt
- **Kein Cross-Zone-Traffic** für normale IOT-Operationen
- **Minimale Firewall-Regeln** zwischen Zonen
- **Blast Radius Reduction** - Kompromittierung bleibt in einer Zone

Performance

- **Lokaler Traffic** zwischen IOT-Geräten und Services
- **Reduzierte Latenz** für Smart Home Operationen
- **Keine VLAN-Routing** Overhead für häufige Zugriffe

Verwaltung

- **Klare Service-Zuordnung** pro Zone
- **Einfachere Troubleshooting** - Services sind dort wo sie gebraucht werden
- **Zonenbezogene Backups** möglich

Nachteile der dedizierten IOT-Services

Ressourcen-Overhead

- **Doppelte Services** (z.B. InfluxDB in beiden Zonen)
- **Mehr VMs/Container** zu verwalten
- **Höherer Speicher/CPU-Verbrauch**

Komplexität

- **Service-Synchronisation** zwischen Zonen bei Bedarf

- **Zentrale Überwachung** wird schwieriger
- **Backup-Strategie** muss zonenbezogen sein

Monitoring-Herausforderung

- **Grafana in Internal** kann nicht direkt auf IOT-InfluxDB zugreifen
- **Separate Monitoring-Stacks** oder komplexe Datenreplikation nötig

Hybrid-Ansatz (Empfehlung)

Core Services bleiben Internal

```bash
Proxmox, NAS, Pi-hole, Traefik → Internal Zone
Zentrale Überwachung (Grafana, InfluxDB) → Internal Zone
```

IOT-spezifische Services in IOT Zone

```bash
Home Assistant, MQTT, Node-RED → IOT Zone
IOT-Hardware (Hue, Homematic) → IOT Zone
Mobile Clients → IOT Zone
```

Minimale Cross-Zone-Kommunikation

```bash
IOT Home Assistant → Internal Grafana (für Dashboards)
Internal Backup → IOT Services (für Datensicherung)
Mobile Clients → Internal Traefik (für Admin-Interfaces)
```

Praktische Umsetzung

1. **Phase 1:** Services identifizieren die nur IOT brauchen
2. **Phase 2:** Diese Services in IOT-Zone migrieren
3. **Phase 3:** Firewall-Regeln entsprechend reduzieren
4. **Phase 4:** Monitoring und Backup anpassen

WiFi-Netzwerke Übersicht

WiFi-Name	VLAN	Zweck	Geräte	Passwort-Typ
Enzian	Standard-LAN (Default)	Administration, Laptops, Homelab	Admin-Laptops, Management-Geräte	Starkes Passwort
Enzian-IOT	IOT-VLAN (100)	Smart Home + Mobile Clients	Smartphones, Tablets, Smart-TVs	Mittleres Passwort
Enzian-Gast	Gäste-VLAN (200)	Gäste-Zugang	Gäste-Geräte	Einfaches Passwort

Spezifische Regeln für Mobile Geräte (IOT-VLAN)

- **Port 8123:** IOT → Standard-LAN (Home Assistant Web-Interface)
- **mDNS:** Bidirektional zwischen Standard-LAN ↔ IOT für Device Discovery
- **MQTT:** IOT → Standard-LAN Port 1883/8883
- **Chromecast/AirPlay:** IOT-interne Kommunikation erlaubt

Spezifische Regeln für Gäste-VLAN

- **DNS:** Gäste-VLAN → Standard-LAN Port 53 (zu Pi-hole 192.168.1.3)
- **NTP:** Gäste-VLAN → Standard-LAN Port 123 (Zeitserver)
- **Alle anderen Ports:** Gäste-VLAN → Standard-LAN/IOT-VLAN blockiert

Lokales DNS mit Pi-hole + Unbound auf Raspberry Pi

Übersicht

Lokale DNS-Auflösung erfolgt über Pi-hole anstatt öffentlicher DNS-Einträge bei netcup:

- **Sicherheit:** Keine internen Strukturen öffentlich sichtbar
- **Performance:** Lokale Auflösung ohne Internet-Abhängigkeit
- **Zusatznutzen:** Ad-Blocking, Malware-Schutz, DNS-Statistiken
- **Flexibilität:** Einfache Verwaltung über Web-Interface

Architektur-Entscheidung: Dedizierte Hardware

Warum Raspberry Pi statt VMs?

- **Bootstrap-Problem vermeiden:** VMs brauchen DNS zum Starten
- **Unabhängigkeit:** DNS läuft getrennt vom Proxmox Cluster
- **Hochverfügbarkeit:** Zwei Raspberry Pis für Redundanz
- **Kostengünstig:** ~€160 für zwei Pis vs. VM-Ressourcen

Hardware-Spezifikation (pro Pi)

bash

```
Raspberry Pi 4B (4GB RAM)
SSD via USB 3.0 (bessere Performance als SD-Karte)
Gigabit Ethernet (kein WiFi für kritische Infrastruktur)
USV/Powerbank (optional für Stromausfälle)
```

Raspberry Pi DNS-Cluster Setup

IP-Adresszuweisung

bash

```
Pi-hole Primary:   192.168.1.3 → pihole-01.lab.enzmann.online
Pi-hole Secondary: 192.168.1.4 → pihole-02.lab.enzmann.online

# UniFi DHCP DNS-Server Einstellungen:
Primary DNS:   192.168.1.3
Secondary DNS: 192.168.1.4
Tertiary DNS:  8.8.8.8 (ultimativer Fallback)
```

Docker Compose Konfiguration (pro Pi)

yaml

```yaml
# /opt/dns-stack/docker-compose.yml (identisch auf beiden Pis)
version: '3.8'

services:
  unbound:
    image: mvance/unbound-rpi:latest  # ARM-optimiert
    hostname: unbound-${PI_NUMBER}
    environment:
      TZ: 'Europe/Berlin'
    volumes:
      - unbound_config:/opt/unbound/etc/unbound
      - ./unbound.conf:/opt/unbound/etc/unbound/unbound.conf:ro
    networks:
      dns-internal:
        ipv4_address: 172.20.0.2
    restart: unless-stopped

  pihole:
    image: pihole/pihole:latest
    hostname: pihole-${PI_NUMBER}
    environment:
      TZ: 'Europe/Berlin'
      WEBPASSWORD: '${PIHOLE_PASSWORD}'
      VIRTUAL_HOST: 'pihole-${PI_NUMBER}.lab.enzmann.online'
      FTLCONF_LOCAL_IPV4: '${PI_IP}'
      PIHOLE_DNS_: '172.20.0.2#5053'  # Lokaler Unbound
    volumes:
      - pihole_config:/etc/pihole
      - pihole_dnsmasq:/etc/dnsmasq.d
    ports:
      - "53:53/tcp"
      - "53:53/udp"
      - "80:80/tcp"  # Web-Interface
    networks:
      dns-internal:
        ipv4_address: 172.20.0.3
    depends_on:
      - unbound
    restart: unless-stopped

  gravity-sync:
    image: vmstan/gravity-sync:latest
    hostname: gravity-sync-${PI_NUMBER}
    environment:
      GS_REMOTE_HOST: "${REMOTE_PI_IP}"
      GS_REMOTE_USER: "pi"
```

```
      GS_AUTO_MODE: "true"
    volumes:
      - pihole_config:/etc/pihole
      - ./gravity-sync:/root/gravity-sync
      - ~/.ssh:/root/.ssh:ro
    depends_on:
      - pihole
    restart: unless-stopped

volumes:
  pihole_config:
  pihole_dnsmasq:
  unbound_config:

networks:
  dns-internal:
    ipam:
      config:
        - subnet: 172.20.0.0/24
```

Environment-Konfiguration

bash

```
# Pi #1: /opt/dns-stack/.env
PI_NUMBER=01
PI_IP=192.168.1.3
REMOTE_PI_IP=192.168.1.4
PIHOLE_PASSWORD=secure-admin-password

# Pi #2: /opt/dns-stack/.env
PI_NUMBER=02
PI_IP=192.168.1.4
REMOTE_PI_IP=192.168.1.3
PIHOLE_PASSWORD=secure-admin-password
```

Unbound Konfiguration

bash

```
# ./unbound.conf (identisch auf beiden Pis)
server:
    interface: 0.0.0.0
    port: 5053
    do-ip4: yes
    do-ip6: no
    do-udp: yes
    do-tcp: yes
    harden-glue: yes
    harden-dnssec-stripped: yes
    use-caps-for-id: no
    edns-buffer-size: 1232
    prefetch: yes
    num-threads: 2
    so-rcvbuf: 1m
    private-address: 192.168.0.0/16
    private-address: 172.16.0.0/12
    private-address: 10.0.0.0/8
    verbosity: 1
    log-queries: no
    hide-identity: yes
    hide-version: yes
    qname-minimisation: yes
    minimal-responses: yes
    msg-cache-size: 50m
    rrset-cache-size: 100m
    cache-max-ttl: 86400

# Forward zones für lokale Domains
forward-zone:
    name: "lab.enzmann.online"
    forward-addr: 172.20.0.3@53

forward-zone:
    name: "iot.enzmann.online"
    forward-addr: 172.20.0.3@53

forward-zone:
    name: "guest.enzmann.online"
    forward-addr: 172.20.0.3@53
```

Deployment-Strategie

Phase 1: Erstes Raspberry Pi

bash

```
# 1. Raspberry Pi OS installieren
# 2. Docker installieren
sudo curl -fsSL https://get.docker.com | sh
sudo usermod -aG docker pi

# 3. DNS-Stack deployen
git clone <your-dns-config-repo>
cd dns-stack
docker-compose up -d

# 4. Als Primary DNS in UniFi eintragen
# 5. Stabilität für 1-2 Wochen testen
```

Phase 2: Zweites Raspberry Pi

bash

```
# 1. Identische Installation wie Pi #1
# 2. SSH-Keys für Gravity Sync einrichten
# 3. Als Secondary DNS in UniFi hinzufügen
# 4. Gravity Sync zwischen beiden Pis aktivieren
# 5. Load Balancing testen
```

Hochverfügbarkeit und Synchronisation

Automatische Konfigurationssync

- **Gravity Sync** synchronisiert Pi-hole Einstellungen zwischen beiden Pis
- **Blocklisten, DNS-Einträge, Whitelist** werden automatisch abgeglichen
- **Query-Logs** bleiben lokal pro Pi für bessere Performance

Failover-Verhalten

bash

```
# Primärer Pi (192.168.1.3) fällt aus:
# → Clients nutzen automatisch sekundären Pi (192.168.1.4)
# → Keine Unterbrechung der DNS-Auflösung
# → Pi-hole Web-Interface über sekundären Pi verfügbar

# Sekundärer Pi (192.168.1.4) fällt aus:
# → Primärer Pi übernimmt alle Anfragen
# → Gravity Sync pausiert automatisch
# → Nach Wiederherstellung: Automatische Resync
```

Wartung und Updates

Docker Container Updates

bash

```
# Einzelner Pi (Rolling Update):
cd /opt/dns-stack
docker-compose pull
docker-compose up -d

# Automatisierung via Cron:
0 3 * * 0 cd /opt/dns-stack && docker-compose pull && docker-compose up -d
```

Backup-Strategie

bash

```
# Pi-hole Konfiguration backup
docker-compose exec pihole pihole -a -t

# Volume Backup
sudo cp -r /var/lib/docker/volumes/dns-stack_pihole_config /backup/
```

Pi-hole + Unbound Setup

Docker Compose Konfiguration

yaml

```yaml
# pihole/docker-compose.yml
version: '3.8'

services:
  unbound:
    image: mvance/unbound:latest
    hostname: unbound-01
    environment:
      TZ: 'Europe/Berlin'
    volumes:
      - unbound_config:/opt/unbound/etc/unbound
    networks:
      - pihole-internal
    deploy:
      placement:
        constraints:
          - node.role == manager

  pihole:
    image: pihole/pihole:latest
    hostname: pihole-01
    environment:
      TZ: 'Europe/Berlin'
      WEBPASSWORD: 'secure-admin-password'
      VIRTUAL_HOST: 'pihole-01.lab.enzmann.online'
      PROXY_LOCATION: 'pihole-01'
      FTLCONF_LOCAL_IPV4: '192.168.1.3'
      PIHOLE_DNS_: '10.0.1.2#5053'  # Unbound Container IP
    volumes:
      - pihole_config:/etc/pihole
      - pihole_dnsmasq:/etc/dnsmasq.d
      - pihole_custom_conf:/etc/pihole/custom.conf
    ports:
      - "53:53/tcp"
      - "53:53/udp"
      - "67:67/udp"  # DHCP (optional)
    networks:
      pihole-internal:
        ipv4_address: 10.0.1.3
      traefik:
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.pihole.rule=Host(`pihole-01.lab.enzmann.online`)"
      - "traefik.http.routers.pihole.tls.certresolver=letsencrypt"
      - "traefik.http.services.pihole.loadbalancer.server.port=80"
    depends_on:
```

```
      - unbound
    deploy:
      placement:
        constraints:
          - node.role == manager

volumes:
  pihole_config:
  pihole_dnsmasq:
  pihole_custom_conf:
  unbound_config:

networks:
  pihole-internal:
    ipam:
      config:
        - subnet: 10.0.1.0/24
  traefik:
    external: true
```

Unbound Konfiguration

bash

```
# Unbound Config erstellen (einmalig)
docker exec -it $(docker ps | grep unbound | cut -d' ' -f1) sh -c 'cat > /opt/unbound/etc/unbou
server:
    # Listening
    interface: 0.0.0.0
    port: 5053
    do-ip4: yes
    do-ip6: no
    do-udp: yes
    do-tcp: yes

    # Trust glue only if it is within the server's authority
    harden-glue: yes

    # Require DNSSEC data for trust-anchored zones
    harden-dnssec-stripped: yes

    # Don't use Capitalization randomization
    use-caps-for-id: no

    # Reduce EDNS reassembly buffer size.
    edns-buffer-size: 1232

    # Perform prefetching of close to expired message cache entries
    prefetch: yes

    # One thread should be sufficient
    num-threads: 1

    # Ensure kernel buffer is large enough
    so-rcvbuf: 1m

    # Ensure privacy of local IP ranges
    private-address: 192.168.0.0/16
    private-address: 169.254.0.0/16
    private-address: 172.16.0.0/12
    private-address: 10.0.0.0/8
    private-address: fd00::/8
    private-address: fe80::/10

    # Logging
    verbosity: 1
    log-queries: no
    log-replies: no

    # Performance tuning
```

```
    msg-cache-slabs: 2
    rrset-cache-slabs: 2
    infra-cache-slabs: 2
    key-cache-slabs: 2
    msg-cache-size: 50m
    rrset-cache-size: 100m
    cache-max-ttl: 86400
    cache-min-ttl: 300

    # Security
    hide-identity: yes
    hide-version: yes
    qname-minimisation: yes
    minimal-responses: yes

# Forward zones for local domains
forward-zone:
    name: "lab.enzmann.online"
    forward-addr: 10.0.1.3@53  # Pi-hole IP

forward-zone:
    name: "iot.enzmann.online"
    forward-addr: 10.0.1.3@53  # Pi-hole IP
EOF'
```

DNS-Konfiguration in Pi-hole

Lokale DNS-Einträge (via Web-Interface)

bash

```
# Standard-LAN (Homelab) - Core Infrastructure
192.168.1.2     unifi-controller-01.lab.enzmann.online
192.168.1.3     pihole-01.lab.enzmann.online
192.168.1.10    switch-main-01.lab.enzmann.online
192.168.1.11    ap-wz-01.lab.enzmann.online
192.168.1.12    ap-sz-01.lab.enzmann.online

# Homelab Core
192.168.1.21    pve-01.lab.enzmann.online
192.168.1.22    pve-02.lab.enzmann.online
192.168.1.25    nas-01.lab.enzmann.online

# Homelab Services (Beispiele)
192.168.1.41    ha-prod-01.lab.enzmann.online
192.168.1.42    ha-test-01.lab.enzmann.online
192.168.1.45    docker-01.lab.enzmann.online
192.168.1.48    traefik-01.lab.enzmann.online
192.168.1.50    portainer-01.lab.enzmann.online
192.168.1.51    grafana-01.lab.enzmann.online
192.168.1.52    influx-01.lab.enzmann.online
192.168.1.55    mqtt-01.lab.enzmann.online
192.168.1.56    prometheus-01.lab.enzmann.online

# Client Devices
192.168.1.205   desktop-admin-01.lab.enzmann.online
192.168.1.206   laptop-admin-01.lab.enzmann.online

# IOT-VLAN (Smart Home) - wichtigste Geräte
192.168.100.10  hm-ccu-uv-01.iot.enzmann.online
192.168.101.1   hue-wz-bridge01.iot.enzmann.online
192.168.101.2   sonos-wz-bridge01.iot.enzmann.online
```

Wildcard-Domains (via dnsmasq config)

bash

```
# /etc/dnsmasq.d/02-lab-wildcard.conf
address=/lab.enzmann.online/192.168.1.48

# /etc/dnsmasq.d/03-iot-wildcard.conf
address=/iot.enzmann.online/192.168.1.48

# /etc/dnsmasq.d/04-guest-wildcard.conf
address=/guest.enzmann.online/192.168.1.48
address=/guest.enzmann.online/192.168.1.48
```

UniFi Integration

DHCP-Einstellungen ändern

1. **Standard-LAN Netzwerk bearbeiten**
2. **DHCP → DNS Server:** `192.168.1.3` (Pi-hole IP)
3. **DHCP → Domain Name:** `lab.enzmann.online`

UniFi Integration

Standard-LAN DHCP-Einstellungen

1. **Standard-Netzwerk (Default) bearbeiten**
2. **DHCP → DNS Server:** `192.168.1.3` (Pi-hole IP)
3. **DHCP → Domain Name:** `lab.enzmann.online`

IOT-VLAN DHCP-Einstellungen

1. **IOT-VLAN Netzwerk bearbeiten**
2. **DHCP → DNS Server:** `192.168.1.3` (Pi-hole IP)
3. **DHCP → Domain Name:** `iot.enzmann.online`

Gäste-VLAN DHCP-Einstellungen

1. **Gäste-VLAN Netzwerk bearbeiten**
2. **DHCP → DNS Server:** `192.168.1.3` (Pi-hole IP)
3. **DHCP → Domain Name:** `guest.enzmann.online`
4. **DHCP → Lease Time:** 4 Stunden (kürzer für Gäste)

Vorteile der Pi-hole + Unbound Lösung

Sicherheit & Privatsphäre

- **Keine externen DNS-Provider** - alle Anfragen bleiben lokal bis zu den Root-Servern
- **DNSSEC-Validierung** durch Unbound für sichere DNS-Auflösung
- **Kein DNS-Logging** bei externen Anbietern (Google, Cloudflare)
- **Qname-Minimisation** reduziert Datenleckage

Performance

- **Lokales Caching** auf zwei Ebenen (Pi-hole + Unbound)
- **Prefetching** von häufig genutzten Domains durch Unbound
- **Rekursive Auflösung** direkt zu autoritativen Servern
- **Optimierte Cache-Größen** für Homelab-Umgebung

Zusatzfunktionen

- **Ad-Blocking** für alle Geräte im Netzwerk (Pi-hole)
- **Malware-Schutz** über Blocklisten (Pi-hole)
- **Query-Logging** für Troubleshooting (Pi-hole)
- **Statistiken** über DNS-Nutzung (Pi-hole)
- **Lokale Domain-Auflösung** für `.lab` und `.iot` Subdomains

HTTPS & Zertifikate mit Traefik

Übersicht

Alle Homelab-Services werden über HTTPS mit echten Let's Encrypt Zertifikaten bereitgestellt:

- **Domain:** enzmann.online (gehostet bei netcup)
- **Reverse Proxy:** Traefik mit automatischer SSL-Terminierung
- **Zertifikate:** Let's Encrypt Wildcard via DNS-Challenge (netcup API)

DNS-Struktur bei netcup

```
# A-Records (zeigen auf lokale IPs)
ha.enzmann.online          → 192.168.1.41
grafana.enzmann.online     → 192.168.1.51
portainer.enzmann.online   → 192.168.1.50
traefik.enzmann.online     → 192.168.1.48

# Wildcard für alle Services
*.enzmann.online           → 192.168.1.48  (Traefik)
```

Traefik Konfiguration

Docker Compose Setup

yaml

```yaml
# traefik/docker-compose.yml
version: '3.8'

services:
  traefik:
    image: traefik:v3.0
    command:
      # API und Dashboard
      - "--api.dashboard=true"
      - "--api.insecure=false"

      # Provider
      - "--providers.docker=true"
      - "--providers.docker.swarmMode=true"
      - "--providers.docker.exposedbydefault=false"

      # Entrypoints
      - "--entrypoints.web.address=:80"
      - "--entrypoints.websecure.address=:443"
      - "--entrypoints.web.http.redirections.entrypoint.to=websecure"
      - "--entrypoints.web.http.redirections.entrypoint.scheme=https"

      # Let's Encrypt mit netcup DNS-Challenge für Wildcards
      - "--certificatesresolvers.letsencrypt.acme.dnschallenge=true"
      - "--certificatesresolvers.letsencrypt.acme.dnschallenge.provider=netcup"
      - "--certificatesresolvers.letsencrypt.acme.email=admin@enzmann.online"
      - "--certificatesresolvers.letsencrypt.acme.storage=/letsencrypt/acme.json"

      # Logging
      - "--log.level=INFO"
      - "--accesslog=true"

    ports:
      - "80:80"
      - "443:443"

    environment:
      # netcup API Credentials
      NETCUP_CUSTOMER_NUMBER: "${NETCUP_CUSTOMER_NUMBER}"
      NETCUP_API_KEY: "${NETCUP_API_KEY}"
      NETCUP_API_PASSWORD: "${NETCUP_API_PASSWORD}"

    volumes:
      - /var/run/docker.sock:/var/run/docker.sock:ro
      - traefik_letsencrypt:/letsencrypt
```

```
    labels:
      # Traefik Dashboard
      - "traefik.enable=true"
      - "traefik.http.routers.dashboard.rule=Host(`traefik-01.lab.enzmann.online`)"
      - "traefik.http.routers.dashboard.service=api@internal"
      - "traefik.http.routers.dashboard.tls.certresolver=letsencrypt"
      - "traefik.http.routers.dashboard.middlewares=auth"

      # Basic Auth für Dashboard
      - "traefik.http.middlewares.auth.basicauth.users=admin:$2y$10$..."  # htpasswd generiert

    networks:
      - traefik

    deploy:
      placement:
        constraints:
          - node.role == manager

volumes:
  traefik_letsencrypt:

networks:
  traefik:
    external: true
```

Environment File (.env)

bash

```
# netcup API Credentials (von netcup CCP)
NETCUP_CUSTOMER_NUMBER=123456
NETCUP_API_KEY=abcdefghijklmnopqrstuvwxyz
NETCUP_API_PASSWORD=your-api-password
```

Service-Konfiguration Beispiele

Home Assistant

yaml

```yaml
# homeassistant/docker-compose.yml
services:
  homeassistant:
    image: homeassistant/home-assistant:stable
    volumes:
      - ha_config:/config
    networks:
      - traefik
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.homeassistant.rule=Host(`ha-prod-01.lab.enzmann.online`)"
      - "traefik.http.routers.homeassistant.tls.certresolver=letsencrypt"
      - "traefik.http.services.homeassistant.loadbalancer.server.port=8123"

networks:
  traefik:
    external: true
```

Grafana

yaml

```yaml
# monitoring/docker-compose.yml
services:
  grafana:
    image: grafana/grafana:latest
    environment:
      - GF_SERVER_ROOT_URL=https://grafana-01.lab.enzmann.online
      - GF_SECURITY_ADMIN_PASSWORD=secure-password
    networks:
      - traefik
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.grafana.rule=Host(`grafana-01.lab.enzmann.online`)"
      - "traefik.http.routers.grafana.tls.certresolver=letsencrypt"
      - "traefik.http.services.grafana.loadbalancer.server.port=3000"
```

Portainer

yaml

```yaml
# portainer/docker-compose.yml
services:
  portainer:
    image: portainer/portainer-ce:latest
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - portainer_data:/data
    networks:
      - traefik
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.portainer.rule=Host(`portainer-01.lab.enzmann.online`)"
      - "traefik.http.routers.portainer.tls.certresolver=letsencrypt"
      - "traefik.http.services.portainer.loadbalancer.server.port=9000"
```

netcup DNS API Setup

1. API-Zugang aktivieren

1. Bei netcup im Customer Control Panel anmelden
2. **Stammdaten** → **API** aufrufen
3. **API-Key** und **API-Password** generieren
4. **DNS-API** Berechtigung aktivieren

2. DNS-Einträge bei netcup

Wichtig: Keine A-Records für lokale Services erstellen!

bash

```bash
# Nur für DNS-Challenge erforderlich - keine manuellen Einträge nötig
# Traefik erstellt automatisch TXT-Records für Let's Encrypt
```

Optional: Falls später externe VPN-Services gewünscht:

bash

```bash
vpn.enzmann.online        A  [Externe-IP]  # Nur bei VPN-Setup
```

3. Deployment

bash

```
# Traefik Network erstellen
docker network create --driver overlay traefik

# Pi-hole + Unbound Stack deployen
docker stack deploy -c pihole/docker-compose.yml pihole

# Warten bis Container gestartet sind
sleep 30

# Unbound Konfiguration anwenden (einmalig)
docker exec -it $(docker ps | grep unbound | cut -d' ' -f1) sh -c 'cat > /opt/unbound/etc/unbou
server:
    interface: 0.0.0.0
    port: 5053
    do-ip4: yes
    do-ip6: no
    do-udp: yes
    do-tcp: yes
    harden-glue: yes
    harden-dnssec-stripped: yes
    use-caps-for-id: no
    edns-buffer-size: 1232
    prefetch: yes
    num-threads: 1
    so-rcvbuf: 1m
    private-address: 192.168.0.0/16
    private-address: 172.16.0.0/12
    private-address: 10.0.0.0/8
    verbosity: 1
    log-queries: no
    hide-identity: yes
    hide-version: yes
    qname-minimisation: yes
    minimal-responses: yes
    msg-cache-size: 50m
    rrset-cache-size: 100m
    cache-max-ttl: 86400

forward-zone:
    name: "lab.enzmann.online"
    forward-addr: 10.0.1.3@53

forward-zone:
    name: "iot.enzmann.online"
    forward-addr: 10.0.1.3@53
```

```
forward-zone:
    name: "guest.enzmann.online"
    forward-addr: 10.0.1.3@53

forward-zone:
    name: "guest.enzmann.online"
    forward-addr: 10.0.1.3@53
EOF'

# Unbound neu starten für Konfiguration
docker exec -it $(docker ps | grep unbound | cut -d' ' -f1) unbound-control reload

# UniFi DHCP auf Pi-hole umstellen (192.168.1.3 als DNS)

# Pi-hole lokale DNS-Einträge konfigurieren (Web-Interface)

# Environment für Traefik setzen
echo "NETCUP_CUSTOMER_NUMBER=123456" > .env
echo "NETCUP_API_KEY=your-api-key" >> .env
echo "NETCUP_API_PASSWORD=your-api-password" >> .env

# Traefik deployen
docker stack deploy -c traefik/docker-compose.yml traefik

# Services deployen
docker stack deploy -c homeassistant/docker-compose.yml homeassistant
docker stack deploy -c monitoring/docker-compose.yml monitoring
```

Wildcard-Zertifikat Vorteile

- **Ein Zertifikat** für alle *.enzmann.online Subdomains
- **Automatische Erneuerung** alle 60 Tage
- **Keine Rate-Limits** von Let's Encrypt
- **Einfache Service-Erweiterung** ohne zusätzliche Zertifikatskonfiguration

Zugriff auf Services

Nach dem Setup sind alle Services sicher über HTTPS erreichbar:

```
https://ha-prod-01.lab.enzmann.online       → Home Assistant
https://grafana-01.lab.enzmann.online       → Grafana Dashboard
https://portainer-01.lab.enzmann.online     → Docker Management
https://traefik-01.lab.enzmann.online       → Traefik Dashboard
https://pihole-01.lab.enzmann.online        → Pi-hole Admin Interface
```

Zusätzlich: Alle IOT-Geräte sind über ihre Subdomains erreichbar:

```
https://hm-ccu-uv-01.iot.enzmann.online     → Homematic CCU
https://shelly-dimmer-flur-01.iot.enzmann.online → Shelly Dimmer
```

DHCP-Reservierungen

Standard-LAN (Homelab)

```
UniFi Controller: 192.168.1.2 → unifi-controller-01.lab.enzmann.online
Proxmox Host 1: 192.168.1.21 → pve-01.lab.enzmann.online
Proxmox Host 2: 192.168.1.22 → pve-02.lab.enzmann.online
Pi-hole DNS: 192.168.1.3 → pihole-01.lab.enzmann.online
Home Assistant: 192.168.1.41 → ha-prod-01.lab.enzmann.online
Docker Swarm Manager: 192.168.1.45 → docker-01.lab.enzmann.online
Traefik Reverse Proxy: 192.168.1.48 → traefik-01.lab.enzmann.online
```

IOT-VLAN (Smart Home + Mobile Clients)

```
Homematic CCU: 192.168.100.10 → hm-ccu-uv-01.iot.enzmann.online
Hue Bridge: 192.168.101.1 → hue-wz-bridge01.iot.enzmann.online
Sonos Bridge: 192.168.101.2 → sonos-wz-bridge01.iot.enzmann.online
iPhone Admin: 192.168.101.200 → iphone-admin-01.iot.enzmann.online
iPad Wohnzimmer: 192.168.101.201 → ipad-wz-01.iot.enzmann.online
Samsung TV: 192.168.101.210 → tv-wz-01.iot.enzmann.online
```

Gäste-VLAN (Gast-Zugang)

```
# Automatische DHCP-Zuweisung (192.168.200.10-250)
# Keine statischen Reservierungen für Gäste
```

Geräte-Inventar

Standard-LAN - Homelab & Infrastructure

UniFi Infrastructure (192.168.1.2 - 192.168.1.20)

Gerät	IP	DNS-Name	Öffentlicher Zugang	Notizen
UniFi Controller	192.168.1.2	unifi-controller-01.lab.enzmann.online	-	Controller VM/Hardware
Pi-hole + Unbound	192.168.1.3	pihole-01.lab.enzmann.online	https://pihole-01.lab.enzmann.online	DNS + Ad-Blocking + rekursiver Resolver
UniFi Switch Pro 24	192.168.1.10	switch-main-01.lab.enzmann.online	-	Hauptswitch Arbeitszimmer
UniFi AP Pro 6	192.168.1.11	ap-wz-01.lab.enzmann.online	-	Access Point Wohnzimmer
UniFi AP Pro 6	192.168.1.12	ap-sz-01.lab.enzmann.online	-	Access Point Schlafzimmer

Homelab Core (192.168.1.21 - 192.168.1.40)

Gerät	IP	DNS-Name	Öffentlicher Zugang	Notizen
Proxmox Host 1	192.168.1.21	pve-01.lab.enzmann.online	-	Hauptserver
Proxmox Host 2	192.168.1.22	pve-02.lab.enzmann.online	-	Backup/Cluster
TrueNAS Scale	192.168.1.25	nas-01.lab.enzmann.online	-	Zentraler Storage

Homelab Services (192.168.1.41 - 192.168.1.99)

Gerät	IP	DNS-Name	Öffentlicher Zugang	Notizen
Home Assistant Prod	192.168.1.41	ha-prod-01.lab.enzmann.online	https://ha-prod-01.lab.enzmann.online	Produktiv HA Instance
Home Assistant Test	192.168.1.42	ha-test-01.lab.enzmann.online	-	Test/Development
Docker Swarm Manager	192.168.1.45	docker-01.lab.enzmann.online	-	Swarm Leader
Docker Swarm Worker 1	192.168.1.46	docker-02.lab.enzmann.online	-	Swarm Worker
Docker Swarm Worker 2	192.168.1.47	docker-03.lab.enzmann.online	-	Swarm Worker
Traefik Reverse Proxy	192.168.1.48	traefik-01.lab.enzmann.online	https://traefik-01.lab.enzmann.online	SSL-Terminierung
Portainer	192.168.1.50	portainer-01.lab.enzmann.online	https://portainer-01.lab.enzmann.online	Docker Management
Grafana	192.168.1.51	grafana-01.lab.enzmann.online	https://grafana-01.lab.enzmann.online	Monitoring Dashboard
InfluxDB	192.168.1.52	influx-01.lab.enzmann.online	-	Time Series DB
MQTT Broker	192.168.1.55	mqtt-01.lab.enzmann.online	-	Mosquitto
Prometheus	192.168.1.56	prometheus-01.lab.enzmann.online	-	Metrics Collection
Node Exporter	192.168.1.57	nodeexp-01.lab.enzmann.online	-	System Metrics
Loki	192.168.1.58	loki-01.lab.enzmann.online	-	Log Aggregation
Jaeger	192.168.1.59	jaeger-01.lab.enzmann.online	-	Distributed Tracing
Zusätzliche Services	192.168.1.60-99	-	-	**40 weitere IPs verfügbar**

Client Devices (192.168.1.201 - 192.168.1.220)

Gerät	IP	DNS-Name	Öffentlicher Zugang	Notizen
Admin Desktop	192.168.1.205	desktop-admin-01.lab.enzmann.online	-	Management PC (kabelgebunden)
Admin Laptop	192.168.1.206	laptop-admin-01.lab.enzmann.online	-	Mobile Management (WiFi: "Enzian")
Weitere Clients	192.168.1.207-220	-	-	**Laptops, Drucker (kabelgebunden + WiFi)**

IOT-VLAN - Smart Home Geräte

Unterverteilung (192.168.10.1 - 192.168.10.62)

Gerät	IP	DNS-Name	MAC	Notizen
Homematic CCU	192.168.10.10	hm-ccu-uv-01.iot.local	-	Zentrale
UniFi Switch	192.168.10.11	switch-uv-01.iot.local	-	Hauptverteiler

Flur (192.168.10.65 - 192.168.10.126)

Gerät	IP	DNS-Name	MAC	Notizen
Shelly 1 (Deckenlampe)	192.168.10.70	shelly-1-flur-01.iot.local	-	Hauptlicht
Homematic Bewegungsmelder	192.168.10.71	hm-motion-flur-01.iot.local	-	Eingang

Arbeitszimmer (192.168.10.129 - 192.168.10.190)

Gerät	IP	DNS-Name	MAC	Notizen
Shelly Dimmer	192.168.10.135	shelly-dimmer-az-01.iot.local	-	Schreibtischlampe
Hue Strip	192.168.10.136	hue-az-01.iot.local	-	Monitor-Backlight

Schlafzimmer (192.168.10.193 - 192.168.10.254)

Gerät	IP	DNS-Name	MAC	Notizen
Hue Lampe Links	192.168.10.200	hue-sz-01.iot.local	-	Nachttischlampe
Hue Lampe Rechts	192.168.10.201	hue-sz-02.iot.local	-	Nachttischlampe
Homematic Fensterkontakt	192.168.10.202	hm-window-sz-01.iot.local	-	Fenster Straßenseite

Wohnzimmer (192.168.11.1 - 192.168.11.62)

Gerät	IP	DNS-Name	MAC	Notizen
Hue Bridge	192.168.11.1	hue-wz-bridge01.iot.local	-	Zentrale Bridge
Sonos One	192.168.11.10	sonos-wz-01.iot.local	-	Musikwiedergabe
Hue Deckenlampe	192.168.11.11	hue-wz-01.iot.local	-	Hauptbeleuchtung
Hue Stehlampe	192.168.11.12	hue-wz-02.iot.local	-	Ambientelicht

Küche (192.168.11.65 - 192.168.11.126)

Gerät	IP	DNS-Name	MAC	Notizen
Shelly 1PM (Dunstabzug)	192.168.11.70	shelly-pro1pm-kueche-01.iot.local	-	Dunstabzugsteuerung
Hue Unterbauleuchte	192.168.11.71	hue-kueche-01.iot.local	-	Arbeitsplatte
Sonos One SL	192.168.11.72	sonos-kueche-01.iot.local	-	Küchenmusik
Homematic Temperatursensor	192.168.11.73	hm-temp-kueche-01.iot.local	-	Raumtemperatur

Bad (192.168.11.129 - 192.168.11.190)

Gerät	IP	DNS-Name	MAC	Notizen
Shelly 1 (Lüftung)	192.168.11.135	shelly-1-bad-01.iot.local	-	Lüftungssteuerung
Homematic Feuchtigkeitssensor	192.168.11.136	hm-humid-bad-01.iot.local	-	Luftfeuchtigkeit
Hue Spiegellampe	192.168.11.137	hue-bad-01.iot.local	-	Spiegelbeleuchtung

[Weitere Räume nach gleichem Schema]

Wartungshinweise

Backup-Strategie

- **UniFi Controller:** Täglich automatisch + wöchentlich manuell
- **Proxmox:** Wöchentlich (VMs + Konfiguration)
- **Home Assistant:** Täglich automatisch
- **Docker Swarm:** Backup der compose files + Volumes

Update-Fenster

- **Infrastruktur (UniFi, Proxmox):** Sonntag 02:00-04:00 Uhr
- **Services (Home Assistant, Docker):** Sonntag 04:00-06:00 Uhr
- **IOT-Geräte:** Nach Bedarf, rollierend

Monitoring

- **Homelab:** Grafana + InfluxDB für alle Services

- **IOT:** Home Assistant Device Tracker + Ping-Tests alle 5 Minuten
- **Network:** UniFi Controller Statistiken

Dokumentation aktualisieren

- Bei jeder Geräteerweiterung (IOT)
- Bei Service-Änderungen (Homelab)
- Nach größeren Netzwerkänderungen

Troubleshooting

Homelab-spezifische Probleme

1. **VM nicht erreichbar:**
   - Proxmox Host-Status prüfen
   - VM-Status in Proxmox GUI kontrollieren
   - Network Bridge Konfiguration überprüfen
2. **Docker Service nicht verfügbar:**
   - Swarm Status: `docker node ls`
   - Service Status: `docker service ps <service>`
   - Container Logs: `docker service logs <service>`
3. **Home Assistant Verbindungsprobleme zu IOT:**
   - Firewall-Regeln Standard-LAN → IOT prüfen
   - mDNS-Reflector Status kontrollieren
   - MQTT Broker Erreichbarkeit testen
4. **HTTPS/Traefik Probleme:**
   - **Zertifikat nicht erstellt:**

     bash

     ```
     # Traefik Logs prüfen
     docker service logs traefik_traefik

     # netcup API Credentials testen
     curl -X POST https://ccp.netcup.net/run/webservice/servers/endpoint.php \
       -d '{"action":"login","param":{"customernumber":"123456","apikey":"...","apipassword"
     ```

   - **Service nicht erreichbar über HTTPS:**

bash

```
# DNS Auflösung testen (lokal)
nslookup ha-prod-01.lab.enzmann.online 192.168.1.3

# Traefik Dashboard prüfen: https://traefik-01.lab.enzmann.online
# Router und Services Status kontrollieren
```

- **Wildcard-Zertifikat Probleme:**

bash

```
# ACME Logs prüfen
docker exec -it $(docker ps | grep traefik | cut -d' ' -f1) cat /letsencrypt/acme.json

# DNS Challenge manuell testen
dig TXT _acme-challenge.lab.enzmann.online
dig TXT _acme-challenge.iot.enzmann.online
```

5. **Pi-hole + Unbound DNS-Probleme:**
   - **Lokale Domain nicht auflösbar:**

bash

```
# Pi-hole Status prüfen
docker service logs pihole_pihole

# Unbound Status prüfen
docker service logs pihole_unbound

# DNS-Auflösung manuell testen
nslookup ha-prod-01.lab.enzmann.online 192.168.1.3

# Pi-hole Query-Log prüfen: https://pihole-01.lab.enzmann.online
```

   - **Unbound nicht erreichbar:**

bash

```
# Unbound Container IP prüfen
docker exec -it $(docker ps | grep pihole | cut -d' ' -f1) nslookup google.com 10.0.1.2

# Unbound Konfiguration prüfen
docker exec -it $(docker ps | grep unbound | cut -d' ' -f1) unbound-checkconf

# Unbound Cache-Statistiken
docker exec -it $(docker ps | grep unbound | cut -d' ' -f1) unbound-control stats_nores
```

- **DNS-Auflösung langsam:**

```bash
# Cache-Hit-Rate prüfen
docker exec -it $(docker ps | grep unbound | cut -d' ' -f1) unbound-control stats | gre

# DNS-Query-Zeit testen
dig @192.168.1.3 google.com +stats

# Pi-hole Cache leeren
docker exec -it $(docker ps | grep pihole | cut -d' ' -f1) pihole restartdns
```

- **Wildcard-Domains funktionieren nicht:**

```bash
# dnsmasq Konfiguration prüfen
docker exec -it $(docker ps | grep pihole | cut -d' ' -f1) cat /etc/dnsmasq.d/02-lab-wi

# dnsmasq neu starten
docker exec -it $(docker ps | grep pihole | cut -d' ' -f1) pihole restartdns

# Unbound Forward-Zonen prüfen
docker exec -it $(docker ps | grep unbound | cut -d' ' -f1) cat /opt/unbound/etc/unboun
```

6. **Gäste-VLAN Probleme:**
   - **Gäste haben keinen Internet-Zugang:**

```bash
# VLAN-Zuordnung prüfen
# UniFi Controller → Clients → VLAN-Status kontrollieren

# Firewall-Regeln für Gäste-VLAN → Internet prüfen
# Gateway-Routing für 192.168.200.0/24 kontrollieren
```

   - **Gäste können auf lokale Ressourcen zugreifen:**

```bash
# Firewall-Regeln überprüfen:
# Gäste-VLAN → Standard-LAN: Blockiert (außer DNS)
# Gäste-VLAN → IOT-VLAN: Blockiert

# WiFi Gast-Isolation prüfen ("Enzian-Gast")
# VLAN-Zuordnung von "Enzian-Gast" → VLAN 200 kontrollieren
```

   - **DNS funktioniert nicht für Gäste:**

```bash
# Pi-hole Firewall-Regel prüfen
# Gäste-VLAN → 192.168.1.3:53 erlaubt?

# DNS-Auflösung von Gäste-VLAN testen
nslookup google.com 192.168.1.3
```

IOT-spezifische Probleme

1. **Gerät nicht erreichbar:**
   - VLAN-Zuordnung prüfen
   - DHCP-Lease erneuern
   - Firewall-Regeln überprüfen
2. **DNS-Auflösung funktioniert nicht:**
   - Controller-DNS-Einstellungen prüfen
   - mDNS-Reflector aktivieren
3. **Home Assistant kann IOT-Geräte nicht finden:**
   - Firewall-Regel Standard-LAN → IOT prüfen
   - Integration-spezifische Ports freischalten
   - Network Discovery Settings in HA prüfen

Netzwerk-übergreifende Probleme

1. **Keine Inter-VLAN Kommunikation:**
   - Gateway-Konfiguration prüfen
   - Routing-Tabellen kontrollieren
   - Firewall-Regeln step-by-step testen
2. **Performance-Probleme:**
   - Switch-Auslastung in UniFi Controller prüfen
   - QoS-Einstellungen anpassen
   - Bandbreiten-Limits überprüfen

Erstellt: [Datum]

Letzte Aktualisierung: [Datum]

Version: 4.0 (erweitert um Gäste-VLAN und Mobile Clients)