WolfISMS

Wolf's Informations-Sicherheits-Management-System



Docker-Installation

Version 4.00

Stand: November 2026

© 2012–2026 WolfInfoSec GmbH

1. Überblick

Diese Anleitung beschreibt den Betrieb von WolfISMS als Docker-Container hinter dem Nginx Proxy Manager (NPM). NPM übernimmt dabei die TLS-Terminierung und die Zertifikatsverwaltung, WolfISMS selbst spricht ausschließlich HTTP innerhalb des Docker-Netzwerks.

Sie richtet sich an Administratoren, die eine Docker-Umgebung betreiben oder neu aufsetzen; wie ein Ubuntu Server als Docker-Host eingerichtet wird, beschreibt Kapitel 3. Für die klassische Installation direkt auf einem Ubuntu- oder Debian-Server mit Apache und mod_wsgi siehe stattdessen Kapitel 3 der Administrationsdokumentation.

1.1 Zielarchitektur

Komponente	Aufgabe
Nginx Proxy Manager	Nimmt HTTP (80) und HTTPS (443) von außen entgegen, terminiert TLS, verwaltet die Let's-Encrypt-Zertifikate und leitet die Anfragen im internen Netz an den WolfISMS-Container weiter. Die Administrationsoberfläche läuft auf Port 81
WolfISMS-Container	Die Anwendung selbst. Sie ist von außen nicht direkt erreichbar, sondern ausschließlich über das Docker-Netzwerk proxy
Datenbank	MariaDB bzw. MySQL – als eigener Container oder als bereits vorhandener Datenbankserver, umgeschaltet über DB_MODUS (siehe Kapitel 6)
Docker-Netzwerk "proxy"	Verbindet NPM und WolfISMS. Es wird einmalig manuell angelegt und von beiden Compose-Projekten als externes Netz eingebunden

1.2 Warum HTTPS zwingend erforderlich ist

WolfISMS setzt seine Session-Cookies mit dem Attribut Secure=True. Ein Browser überträgt solche Cookies ausschließlich über verschlüsselte Verbindungen. Über reines HTTP kommt daher zwar die Loginseite zustande, die Anmeldung schlägt aber fehl, weil die Session nicht gehalten werden kann. Ein gültiges Zertifikat ist damit keine Empfehlung, sondern Betriebsvoraussetzung.

Reverse-Proxy-Unterstützung ist eingebaut:
Die Anwendung wertet die Header X-Forwarded-For, X-Forwarded-Proto, X-Forwarded-Host und X-Forwarded-Port des Proxys aus (ProxyFix in isms.py). Sie erkennt dadurch korrekt, dass sie hinter einem TLS-Terminierungspunkt läuft, erzeugt absolute Links mit https und dem öffentlichen Hostnamen und zählt das Rate-Limit je Client statt je Proxy. Wie viele Proxys vor ihr stehen, sagt ISMS_PROXY_HOPS – das Image setzt 1 (Kapitel 9.6). Der Nginx Proxy Manager setzt diese Header standardmäßig – eine zusätzliche Konfiguration ist nicht nötig.

© 2012–2026 WolfInfoSec GmbH

2. Voraussetzungen

Komponente	Anforderung
Docker Engine	Aktuelle Version mit dem Plug-in Compose V2 ab 2.20 (docker compose, nicht docker-compose)
Betriebssystem	Ubuntu Server 24.04 LTS empfohlen (Einrichtung siehe Kapitel 3); jeder andere Linux-Docker-Host ist ebenso möglich
Öffentlicher DNS-Eintrag	Der gewünschte Hostname, z. B. isms.ihre-domain.de, muss auf die öffentliche IP-Adresse des Docker-Hosts zeigen
Portfreigaben	Die Ports 80 und 443 müssen von außen auf den Docker-Host erreichbar sein. Port 80 wird für die ACME-Challenge von Let's Encrypt benötigt
Port 81	Die NPM-Administrationsoberfläche. Sie sollte nicht öffentlich erreichbar sein, sondern nur aus dem internen Netz oder über VPN
Installationspaket	Das WolfISMS-Repository bzw. Docker-Paket mit docker-compose.yml und .envdefault (vom Hersteller bereitgestellt)
Datenbank	MariaDB 10.11+ oder MySQL 8.0+ – als Container oder als externer Server

Port 81 nicht ins Internet veröffentlichen:
Die Administrationsoberfläche des Nginx Proxy Managers verwaltet sämtliche Zertifikate und Weiterleitungen. Beschränken Sie den Zugriff über die Firewall des Hosts auf das interne Netz und ändern Sie die Standard-Zugangsdaten unmittelbar nach der ersten Anmeldung.

© 2012–2026 WolfInfoSec GmbH

3. Vorbereitung – Ubuntu Server und Docker installieren

Dieses Kapitel richtet einen neuen Docker-Host ein. Wer bereits einen Linux-Server mit Docker Engine und Compose-Plug-in (2.20 oder neuer) betreibt, kann es überspringen und direkt mit Schritt 1 fortfahren. Beschrieben ist Ubuntu Server 24.04 LTS; die Befehle gelten unverändert auch für 22.04 LTS.

3.1 Server-Ressourcen

Ressource	Empfehlung
CPU	2 virtuelle Kerne, für mehr als 20 gleichzeitige Benutzer 4
Arbeitsspeicher	4 GB, mit internem Datenbankcontainer besser 8 GB
Festplatte	40 GB für System, Images und Datenbank; dazu Platz für hochgeladene Dokumente und Sicherungen
Netzwerk	Feste IPv4-Adresse (statisch oder per DHCP-Reservierung) und Internetzugang für Paketquellen, Docker-Images und Let's Encrypt

3.2 Ubuntu Server installieren

Das Installationsabbild „Ubuntu Server 24.04 LTS“ steht unter https://ubuntu.com/download/server bereit. Es wird auf einen USB-Stick geschrieben oder in der Virtualisierung (VMware, Proxmox, Hyper-V) als virtuelles Laufwerk eingebunden. Der Installer fragt die folgenden Punkte ab:

- **Sprache und Tastatur**: Sprache nach Wahl, Tastaturlayout German
- **Installationsart**: Ubuntu Server (nicht die minimierte Variante)
- **Netzwerk**: die Schnittstelle auf „Edit IPv4“ → Manual stellen und Adresse, Gateway und DNS-Server eintragen – oder DHCP lassen, wenn der DHCP-Server die Adresse fest reserviert
- **Proxy und Paketquelle**: leer lassen bzw. die Vorgabe übernehmen
- **Speicher**: Use an entire disk mit „Set up this disk as an LVM group“. Der Installer nutzt von der Volume Group zunächst nur einen Teil; die logische Partition im Speicherüberblick auf die volle Größe ziehen
- **Profil**: Name, Servername (z. B. isms), Benutzername und Passwort des Administrationskontos. root bleibt ohne Passwort; administriert wird mit sudo
- **Ubuntu Pro**: überspringen (Skip for now)
- **SSH**: „Install OpenSSH server“ aktivieren, damit der Server sich anschließend aus der Ferne verwalten lässt
- **Featured Server Snaps**: nichts auswählen – insbesondere nicht das Snap docker

Docker nicht als Snap installieren:
Das Snap-Paket docker aus dem Installer legt Daten und Konfiguration unter /var/snap ab, kennt nicht alle Compose-Funktionen und ist mit dieser Anleitung nicht getestet. Docker wird in Abschnitt Docker Engine installieren aus dem offiziellen Docker-Repository eingerichtet.

Nach dem Neustart meldet sich der Administrator per SSH an (ssh <benutzer>@<server-ip>).

3.3 System aktualisieren und Grundeinstellungen

Zuerst alle Pakete auf den aktuellen Stand bringen und die Zeitzone setzen. Die Zeitzone bestimmt die Uhrzeit im Protokoll des Hosts; Zertifikate und Zwei-Faktor-Codes verlangen außerdem eine synchrone Uhr, die Ubuntu über systemd-timesyncd bereits hält.

© 2012–2026 WolfInfoSec GmbH

```
sudo apt update && sudo apt full-upgrade -y
sudo timedatectl set-timezone Europe/Berlin
timedatectl
sudo reboot
```

Sicherheitsaktualisierungen spielt Ubuntu über unattended-upgrades automatisch ein. Ob der Dienst aktiv ist, zeigt:

```
sudo dpkg-reconfigure -plow unattended-upgrades
```

3.4 Firewall einrichten

Die Host-Firewall ufw lässt SSH sowie HTTP und HTTPS für den Nginx Proxy Manager zu. SSH unbedingt vor dem Einschalten freigeben, sonst sperrt man sich aus:

```
sudo ufw allow OpenSSH
sudo ufw allow 80/tcp
sudo ufw allow 443/tcp
sudo ufw enable
sudo ufw status verbose
```

ufw schützt keine Docker-Ports:
Docker trägt veröffentlichte Ports (ports: in der docker-compose.yml) direkt in iptables ein, an ufw vorbei. Eine ufw-Regel für Port 81 bleibt deshalb wirkungslos. Die NPM-Administrationsoberfläche wird stattdessen nur an die interne Adresse gebunden, z. B. "10.0.0.5:81:81" statt "81:81" in Schritt 2, oder über die vorgelagerte Firewall abgeschottet.

3.5 Docker Engine installieren

Die Ubuntu-Pakete docker.io und docker-compose-v2 sind älter als die Docker-eigenen und werden – falls vorhanden – zuerst entfernt. Auf einem frisch installierten Server meldet der Befehl nur, dass die Pakete nicht installiert sind:

```
for pkg in docker.io docker-doc docker-compose docker-compose-v2 \
           podman-docker containerd runc; do
  sudo apt-get remove -y $pkg
done
```

Danach das offizielle Docker-Repository samt Signaturschlüssel einbinden:

```
sudo apt-get update
sudo apt-get install -y ca-certificates curl git
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg \
     -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
echo "deb [arch=$(dpkg --print-architecture) \
  signed-by=/etc/apt/keyrings/docker.asc] \
  https://download.docker.com/linux/ubuntu \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" \
  | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Und die Docker Engine mit dem Compose-Plug-in installieren:

© 2012–2026 WolfInfoSec GmbH

```
sudo apt-get update
sudo apt-get install -y docker-ce docker-ce-cli containerd.io \
     docker-buildx-plugin docker-compose-plugin
```

3.6 Installation prüfen und Benutzer berechtigen

```
sudo systemctl enable --now docker
docker --version
docker compose version
sudo docker run --rm hello-world
```

docker compose version muss 2.20 oder neuer melden (Voraussetzung für die Datenbankumschaltung in Schritt 3). hello-world gibt „Hello from Docker!“ aus. Damit der Administrator Docker ohne sudo bedienen kann, wird er in die Gruppe docker aufgenommen; die Gruppenmitgliedschaft greift erst nach einer neuen Anmeldung:

```
sudo usermod -aG docker $USER
exit
# neu per SSH anmelden, dann:
docker ps
```

Gruppe docker bedeutet Root-Rechte:
Wer Docker bedienen darf, kann jeden Container mit Zugriff auf das Dateisystem des Hosts starten. Nur Administratoren in die Gruppe aufnehmen.

Der Host ist damit vorbereitet. Weiter geht es mit dem Docker-Netzwerk in Schritt 1.

© 2012–2026 WolfInfoSec GmbH

4. Schritt 1 – Docker-Netzwerk anlegen

Beide Compose-Projekte – der Nginx Proxy Manager und WolfISMS – binden dasselbe externe Netzwerk ein. Es wird auf dem Docker-Host einmalig angelegt:

```
docker network create proxy
```

Die Prüfung, ob das Netz existiert:

```
docker network ls | grep proxy
```

Das Netzwerk muss vor dem ersten docker compose up bestehen. Fehlt es, brechen beide Projekte mit der Meldung „network proxy declared as external, but could not be found“ ab.

© 2012–2026 WolfInfoSec GmbH

5. Schritt 2 – Nginx Proxy Manager bereitstellen

Legen Sie auf dem Docker-Host ein eigenes Verzeichnis für den Reverse Proxy an und darin die Datei docker-compose.yml:

```
mkdir -p /opt/npm
cd /opt/npm
```

Inhalt der Datei /opt/npm/docker-compose.yml:

```
services:
  npm:
    image: jc21/nginx-proxy-manager:2
    restart: unless-stopped
    ports:
      - "80:80"     # HTTP + ACME-Challenge
      - "443:443"   # HTTPS
      - "81:81"     # Administrationsoberflaeche
    volumes:
      - npm_data:/data
      - npm_letsencrypt:/etc/letsencrypt
    networks:
      - proxy

networks:
  proxy:
    external: true
    name: proxy

volumes:
  npm_data:
  npm_letsencrypt:
```

Anschließend den Container starten:

```
docker compose up -d
```

5.1 Erste Anmeldung

Die Administrationsoberfläche ist danach unter http://<docker-host>:81 erreichbar. Der Nginx Proxy Manager legt beim ersten Start ein Standardkonto an, dessen Zugangsdaten in der Dokumentation des Images genannt werden.

> Ändern Sie unmittelbar nach der ersten Anmeldung E-Mail-Adresse und Passwort des Standardkontos. Solange die Vorgabewerte gelten, kann jeder mit Netzzugriff auf Port 81 die Zertifikate und Weiterleitungen der gesamten Umgebung verändern.

© 2012–2026 WolfInfoSec GmbH

6. Schritt 3 – WolfISMS bereitstellen

Das WolfISMS-Paket enthält die docker-compose.yml sowie die Vorlage .envdefault. Die eigentliche Konfiguration erfolgt ausschließlich über Umgebungsvariablen in der Datei .env.

6.1 Paket beziehen und Konfiguration anlegen

```
git clone https://github.com/wolfinfosec/wolfisms.git
cd wolfisms
cp .envdefault .env
```

Danach die Datei .env öffnen und die Werte an die eigene Umgebung anpassen. Die vollständige Referenz aller Variablen enthält Kapitel 9.

Werden eigene Zertifikate benötigt – etwa für die Verbindung zu einem externen Datenbankserver –, werden diese in das Verzeichnis db-certs/ kopiert (siehe Datenbank: intern oder extern). Für die TLS-Terminierung nach außen ist das nicht erforderlich; darum kümmert sich der Nginx Proxy Manager.

6.2 Datenbank: intern oder extern

Die Variable DB_MODUS in der .env entscheidet, ob MariaDB als Container im Stack mitläuft. docker-compose.yml bindet dazu per include die Datei docker-compose.db-<DB_MODUS>.yml ein. Fehlt der Eintrag – etwa in einer .env aus einer älteren Fassung –, gilt intern; bestehende Installationen laufen also unverändert weiter.

DB_MODUS	Verhalten
intern (Vorgabe)	docker-compose.db-intern.yml: Container db mit MariaDB 11.4 und Volume db_data. Beim ersten Start legt er Datenbank und Benutzer an und spielt einen Dump aus ./db-init ein. DB_HOST bleibt leer, DB_ROOT_PASSWORD ist Pflicht
extern	docker-compose.db-extern.yml: Es startet kein Datenbankcontainer. App und Cron verbinden sich mit DB_HOST und DB_PORT; DB_ROOT_PASSWORD wird nicht benutzt

Für eine externe Datenbank müssen Datenbank und Benutzer auf dem Server bereits angelegt sein, zum Beispiel:

```
CREATE DATABASE isms CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
CREATE USER 'isms'@'%' IDENTIFIED BY '<passwort>';
GRANT ALL PRIVILEGES ON isms.* TO 'isms'@'%';
```

- **Verbindung**: DB_HOST, DB_PORT, DB_NAME, DB_USER und DB_PASSWORD in der .env; DB_TYPE bleibt mariadb (oder mysql). Die Werte gelten für App und Cron gleichermaßen
- **Datenbank auf demselben Host**: DB_HOST=localhost zeigt im Container auf den Container selbst – die LAN-IP des Hosts eintragen. MariaDB muss auf dieser Adresse lauschen (bind-address), Port 3306 in der Firewall für das Docker-Netz offen sein, und der Benutzer muss aus dem Docker-Netz kommen dürfen, etwa 'isms'@'172.16.0.0/255.240.0.0' mit demselben GRANT wie oben. Teilt sich der Container die Datenbank mit einer bestehenden Installation, den Cron-Dienst nicht mitstarten (docker compose up -d --scale cron=0) – sonst laufen alle Cron-Aufgaben doppelt und Erinnerungsmails gehen zweimal hinaus
- **TLS**: DB_SSL_MODE=required verschlüsselt ohne Zertifikatsprüfung, verify-full prüft Zertifikat und Hostnamen gegen DB_SSL_CA. Die Zertifikatsdateien gehören in den Ordner db-certs/ neben der

© 2012–2026 WolfInfoSec GmbH

docker-compose.yml; im Container liegen sie unter /var/www/isms/db-certs (schreibgeschützt), z. B. DB_SSL_CA=/var/www/isms/db-certs/db-ca.pem

- **Datensicherung**: Das Backup im Adminbereich nutzt dieselbe Verbindung. Der Benutzer braucht dafür auch Leserechte auf Routinen, Trigger und Events – GRANT ALL auf die Datenbank genügt
- **Daten übernehmen**: Eine bestehende Installation zieht um, indem ihr Dump (Backup im Adminbereich) auf dem neuen Server eingespielt wird, bevor DB_MODUS umgestellt wird
- **Voraussetzung**: include und die optionale Abhängigkeit (depends_on … required: false) verlangen Docker Compose 2.20 oder neuer

Betriebsart wechseln:
Erst docker compose down (das Volume db_data bleibt erhalten), dann DB_MODUS in der .env ändern, dann docker compose up -d. Die Datei docker-compose.db-extern.yml enthält den Dienst db nur als abgeschalteten Platzhalter (Profil nie-starten) – das Profil bitte nicht in COMPOSE_PROFILES eintragen.

6.3 Container starten

```
docker compose up -d
```

Den Start kontrollieren:

```
docker compose ps
docker compose logs -f
```

6.4 Anbindung an den Reverse Proxy

Damit der Nginx Proxy Manager den Anwendungscontainer erreicht, müssen beide im Netzwerk proxy liegen und der Container unter einem stabilen Namen ansprechbar sein. Dieser Name wird über die Variable PROXY_ALIAS gesetzt und im Proxy Host als Forward Hostname eingetragen. Die mitgelieferte docker-compose.yml bindet das Netzwerk bereits ein; das folgende Beispiel zeigt die maßgeblichen Abschnitte:

```
services:
  wolfisms:
    # ... image, volumes, depends_on ...
    env_file: .env
    restart: unless-stopped
    networks:
      proxy:
        aliases:
          - ${PROXY_ALIAS}    # z. B. wolfisms-app

networks:
  proxy:
    external: true
    name: proxy
```

© 2012–2026 WolfInfoSec GmbH

Port der Anwendung nur auf 127.0.0.1:
Der WolfISMS-Container wird ausschließlich über das interne Netzwerk proxy angesprochen. Die docker-compose.yml veröffentlicht Port 8000 nur auf 127.0.0.1 des Docker-Hosts – zur Diagnose, etwa mit curl auf dem Host oder über einen SSH-Tunnel; von außen ist er nicht erreichbar. Diese Bindung bitte nicht ändern: Ein von außen erreichbarer Port würde die TLS-Terminierung umgehen, die Anmeldung wegen der Secure-Cookies unbrauchbar machen und – weil die Anwendung die X-Forwarded-*-Header des Proxys übernimmt – jedem Client erlauben, seine IP-Adresse oder eine HTTPS-Verbindung vorzutäuschen.

Die verbindliche docker-compose.yml ist Bestandteil des Installationspakets. Das obige Beispiel zeigt nur die für die Proxy-Anbindung maßgeblichen Abschnitte und ersetzt die mitgelieferte Datei nicht.

© 2012–2026 WolfInfoSec GmbH

7. Schritt 4 – Proxy Host und Zertifikat einrichten

Die Veröffentlichung erfolgt anschließend in der NPM-Oberfläche unter Hosts → Proxy Hosts → Add Proxy Host.

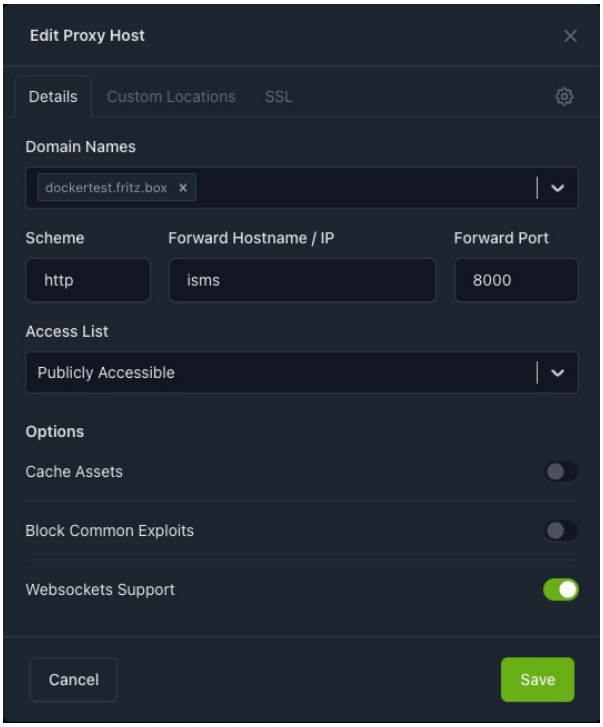
7.1 Reiter „Details“

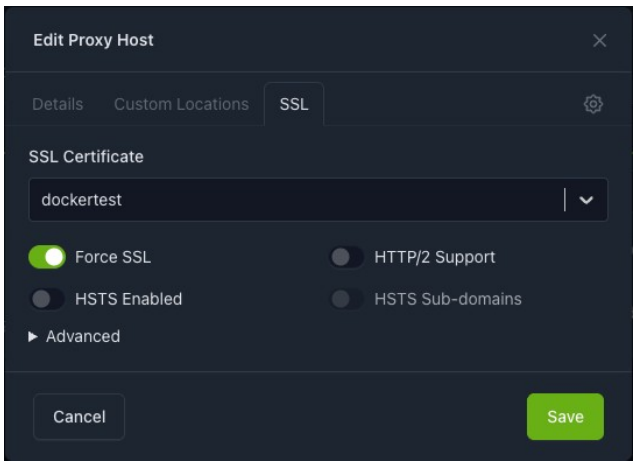
Feld	Wert
Domain Names	isms.ihre-domain.de
Scheme	http
Forward Hostname	wolfisms-app (der in PROXY_ALIAS gesetzte Name)
Forward Port	8000
Block Common Exploits	aktiviert
Websockets Support	aktiviert

7.2 Reiter „SSL“

- **SSL Certificate**: „Request a new SSL Certificate“ auswählen
- **Force SSL**: aktivieren – leitet HTTP-Aufrufe automatisch auf HTTPS um
- **HTTP/2 Support**: aktivieren
- **E-Mail-Adresse**: für die Ablaufbenachrichtigungen von Let's Encrypt hinterlegen und die Nutzungsbedingungen bestätigen

Let's Encrypt setzt für die Standard-Challenge voraus, dass der Domainname öffentlich auf die IP-Adresse des Hosts zeigt und Port 80 von außen erreichbar ist. Ist das nicht gegeben – etwa in einer rein internen Umgebung –, muss stattdessen eine DNS-Challenge über den DNS-Anbieter konfiguriert oder ein eigenes Zertifikat hochgeladen werden.





7.3 Installation prüfen

Prüfpunkt	Erwartetes Ergebnis
https://isms.ihre-domain.de aufrufen	Die Loginseite erscheint, das Zertifikat wird als gültig angezeigt
HTTP-Aufruf derselben Adresse	Automatische Weiterleitung auf HTTPS (Force SSL)
Anmeldung	Weiterleitung auf das Dashboard – gelingt die Anmeldung nicht, siehe Kapitel 10
Containerprotokoll	docker compose logs ohne Traceback-Einträge
Adminbereich	Der Menüpunkt „Updates“ fehlt – das ist im Container-Modus so vorgesehen (siehe Kapitel 8)

Unmittelbar nach der ersten Anmeldung:
1) Standardpasswort des Admin-Benutzers ändern
2) Lizenzdatei customer.lic einspielen (Admin → Lizenz)
3) Container neu starten, damit die Lizenz und der daraus abgeleitete Secret Key wirksam werden: docker compose restart

© 2012–2026 WolfInfoSec GmbH

8. Betrieb im Container-Modus

Mit der Umgebungsvariablen WOLFISMS_CONTAINER=1 erkennt die Anwendung, dass sie in einem Container läuft (config.container_mode). Das ändert das Verhalten an zwei Stellen.

8.1 In-App-Updater gesperrt

Der gesamte Update-Bereich einschließlich /admin/updates ist gesperrt und antwortet mit HTTP 404; die Kachel im Adminbereich wird ausgeblendet. Das ist beabsichtigt: Ein Update im laufenden Container würde beim nächsten Neustart des Images verloren gehen.

Aktualisierungen erfolgen im Docker-Betrieb ausschließlich über ein neues Image:

```
docker compose pull
docker compose up -d
```

8.2 Konfiguration ausschließlich aus der Umgebung

Außerhalb von Docker liest config.py die Zugangsdaten aus der Datei /etc/wolfisms/secret.env. Im Container existiert diese Datei in der Regel nicht – das ist kein Fehler: Fehlt sie, verwendet die Anwendung ausschließlich die gesetzten Umgebungsvariablen. Genau diese liefert die Datei .env über docker compose.

Abweichender Pfad:
Soll doch eine Env-Datei im Container verwendet werden, lässt sich ihr Pfad über die Variable WOLFISMS_ENV_FILE setzen. Bereits gesetzte Umgebungsvariablen haben dabei immer Vorrang vor den Werten aus der Datei.

8.3 Datenhaltung und Sicherung

Persistent zu halten sind die Datenbank sowie das Upload-Verzeichnis. Beide müssen als Volume eingebunden sein, sonst gehen sie beim Austausch des Containers verloren.

Daten	Ort	Hinweis
Datenbank	Volume des Datenbankcontainers bzw. externer Datenbankserver	Enthält sämtliche ISMS-Inhalte
Hochgeladene Dateien	UPLOAD_DIR (Standard /var/www/isms/wolfisms/uploads)	Anhänge, Playbooks, Logo, Vorlagen und Briefpapier
Sessions	SESSION_DIR (Standard /var/www/isms/wolfisms/sessions)	Muss beschreibbar sein; ein Verlust meldet lediglich alle Benutzer ab. Die Rechte setzt die Anwendung beim Start selbst auf 0700, die Sitzungsdateien auf 0600
Lizenzdatei	license/customer.lic im Anwendungsverzeichnis	Nach einem Image-Wechsel prüfen, ob sie noch vorhanden ist

Die Sicherung selbst erfolgt wie bei der klassischen Installation über den Adminbereich unter /admin/backup oder über den Direktlink /backup/download. Das Archiv enthält den Datenbank-

© 2012–2026 WolfInfoSec GmbH

Dump und das Upload-Verzeichnis; Anwendungscode, Konfiguration und Lizenzdatei sind nicht enthalten und müssen separat gesichert werden. Einzelheiten enthält das Kapitel „Datensicherung (Backup)" der Administrationsdokumentation.

8.4 Protokolle

Zweck	Befehl
Laufende Ausgabe verfolgen	docker compose logs -f
Nur die Anwendung	docker compose logs -f wolfisms
Letzte 200 Zeilen	docker compose logs --tail=200
Zugriffe und Zertifikate des Proxys	docker compose logs -f npm (im Verzeichnis /opt/npm)

© 2012–2026 WolfInfoSec GmbH

9. Konfigurationsreferenz (.env)

Alle Parameter werden als Umgebungsvariablen gesetzt. Die Auflösung erfolgt in dieser Reihenfolge: bereits gesetzte Umgebungsvariablen, danach die Env-Datei, zuletzt die im Code hinterlegten Vorgabewerte. Für DB_PASSWORD gibt es bewusst keinen Vorgabewert – fehlt die Variable, startet die Anwendung nicht.

9.1 Pflichtangaben

Variable	Beispiel	Bedeutung
DB_PASSWORD	(starkes Passwort)	Datenbankpasswort – einziger Wert ohne Vorgabe
DB_USER	isms	Datenbankbenutzer (Vorgabe: wolf)
DB_HOST	db	Leer = interner Container db; bei DB_MODUS=extern Hostname oder IP des Datenbankservers
DB_NAME	isms	Name der Datenbank
WEBPATH	https://isms.ihre-domain.de	Öffentliche Adresse der Instanz mit https:// und ohne Schrägstrich am Ende – genau so, wie sie im Browser steht. Nach der Anmeldung leitet die Anwendung dorthin weiter, Links in E-Mails bauen darauf auf. Ein Unterpfad wie /isms wird nicht unterstützt, dafür eine eigene (Sub-)Domain verwenden
ISMS_SECRET_KEY	(leer)	Optional. Leer lassen genügt: Die Anwendung erzeugt beim ersten Start einen Schlüssel für die Sitzungen und legt ihn im Volume sessions ab, er übersteht Neustarts und Updates. Nur setzen, wenn mehrere Instanzen denselben Schlüssel brauchen
WOLFISMS_CONTAINER	1	Aktiviert den Container-Modus (siehe Kapitel 8)
PROXY_ALIAS	wolfisms-app	Netzwerk-Alias des Containers; im Proxy Host als Forward Hostname einzutragen

Soll ISMS_SECRET_KEY gesetzt werden, einen Zufallswert so erzeugen:

```
python3 -c "import secrets; print(secrets.token_hex(32))"
```

9.2 Datenbank

Variable	Vorgabe	Bedeutung
DB_MODUS	intern	intern = MariaDB-Container im Stack, extern = vorhandener Datenbankserver (siehe Abschnitt Datenbank: intern oder extern)
DB_TYPE	mariadb	mariadb oder mysql; mssql ist vorgesehen und wird derzeit beim Start abgelehnt
DB_PORT	(leer)	Leer = 3306
DB_ROOT_PASSWORD	–	Nur bei intern: root-Passwort des Datenbankcontainers
DB_SSL_MODE	(leer)	leer = unverändert, disabled, required oder verify-

© 2012–2026 WolfInfoSec GmbH

Variable	Vorgabe	Bedeutung
		full
DB_SSL_CA	(leer)	CA-Datei im Container, Pflicht bei verify-full
DB_SSL_CERT / DB_SSL_KEY	(leer)	Client-Zertifikat und Schlüssel, falls der Server sie verlangt

Ungültige Werte – ein unbekannter DB_TYPE, ein falscher DB_SSL_MODE oder verify-full ohne DB_SSL_CA – lassen die Anwendung mit einer klaren Meldung im Protokoll gar nicht erst starten.

9.3 Pfade und Instanz

Variable	Vorgabe	Bedeutung
ISMS_BASE_URL	(leer)	Nur nötig, wenn WEBPATH keine vollständige Adresse ist: Basis der Links in E-Mails. Steht in WEBPATH eine Adresse mit https://, wird diese verwendet und ISMS_BASE_URL ignoriert
PATHNAME	isms	Verzeichnisname der Instanz
UPLOAD_DIR	/var/www/isms/wolfisms/uploads	Pfad für hochgeladene Dateien – als Volume einbinden
SESSION_DIR	/var/www/isms/wolfisms/sessions	Pfad für serverseitige Sessions – muss beschreibbar sein
WOLFISMS_ENV_FILE	/etc/wolfisms/secret.env	Abweichender Pfad einer Env-Datei im Container

9.4 Sprache

Variable	Vorgabe	Bedeutung
DEUTSCH	True	Deutsche Oberfläche zulassen
ENGLISCH	False	Englische Oberfläche zulassen. Mit DEUTSCH=0 und ENGLISCH=1 läuft die Installation rein englisch

9.5 Single Sign-On (optional)

Variable	Bedeutung
MS_AUTH	1 = Anmeldung über Microsoft Entra ID aktivieren
GOOGLE_AUTH	1 = Anmeldung über Google aktivieren
SSO_ALLOWED_DOMAINS	Kommaseparierte Allowlist zugelassener E-Mail-Domains. Eine leere Liste lässt niemanden zu – das ist der sichere Vorgabewert
SSO_AUTO_CREATE	1 = unbekannte SSO-Benutzer automatisch anlegen (Vorgabe 0)
MS_TENANT_ID / MS_CLIENT_ID / MS_CLIENT_SECRET / MS_REDIRECT_URI	Zugangsdaten und Callback-URL für Microsoft
GOOGLE_CLIENT_ID / GOOGLE_CLIENT_SECRET / GOOGLE_REDIRECT_URI	Zugangsdaten und Callback-URL für Google

© 2012–2026 WolfInfoSec GmbH

Die Callback-URL muss auf den öffentlichen Hostnamen zeigen, den der Reverse Proxy bedient – nicht auf den Containernamen.

9.6 Weitere Variablen

Variable	Bedeutung
UPDATE_PROXY	Proxy für ausgehende Verbindungen. Im Container-Modus ohne Wirkung auf den gesperrten Updater
RATELIMIT_STORAGE_URI	Speicher für die Rate-Limit-Zählung. Die Vorgabe memory:// zählt je Arbeitsprozess – mit mehreren Workern vervielfacht sich damit jedes Limit. Das Image startet drei Worker; solange memory:// gilt, steht deshalb beim Start eine Warnung im Protokoll. Auf einen gemeinsamen Speicher setzen, etwa redis://redis:6379
ISMS_PROXY_HOPS	Anzahl der Reverse-Proxys vor der Anwendung, deren X-Forwarded-*-Header gelten. Das Image setzt 1 (der Nginx Proxy Manager). Stehen zwei Proxys hintereinander, 2 eintragen; 0 schaltet die Auswertung ab. Zu hoch gesetzt könnte ein Client die Header selbst mitschicken und so seine IP oder den Hostnamen fälschen
ISMS_DEBUG	1 = Flask-Debug-Modus einschalten. Nur zur Fehlersuche und danach wieder entfernen

ISMS_DEBUG darf im Produktivbetrieb nicht gesetzt bleiben. Der Debug-Modus schaltet PROPAGATE_EXCEPTIONS ein, deaktiviert das Template-Caching und ist in jeder Härtungs-Baseline ein Finding.

© 2012–2026 WolfInfoSec GmbH

10. Fehlersuche

Symptom	Ursache und Abhilfe
„network proxy declared as external, but could not be found“	Das Docker-Netzwerk fehlt. Mit docker network create proxy anlegen (Kapitel 4) und die Container erneut starten
„permission denied while trying to connect to the Docker daemon socket“	Der Benutzer ist nicht in der Gruppe docker oder hat sich seit usermod nicht neu angemeldet (Kapitel 3). Neu anmelden oder den Befehl mit sudo ausführen
NPM meldet „502 Bad Gateway“	Der Anwendungscontainer ist über den eingetragenen Forward Hostname nicht erreichbar. Prüfen, ob der Container läuft, ob er im Netzwerk proxy liegt und ob PROXY_ALIAS mit dem im Proxy Host eingetragenen Namen übereinstimmt
Loginseite erscheint, die Anmeldung schlägt aber wortlos fehl	Der Zugriff erfolgt über HTTP statt HTTPS. Session-Cookies sind Secure=True und werden über unverschlüsselte Verbindungen nicht gesetzt. „Force SSL“ im Proxy Host aktivieren
Container startet nicht, Protokoll nennt die Datenbank	DB_PASSWORD fehlt oder die Datenbank ist noch nicht bereit. Die Variable ist Pflicht und hat bewusst keinen Vorgabewert; die Startreihenfolge über depends_on prüfen
„depends on undefined service db“ oder Fehler bei include	Docker Compose ist älter als 2.20, oder eine der Dateien docker-compose.db-intern.yml / docker-compose.db-extern.yml fehlt bzw. DB_MODUS hat einen anderen Wert als intern oder extern
Protokoll: „WARNUNG: Datenbank-Host ‚db‘ ist nicht auflösbar“	DB_MODUS=extern ist gesetzt, DB_HOST aber leer – oder DB_HOST=db bei externer Datenbank. DB_HOST auf den Datenbankserver setzen bzw. für die interne Datenbank DB_MODUS=intern
Zertifikatsanforderung schlägt fehl	Der DNS-Eintrag zeigt nicht auf den Host oder Port 80 ist von außen nicht erreichbar. Beides prüfen oder auf eine DNS-Challenge ausweichen
Links im ISMS zeigen auf http:// oder den Containernamen	Der Proxy übergibt X-Forwarded-Proto bzw. X-Forwarded-Host nicht, oder WEBPATH ist falsch gesetzt. Beide Angaben prüfen
Der Menüpunkt „Updates“ fehlt im Adminbereich	Kein Fehler: Im Container-Modus ist der In-App-Updater gesperrt. Aktualisierung über ein neues Image (Kapitel 8.1)
Nach dem Einspielen der Lizenz gilt weiterhin der Demo-Modus	Die Lizenzprüfung ist zwischengespeichert. Container neu starten: docker compose restart
Hochgeladene Dateien sind nach einem Update verschwunden	UPLOAD_DIR war nicht als Volume eingebunden. Volume ergänzen und die Dateien aus der letzten Sicherung wiederherstellen
Anmeldung klappt, die Seite springt aber sofort zur Loginseite zurück	WEBPATH weicht von der Adresse im Browser ab – anderer Hostname, Zugriff über die IP oder noch der Vorgabewert aus .envdefault. Die Weiterleitung nach der Anmeldung führt dann auf einen Host ohne Sitzungscookie. WEBPATH genau auf die aufgerufene Adresse setzen und docker compose up -d
Externe Datenbank: „Access denied for user ‚isms‘@‚172.…‘“	Der Datenbankbenutzer ist nur für localhost oder eine andere Adresse angelegt; die Verbindung aus dem Container kommt aus dem Docker-Netz. Benutzer auch für dieses Netz anlegen (Kapitel 6.2) und prüfen, was im Container ankommt: docker compose exec app printenv DB_USER DB_PASSWORD DB_HOST
Proxy und Anwendung reagieren sehr langsam	Meist hängt eine Namensauflösung. MariaDB löst die Adresse jedes Clients rückwärts auf: skip-name-resolve in der MariaDB-Konfiguration setzen (Benutzer dann mit IP, Netz oder % anlegen, nicht mit Hostnamen). Wartet der Proxy Manager auf IPv6-Antworten, in seinem Dienst unter environment DISABLE_IPV6: 'true' eintragen und ihn mit docker compose up -d --force-recreate neu erstellen
Anmeldeversuche enden mit „zu	Das Rate-Limit zählt je Client-IP. Kommt die Client-Adresse nicht an –

© 2012–2026 WolfInfoSec GmbH

Symptom	Ursache und Abhilfe
viele Anfragen“ (429), obwohl kaum jemand arbeitet	etwa weil ISMS_PROXY_HOPS=0 gesetzt wurde oder ein zweiter Proxy davorsteht –, sieht die Anwendung nur die Proxy-IP, und alle Benutzer teilen sich ein Kontingent. ISMS_PROXY_HOPS auf die Zahl der Proxys setzen und prüfen, dass der Proxy X-Forwarded-For weitergibt

© 2012–2026 WolfInfoSec GmbH

11. Checkliste

✓	Aufgabe
☐	DNS-Eintrag zeigt auf die öffentliche IP-Adresse des Docker-Hosts
☐	Ports 80 und 443 sind von außen erreichbar
☐	Port 81 ist auf das interne Netz beschränkt
☐	Ubuntu Server installiert und aktualisiert, ufw aktiv, Docker Engine und Compose-Plug-in ab 2.20 aus dem Docker-Repository installiert
☐	Docker-Netzwerk proxy angelegt
☐	Nginx Proxy Manager gestartet, Standard-Zugangsdaten geändert
☐	.env aus .envdefault erstellt und vollständig ausgefüllt
☐	DB_PASSWORD gesetzt
☐	WOLFISMS_CONTAINER=1 gesetzt
☐	PROXY_ALIAS gesetzt und im Proxy Host als Forward Hostname eingetragen
☐	WEBPATH auf die öffentliche Adresse gesetzt (mit https://, ohne Schrägstrich am Ende)
☐	Volumes für Datenbank und UPLOAD_DIR eingebunden
☐	Bei externer Datenbank: DB_MODUS=extern, DB_HOST/DB_PORT gesetzt, Datenbank und Benutzer angelegt, Protokoll ohne Verbindungsfehler
☐	Proxy Host angelegt, Zertifikat angefordert, Force SSL und HTTP/2 aktiviert
☐	Anmeldung über HTTPS erfolgreich getestet
☐	Standardpasswort des Admin-Benutzers geändert
☐	Lizenzdatei eingespielt und Container neu gestartet
☐	Zweiter Admin-Account als Notfallzugang angelegt
☐	Backup über /admin/backup getestet und extern abgelegt
☐	ISMS_DEBUG nicht gesetzt
☐	.env nicht in ein Git-Repository eingecheckt

© 2012–2026 WolfInfoSec GmbH

12. Weiterführende Dokumentation

Dokument	Inhalt
WolfISMS Admindokumentation	Vollständige Systemadministration: Benutzer- und Rechteverwaltung, Datensicherung, Lizenzverwaltung, REST-API, Zwei-Faktor-Authentifizierung, Single Sign-On, KI-Einstellungen und Sicherheitshinweise. Dort steht auch, was im Containerbetrieb anders läuft als bei der klassischen Installation auf Apache mit mod_wsgi
WolfISMS Benutzerdokumentation	Beschreibung aller Fachmodule für Anwender
WolfISMS REST-API	Schnittstellenbeschreibung für die Anbindung externer Systeme: Incidents, Maßnahmen, Assets, Risiken und Lieferanten – mit allen Feldern, Filtern, Wertelisten und einem vollständigen Beispiel-Sync-Client

WolfISMS – Wolf's Informations-Sicherheits-Management-System
© 2012–2026 WolfInfoSec GmbH
Update-Server: https://update.wolfisms.de
Version: 4.00 | Stand: November 2026