Aegilock – Intelligenter Bot-Schutz für Ihre Website

Willkommen bei **Aegilock**, Ihrer Lösung für automatisierten Schutz gegen Bots, Scraper und unerwünschten Traffic. Diese Anleitung hilft Ihnen, Aegilock in wenigen Minuten auf Ihrem Server zu installieren.

Es gibt 2 Optionen Aegilock zu installieren:

Variante 1: Docker

Variante 2: Aegilock ohne Docker installieren und betreiben

Variante 1: Docker Installation

Voraussetzungen

- Ein Server mit **Docker** und **Docker Compose** installiert
- Grundkenntnisse im Umgang mit der Kommandozeile (SSH, Terminal)

Schnellstart

1. Repository herunterladen

git clone https://github.com/IHR-BENUTZER/aegilock-docker.git

cd aegilock-docker

2. Umgebungsdatei erstellen

cp .env.example .env

Sie können die .env-Datei anpassen, um eigene Einstellungen vorzunehmen (Port, API-Key etc.).

3. Aegilock starten

docker compose up -d

Aegilock läuft jetzt im Hintergrund.

Zugriff

Wenn alles läuft, erreichen Sie Aegilock unter:

http://IHRE-DOMAIN.de:8080

Oder lokal auf:

http://localhost:8080

Konfiguration

Die Konfiguration befindet sich im Ordner:

aegilock/config/

Dort finden Sie zum Beispiel:

- blocklist.json – blockierte IPs, User-Agents, Länder etc.
- config.json – weitere Einstellungen für das Verhalten
- ml-service – verhaltensbasierte, selbstlernende Bot-Erkennung

Nach Änderungen bitte den Container neu starten:

docker compose restart

Sicherheitshinweise

- Achten Sie darauf, dass die .env-Datei nicht öffentlich zugänglich ist.
- Verwenden Sie einen Reverse-Proxy (z. B. nginx oder Caddy) für HTTPS-Zugriff.
- Führen Sie regelmäßige Updates durch, um Sicherheitslücken zu vermeiden.

Variante 2: Aegilock ohne Docker installieren und betreiben

Wenn Sie Aegilock auf einem Server ohne Docker einsetzen möchten, gehen Sie bitte folgendermaßen vor:

Voraussetzungen:

Ein Server (Linux) mit Root- oder sudo-Rechten
Node.js (Version 16 oder höher)
npm (Node Package Manager)
Eine laufende Redis-Instanz (lokal oder extern)
Optional: Ein Reverse-Proxy (zum Beispiel nginx oder Caddy) für HTTPS

Schritt 1: Repository klonen

1. Melden Sie sich per SSH auf Ihrem Server an.
2. Wechseln Sie in das Verzeichnis, in dem Sie Aegilock installieren möchten, zum Beispiel /opt oder /srv:
   cd /opt
3. Klonen Sie das GitHub-Repository:
   git clone:
4. Wechseln Sie in das neue Verzeichnis:
   cd aegilock

Schritt 2: Node.js und Abhängigkeiten installieren

1. Prüfen Sie, ob Node.js und npm installiert sind:
   node --version
   npm --version
2. Wenn Node.js nicht installiert ist, folgen Sie den Anleitungen Ihrer Linux-Distribution, zum Beispiel für Debian/Ubuntu:
   curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
   sudo apt-get install -y nodejs
3. Installieren Sie alle benötigten Pakete:
   npm install

Schritt 3: Redis einrichten

Option A: Lokale Redis-Installation (Debian/Ubuntu)

1. Redis-Paket installieren:
   sudo apt-get update
   sudo apt-get install -y redis-server
2. Redis so konfigurieren, dass es als Dienst startet (Standard unter Debian/Ubuntu):
   sudo systemctl enable redis-server
   sudo systemctl start redis-server
3. Prüfen, ob Redis läuft:
   redis-cli ping
   ↳ Antwort sollte „PONG“ sein

Option B: Externe Redis-Instanz
Wenn Sie bereits eine verwaltete Redis-Instanz (Cloud) haben, notieren Sie sich Host und Port sowie ein eventuelles Passwort.

Schritt 4: Umgebungsvariablen (Konfiguration)
Erstellen Sie im Aegilock-Verzeichnis eine Datei .env und fügen Sie dort mindestens folgende Werte ein:

makefile

KopierenBearbeiten

PORT=3000

NODE_ENV=production

REDIS_HOST=127.0.0.1

REDIS_PORT=6379

REDIS_PASSWORD= # nur, wenn Ihre Redis-Instanz ein Passwort erfordert

BLOCKLIST_FILE=./config/blocklist.json

API_KEY=IhrGeheimerAPIKey

- PORT: Port, auf dem Aegilock später lauschen soll (Standard: 3000).
- REDIS_HOST, REDIS_PORT, REDIS_PASSWORD: Verbindung zu Redis.
- BLOCKLIST_FILE: Pfad zur Blocklist-Datei im Verzeichnis config.

- API_KEY: Geheimer Schlüssel, der nur intern verwendet wird (z. B. im ML-Service).

Schritt 5: Konfigurationsdateien anpassen
Im Unterordner config/ finden Sie folgende Dateien:

- blocklist.json: Beispiel für IP-Blocklisten, User-Agent-Filter und Länder-Sperren. Passen Sie hier an, was Sie blockieren oder erlauben möchten.
- config.json: Weitere Einstellungen, etwa Zeitlimits, Rate-Limits oder GeoIP-Optionen. Lesen Sie die Kommentare in der Datei und passen Sie an, wenn nötig.
- ml-service/: (Optional) Hier liegen Modell-Dateien und Settings für die verhaltensbasierte Bot-Erkennung. Wenn Sie das ML-Feature nicht verwenden, können Sie diesen Ordner ignorieren.

Schritt 6: Aegilock erstmals starten

1. Wechseln Sie in das Hauptverzeichnis von Aegilock (dort, wo server.js liegt).
2. Starten Sie die App im Produktionsmodus:
   node server.js
3. Die Konsole sollte anzeigen, dass Aegilock auf Port 3000 (oder dem Wert in .env) gestartet wurde und sich mit Redis verbinden konnte.
4. Öffnen Sie im Browser http://IHRE-SERVER-IP:3000 oder http://IHRE-DOMAIN:3000, um zu prüfen, ob Aegilock erreichbar ist.

Schritt 7: Prozessverwaltung einrichten
Damit Aegilock auch nach einem Server-Neustart oder bei Absturz automatisch wieder startet, verwenden Sie einen Process Manager wie pm2 oder richten einen Systemdienst ein.

Variante A: pm2 (empfohlen für einfache Einrichtung)

1. Installieren Sie pm2 global:
   sudo npm install -g pm2
2. Starten Sie Aegilock unter pm2:
   pm2 start server.js --name aegilock
3. Speichern Sie die Prozessliste, damit pm2 nach einem Neustart wieder alles startet:
   pm2 save

4. Richten Sie den pm2-Startupservice ein (je nach Distribution kann das variieren, hier ein Beispiel für Ubuntu):
   pm2 startup systemd
   (pm2 zeigt anschließend den genauen Befehl an, den Sie ausführen müssen.)

Variante B: Systemd-Dienst (für fortgeschrittene Admins)

1. Erstellen Sie eine Datei /etc/systemd/system/aegilock.service mit folgendem Inhalt (passen Sie Pfade an):

ini

KopierenBearbeiten

[Unit]

Description=Aegilock Bot-Schutz

After=network.target

[Service]

Type=simple

User=www-data

WorkingDirectory=/opt/aegilock

ExecStart=/usr/bin/node /opt/aegilock/server.js

Restart=on-failure

EnvironmentFile=/opt/aegilock/.env

[Install]

WantedBy=multi-user.target

2. Aegilock als Dienst registrieren:
   sudo systemctl daemon-reload
   sudo systemctl enable aegilock
   sudo systemctl start aegilock
3. Prüfen Sie den Status:
   sudo systemctl status aegilock

Schritt 8: Optionaler Reverse-Proxy und HTTPS einrichten
Damit Aegilock unter Ihrer Domain per HTTPS erreichbar ist, richten Sie einen Reverse-Proxy ein (Beispiel nginx):

1. Installieren Sie nginx (Debian/Ubuntu):
   sudo apt-get install -y nginx
2. Erstellen Sie unter /etc/nginx/sites-available/aegilock.conf folgende Konfiguration (passen Sie IHRE-DOMAIN.de an):

nginx

KopierenBearbeiten

```
server {
  listen 80;
  server_name IHRE-DOMAIN.de;
  return 301 https://$host$request_uri;
}

server {
  listen 443 ssl;
  server_name IHRE-DOMAIN.de;

  ssl_certificate     /etc/letsencrypt/live/IHRE-DOMAIN.de/fullchain.pem;
  ssl_certificate_key /etc/letsencrypt/live/IHRE-DOMAIN.de/privkey.pem;

  location / {
    proxy_pass http://127.0.0.1:3000;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
  }
}
```

3. Stellen Sie sicher, dass Let's Encrypt-Zertifikate vorhanden sind (zum Beispiel mit Certbot):
   sudo apt-get install -y certbot python3-certbot-nginx
   sudo certbot --nginx -d IHRE-DOMAIN.de
4. Aktivieren Sie die neue Site und testen Sie nginx:
   sudo ln -s /etc/nginx/sites-available/aegilock.conf /etc/nginx/sites-enabled/
   sudo nginx -t
   sudo systemctl reload nginx

Nun leitet nginx alle Anfragen an Aegilock weiter und stellt HTTPS bereit.

Schritt 9: Überwachung und Logs

- Logdateien finden Sie standardmäßig in der Konsole oder in /var/log/syslog, wenn Sie Systemd nutzen.
- Für detailliertere Logs können Sie zusätzlich einen Dienst wie Graylog, ELK-Stack (Elasticsearch, Logstash, Kibana) oder lokales File-Logging per Winston/Log4js konfigurieren.
- Beobachten Sie die Redis-Verbindung, API-Fehler und Performance-Metriken.

Schritt 10: Updates und Wartung

1. Um Aegilock auf eine neue Version zu aktualisieren, stoppen Sie zuerst den Dienst (pm2 oder Systemd):
   pm2 stop aegilock (oder) sudo systemctl stop aegilock
2. Ziehen Sie die neuesten Änderungen aus Git:
   cd /opt/aegilock
   git pull origin main
3. Installieren Sie ggf. neue Abhängigkeiten:
   npm install
4. Starten Sie den Dienst wieder:
   pm2 start aegilock (oder) sudo systemctl start aegilock

Fertig. Aegilock läuft jetzt ohne Docker direkt auf Ihrem System. Kontrollieren Sie regelmäßig Logs und Versionen, um Sicherheit und Stabilität zu gewährleisten.

Support & Kontakt

Bei Fragen oder Problemen erreichen Sie uns unter:

kontakt@aegilock.de
https://aegilock.de