Authentik

Identity Provider self hosted on Internal Services VM

- Authentik Docker Compose Install
- Authentik Passwordless Login
- Authentik OAuth/ OIDC Setup - Portainer
- Authentik OAuth/ OIDC Setup - Home Assistnat

Authentik Docker Compose Install

Authentik is an open-source Identity Provider (IdP) that helps you manage authentication and authorization across your apps and infrastructure. It supports:

- Single Sign-On (SSO) via OAuth2, OpenID Connect, SAML
- LDAP & SCIM integration
- Multi-factor authentication
- Reverse proxy for seamless app protection

Think of it as your self-hosted alternative to services like Okta or Auth0, but with full control and flexibility.

Prerequisites:

- Docker & Docker Compose

[Authentik Docker Compose Installation Guide]

Install Steps:

1. Open SSH and get to the device you want to run it on. (my case Overseer)
2. grab preconfigured yml

```
wget https://goauthentik.io/docker-compose.yml
```

If this is a fresh authentik installation, you need to generate a password and a secret key.

3. Run the following commands to generate a password and secret key and write them to your `.env` file:

```
echo "PG_PASS=$(openssl rand -base64 36 | tr -d '\n')" >> .env
echo "AUTHENTIK_SECRET_KEY=$(openssl rand -base64 60 | tr -d '\n')" >> .env
```

4. To enable error reporting, run the following command:

```
echo "AUTHENTIK_ERROR_REPORTING__ENABLED=true" >> .env
```

5. By default, authentik listens internally on port 9000 for HTTP and 9443 for HTTPS.

```
cd /docker/authentik/.env
```

6. To change the exposed ports to 80 and 443, you can set the following variables in `.env`:

```
COMPOSE_PORT_HTTP=80
COMPOSE_PORT_HTTPS=443
```

7. Startup docker compose

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

To start the initial setup, navigate to **http://<your server's IP or hostname>:9000/if/flow/initial-setup/**

Alternative Install Steps:

1. Open SSH and get to the device you want to run it on. (my case Overseer)
2. Create Directory

```
mkdir /docker/authentik
cd /docker/authentik
```

3. Create docker-compose.yml and edit it

```
nano docker-compose.yml #might need to use sudo if it doesn't give you access
```

```
version: '3.8'

services:
  postgresql:
    image: postgres:15
    environment:
      POSTGRES_DB: authentik
```

```
    POSTGRES_USER: authentik
    POSTGRES_PASSWORD: authentik
  volumes:
    - postgresql_data:/var/lib/postgresql/data

redis:
  image: redis:7
  volumes:
    - redis_data:/data

server:
  image: ghcr.io/goauthentik/server:latest
  depends_on:
    - postgresql
    - redis
  environment:
    AUTHENTIK_SECRET_KEY: "supersecretkey"
    AUTHENTIK_POSTGRESQL__HOST: postgresql
    AUTHENTIK_POSTGRESQL__USER: authentik
    AUTHENTIK_POSTGRESQL__PASSWORD: authentik
    AUTHENTIK_POSTGRESQL__NAME: authentik
    AUTHENTIK_REDIS__HOST: redis
  ports:
    - "8080:8000"  # Web UI
    - "9444:9443"  # Proxy port
  volumes:
    - authentik_media:/media
    - authentik_static:/static

worker:
  image: ghcr.io/goauthentik/worker:latest
  depends_on:
    - server
  environment:
    AUTHENTIK_SECRET_KEY: "supersecretkey"
    AUTHENTIK_POSTGRESQL__HOST: postgresql
    AUTHENTIK_POSTGRESQL__USER: authentik
    AUTHENTIK_POSTGRESQL__PASSWORD: authentik
    AUTHENTIK_POSTGRESQL__NAME: authentik
    AUTHENTIK_REDIS__HOST: redis
```

```
    volumes:
      - authentik_media:/media
      - /var/run/docker.sock:/var/run/docker.sock

volumes:
  postgresql_data:
  redis_data:
  authentik_media:
  authentik_static:
```

4. Create the .env file

```
nano .env #might need to run it with sudo
```

```
# Database credentials
PG_USER=authentik
PG_PASS=supersecurepassword123
PG_DB=authentik

# Authentik image tag
AUTHENTIK_IMAGE=ghcr.io/goauthentik/server
AUTHENTIK_TAG=2025.6

# Optional: HTTP/HTTPS ports (not forwarded externally)
COMPOSE_PORT_HTTP=9000
COMPOSE_PORT_HTTPS=9444

# Secret Key
AUTHENTIK_SECRET_KEY=your-super-secret-key
```

5. Start the stack

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

Once the stack is up, everything is finished installing you can check it with

```
docker-compose ps
```

To start the initial setup, navigate to http://<your server's IP or hostname>:9000/if/flow/initial-setup/.

Authentik Passwordless Login

Passwordless Login in Authentik allows us to login using passkey instead of password. This option provides higher security and faster authentication.

At the moment Passwordless Authentication only supposts WebAuth devicees (tokens, yubkey, 1password passkey).

Authentik Documentation on Passwordless Login

Steps to Set Up Passwordless Login Flow

1. Login to Authentik as Administrator
2. Click on F**lows and Stages** and click on **Flows**
3. Click **Create**
    1. Keep the name similar across the process for easier setup
    2. For Designation choose Authentication
4. Click on new created Flow
5. Click on **Stage Bindings** and choose **Create & Bind Stage**
    1. Choose Authenticator Validation Stage
    2. Click Next and add name similar to previous one
    3. Choose WebAuthn Authentication
    4. For not configured action choose **Force the user to configure an authenticator**
    5. For configuration stage find **default-autenticator-webauth-setup** and push over to the right
    6. Click Next and Finish
6. Click **Bind existing Stage**
    1. For Stage select **default-authentication-login** (or personal one)
    2. If you add Order number for previous part, add a higher number
    3. Click **Create**
7. Go back to Flows and select your **Welcome Page** or **defaulth-authentication-flow**
8. Go to **Stage Bindings** and for **Identification Stage** click **Edit Stage**
    1. Go to flow Settings
    2. Select **passwordless flow**

You should be ready to go

Authentik OAuth/ OIDC Setup - Portainer

Authentik uses many ways to connect to services, one being OAuth or Open ID Connect. This method is widely used on many services, such as Portainer.

Please follow Authentik and Portainer documentation

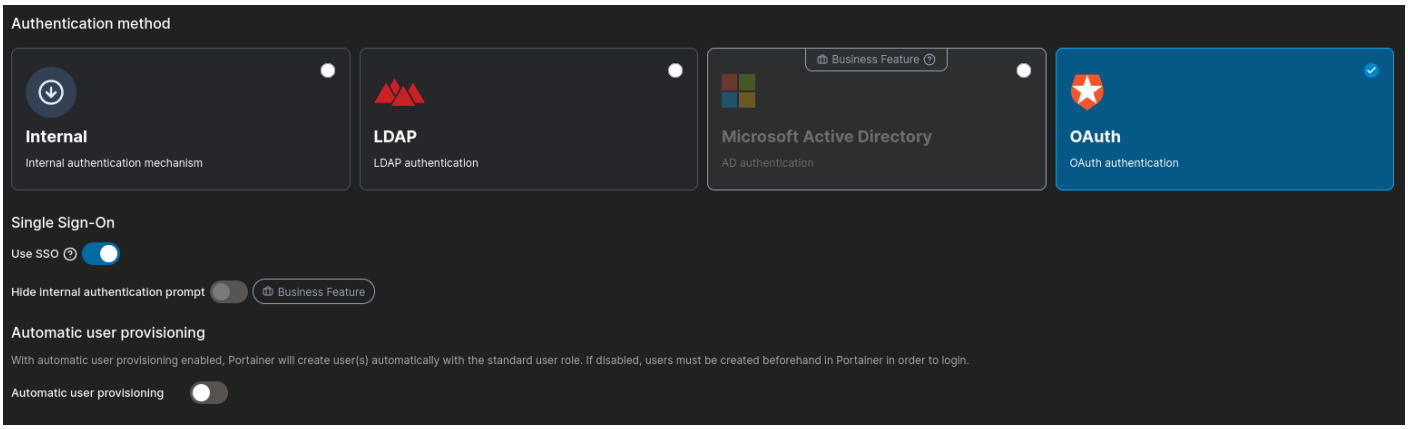
- Portainer OAuth Setup Documentation
- Authentik Portainer Integration Documentation

Authentik OAuth:

1. Login to Authentik **Admin Interface**
2. Go to **Applications** and select **Create with Provider**
    1. Choose a name and group
    2. Under **URI** in **Launch URL** enter **https://portainer.cyberpaw.org**
    3. Choose **Oauth2** Provider
    4. Name the provider same as application
    5. For Authorization Flow choose **Cyberpaw-authorization-flow** (or default one)
    6. Make sure **Confidential** is selected for Client Type
    7. Copy Client ID and Client Key
    8. In Redirect URIs enter **https://portainer.cyberpaw.org** (check portainer instructions for more detail)
    9. For Encryption key choose **default-authentic-self-signed-certificate**
    10. Under **Advanced flow** settings choose **Welcome to Authentick** (or default one)
    11. Under **Configure Bindings** click **Bind existing policy/group/users**
    12. Select **Group** and choose existing group that is authorized to use this service
    13. Review and Submit
3. The provider is created and should say it's connected to application

Portainer Steps:

1. Navigate to Portainer page and login
2. Under **Settings** go to **Authentication** and select **OAuth**
3. Enable **use SSO**



4. Choose **Automatic User Provisioning** allowing other Authentik users that don't have Portainer user can login
    1. If not selected you will need to create an account with same email as Authentik user
5. Scroll down to **OAuth Configuration**
    1. Copy and Paste all the field ID, secret and URLs from Provider information in Authentik
        1. Go back to Authentik **Admin Interface**
        2. Lower **Application** Section and click **Providers**
        3. Click on **Portainer** Provider and copy all the required information to Portainers OAuth Configuration
    2. For User Identification type "**email**"
    3. For Scope type "**email oauth provider**" **-**Portainer documentation says to use dashes but use space instead
    4. Save
6. Logout and you should see **Login with OAuth** button

Authentik OAuth/ OIDC Setup - Home Assistnat

Authentik uses many ways to connect to services, one being OAuth or Open ID Connect. This method is widely used on many services, such as Home Assistant. Home Assistant doesn't have native Open ID Connection, so we will need to use HACS for setup

Please follow Authentik and Portainer documentation

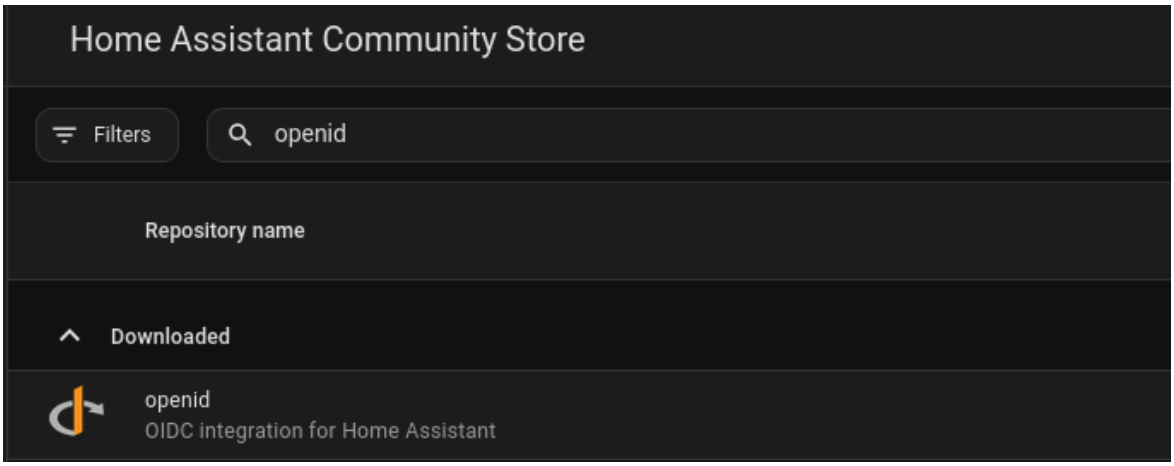
- Home Assistant OAuth Setup Documentation
- GitHub Hass-openid Setup with HACS

Authentik OAuth:

1. Login to Authentik **Admin Interface**
2. Go to **Applications** and select **Create with Provider**
    1. Choose a name and group
    2. Under **URI** in **Launch URL** enter **https://portainer.cyberpaw.org**
    3. Choose **Oauth2** Provider
    4. Name the provider same as application
    5. For Authorization Flow choose **Cyberpaw-authorization-flow** (or default one)
    6. Make sure **Confidential** is selected for Client Type
    7. Copy Client ID and Client Key
    8. In Redirect URIs enter http://overseer.cyberpaw.org:8123/auth/openid/callback
    9. For Encryption key choose **default-authentic-self-signed-certificate**
    10. Under **Advanced flow** settings choose **Welcome to Authentick** (or default one)
    11. Under **Configure Bindings** click **Bind existing policy/group/users**
    12. Select **Group** and choose existing group that is authorized to use this service
    13. Review and Submit
3. The provider is created and should say it's connected to application

Home Assistant Steps:

1. Login to Home Assistant with Admin
2. Open **HACS**
3. Search for hass-openid



4. Go to **Terminal** app on HA
5. Navigate to Your Home Assistant Config Directory

```
cd /config
```

6. Create custom_components Directory

```
mkdir -p /config/custom_components/openid
```

7. Download the Files from GitHub

```
git clone https://github.com/cavefire/hass-openid.git
cp -r hass-openid/custom_components/openid /config/custom_components/
```

8. Restart Home Assistant
9. Go back to Terminal and add following configuration to configuration.yaml file

```
  #OAuth with Authentik
openid:
  client_id: YOUR_CLIENT_ID
  client_secret: YOUR_CLIENT_SECRET
  configure_url: "https://auth.cyberpaw.org/application/o/home-assistant/.well-known/openid-
configuration"  # Replace with your Identity Provider's URL
  username_field: "email"  # Adjust based on your IdP's user info response
  scope: "openid profile email"
  block_login: false
  openid_text: "Login with Authentik"  # Text to display on the login page
```

10. Restart Home Assistant

If you want to disable the default Home Assistant login and only allow OpenID authentication, set `block_login` to `true` in your configuration