Authentik

Authentik: The Versatile Open-Source Identity Provider

In today’s digital landscape, managing identity and access is a critical component of ensuring the security and efficiency of online services. Authentik, an open-source identity provider, offers a powerful and flexible solution to meet these needs. This article provides an in-depth look at Authentik, its features, and detailed instructions on how to install and set it up using Docker Compose.

What is Authentik?

Authentik is an open-source identity provider that focuses on providing flexible and scalable authentication and authorization solutions. It supports a wide range of authentication protocols and integrates seamlessly with various applications and services. Authentik aims to simplify identity management while ensuring robust security.

Key Features of Authentik

1. **Flexible Authentication**: Supports multiple authentication methods, including LDAP, OAuth2, SAML, and more.
2. **User Management**: Comprehensive user management features, including user self-service and admin-controlled user provisioning.
3. **Authorization**: Fine-grained access control policies to ensure users have the right level of access.
4. **Integrations**: Integrates with numerous third-party applications and services, making it a versatile solution for diverse environments.
5. **Custom Workflows**: Allows for the creation of custom authentication and authorization workflows tailored to specific requirements.
6. **Open Source**: As an open-source project, Authentik benefits from community contributions and transparency in development.

Installing Authentik Using Docker Compose

Setting up Authentik with Docker Compose is straightforward and ensures a consistent environment across different deployments. Follow these steps to get started.

Prerequisites

Ensure you have Docker and Docker Compose installed on your system. You can download and install them from the Docker website.

Docker Compose Configuration

1. **Create a Docker Compose file**: Create a file named `docker-compose.yml` in your desired directory.
2. **Add the following content to the file**:

```
---

services:
  postgresql:
    image: docker.io/library/postgres:16-alpine
    restart: unless-stopped
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -d $${POSTGRES_DB} -U $${POSTGRES_USER}"]
      start_period: 20s
      interval: 30s
      retries: 5
      timeout: 5s
    volumes:
      - ${DOCKER}/authentik/database:/var/lib/postgresql/data
    environment:
      POSTGRES_PASSWORD: ${PG_PASS:?database password required}
      POSTGRES_USER: ${PG_USER:-authentik}
      POSTGRES_DB: ${PG_DB:-authentik}
    env_file:
      - .env
  redis:
    image: docker.io/library/redis:alpine
    command: --save 60 1 --loglevel warning
    restart: unless-stopped
    healthcheck:
      test: ["CMD-SHELL", "redis-cli ping | grep PONG"]
      start_period: 20s
      interval: 30s
      retries: 5
      timeout: 3s
```

```
    volumes:
      - ${DOCKER}/authentik/redis:/data
  server:
    image: ${AUTHENTIK_IMAGE:-ghcr.io/goauthentik/server}:${AUTHENTIK_TAG:-2024.4.2}
    restart: unless-stopped
    command: server
    environment:
      AUTHENTIK_REDIS__HOST: redis
      AUTHENTIK_POSTGRESQL__HOST: postgresql
      AUTHENTIK_POSTGRESQL__USER: ${PG_USER:-authentik}
      AUTHENTIK_POSTGRESQL__NAME: ${PG_DB:-authentik}
      AUTHENTIK_POSTGRESQL__PASSWORD: ${PG_PASS}
    volumes:
      - ${DOCKER}/authentik/media:/media
      - ${DOCKER}/authentik/custom-templates:/templates
    env_file:
      - .env
    ports:
      - "${HTTP_PORT}:9000"
      - "${HTTPS_PORT}:9443"
    depends_on:
      - postgresql
      - redis
  worker:
    image: ${AUTHENTIK_IMAGE:-ghcr.io/goauthentik/server}:${AUTHENTIK_TAG:-2024.4.2}
    restart: unless-stopped
    command: worker
    environment:
      AUTHENTIK_REDIS__HOST: redis
      AUTHENTIK_POSTGRESQL__HOST: postgresql
      AUTHENTIK_POSTGRESQL__USER: ${PG_USER:-authentik}
      AUTHENTIK_POSTGRESQL__NAME: ${PG_DB:-authentik}
      AUTHENTIK_POSTGRESQL__PASSWORD: ${PG_PASS}
    # `user: root` and the docker socket volume are optional.
    # See more for the docker socket integration here:
    # https://goauthentik.io/docs/outposts/integrations/docker
    # Removing `user: root` also prevents the worker from fixing the permissions
    # on the mounted folders, so when removing this make sure the folders have the
correct UID/GID
```

```
    # (1000:1000 by default)
    user: root
    volumes:
      - ${DOCKER}/authentik/var/run/docker.sock:/var/run/docker.sock
      - ${DOCKER}/authentik/media:/media
      - ${DOCKER}/authentik/certs:/certs
      - ${DOCKER}/authentik/custom-templates:/templates
    env_file:
      - .env
    depends_on:
      - postgresql
      - redis
```

3. **Adjust the environment variables**: Ensure you replace placeholder values like `your-secret-key` with appropriate values for your setup.

Running Authentik

1. **Navigate to the directory containing your `docker-compose.yml` file**.

Run the following command to start Authentik:

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

This command will pull the necessary Docker images and start the containers in detached mode. Authentik will be accessible at `http://localhost:9000`.

Basic Setup Instructions

Once Authentik is up and running, follow these steps to complete the basic setup:

1. **Access the Authentik Web Interface**: Open a web browser and navigate to `http://localhost:9000/if/flow/initial-setup/`.
2. **Initial Setup Wizard**: Authentik will present an initial setup wizard. Follow the prompts to configure the initial admin user and basic settings.
3. **Configure Authentication Sources**: Navigate to the "Sources" section in the admin interface to add authentication sources such as LDAP, OAuth2, or SAML providers.
4. **Create Applications**: In the "Applications" section, add the applications you want to protect with Authentik. Configure the necessary authentication flows and policies.
5. **User Management**: Use the "Users" section to manage user accounts, groups, and permissions.
6. **Policies and Flows**: Define custom policies and authentication flows to meet your specific access control requirements.

Helpful Resources

- Authentik Documentation: Comprehensive guides and references.
- Authentik GitHub Repository: Source code and community contributions.

Conclusion

Authentik provides a powerful and flexible solution for managing authentication and authorization across a variety of environments. Its open-source nature, combined with robust features and ease of deployment using Docker Compose, makes it an excellent choice for organizations of all sizes. By following the setup instructions provided, you can quickly get started with Authentik and leverage its capabilities to enhance your identity management strategy.

For more information and advanced configurations, visit the Authentik homepage.

Revision #6
Created 2024-07-01 06:35:53 UTC by thesabear
Updated 2024-07-21 18:58:44 UTC by thesabear