Docker: Authentik

Authentik is a middleware expanding the basic authentication- features of FreeIPA by many additional, modern ways of authentication which is used by modern Software. It will make SSO possible (Single-Sign-On: only logging into your pc will be enough to open all apps) by using the Kerberos-Credentials to login to authentik and than to authenticate the third-aprty app like Nextcloud without any additional User or Password- Dialog.

Dockerfile

Usually, i link the env-File to some central File. This time, as there are Passwords included, i will setup an own file. File `.env`:

```
#Common Docker-Parameters
COMPOSE_PROJECT_NAME=servername
COMPOSE_HTTP_TIMEOUT=180

#Authentik: https://goauthentik.io/docs/installation/docker-compose
AUTHENTIK_TAG=latest
#use maybe: openssl rand -base64 60 | tr -d '\n'
AUTHENTIK_SECRET_KEY=SEEDOCS
AUTHENTIK_ERROR_REPORTING__ENABLED=true
#
#Postgres-DB Authentik
AUTHENTIK_POSTGRESQL__HOST=servername-authentik_pgsql-1
AUTHENTIK_POSTGRESQL__USER=authentik
AUTHENTIK_POSTGRESQL__NAME=authentik
AUTHENTIK_POSTGRESQL__PASSWORD=YOURPGPASS
#
#Redis - we are using nextcloud here
AUTHENTIK_REDIS__HOST=nextcloud-aio-redis
AUTHENTIK_REDIS__PASSWORD=SeeInNextcloudSettings_nextcloud_data_config/config.php
#
## SMTP Host Emails are sent to
#AUTHENTIK_EMAIL__HOST=mailserver.smtp.de
#AUTHENTIK_EMAIL__PORT=465
## Optionally authenticate (don't add quotation marks to your password)
```

```
#AUTHENTIK_EMAIL__USERNAME=mailuser@host.de
#AUTHENTIK_EMAIL__PASSWORD=MAILPWD
## Use StartTLS
#AUTHENTIK_EMAIL__USE_TLS=false
## Use SSL
#AUTHENTIK_EMAIL__USE_SSL=true
#AUTHENTIK_EMAIL__TIMEOUT=10
## Email address authentik will send from, should have a correct @domain
#AUTHENTIK_EMAIL__FROM=mailuser@host.de
```

The File `docker-compose.yml` contains:

```

# Source: https://docs.goauthentik.io/docs/releases/2024.10
# wget -O docker-compose.yml https://goauthentik.io/version/2024.10/docker-compose.yml

services:
  authentik_pgsql:
    image: docker.io/library/postgres:16-alpine
    restart: always
    healthcheck:
      test: ["CMD-SHELL", "pg_isready -d $${POSTGRES_DB} -U $${POSTGRES_USER}"]
      start_period: 20s
      interval: 30s
      retries: 5
      timeout: 5s
    volumes:
      - authentik_pgsql_data:/var/lib/postgresql/data
      - authentik_backup:/tmp/backup
#Defined in .env
#    environment:
#      POSTGRES_PASSWORD: ${PG_PASS:?database password required}
#      POSTGRES_USER: ${PG_USER:-authentik}
#      POSTGRES_DB: ${PG_DB:-authentik}
    env_file:
      - .env

# Using nextcloud-aio-redis
```

```
#  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:
#      - redis:/data

# Authentik Server
  authentik_server:
    image: ${AUTHENTIK_IMAGE:-ghcr.io/goauthentik/server}:${AUTHENTIK_TAG:-2025.2.1}
    restart: always
    command: server
# When Upgrading: Check for new Parameters and add to env, not here
# Possible Parameters: https://docs.goauthentik.io/docs/install-config/configuration/
    environment:
#      AUTHENTIK_REDIS__HOST: nextcloud-aio-redis # see .env for password
#      AUTHENTIK_POSTGRESQL__HOST: nextcloud-aio-database
#      AUTHENTIK_POSTGRESQL__USER: ${AUTHENTIK_PG_USER:-authentik}
#      AUTHENTIK_POSTGRESQL__NAME: ${AUTHENTIK_PG_DB:-authentik}
#      AUTHENTIK_POSTGRESQL__PASSWORD: ${AUTHENTIK_PG_PASS}
      KRB5_TRACE: /dev/stderr
    volumes:
      - authentik_media:/media
      - authentik_custom_templates:/templates
    env_file:
      - .env
# Caddy virtualised
#    ports:
#      - "${COMPOSE_PORT_HTTP:-9000}:9000"
#      - "${COMPOSE_PORT_HTTPS:-9443}:9443"
   networks:
     - nextcloud-aio
```

```
     - default

# Authentik Worker
  authentik_worker:
    image: ${AUTHENTIK_IMAGE:-ghcr.io/goauthentik/server}:${AUTHENTIK_TAG:-2024.10.0}
    restart: always
    command: worker
#Se above
#    environment:
#      AUTHENTIK_REDIS__HOST: nextcloud-aio-redis
#      AUTHENTIK_POSTGRESQL__HOST: nextcloud-aio-database
#      AUTHENTIK_POSTGRESQL__USER: ${AUTHENTIK_PG_USER:-authentik}
#      AUTHENTIK_POSTGRESQL__NAME: ${AUTHENTIK_PG_DB:-authentik}
#      AUTHENTIK_POSTGRESQL__PASSWORD: ${AUTHENTIK_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:
# No Docker integration / LDAP- Outpost not needed (will be freeipa)
#      - /var/run/docker.sock:/var/run/docker.sock
      - authentik_media:/media
      - authentik_certs:/certs
      - authentik_custom_templates:/templates
    depends_on:
      - authentik_server
    env_file:
      - .env
   networks:
     - nextcloud-aio
     - default

volumes:
  authentik_pgsql_data:
  authentik_backup:
```

```
  authentik_media:
  authentik_custom_templates:
  authentik_certs:
# redis:

networks:
 nextcloud-aio:
   external: true
# Still needs to be defined while without it won't enable ipv6
  default:
    driver: bridge
    enable_ipv6: true
```

Carefully look at each line to fit your needs.

Caddy

in docker Caddy- Service enhance the lines:

```
# Authentik
https://authentik.domain.tld:443 {
       header Strict-Transport-Security max-age=31536000;
       reverse_proxy servername-authentik_server-1:9000
}
```

First start

After doing `docker compose up -d` and restarting Caddy you should be able to Navigate to

https://authentik.domain.tld/if/flow/initial-setup/

and set the Password for the admin user `akadmin`.

Backup PostgresSQL Database

This is really VERY Important! The reason is, that every PostgresDB- Version has it own Database- File- Format and if you update Postgres from on Major Version to the next, you WILL NOT BE ABLE to start your DB with the old Volume- Data!

You NEED to have the DB-Backed up and restored to the next Version!

Create a File in your Docker- Dir named e.g. `docker_backup_authentik_db.sh:`

```
#!/bin/bash
# Make Postgres-Backup of Authentik
echo "Backup of authentik-postgres to servername-authentik_pgsql-1:/tmp/backup"
/home/docker/bin/docker exec -t -e PGPASSWORD=YOURPASSWORD servername-authentik_pgsql-1 pg_dump --compress=zstd:8 -f
/tmp/backup/pg_dump_authentik.sql -U authentik authentik
echo "Done Backup of Authentik-DB."
```

Now add this script to `backup/docker_backup_all.sh` .

Don't forget to extend your `backup/docker_backup_all.sh` by adding `authentik` to the service- list if you have not done so far.

If you Update the Major- Version, make sure to create a new Volume for your pgsql- data.

Configuring

Now that Authentik is working, we are glueing all services together.

Sync of FreeIPA/LDAP

Don't use Kerberos-Sync, because the kadmin-interface of FreeIPA is blocked to not have someone messing around with kerberos without FreeIPA not beeing informed. So use LDAP- Sync for the Users.

To Sync FreeIPA with Authentik, follow this Guide: https://docs.goauthentik.io/docs/users-sources/sources/directory-sync/freeipa/

When the Sync has been configured, all FreeIPA- Users should show up in Authentik.

After SVC- user is created, use the following commands to modify password reset as written in the doc:

```
ldapmodify -x -D "cn=Directory Manager" -H ldaps://ipa.domain.tld:636 -W
dn: cn=ipa_pwd_extop,cn=plugins,cn=config
changetype: modify
add: passSyncManagersDNs
passSyncManagersDNs: uid=svc_authentik,cn=users,cn=accounts,dc=domain,dc=tld
```

At the next line, hit CTRL+D and the modification should be set, check with:

```
ldapsearch -xv -Z -W -H ldap://ipa.domain.tld -b "cn=ipa_pwd_extop,cn=plugins,cn=config" -D "cn=Directory Manager"
```

which should show the entry for `passsyncmanagersdns` .

Secure LDAP- Users with TOTP

Now any User can login with its FreeIPA- Password, also if SPNEGO/ kerberos as beneath is not setup yet.

This is quite insecure, so you shoul add a second factor for that type of Login (for SPNEGO the second factor is your integrated Machine, which has the key stored already).

To do so, in the Autentik Admin- Panel go to Stages and edit the Stage „default-authentication-mfa-validation“
Change „Not configured action“ → Force…
At „Configuration stages“ → default.authenticator-totp-setup

The Next time you are logging in with User and Password in Authentik, it will ask to setup a TOTP- Device. You can for example use
https://f-droid.org/de/packages/org.liberty.android.freeotpplus/

Hint: There is also an default Flow for this to import in Authentik here
https://docs.goauthentik.io/docs/add-secure-apps/flows-stages/flow/examples/flows#two-factor-login

Current Bug

At the time of writing this, there was a bug here: https://github.com/goauthentik/authentik/issues/5972#issuecomment-2075631779

So if you have to enter the OTP twice, than go to Flows, click on `default-authentication-flow`, Tab Stage Bindings and delete the `default-authentication-mfa-validation` Stage (not the other MFA!)

Attaching SPNEGO

With SPNEGO, you gain access to SSO in Authentik.

Here is the link to the Docs: https://docs.goauthentik.io/docs/users-sources/sources/protocols/kerberos/

You need to logon to FreeIPA as Admin and do the following:

- Go to Hosts, add Host `authentik.domain.tld`
- Go to Services, Add an new HTTP- Service for that Host, called HTTP/ipa.domain.tld@DOMAIN.TLD
- Add the Ipa- User admin to the „Allowed to create Keytab“- Section

After that, you need to the Docker- Console into the running FreeIPA-Container and use the commands there:

```
#~> docker exec -it servername-ipa-1 bash
# Logon as Admin
kinit admin
# Create and read the Keytab for that service
ipa-getkeytab -s ipa.domain.tld -p HTTP/authentik.domain.tld -k /tmp/authentik.keytab
cat /tmp/authentik.keytab | base64
rm /tmp/authentik.keytab
```

This is the Keytab (a better „Password“) that you will use for SPNEGO in Authentik.

Use the Servicename HTTP@`authentik.domain.tld` as the Servername!

Important: Use „User matching mode“ = „Link to User with identical Username. …“ - otherwise Kerberos may fail!

Than activate Kerberos in Flows and Stages > Stages > default-authentication-identification > Source settings

Make sure, that your client is able to login with FreeIPA using SSSD/Kerberos as written in docker-freeipa - or at least, that krb5 client is installed and is able to login to your IPA- Instance with `kinit username` and that you have configured your Browser to work with the certificate. You can also check your Browser on the site ipa.damon.tld after using kinit - it should automatically log you into ipa without a new Password.