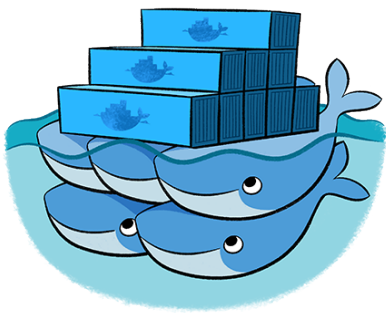


Add OIDC-Integration for Linkwarden



Overview

This page will show you how to add Linkwarden as a client (integration) for Authelia using OpenID Connect (OIDC). We'll walk through configuring the `clients` section in Authelia's `configuration.yml`, updating your access control rules, and finally setting up Linkwarden with the correct environment variables.

Step 1: Authelia Configuration for the New Client

In the OIDC setup guide, you've already configured the basic OIDC parameters (`hmac_secret`, `jwtks`, etc.). Now, we need to add the `clients` block to your OIDC configuration to allow Linkwarden to authenticate via Authelia.

Open your Authelia configuration file:

```
nano /mnt/glustermount/data/authelia_data/config/configuration.yml
```

Within the `identity_providers` > `oidc` section, add or edit the clients block as follows:

```
identity_providers:
  oidc:
    hmac_secret: 'this_is_a_secret_abc123abc123abc'
    jwtks:
      - key_id: 'F2H5xqbYsa3AssEZTU'
        algorithm: 'RS256'
        use: 'sig'
        key: '{{ secret "/secrets/rsa_2048_private.txt" | mindent 10 "|" | msquote }}'
    lifespans:
      access_token: '1 hour'
```

```
authorize_code: '1 minute'
id_token: '1 hour'
refresh_token: '90 minutes'
enable_client_debug_messages: false
clients:
  - client_id: 'linkwarden'
    client_name: 'Linkwarden'
    client_secret: '$pbkdf2-sha512$310000$c8p78n7pUMln0jzvd4aK4Q$JNRBzwAo0ek5qKn50cFzzvE9RXV88h1wJn5KGiHrD0YKtZaR/nCb2CJPOsKaPK0hjF.9yHxzQGZziziccp6Yng' #insecure_secret
    public: false
    authorization_policy: 'two_factor'
    redirect_uris:
      - 'https://linkwarden.YOURDOMAIN.com/api/v1/auth/callback/authelia'
    scopes:
      - 'openid'
      - 'groups'
      - 'email'
      - 'profile'
    userinfo_signed_response_alg: 'none'
    token_endpoint_auth_method: 'client_secret_basic'
```

Step 1.1: `client_id`

This references the unique identifier for the client (RFC: [RFC3986, Section 2.3](#)). In this example, it's set to `linkwarden`. You could instead use a random string to avoid collisions or for security best practices. Tools like [it-tools token generator](#) can be used to generate a random client ID.

Step 1.2: `client_name`

This is a human-readable name for your application. Here, we're calling it `Linkwarden`. You can name it anything that will help you identify this application in the future.

Step 1.3: `client_secret`

This is the secret used by the client to authenticate to Authelia (RFC: [RFC3986, Section 2.3](#)). In the example above, we're using a hashed secret labeled `insecure_secret` for testing.

You can generate a more secure secret by running:

```
docker exec -it authelia authelia crypto hash generate pbkdf2 --variant sha512 --random --random.length 72 --random.charset rfc3986
```

When using Docker-Swarm Change the Containername by adding your Stackname infront or using the ContainerID:

```
docker exec -it traefik_authelia authelia crypto hash generate pbkdf2 --variant sha512 --random --random.length 72 --random.charset rfc3986  
docker exec -it 337adb14377e authelia crypto hash generate pbkdf2 --variant sha512 --random --random.length 72 --random.charset rfc3986
```

Example output:

```
Random Password: Xh.nVAMt3P5m~fUTBj4issbKc38Xx5E47nUN7YvTzSntJv0DK2_EKdURzZFYhhs4LE4oKf~c  
Digest: $pbkdf2-  
sha512$310000$fK3IAD7WgJ147IBgnUdC9g$F1QMc0kpTwVIUNIdTaAGG8uD0EoQRxham7nN8HUXHVhNNUh2ubP  
u/wgo.YxXYC5ewNL.j3WPqnFLCB/mwfWSgA
```

You'd then place the `Digest` portion into the `client_secret` field in Authelia and use the corresponding "Random Password" value (or the "insecure_secret" equivalent) in the Linkwarden environment variable.

Step 1.4: `redirect_uris` and `authorization_policy`

Make sure the redirect URI matches `https://linkwarden.YOURDOMAIN.com/api/v1/auth/callback/authelia` (or whichever endpoint Linkwarden expects). Adjust your `authorization_policy` (such as `one_factor` or `two_factor`) depending on your security needs.

Step 2: Add Domain to Authelia

Access Control

Next, you'll want to allow access for `linkwarden.YOURDOMAIN.com` in your Authelia rules. Typically, this is done in the `access_control` section:

```
access_control:
  default_policy: 'deny'
  rules:
    - domain: 'whoami-secure.YOURDOMAIN.com'
      policy: 'two_factor'
    - domain: 'links.YOURDOMAIN.com'
      policy: 'two_factor'
```

Replace `links.YOURDOMAIN.com` or add another rule for the domain or subdomain where Linkwarden resides (`linkwarden.YOURDOMAIN.com`).

Step 3: Add OIDC Settings to

Linkwarden

If using a Docker Compose setup, you can add the following environment variables in your `docker-compose.yml` (or the equivalent setup in Portainer):

```
services:
  linkwarden:
    image: ghcr.io/linkwarden/linkwarden:v2.9.3
    environment:
      # SSO - Authelia
      - NEXT_PUBLIC_AUTHELIA_ENABLED=true
      - AUTHELIA_WELLKNOWN_URL=https://auth.YOURDOMAIN.com/.well-known/openid-configuration
      - AUTHELIA_CLIENT_ID=${AUTHELIA_CLIENT_ID}
      - AUTHELIA_CLIENT_SECRET=${AUTHELIA_CLIENT_SECRET}
      # SSO - Accounts
```

```
- DISABLE_NEW_SSO_USERS=false
```

Important: Make sure `DISABLE_NEW_SSO_USERS` is set to `false` or new users will be blocked from logging in via SSO.

Environment Variables Table

Environment Variable	Default	Description
NEXT_PUBLIC_AUTHELIA_ENABLED	-	If set to true, Authelia will be enabled and you'll need to define the variables below.
AUTHELIA_WELLKNOWN_URL	-	<code>https://{authelia.domain.com}/.well-known/openid-configuration</code>
AUTHELIA_CLIENT_ID	-	Client ID
AUTHELIA_CLIENT_SECRET	-	Client Secret. (Random Password from command below)

`AUTHELIA_WELLKNOWN_URL`: This is an OIDC discovery URL that describes what Authelia supports (endpoints, claims, etc.)

`AUTHELIA_CLIENT_ID` (RFC [RFC3986, Section 2.3](#)): Again, best practice is a random unique string.

`AUTHELIA_CLIENT_SECRET` (RFC [RFC3986, Section 2.3](#)): Generated secret value (either from the example above or your own generator).

Step 4: Restart Linkwarden

After updating the environment variables in your Docker Compose or Portainer configuration, restart Linkwarden to apply the changes:

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

Or the equivalent command depending on your environment. Once Linkwarden is back online, you can head to its login page and try logging in with Authelia.

Step 5: Troubleshooting

If something goes wrong:

- Check your Authelia logs to ensure the client is configured correctly.
 - Verify your redirect URI in both Authelia's `redirect_uris` and Linkwarden's OIDC settings.
 - Make sure your `authorization_policy` is not blocking you (e.g., requiring two-factor when you haven't set it up).
 - Confirm `DISABLE_NEW_SSO_USERS=false` if you want new accounts to be created in Linkwarden via SSO.
-

Conclusion

By adding Linkwarden as an OIDC client in Authelia and configuring the environment variables in Linkwarden, you can centralize authentication and enable secure, convenient login. Be sure to use secure secrets, update your access control lists carefully, and confirm everything is functioning by testing the login flow. Once everything is tested and stable, you're ready to enjoy single sign-on with Authelia for Linkwarden.

Revision #8

Created 1 February 2025 21:28:31 by aeoneros

Updated 4 February 2025 12:15:58 by aeoneros