Install Authelia for Authentication in front of Your Web Apps

https://www.youtube.com/embed/5KtbmrUwYNQ

Self hosting amazing open source software is the best feeling in the world. Knowing you're not tied to someone else's servers, whims, or quirks. The ability to control how you use your own apps, and how and when you add more is what open source is all about. But what about security?

I self host a lot of apps, and many of them don't have any built-in authentication. While that makes it quite convenient to use them, it also makes me a bit fearful of what others may be able to do with *my* applications and websites.

This is where Authelia comes in. Authelia provides a web application for authentication (make sure you are somone who should be using an application) and authorization (make sure you're permitted to use it) in front of your existing web applications.

As an example. I run the Homer Dashboard as a great way to link to all of my self hosted web applications. Homer itself does not (yet) have authentication. It just comes up, which is super useful and somewhat ideal, but I want to have access to my dashboard no matter where I am, so I've created a reverse proxy to it.

Yes, yes, there is always a VPN, or overlay network, or a dozen other ways to do it, but honestly, I just want to access my dashboard from anywhere, anytime. The reverse proxy is a great way to do that. To provide an authentication layer, though, makes me feel so much better about it. This is where something like Authelia comes into play. Now I can put Authelia in front of my Homer dash, and before the dash can be reached, a user must first authenticate.



Authelia Prompt in front of Homer Dash

This doesn't just have to be for web applications that don't have authentication built in. You could opt to use Authelia in front of an application with built in authentication as well. It just adds one more layer of security to your sites.

Today, we'll go through getting authelia setup and running.

What you'll need

- A server that you control
- Docker, Docker-Compose, and NGinX Proxy Manager Installed and Ready
- A Domain / Sub-domain you want to use for your Authelia install (e.g. auth.example.com)
- An A-record pointing to the Public IP address where your server is.
- About 1 hour of time.

Installing Docker, Docker-Compose, and NGinX Proxy Manager

NOTE: if you already have docker, docker-compose, and NGinX Proxy Manager installed, you can skip down to the section that says "Create your Authelia Entry in NGinX Proxy Manager".

In order to make this as painless as possible, I have built a script to install Docker and Docker-Compose for Ubuntu 18.04, 20.04, Debian, and CentOS. You can find these out on github at

https://github.com/bmcgonag/docker_installs

Additionally, I have modified the Ubuntu 20.04 script into a new script that will also install NGinX Proxy Manager, and get it running in Docker for you. You can get it direclty by going to

https://gitlab.com/bmcgonag/docker_installs/-/raw/main/install_docker_nproxyman.sh

To use the script above, just open a terminal, SSH to your server (if you aren't already on it), and create a new file called install-docker.sh

```
nano install-docker.sh
```

Copy and paste the script from the github site. CTRL + C after you highlight it. Then use CTRL + Shift + V to paste it into the terminal window in your file.

Save it with CTRL + O, then press Enter to confirm, and then exit the nano editor with CTRL + X.

Now, change the permissions on the script to make it executable with:

```
chmod +x install-docker.sh
```

Finally, you can run the script with:

```
./install-docker.sh
```

This will pull down Docker-CE (Community Edition), and install it, then install Docker-Compose, and finally pull down a default docker-compose.yml file to setup and make NGinX Proxy Manager run for you.

NOTE: The default values in the docker-compose file are straight from the NGinX Proxy Manager Quick Start, so I highly recommend, stopping the container, and changing the DB user and password values (making sure they match in both sections of the compose file), and then restarting it.

Login to NGinX Proxy Manager by going to http://your-ip-or-domain:81 and use the default credentials of:

- username: admin@example.com
- password: changeme

the first time you login, you'll be prompted to change the username email, and password to something stronger.

Create your Authelia Entry in NGinX Proxy Manager

Now, you'll need to create an entry for the authelia front end in NGinX Proxy Manager. Somthing like `auth.your-domain.org`, of course replacing `your-domain.org` with your actual domain. Go ahead nad make sure it opens properly without SSL, then setup the LetsEncrypt SSL on your new auth domain.

Now, test it again, and make sure it comes up with SSL. Once that's working we'll edit it one more time, and need to add some specific text / configuration to the Advanced tab of our auth entry. For the most part this will be copy paste, with one modification.

```
location / {
        set $upstream_authelia http://<authelia-server-ip-and:port>; # e.g.
http://192.168.1.13:9091
        proxy_pass $upstream_authelia;
        client_body_buffer_size 128k;

        #Timeout if the real server is dead
        proxy_next_upstream error timeout invalid_header http_500 http_502 http_503;

        # Advanced Proxy Config
        send_timeout 5m;
        proxy_read_timeout 360;
        proxy_send_timeout 360;
        proxy_connect_timeout 360;

        # Basic Proxy Config
```

```
        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;
        proxy_set_header X-Forwarded-Host $http_host;
        proxy_set_header X-Forwarded-Uri $request_uri;
        proxy_set_header X-Forwarded-Ssl on;
        proxy_redirect  http://  $scheme://;
        proxy_http_version 1.1;
        proxy_set_header Connection "";
        proxy_cache_bypass $cookie_session;
        proxy_no_cache $cookie_session;
        proxy_buffers 64 256k;

        # If behind reverse proxy, forwards the correct IP
        set_real_ip_from 10.0.0.0/8;
        set_real_ip_from 172.0.0.0/8;
        set_real_ip_from 192.168.0.0/16;
        set_real_ip_from fc00::/7;
        real_ip_header X-Forwarded-For;
        real_ip_recursive on;
    }
```

In the above text, make sure to change the part where it says `<authelia-server-ip-and:port>` to be your authelia server's IP and Port number, then Save.

Installing Authelia

For Authelia, you'll need 3 files:

1. docker-compose.yml
2. configuration.yml
3. users_database.yml

Docker-Compose for Authelia

In your server, create a new folder called "authelia", and move into that folder:

`mkdir authelia`

`cd authelia`

Now, create two more folders called "config" and "redis"

`mkdir config`

`mkdir redis`

Finally, create a new file called "docker-compose.yml":

`nano docker-compose.yml`

And paste the following text into it:

```
version: '3.3'
services:
  authelia:
    image: authelia/authelia
    volumes:
      - ./config:/config
    ports:
      - 9091:9091
    restart: unless-stopped
    healthcheck:
      disable: true
    environment:
      - TZ=America/Chicago
    depends_on:
      - redis

  redis:
    image: redis:alpine
    volumes:
      - ./redis:/data
    expose:
      - 6379
    restart: unless-stopped
    environment:
      - TZ=America/Chicago
```

Once pasted, make sure to change the left side of the port mapping **if necessary** from 9091 to any free port you have on your server. If 9091 is free, then leave it as is. NOTE: if you change the port here, you must go back and change it in your NGinX Proxy Manager advanced tab, as well as the main tab for your "auth.your-dormain.org" entry.

Next, change the timezone (TZ) environment variable to your timezone.

Now save the file with CTRL + O, then press Enter to confirm, and use CTRL + X to exit.

We can't run the compose file just yet, as we have a couple of other files to create first.

The Configuration File

Now we need to create our Authelia main configuration file. Change into the "config" directory:

`cd config`

and create a new file called "configuration.yml":

`nano configuration.yml`

In that file, paste the following: NOTE: This text has been updated since the original video and blog-post to reflect changes to the yaml for Authelia.

```
###############################################################
#                   Authelia configuration                    #
###############################################################

server:
  host: 0.0.0.0
  port: 9091

jwt_secret: a-super-long-strong-string-of-letters-numbers-characters
log:
  level: debug

default_redirection_url: https://auth.routemehome.org
totp:
  issuer: routemehome.org
  period: 30
  skew: 1

#duo_api:     ## You can use this api if you want push notifications of auth attempts
#  hostname: api-123456789.example.com
#  integration_key: ABCDEF
#  secret_key: yet-another-long-string-of-characters-and-numbers-and-symbols
```

```
authentication_backend:
  disable_reset_password: false
  file:
    path: /config/users_database.yml
    password:
      algorithm: argon2id
      iterations: 1
      salt_length: 16
      parallelism: 8
      memory: 64

access_control:
  default_policy: deny
  rules:
    # Rules applied to everyone
    - domain:
        - "noauth.domain.org"
        - "another-no-auth.domain.org"
      policy: bypass
    - domain:
        - "my1st1factor.domain.org"
        - "my2nd1factor.domain.org"
        - "domain.org"
      policy: one_factor
#      networks:
#        - 192.168.1.0/24
    - domain:
        - "a2factor.domain.org"
      policy: two_factor
#     networks:
#       - 192.168.1.0/24

session:
  name: authelia_session
  # This secret can also be set using the env variables AUTHELIA_SESSION_SECRET_FILE
  secret: a-really-L0ng_s7r0ng-secr3t-st1nggggg-shoul0-be-used
  expiration: 3600 # 1 hour
  inactivity: 7200 # 2 hours
  domain: <your.domain-here.org> # Should match whatever your root protected domain is
```

```
  redis:
    host: authelia_redis_1
    port: 6379
    # This secret can also be set using the env variables AUTHELIA_SESSION_REDIS_PASSWORD_FILE
#    password: authelia

regulation:
  max_retries: 5
  find_time: 2m
  ban_time: 10m

theme: dark

storage:
  encryption_key: a-very-long-strong-key-should-be-used-here
  local:
    path: /config/db.sqlite3

notifier:
#  filesystem:
#    filename: /config/notification.txt
  smtp:
    username: <your@email-here.com>
    password: <your-smtp-password>
    host: <your-smtp-server.url>
    port: 25,465,or 587
    sender: <sender@email-here.com>
    subject: "[Authelia] {title}"
    disable_require_tls: false
    disable_html_emails: false
    tls:
      server_name: <your-smtp-server.url>
      skip_verify: false
      minimum_version: TLS1.2
```

In the above file, make sure to change any field with "<" and ">" symbols around the text. Make sure you have updated all values to meet your needs. For more information on what each of these options are, watch my video, or check the Authelia documentation.

Once you have pasted, and updated the file above, save it with CTRL + O, then Enter to confirm, and exit with CTRL + X.

The Users Database file

Finally, we need to create a file for our users. This wile will be called "users_database.yml".

While still in the "config" directory, create the new file:

`nano users_database.yml`

then paste the following into the file, and modify according to your needs.

```
users:
  john:
    displayname: "John Doe"
    password:
"$argon2id$v=19$m=65536,t=3,p=2$BpLnfgDsc2WD8F2q$o/vzA4myCqZZ36bUGsDY//8mKUYNZZaR0t4MFFSs+iM"
    email: john.doe@authelia.com
    groups:
      - admins
      - dev
  harry:
    displayname: "Harry Potter"
    password:
"$argon2id$v=19$m=65536,t=3,p=2$BpLnfgDsc2WD8F2q$o/vzA4myCqZZ36bUGsDY//8mKUYNZZaR0t4MFFSs+iM"
    email: harry.potter@authelia.com
    groups: []
```

In the example file above, we have two users. John and Harry. If you only need 1 user, you would simply remove the section for Harry, and modify the section for John.

The password for the user you create must be entered as a hashed password, and not in plain text. In order to get the hashed password, you'll use the following command:

`docker run authelia/authelia:latest authelia hash-password 'yourpassword'`

Replace 'yourpassword' in the above command with the actual plain text password you want for your user.

Press enter, and allow the command to run. If you've not already pulled down the authelia/authelia image, it will pull down at this point from dockerhub, so be patient the first time you run this command.

Once complete, copy the hashed password from the command line by highlighting it, and using a right-click >> copy, or CTRL + Shift + C to copy it, and then re-open your users_database.yml file and paste the full hash between the double quotation marks.

Save the file again, and you're set.

Test Our Authelia Setup

We can finally test the authelia setup. We don't have any web applications protected by it just yet, but we can make sure that it's working, and that we can authenticate.

Move back out of the "config" directory one level with:

`cd ..`

and then run `docker-compose up`

This will pull down Redis, and startup Authelia and Redis. Watch the output for errors, but if everything is setup properly, you sould see a message that says Authelia is running at 0.0.0.0:9091.

Open your browser (or a new tab) and go to https://auth.<your-domain.org> <– of course using your actual domain.

You should be prompted with the login screen. You can now enter your username and password, and make sure you're able to login.

Setup our NGinX Proxy Manager sites for Authelia

For each site you put in the Authelia configuration file, you need to add a proxy host entry in NGinX Proxy Manager. Make sure you've setup each site in NPM, then on the Advanced tab for each one you'll need to add the following JSON. This long text only needs a few changes, and once you do it for the first site, copy and paste it from one to the next, and you'll only need to make 1 or 2 changes each time.

```
location /authelia {
    internal;
    set $upstream_authelia http://<your-authelia-server-ip-and:port>/api/verify;
#ADD YOUR IP AND PORT OF AUTHELIA
    proxy_pass_request_body off;
    proxy_pass $upstream_authelia;
    proxy_set_header Content-Length "";
```

```
    # Timeout if the real server is dead
    proxy_next_upstream error timeout invalid_header http_500 http_502 http_503;
    client_body_buffer_size 128k;
    proxy_set_header Host $host;
    proxy_set_header X-Original-URL $scheme://$http_host$request_uri;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $remote_addr;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-Forwarded-Host $http_host;
    proxy_set_header X-Forwarded-Uri $request_uri;
    proxy_set_header X-Forwarded-Ssl on;
    proxy_redirect  http://  $scheme://;
    proxy_http_version 1.1;
    proxy_set_header Connection "";
    proxy_cache_bypass $cookie_session;
    proxy_no_cache $cookie_session;
    proxy_buffers 4 32k;

    send_timeout 5m;
    proxy_read_timeout 240;
    proxy_send_timeout 240;
    proxy_connect_timeout 240;
}

    location / {
        set $upstream_<appname> http://<application-ip-and:port>;  #ADD IP AND PORT
OF SERVICE
        proxy_pass $upstream_<appname>;  #change name of the service

        auth_request /authelia;
        auth_request_set $target_url $scheme://$http_host$request_uri;
        auth_request_set $user $upstream_http_remote_user;
        auth_request_set $groups $upstream_http_remote_groups;
        proxy_set_header Remote-User $user;
        proxy_set_header Remote-Groups $groups;
        error_page 401 =302 https://auth.<example.com>/?rd=$target_url;

        client_body_buffer_size 128k;
```

```
        proxy_next_upstream error timeout invalid_header http_500 http_502
http_503;

        send_timeout 5m;
        proxy_read_timeout 360;
        proxy_send_timeout 360;
        proxy_connect_timeout 360;

        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;
        proxy_set_header X-Forwarded-Host $http_host;
        proxy_set_header X-Forwarded-Uri $request_uri;
        proxy_set_header X-Forwarded-Ssl on;
        proxy_redirect  http://  $scheme://;
        proxy_http_version 1.1;
        proxy_set_header Connection "";
        proxy_cache_bypass $cookie_session;
        proxy_no_cache $cookie_session;
        proxy_buffers 64 256k;

        # add your ip range here, and remove this comment!
        set_real_ip_from 192.168.7.0/16;
        set_real_ip_from 172.0.0.0/8;
        real_ip_header X-Forwarded-For;
        real_ip_recursive on;
    }
```

JSON for Advanced Tab of each site in NGinX Proxy Manager you want behind Authelia

In the above file, you need to change the 5 portions with the "<" and ">" around them.

On part 1, enter the IP and Port of your authelia server (auth.your-domain.org).

`<your-authelia-server-ip-and:port>` --> put your actual authelia ip and port number here. You only need to edit this the first time, then copy and paste the same config to each site you are setting up as this value will always be the same.

In part 2, you need to change the following:

`<appname>` --> whatever simple name defines the app (e.g. homer, ntop, jitsi, npmui, etc). This will be different for each entry you make in NGinX Proxy Manager for different sites.

`application-ip-and:port` --> The IP address and port number for the application you are placing behind Authelia. This will be different for each entry in NGinX Proxy Manager for different sites.

`auth.<example.com>` should be changed to the domain / subdomain for your authelia server. Change this the first time you make this file, then just use it over and over as this value will remain the same.

Finally, under the '*real ip*' section, you **may need to** add your local network IP range. For instance if your network is 10.21.0.0 based, then you'll want to add a line like:

```
set_real_ip_from 10.21.0.0/16
```

That's it. Save your configuration, and give it a try. NOTE: You may need to use CTRL + F5 to reresh your browser the fist time you open a page you've been to previously. Browsers tend to cache information, so making it refresh once can help.

Test your site, if all is setup correctly, you'll be taken to the Authelia sign in page. Depending on whether you set the site for one_factor or two_factor, you may also be asked to setup your TOTP app, and use your one time pin as well.

I hope this tutorial is helpful for you all, and look forward to using more Authelia going forward!

Support My Channel and Content

Patreon is a membership platform that makes it easy for artists and creators to get paid. Join over 200,000 creators earning salaries from over 6 million monthly patrons.

Revision #2
Created 2022-12-12 19:25:36 UTC by Brian McGonagill
Updated 2022-12-12 19:27:17 UTC by Brian McGonagill