Authelia and NPM

This article describes how Authelia and Nginx Proxy Manager (NPM) can be configured to provide both single sign on protection for various internal services via ssl certificate and two factor authentication via passkeys and/or TOTP (time based one time password).

This requires either a domain pointing to a fixed address or a dynamic dns provider that ideally allows wildcard subdomains. This also requires ports 80/443 to be redirected to the internal server running the npm docker instance ports 70 and 743 respectively. Adjust as necessary.

Replace all example.com domains and password references with your domain and random passwords!

The examples use simple local yaml file for authelia users and local file for notifications (used for 2FA setup) and local sqlite database. Any complex SSO, Redis, LDAP, mysql/postgresql database or other integration is out of scope of this article.

Docker setup

docker-npm.yml

```yaml
name: npm
services:
  npm:
    image: 'jc21/nginx-proxy-manager:latest'
    restart: unless-stopped
    container_name: npm

    ports:
      # These ports are in format <host-port>:<container-port>
      - '70:80' # Public HTTP Port
      - '743:443' # Public HTTPS Port
      - '71:81' # Admin Web Port
      # Add any other Stream port you want to expose
      # - '21:21' # FTP

    environment:
      TZ: "Europe/London"

      # Uncomment this if you want to change the location of
      # the SQLite DB file within the container
      # DB_SQLITE_FILE: "/data/database.sqlite"

      # Uncomment this if IPv6 is not enabled on your host
      DISABLE_IPV6: 'true'

      INITIAL_ADMIN_EMAIL: something@example.com
```

```
      INITIAL_ADMIN_PASSWORD: secure_password

    volumes:
      - /opt/npm/data:/data
      - /opt/npm/letsencrypt:/etc/letsencrypt
      - /opt/npm/snippets:/snippets

    healthcheck:
      test: ["CMD", "/usr/bin/check-health"]
      interval: 10s
      timeout: 3s
```

docker-authelia.yml

```
name: authelia
services:
  authelia:
    image: authelia/authelia
    container_name: authelia
    restart: unless-stopped
    volumes:
      - /opt/authelia:/config
    ports:
      - 6091:9091
    environment:
      - TZ=Europe/London
```

Create the data directories:

```
mkdir -p /opt/authelia
mkdir -p /opt/npm/{data,letsencrypt,snippets}
```

Authelia Configuration

/opt/authelia/configuration.yml

```
# yamllint disable rule:comments-indentation
---
##############################################################################
########
##                            Authelia Configuration
##
##############################################################################
########

##
## Notes:
```

```
##
##    - the default location of this file is assumed to be
configuration.yml unless otherwise noted
##    - when using docker the container expects this by default to be
at /config/configuration.yml
##    - the default location where this file is loaded from can be
overridden with the X_AUTHELIA_CONFIG environment var
##    - the comments in this configuration file are helpful but users
should consult the official documentation on the
##      website at https://www.authelia.com/ or
https://www.authelia.com/configuration/prologue/introduction/
##    - this configuration file template is not automatically updated
##

## Certificates directory specifies where Authelia will load trusted
certificates (public portion) from in addition to
## the system certificates store.
## They should be in base64 format, and have one of the following
extensions: *.cer, *.crt, *.pem.
# certificates_directory: '/config/certificates/'

## The theme to display: light, dark, grey, auto.
theme: 'auto'

## Set the default 2FA method for new users and for when a user has a
preferred method configured that has been
## disabled. This setting must be a method that is enabled.
## Options are totp, webauthn, mobile_push.
# default_2fa_method: ''

##
## Server Configuration
##
# server:
  ## The address for the Main server to listen on in the address common
syntax.
  ## Formats:
  ##  - [<scheme>://]<hostname>[:<port>][/<path>]
  ##  - [<scheme>://][hostname]:<port>[/<path>]
  ## Square brackets indicate optional portions of the format. Scheme
must be 'tcp', 'tcp4', 'tcp6', 'unix', or 'fd'.
  ## The default scheme is 'unix' if the address is an absolute path
otherwise it's 'tcp'. The default port is '9091'.
  ## If the path is specified this configures the router to handle both
the `/` path and the configured path.
  # address: 'tcp://:9091/'

  ## Set the path on disk to Authelia assets.
  ## Useful to allow overriding of specific static assets.
  # asset_path: '/config/assets/'
```

```
  ## Disables writing the health check vars to /app/.healthcheck.env
which makes healthcheck.sh return exit code 0.
  ## This is disabled by default if either /app/.healthcheck.env or
/app/healthcheck.sh do not exist.
  # disable_healthcheck: false

  ## Authelia by default doesn't accept TLS communication on the server
port. This section overrides this behaviour.
  # tls:
    ## The path to the DER base64/PEM format private key.
    # key: ''

    ## The path to the DER base64/PEM format public certificate.
    # certificate: ''

    ## The list of certificates for client authentication.
    # client_certificates: []

  ## Server headers configuration/customization.
  # headers:

    ## The CSP Template. Read the docs.
    # csp_template: ''

  ## Server Buffers configuration.
  # buffers:

    ## Buffers usually should be configured to be the same value.
    ## Explanation at https://www.authelia.com/c/server#buffer-sizes
    ## Read buffer size adjusts the server's max incoming request size
in bytes.
    ## Write buffer size does the same for outgoing responses.

    ## Read buffer.
    # read: 4096

    ## Write buffer.
    # write: 4096

  ## Server Timeouts configuration.
  # timeouts:

    ## Read timeout in the duration common syntax.
    # read: '6 seconds'

    ## Write timeout in the duration common syntax.
    # write: '6 seconds'

    ## Idle timeout in the duration common syntax.
    # idle: '30 seconds'
```

```
  ## Server Endpoints configuration.
  ## This section is considered advanced and it SHOULD NOT be
configured unless you've read the relevant documentation.
  # endpoints:
    ## Enables the pprof endpoint.
    # enable_pprof: false

    ## Enables the expvars endpoint.
    # enable_expvars: false

    ## Configure the authz endpoints.
    # authz:
      # forward-auth:
        # implementation: 'ForwardAuth'
        # authn_strategies: []
      # ext-authz:
        # implementation: 'ExtAuthz'
        # authn_strategies: []
      # auth-request:
        # implementation: 'AuthRequest'
        # authn_strategies: []
      # legacy:
        # implementation: 'Legacy'
        # authn_strategies: []

##
## Log Configuration
##
log:
  ## Level of verbosity for logs: info, debug, trace.
  # level: 'debug'
  level: 'info'

  ## Format the logs are written as: json, text.
  # format: 'json'
  format: 'text'

  ## File path where the logs will be written. If not set logs are
written to stdout.
  file_path: '/config/authelia.log'

  ## Whether to also log to stdout when a log_file_path is defined.
  keep_stdout: false

##
## Telemetry Configuration
##
# telemetry:

  ##
  ## Metrics Configuration
```

```
  ##
  # metrics:
    ## Enable Metrics.
    # enabled: false

    ## The address for the Metrics server to listen on in the address
common syntax.
    ## Formats:
    ##  - [<scheme>://]<hostname>[:<port>][/<path>]
    ##  - [<scheme>://][hostname]:<port>[/<path>]
    ## Square brackets indicate optional portions of the format. Scheme
must be 'tcp', 'tcp4', 'tcp6', 'unix', or 'fd'.
    ## The default scheme is 'unix' if the address is an absolute path
otherwise it's 'tcp'. The default port is '9959'.
    ## If the path is not specified it defaults to `/metrics`.
    # address: 'tcp://:9959/metrics'

    ## Metrics Server Buffers configuration.
    # buffers:

      ## Read buffer.
      # read: 4096

      ## Write buffer.
      # write: 4096

    ## Metrics Server Timeouts configuration.
    # timeouts:

      ## Read timeout in the duration common syntax.
      # read: '6 seconds'

      ## Write timeout in the duration common syntax.
      # write: '6 seconds'

      ## Idle timeout in the duration common syntax.
      # idle: '30 seconds'

##
## TOTP Configuration
##
## Parameters used for TOTP generation.
totp:
  ## Disable TOTP.
  disable: false

  ## The issuer name displayed in the Authenticator application of your
choice.
  issuer: 'domain.example.com'

  ## The TOTP algorithm to use.
```

```
  ## It is CRITICAL you read the documentation before changing this
option:
  ## https://www.authelia.com/c/totp#algorithm
  # algorithm: 'SHA1'

  ## The number of digits a user has to input. Must either be 6 or 8.
  ## Changing this option only affects newly generated TOTP
configurations.
  ## It is CRITICAL you read the documentation before changing this
option:
  ## https://www.authelia.com/c/totp#digits
  # digits: 6

  ## The period in seconds a Time-based One-Time Password is valid for.
  ## Changing this option only affects newly generated TOTP
configurations.
  period: 30

  ## The skew controls number of Time-based One-Time Passwords either
side of the current one that are valid.
  ## Warning: before changing skew read the docs link below.
  skew: 1
  ## See: https://www.authelia.com/c/totp#input-validation to read
  ## the documentation.

  ## The size of the generated shared secrets. Default is 32 and is
sufficient in most use cases, minimum is 20.
  # secret_size: 32

  ## The allowed algorithms for a user to pick from.
  # allowed_algorithms:
  # - 'SHA1'

  ## The allowed digits for a user to pick from.
  # allowed_digits:
  # - 6

  ## The allowed periods for a user to pick from.
  # allowed_periods:
  # - 30

  ## Disable the reuse security policy which prevents replays of one-
time password code values.
  # disable_reuse_security_policy: false

##
## WebAuthn Configuration
##
## Parameters used for WebAuthn.
webauthn:
  ## Disable WebAuthn.
```

```
  disable: false

  ## Enables logins via a Passkey.
  enable_passkey_login: true

  ## The display name the browser should show the user for when using
WebAuthn to login/register.
  display_name: 'Authelia'

  ## Conveyance preference controls if we collect the attestation
statement including the AAGUID from the device.
  ## Options are none, indirect, direct.
  attestation_conveyance_preference: 'indirect'

  ## The interaction timeout for WebAuthn dialogues in the duration
common syntax.
  timeout: '60 seconds'

  ## Authenticator Filtering.
  filtering:
    ## Prohibits registering Authenticators that claim they can export
their credentials in some way.
    prohibit_backup_eligibility: false

    ## Permitted AAGUID's. If configured specifically only allows the
listed AAGUID's.
    permitted_aaguids: []

    ## Prohibited AAGUID's. If configured prohibits the use of specific
AAGUID's.
    prohibited_aaguids: []

  ## Selection Criteria controls the preferences for registration.
  selection_criteria:
    ## The attachment preference. Either 'cross-platform' for dedicated
authenticators, or 'platform' for embedded
    ## authenticators.
    #attachment: 'cross-platform'
    #platform required for fingerprint biometric on devices
    attachment: 'platform'

    ## The discoverability preference. Options are 'discouraged',
'preferred', and 'required'.
    discoverability: 'preferred'

    ## User verification controls if the user must make a gesture or
action to confirm they are present.
    ## Options are required, preferred, discouraged.
    user_verification: 'preferred'

  ## Metadata Service validation via MDS3.
```

```
  # metadata:

    ## Enable the metadata fetch behaviour.
    # enabled: false

    ## Configure the Cache Policy for the Metadata Service.
    # cache_policy: 'strict'

    ## Enable Validation of the Trust Anchor. This generally should be
enabled if you're using the metadata. It
    ## ensures the attestation certificate presented by the
authenticator is valid against the MDS3 certificate that
    ## issued the attestation certificate.
    # validate_trust_anchor: true

    ## Enable Validation of the Entry. This ensures that the MDS3
actually contains the metadata entry. If not enabled
    ## attestation certificates which are not formally registered will
be skipped. This may potentially exclude some
    ## virtual authenticators.
    # validate_entry: true

    ## Enabling this allows attestation certificates with a zero AAGUID
to pass validation. This is important if you do
    ## use non-conformant authenticators like Apple ID.
    # validate_entry_permit_zero_aaguid: false

    ## Enable Validation of the Authenticator Status.
    # validate_status: true

    ## List of statuses which are considered permitted when validating
an authenticator's metadata. Generally it is
    ## recommended that this is not configured as any other status the
authenticator's metadata has will result in an
    ## error. This option is ineffectual if validate_status is false.
    # validate_status_permitted: ~

    ## List of statuses that should be prohibited when validating an
authenticator's metadata. Generally it is
    ## recommended that this is not configured as there are safe
defaults. This option is ineffectual if validate_status
    ## is false, or validate_status_permitted has values.
    # validate_status_prohibited: ~

##
## Duo Push API Configuration
##
## Parameters used to contact the Duo API. Those are generated when you
protect an application of type
## "Partner Auth API" in the management panel.
# duo_api:
```

```
  # disable: false
  # hostname: 'api-123456789.example.com'
  # integration_key: 'ABCDEF'
  ## Secret can also be set using a secret:
https://www.authelia.com/c/secrets
  # secret_key: '1234567890abcdefghifjkl'
  # enable_self_enrollment: false

##
## Identity Validation Configuration
##
## This configuration tunes the identity validation flows.
identity_validation:

  ## Reset Password flow. Adjusts how the reset password flow operates.
  reset_password:
    ## Maximum allowed time before the JWT is generated and when the
user uses it in the duration common syntax.
    # jwt_lifespan: '5 minutes'

    ## The algorithm used for the Reset Password JWT.
    # jwt_algorithm: 'HS256'

    ## The secret key used to sign and verify the JWT.
    jwt_secret: 'set_to_secure_password'


  ## Elevated Session flows. Adjusts the flow which require elevated
sessions for example managing credentials, adding,
  ## removing, etc.
  # elevated_session:
    ## Maximum allowed lifetime after the One-Time Code is generated
that it is considered valid.
    # code_lifespan: '5 minutes'

    ## Maximum allowed lifetime after the user uses the One-Time Code
and the user must perform the validation again in
    ## the duration common syntax.
    # elevation_lifespan: '10 minutes'

    ## Number of characters the one-time password contains.
    # characters: 8

    ## In addition to the One-Time Code requires the user performs a
second factor authentication.
    # require_second_factor: false

    ## Skips the elevation requirement and entry of the One-Time Code
if the user has performed second factor
    ## authentication.
    # skip_second_factor: false
```

```
##
## NTP Configuration
##
## This is used to validate the servers time is accurate enough to
validate TOTP.
# ntp:
  ## The address of the NTP server to connect to in the address common
syntax.
  ## Format: [<scheme>://]<hostname>[:<port>].
  ## Square brackets indicate optional portions of the format. Scheme
must be 'udp', 'udp4', or 'udp6'.
  ## The default scheme is 'udp'. The default port is '123'.
  # address: 'udp://time.cloudflare.com:123'

  ## NTP version.
  # version: 4

  ## Maximum allowed time offset between the host and the NTP server in
the duration common syntax.
  # max_desync: '3 seconds'

  ## Disables the NTP check on startup entirely. This means Authelia
will not contact a remote service at all if you
  ## set this to true, and can operate in a truly offline mode.
  # disable_startup_check: false

  ## The default of false will prevent startup only if we can contact
the NTP server and the time is out of sync with
  ## the NTP server more than the configured max_desync. If you set
this to true, an error will be logged but startup
  ## will continue regardless of results.
  # disable_failure: false

##
## Definitions
##
## The definitions are used in other areas as reference points to
reduce duplication.
##
# definitions:
  ## The user attribute definitions.
  # user_attributes:
    ## The name of the definition.
    # definition_name:
      ## The common expression language expression for this definition.
      # expression: ''

  ## The network definitions.
  # network:
    ## The name of the definition followed by the list of CIDR network
```

```
addresses in this definition.
    # internal:
      # - '10.10.0.0/16'
      # - '172.16.0.0/12'
      # - '192.168.2.0/24'
    # VPN:
      # - '10.9.0.0/16'

##
## Authentication Backend Provider Configuration
##
## Used for verifying user passwords and retrieve information such as
email address and groups users belong to.
##
## The available providers are: `file`, `ldap`. You must use only one
of these providers.
authentication_backend:
  ## Password Change Options.
  # password_change:
    ## Disable both the HTML element and the API for password change
functionality.
    # disable: false
  ## Password Reset Options.
  # password_reset:
    ## Disable both the HTML element and the API for reset password
functionality.
    # disable: false

    ## External reset password url that redirects the user to an
external reset portal. This disables the internal reset
    ## functionality.
    # custom_url: ''

  ## The amount of time to wait before we refresh data from the
authentication backend in the duration common syntax.
  ## To disable this feature set it to 'disable', this will slightly
reduce security because for Authelia, users will
  ## always belong to groups they belonged to at the time of login even
if they have been removed from them in LDAP.
  ## To force update on every request you can set this to '0' or
'always', this will increase processor demand.
  ## See the below documentation for more information.
  ## Refresh Interval docs:
https://www.authelia.com/c/1fa#refresh-interval
  # refresh_interval: '5 minutes'



  ##
  ## File (Authentication Provider)
  ##
  ## With this backend, the users database is stored in a file which is
```

```
updated when users reset their passwords.
  ## Therefore, this backend is meant to be used in a dev environment
and not in production since it prevents Authelia
  ## to be scaled to more than one instance. The options under
'password' have sane defaults, and as it has security
  ## implications it is highly recommended you leave the default
values. Before considering changing these settings
  ## please read the docs page below:
  ## https://www.authelia.com/r/passwords#tuning
  ##
  ## Important: Kubernetes (or HA) users must read
https://www.authelia.com/t/statelessness
  ##
  file:
    path: '/config/users_database.yml'
    # watch: false
    # search:
      # email: false
      # case_insensitive: false
    # password:
      # algorithm: 'argon2'
      # argon2:
        # variant: 'argon2id'
        # iterations: 3
        # memory: 65536
        # parallelism: 4
        # key_length: 32
        # salt_length: 16
      # scrypt:
        # variant: 'scrypt'
        # iterations: 16
        # block_size: 8
        # parallelism: 1
        # key_length: 32
        # salt_length: 16
      # pbkdf2:
        # variant: 'sha512'
        # iterations: 310000
        # salt_length: 16
      # sha2crypt:
        # variant: 'sha512'
        # iterations: 50000
        # salt_length: 16
      # bcrypt:
        # variant: 'standard'
        # cost: 12

##
## Password Policy Configuration.
##
# password_policy:
```

```
  ## The standard policy allows you to tune individual settings
manually.
  # standard:
    # enabled: false

    ## Require a minimum length for passwords.
    # min_length: 8

    ## Require a maximum length for passwords.
    # max_length: 0

    ## Require uppercase characters.
    # require_uppercase: true

    ## Require lowercase characters.
    # require_lowercase: true

    ## Require numeric characters.
    # require_number: true

    ## Require special characters.
    # require_special: true

  ## zxcvbn is a well known and used password strength algorithm. It
does not have tunable settings.
  # zxcvbn:
    # enabled: false

    ## Configures the minimum score allowed.
    # min_score: 3

##
## Privacy Policy Configuration
##
## Parameters used for displaying the privacy policy link and drawer.
# privacy_policy:

  ## Enables the display of the privacy policy using the policy_url.
  # enabled: false

  ## Enables the display of the privacy policy drawer which requires
users accept the privacy policy
  ## on a per-browser basis.
  # require_user_acceptance: false

  ## The URL of the privacy policy document. Must be an absolute URL
and must have the 'https://' scheme.
  ## If the privacy policy enabled option is true, this MUST be
provided.
  # policy_url: ''
```

```
##
## Access Control Configuration
##
## Access control is a list of rules defining the authorizations
applied for one resource to users or group of users.
##
## If 'access_control' is not defined, ACL rules are disabled and the
'deny' rule is applied, i.e., access is denied
## to everyone. Otherwise restrictions follow the rules defined.
##
## Note: One can use the wildcard * to match any subdomain.
## It must stand at the beginning of the pattern. (example:
*.example.com)
##
## Note: You must put patterns containing wildcards between simple
quotes for the YAML to be syntactically correct.
##
## Definition: A 'rule' is an object with the following keys: 'domain',
'subject', 'policy' and 'resources'.
##
## - 'domain' defines which domain or set of domains the rule applies
to.
##
## - 'subject' defines the subject to apply authorizations to. This
parameter is optional and matching any user if not
##    provided. If provided, the parameter represents either a user or
a group. It should be of the form
##    'user:<username>' or 'group:<groupname>'.
##
## - 'policy' is the policy to apply to resources. It must be either
'bypass', 'one_factor', 'two_factor' or 'deny'.
##
## - 'resources' is a list of regular expressions that matches a set of
resources to apply the policy to. This parameter
##   is optional and matches any resource if not provided.
##
## Note: the order of the rules is important. The first policy matching
(domain, resource, subject) applies.
access_control:
  ## Default policy can either be 'bypass', 'one_factor', 'two_factor'
or 'deny'. It is the policy applied to any
  ## resource if there is no policy to be applied to the user.
  # default_policy: 'deny'

  rules:
    ## bypass rule
    - domain:
        - "auth.example.com" #This should be your authentication URL
      policy: bypass
    - domain: "example.com" #example domain to protect
```

```
      policy: bypass
        #- domain: "sub1.yourdomain.com" #example subdomain to protect
      #policy: one_factor
        #- domain: "sub2.yourdomain.com" #example subdomain to protect
        #policy: one_factor
    - domain:
       - 'sub1.example.com'
       - 'sub2.example.com'
       - 'sub3.example.com'
      policy: 'two_factor'
    - domain: "*.example.com" #example to protect all subdomains under
top-level domain
      policy: one_factor
      #add or remove additional subdomains as necessary. currenlty only
supports ONE top-level domain
      #any time you add a new subdomain, you will need to restart the
Authelia container to recognize the new settings/rules


  # rules:
    ## Rules applied to everyone
    # - domain: 'public.example.com'
    #   policy: 'bypass'

    ## Domain Regex examples. Generally we recommend just using a
standard domain.
    # - domain_regex: '^(?P<User>\w+)\.example\.com$'
    #   policy: 'one_factor'
    # - domain_regex: '^(?P<Group>\w+)\.example\.com$'
    #   policy: 'one_factor'
    # - domain_regex:
      #   - '^appgroup-.*\.example\.com$'
      #   - '^appgroup2-.*\.example\.com$'
    #   policy: 'one_factor'
    # - domain_regex: '^.*\.example\.com$'
    #   policy: 'two_factor'

    # - domain: 'secure.example.com'
    #   policy: 'one_factor'
    ## Network based rule, if not provided any network matches.
    #   networks:
        # - 'internal'
        # - 'VPN'
        # - '192.168.1.0/24'
        # - '10.0.0.1'

    # - domain:
        # - 'secure.example.com'
        # - 'private.example.com'
    #   policy: 'two_factor'
```

```
    # - domain: 'singlefactor.example.com'
    #   policy: 'one_factor'

    ## Rules applied to 'admins' group
    # - domain: 'mx2.mail.example.com'
    #   subject: 'group:admins'
    #   policy: 'deny'

    # - domain: '*.example.com'
    #   subject:
        # - 'group:admins'
        # - 'group:moderators'
    #   policy: 'two_factor'

    ## Rules applied to 'dev' group
    # - domain: 'dev.example.com'
    #   resources:
        # - '^/groups/dev/.*$'
    #   subject: 'group:dev'
    #   policy: 'two_factor'

    ## Rules applied to user 'john'
    # - domain: 'dev.example.com'
    #   resources:
        # - '^/users/john/.*$'
    #   subject: 'user:john'
    #   policy: 'two_factor'

    ## Rules applied to user 'harry'
    # - domain: 'dev.example.com'
    #   resources:
        # - '^/users/harry/.*$'
    #   subject: 'user:harry'
    #   policy: 'two_factor'

    ## Rules applied to user 'bob'
    # - domain: '*.mail.example.com'
    #   subject: 'user:bob'
    #   policy: 'two_factor'
    # - domain: 'dev.example.com'
    #   resources:
    #     - '^/users/bob/.*$'
    #   subject: 'user:bob'
    #   policy: 'two_factor'

##
## Session Provider Configuration
##
## The session cookies identify the user once logged in.
## The available providers are: `memory`, `redis`. Memory is the
provider unless redis is defined.
```

```
session:
  ## The secret to encrypt the session data. This is only used with
Redis / Redis Sentinel.
  ## Secret can also be set using a secret:
https://www.authelia.com/c/secrets
  secret: 'insecure_session_secret'

  ## Cookies configures the list of allowed cookie domains for sessions
to be created on.
  ## Undefined values will default to the values below.
  cookies:
      -
       ## The name of the session cookie.
       name: 'authelia_session'

       ## The domain to protect.
       ## Note: the Authelia portal must also be in that domain.
       domain: 'example.com'

       ## Required. The fully qualified URI of the portal to redirect
users to on proxies that support redirections.
       ## Rules:
       ##   - MUST use the secure scheme 'https://'
       ##   - The above 'domain' option MUST either:
       ##      - Match the host portion of this URI.
       ##      - Match the suffix of the host portion when prefixed with
'.'.
       authelia_url: 'https://auth.example.com'

       ## Optional. The fully qualified URI used as the redirection
location if the portal is accessed directly. Not
       ## configuring this option disables the automatic redirection
behaviour.
       ##
       ## Note: this parameter is optional. If not provided, user won't
be redirected upon successful authentication
       ## unless they were redirected to Authelia by the proxy.
       ##
       ## Rules:
       ##   - MUST use the secure scheme 'https://'
       ##   - MUST not match the 'authelia_url' option.
       ##   - The above 'domain' option MUST either:
       ##      - Match the host portion of this URI.
       ##      - Match the suffix of the host portion when prefixed with
'.'.
       default_redirection_url: 'https://example.com'

       ## Sets the Cookie SameSite value. Possible options are none,
lax, or strict.
       ## Please read https://www.authelia.com/c/session#same_site
       # same_site: 'lax'
```

```
      ## The value for inactivity, expiration, and remember_me are in
seconds or the duration common syntax.
      ## All three of these values affect the cookie/session validity
period. Longer periods are considered less secure
      ## because a stolen cookie will last longer giving attackers more
time to spy or attack.

      ## The inactivity time before the session is reset. If expiration
is set to 1h, and this is set to 5m, if the user
      ## does not select the remember me option their session will get
destroyed after 1h, or after 5m since the last
      ## time Authelia detected user activity.
      # inactivity: '5 minutes'

      ## The time before the session cookie expires and the session is
destroyed if remember me IS NOT selected by the
      ## user.
      # expiration: '1 hour'

      ## The time before the cookie expires and the session is
destroyed if remember me IS selected by the user. Setting
      ## this value to -1 disables remember me for this session cookie
domain. If allowed and the user uses the remember
      ## me checkbox this overrides the expiration option and disables
the inactivity option.
      # remember_me: '1 month'

  ## Cookie Session Domain default 'name' value.
  # name: 'authelia_session'

  ## Cookie Session Domain default 'same_site' value.
  # same_site: 'lax'

  ## Cookie Session Domain default 'inactivity' value.
  # inactivity: '5m'

  ## Cookie Session Domain default 'expiration' value.
  # expiration: '1h'

  ## Cookie Session Domain default 'remember_me' value.
  # remember_me: '1M'


##
## Regulation Configuration
##
## This mechanism prevents attackers from brute forcing the first
factor. It bans the user if too many attempts are made
## in a short period of time.
regulation:
```

```
  ## Regulation Mode.
  modes:
    - 'user'

  ## The number of failed login attempts before user is banned. Set it
to 0 to disable regulation.
  max_retries: 3

  ## The time range during which the user can attempt login before
being banned in the duration common syntax. The user
  ## is banned if the authentication failed 'max_retries' times in a
'find_time' seconds window.
  find_time: '2 minutes'

  ## The length of time before a banned user can login again in the
duration common syntax.
  ban_time: '10 minutes'

##
## Storage Provider Configuration
##
## The available providers are: `local`, `mysql`, `postgres`. You must
use one and only one of these providers.
storage:
  ## The encryption key that is used to encrypt sensitive information
in the database. Must be a string with a minimum
  ## length of 20. Please see the docs if you configure this with an
undesirable key and need to change it, you MUST use
  ## the CLI to change this in the database if you want to change it
from a previously configured value.
  encryption_key: 'very_secure_encryption_key'

  ##
  ## Local (Storage Provider)
  ##
  ## This stores the data in a SQLite3 Database.
  ## This is only recommended for lightweight non-stateful
installations.
  ##
  ## Important: Kubernetes (or HA) users must read
https://www.authelia.com/t/statelessness
  ##
  local:
    ## Path to the SQLite3 Database.
    path: '/config/db.sqlite3'

##
## Notification Provider
##
## Notifications are sent to users when they require a password reset,
a WebAuthn registration or a TOTP registration.
```

```
## The available providers are: filesystem, smtp. You must use only one
of these providers.
notifier:
  ## You can disable the notifier startup check by setting this to
true.
  disable_startup_check: true

  ##
  ## File System (Notification Provider)
  ##
  ## Important: Kubernetes (or HA) users must read
https://www.authelia.com/t/statelessness
  ##
  filesystem:
    filename: '/config/notification.txt'

  ##
  ## SMTP (Notification Provider)
  ##
  ## Use a SMTP server for sending notifications. Authelia uses the
PLAIN or LOGIN methods to authenticate.
  ## [Security] By default Authelia will:
  ##   - force all SMTP connections over TLS including unauthenticated
connections
  ##       - use the disable_require_tls boolean value to disable this
requirement
  ##         (only works for unauthenticated connections)
  ##   - validate the SMTP server x509 certificate during the TLS
handshake against the hosts trusted certificates
  ##     (configure in tls section)
  # smtp:
    ## The address of the SMTP server to connect to in the address
common syntax.
    # address: 'smtp://127.0.0.1:25'

    ## The connection timeout in the duration common syntax.
    # timeout: '5 seconds'

    ## The username used for SMTP authentication.
    # username: 'test'

    ## The password used for SMTP authentication.
    ## Can also be set using a secret:
https://www.authelia.com/c/secrets
    # password: 'password'

    ## The sender is used to is used for the MAIL FROM command and the
FROM header.
    ## If this is not defined and the username is an email, we use the
username as this value. This can either be just
    ## an email address or the RFC5322 'Name <email address>' format.
```

```
    # sender: 'Authelia <admin@example.com>'

    ## HELO/EHLO Identifier. Some SMTP Servers may reject the default
of localhost.
    # identifier: 'localhost'

    ## Subject configuration of the emails sent. {title} is replaced by
the text from the notifier.
    # subject: '[Authelia] {title}'

    ## This address is used during the startup check to verify the
email configuration is correct.
    ## It's not important what it is except if your email server only
allows local delivery.
    # startup_check_address: 'test@authelia.com'

    ## By default we require some form of TLS. This disables this check
though is not advised.
    # disable_require_tls: false

    ## Disables sending HTML formatted emails.
    # disable_html_emails: false

    # tls:
      ## The server subject name to check the servers certificate
against during the validation process.
      ## This option is not required if the certificate has a SAN which
matches the address options hostname.
      # server_name: 'smtp.example.com'

      ## Skip verifying the server certificate entirely. In preference
to setting this we strongly recommend you add the
      ## certificate or the certificate of the authority signing the
certificate to the certificates directory which is
      ## defined by the `certificates_directory` option at the top of
the configuration.
      ## It's important to note the public key should be added to the
directory, not the private key.
      ## This option is strongly discouraged but may be useful in some
self-signed situations where validation is not
      ## important to the administrator.
      # skip_verify: false

      ## Minimum TLS version for the connection.
      # minimum_version: 'TLS1.2'

      ## Maximum TLS version for the connection.
      # maximum_version: 'TLS1.3'

      ## The certificate chain used with the private_key if the server
requests TLS Client Authentication
```

```
      ## i.e. Mutual TLS.
      # certificate_chain: |
        # -----BEGIN CERTIFICATE-----
        # ...
        # -----END CERTIFICATE-----
        # -----BEGIN CERTIFICATE-----
        # ...
        # -----END CERTIFICATE-----

      ## The private key used with the certificate_chain if the server
requests TLS Client Authentication
      ## i.e. Mutual TLS.
      # private_key: |
        # -----BEGIN PRIVATE KEY-----
        # ...
        # -----END PRIVATE KEY-----

##
## Identity Providers
##
# identity_providers:

  ##
  ## OpenID Connect (Identity Provider)
  ##
  ## It's recommended you read the documentation before configuration
of this section.
  ## See: https://www.authelia.com/c/oidc/provider
  # oidc:
    ## The hmac_secret is used to sign OAuth2 tokens (authorization
code, access tokens and refresh tokens).
    ## HMAC Secret can also be set using a secret:
https://www.authelia.com/c/secrets
    # hmac_secret: 'this_is_a_secret_abc123abc123abc'

    ## The JWK's issuer option configures multiple JSON Web Keys. It's
required that at least one of the JWK's
    ## configured has the RS256 algorithm. For RSA keys (RS or PS) the
minimum is a 2048 bit key.
    # jwks:
    # -
      ## Key ID embedded into the JWT header for key matching. Must be
an alphanumeric string with 7 or less characters.
      ## This value is automatically generated if not provided. It's
recommended to not configure this.
      # key_id: 'example'

      ## The key algorithm used with this key.
      # algorithm: 'RS256'

      ## The key use expected with this key. Currently only 'sig' is
```

```
supported.
      # use: 'sig'

      ## Required Private Key in PEM DER form.
      # key: |
        # -----BEGIN PRIVATE KEY-----
        # ...
        # -----END PRIVATE KEY-----


      ## Optional matching certificate chain in PEM DER form that
matches the key. All certificates within the chain
      ## must be valid and current, and from top to bottom each
certificate must be signed by the subsequent one.
      # certificate_chain: |
        # -----BEGIN CERTIFICATE-----
        # ...
        # -----END CERTIFICATE-----
        # -----BEGIN CERTIFICATE-----
        # ...
        # -----END CERTIFICATE-----

    ## Enables additional debug messages.
    # enable_client_debug_messages: false

    ## SECURITY NOTICE: It's not recommended changing this option and
values below 8 are strongly discouraged.
    # minimum_parameter_entropy: 8

    ## SECURITY NOTICE: It's not recommended changing this option, and
highly discouraged to have it set to 'never'
    ## for security reasons.
    # enforce_pkce: 'public_clients_only'

    ## SECURITY NOTICE: It's not recommended changing this option. We
encourage you to read the documentation and fully
    ## understanding it before enabling this option.
    # enable_jwt_access_token_stateless_introspection: false

    ## The signing algorithm used for signing the discovery and
metadata responses. An issuer JWK with a matching
    ## algorithm must be available when configured. Most clients
completely ignore this and it has a performance cost.
    # discovery_signed_response_alg: 'none'

    ## The signing key id used for signing the discovery and metadata
responses. An issuer JWK with a matching key id
    ## must be available when configured. Most clients completely
ignore this and it has a performance cost.
    # discovery_signed_response_key_id: ''
```

```
    ## Authorization Policies which can be utilized by clients. The
'policy_name' is an arbitrary value that you pick
    ## which is utilized as the value for the 'authorization_policy' on
the client.
    # authorization_policies:
      # policy_name:
        # default_policy: 'two_factor'
        # rules:
          # - policy: 'one_factor'
          #   subject: 'group:services'
          #   networks:
              #  - '192.168.1.0/24'

    ## The lifespans configure the expiration for these token types in
the duration common syntax. In addition to this
    ## syntax the lifespans can be customized per-client.
    # lifespans:
      ## Configures the default/fallback lifespan for given token
types. This behaviour applies to all clients and all
      ## grant types but you can override this behaviour using the
custom lifespans.
      # access_token: '1 hour'
      # authorize_code: '1 minute'
      # id_token: '1 hour'
      # refresh_token: '90 minutes'

    ## Cross-Origin Resource Sharing (CORS) settings.
    # cors:
      ## List of endpoints in addition to the metadata endpoints to
permit cross-origin requests on.
      # endpoints:
        #  - 'authorization'
        #  - 'pushed-authorization-request'
        #  - 'token'
        #  - 'revocation'
        #  - 'introspection'
        #  - 'userinfo'

      ## List of allowed origins.
      ## Any origin with https is permitted unless this option is
configured or the
      ## allowed_origins_from_client_redirect_uris option is enabled.
      # allowed_origins:
        # - 'https://example.com'

      ## Automatically adds the origin portion of all redirect URI's on
all clients to the list of allowed_origins,
      ## provided they have the scheme http or https and do not have
the hostname of localhost.
      # allowed_origins_from_client_redirect_uris: false
```

```
    ## Clients is a list of registered clients and their configuration.
    ## It's recommended you read the documentation before configuration
of a registered client.
    ## See: https://www.authelia.com/c/oidc/registered-clients
    # clients:
      # -
        ## The Client ID is the OAuth 2.0 and OpenID Connect 1.0 Client
ID which is used to link an application to a
        ## configuration.
        # client_id: 'myapp'

        ## The description to show to users when they end up on the
consent screen. Defaults to the ID above.
        # client_name: 'My Application'

        ## The client secret is a shared secret between Authelia and
the consumer of this client.
        # yamllint disable-line rule:line-length
        # client_secret: '$pbkdf2-
sha512$310000$c8p78n7pUMln0jzvd4aK4Q$JNRBzwAo0ek5qKn50cFzzvE9RXV88h1wJn
5KGiHrD0YKtZaR/nCb2CJPOsKaPK0hjf.9yHxzQGZziziccp6Yng'  # The digest of
'insecure_secret'.

        ## Sector Identifiers are occasionally used to generate
pairwise subject identifiers. In most cases this is not
        ## necessary. It is critical to read the documentation for more
information.
        # sector_identifier_uri: 'https://example.com/sector.json'

        ## Sets the client to public. This should typically not be set,
please see the documentation for usage.
        # public: false

        ## Redirect URI's specifies a list of valid case-sensitive
callbacks for this client.
        # redirect_uris:
          # - 'https://oidc.example.com:8080/oauth2/callback'

        ## Request URI's specifies a list of valid case-sensitive TLS-
secured URIs for this client for use as
        ## URIs to fetch Request Objects.
        # request_uris:
          # - 'https://oidc.example.com:8080/oidc/request-object.jwk'

        ## Audience this client is allowed to request.
        # audience: []

        ## Scopes this client is allowed to request.
        # scopes:
          # - 'openid'
          # - 'groups'
```

```
          # - 'email'
          # - 'profile'

        ## Grant Types configures which grants this client can obtain.
        ## It's not recommended to define this unless you know what
you're doing.
        # grant_types:
          # - 'authorization_code'

        ## Response Types configures which responses this client can be
sent.
        ## It's not recommended to define this unless you know what
you're doing.
        # response_types:
          # - 'code'

        ## Response Modes configures which response modes this client
supports.
        # response_modes:
          # - 'form_post'
          # - 'query'

        ## The policy to require for this client; one_factor or
two_factor. Can also be the key names for the
        ## authorization policies section.
        # authorization_policy: 'two_factor'

        ## The custom lifespan name to use for this client. This must
be configured independent of the client before
        ## utilization. Custom lifespans are reusable similar to
authorization policies.
        # lifespan: ''

        ## The consent mode controls how consent is obtained.
        # consent_mode: 'auto'

        ## This value controls the duration a consent on this client
remains remembered when the consent mode is
        ## configured as 'auto' or 'pre-configured' in the duration
common syntax.
        # pre_configured_consent_duration: '1 week'

        ## Requires the use of Pushed Authorization Requests for this
client when set to true.
        # require_pushed_authorization_requests: false

        ## Enforces the use of PKCE for this client when set to true.
        # require_pkce: false

        ## Enforces the use of PKCE for this client when configured,
and enforces the specified challenge method.
```

```
        ## Options are 'plain' and 'S256'.
        # pkce_challenge_method: 'S256'

        ## The signing algorithm used for signing the authorization
responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#authorization_signed
_response_alg
        # authorization_signed_response_alg: 'RS256'

        ## The signing key id used for signing the authorization
responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#authorization_signed
_response_key_id
        # authorization_signed_response_key_id: ''

        ## The content encryption algorithm used for encrypting the
authorization responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#authorization_encryp
ted_response_alg
        # authorization_encrypted_response_alg: 'none'

        ## The encryption algorithm used for encrypting the
authorization responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#authorization_encryp
ted_response_enc
        # authorization_encrypted_response_enc: 'A128CBC-HS256'

        ## The content encryption key id used for encrypting the
authorization responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#authorization_encryp
ted_response_key_id
        # authorization_encrypted_response_key_id: ''

        ## The signing algorithm used for signing the ID Tokens in
Access Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#id_token_signed_resp
onse_alg
        # id_token_signed_response_alg: 'RS256'
```

```
        ## The signing key id used for signing the ID Tokens in Access
Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#id_token_signed_resp
onse_key_id
        # id_token_signed_response_key_id: ''

        ## The content encryption algorithm used for encrypting the ID
Tokens in Access Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#id_token_encrypted_r
esponse_alg
        # id_token_encrypted_response_alg: 'none'

        ## The encryption algorithm used for encrypting the ID Tokens
in Access Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#id_token_encrypted_r
esponse_enc
        # id_token_encrypted_response_enc: 'A128CBC-HS256'

        ## The content encryption key id used for encrypting the ID
Tokens in Access Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#authorization_encryp
ted_response_key_id
        # id_token_encrypted_response_key_id: ''

        ## The signing algorithm used for signing the Access Tokens in
Access Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#access_token_signed_
response_alg
        # access_token_signed_response_alg: 'none'

        ## The signing key id used for signing the Access Tokens in
Access Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#access_token_signed_
response_key_id
        # access_token_signed_response_key_id: ''

        ## The content encryption algorithm used for encrypting the
Access Tokens in Access Request responses.
        ## Please read the documentation before adjusting this option.
```

```
        ## See:
https://www.authelia.com/c/oidc/registered-clients#access_token_encrypt
ed_response_alg
        # access_token_encrypted_response_alg: 'none'

        ## The encryption algorithm used for encrypting the Access
Tokens in Access Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#access_token_encrypt
ed_response_enc
        # access_token_encrypted_response_enc: 'A128CBC-HS256'

        ## The content encryption key id used for encrypting the Access
Tokens in Access Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#access_token_encrypt
ed_response_key_id
        # access_token_encrypted_response_key_id: ''

        ## The signing algorithm used for signing the User Info Request
responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#userinfo_signed_resp
onse_alg
        # userinfo_signed_response_alg: 'none'

        ## The signing key id used for signing the User Info Request
responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#userinfo_signed_resp
onse_key_id
        # userinfo_signed_response_key_id: ''

        ## The content encryption algorithm used for encrypting the
User Info Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#userinfo_encrypted_r
esponse_alg
        # userinfo_encrypted_response_alg: 'none'

        ## The encryption algorithm used for encrypting the User Info
Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#userinfo_encrypted_r
esponse_enc
```

```
        # userinfo_encrypted_response_enc: 'A128CBC-HS256'

        ## The content encryption key id used for encrypting the User
Info Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#userinfo_encrypted_r
esponse_key_id
        # userinfo_encrypted_response_key_id: ''

        ## The signing algorithm used for signing the Introspection
Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#introspection_signed
_response_alg
        # introspection_signed_response_alg: 'none'

        ## The signing key id used for Introspection responses. An
issuer JWK with a matching key id must be available
        ## when configured.
        # introspection_signed_response_key_id: ''

        ## The content encryption algorithm used for encrypting the
Introspection Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#introspection_encryp
ted_response_alg
        # introspection_encrypted_response_alg: 'none'

        ## The encryption algorithm used for encrypting the
Introspection Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#introspection_encryp
ted_response_enc
        # introspection_encrypted_response_enc: 'A128CBC-HS256'

        ## The content encryption key id used for encrypting the
Introspection Request responses.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#introspection_encryp
ted_response_key_id
        # introspection_encrypted_response_key_id: ''

        ## The signature algorithm which must be used for request
objects.
        ## Please read the documentation before adjusting this option.
        ## See:
```

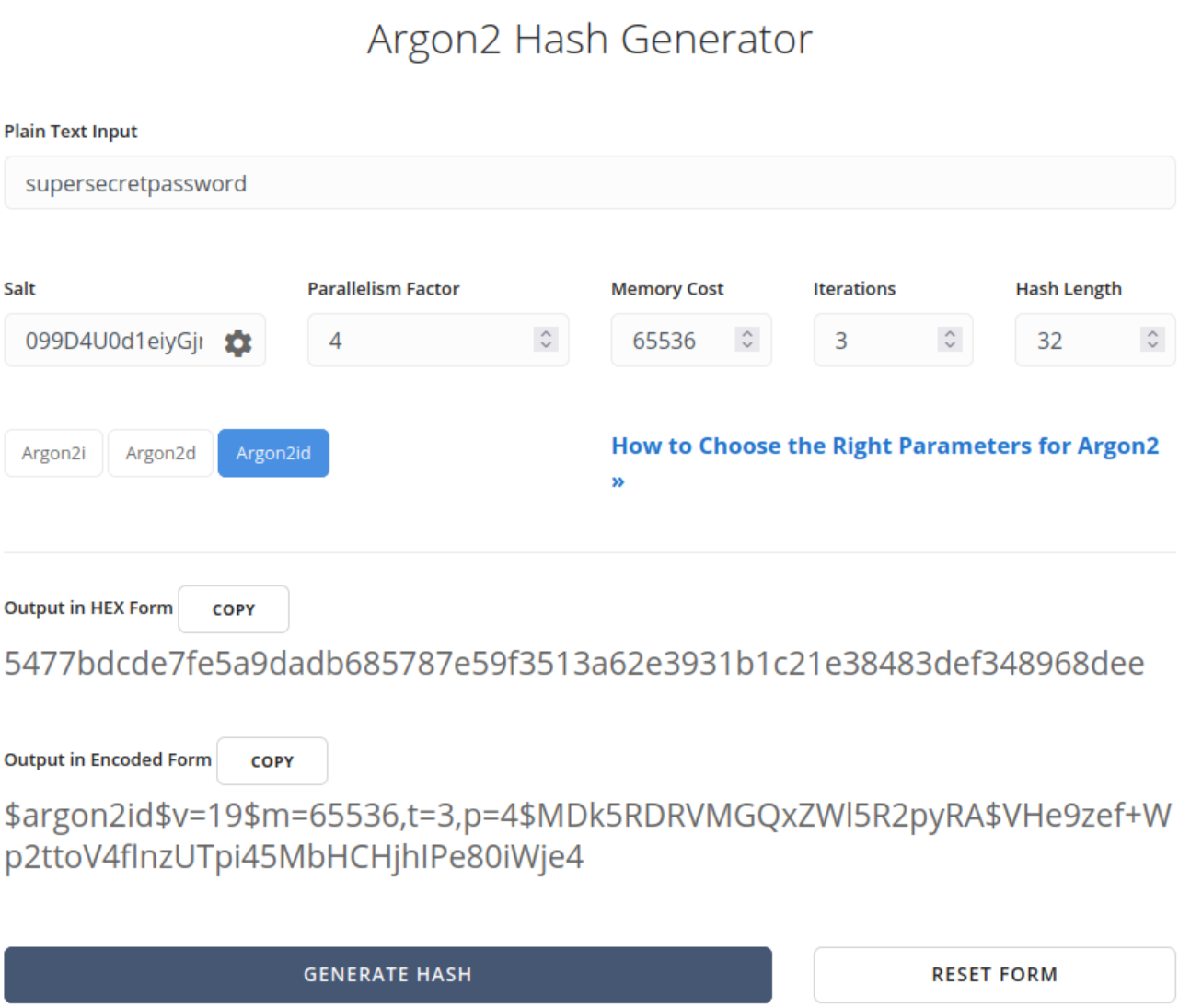
```
https://www.authelia.com/c/oidc/registered-clients#request_object_signi
ng_alg
        # request_object_signing_alg: 'RS256'

        ## The content encryption algorithm which must be used for
request objects.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#request_object_encry
ption_alg
        # request_object_encryption_alg: ''

        ## The encryption algorithm which must be used for request
objects.
        ## Please read the documentation before adjusting this option.
        ## See:
https://www.authelia.com/c/oidc/registered-clients#request_object_encry
ption_enc
        # request_object_encryption_enc: ''

        ## The permitted client authentication method for the Token
Endpoint for this client.
        ## For confidential client types this value defaults to
'client_secret_basic' and for the public client types it
        ## defaults to 'none' per the specifications.
        # token_endpoint_auth_method: 'client_secret_basic'

        ## The permitted client authentication signing algorithm for
the Token Endpoint for this client when using
        ## the 'client_secret_jwt' or 'private_key_jwt'
token_endpoint_auth_method.
        # token_endpoint_auth_signing_alg: 'RS256'

        ## The permitted client authentication method for the
Revocation Endpoint for this client.
        ## For confidential client types this value defaults to
'client_secret_basic' and for the public client types it
        ## defaults to 'none' per the specifications.
        # revocation_endpoint_auth_method: 'client_secret_basic'

        ## The permitted client authentication signing algorithm for
the Revocation Endpoint for this client when using
        ## the 'client_secret_jwt' or 'private_key_jwt'
revocation_endpoint_auth_method.
        # revocation_endpoint_auth_signing_alg: 'RS256'

        ## The permitted client authentication method for the
Introspection Endpoint for this client.
        ## For confidential client types this value defaults to
'client_secret_basic' and for the public client types it
        ## defaults to 'none' per the specifications.
```

```
        # introspection_endpoint_auth_method: 'client_secret_basic'

        ## The permitted client authentication signing algorithm for
the Introspection Endpoint for this client when
        ## using the 'client_secret_jwt' or 'private_key_jwt'
introspection_endpoint_auth_method.
        # introspection_endpoint_auth_signing_alg: 'RS256'

        ## The permitted client authentication method for the Pushed
Authorization Request Endpoint for this client.
        ## For confidential client types this value defaults to
'client_secret_basic' and for the public client types it
        ## defaults to 'none' per the specifications.
        # pushed_authorization_request_endpoint_auth_method:
'client_secret_basic'

        ## The permitted client authentication signing algorithm for
the Pushed Authorization Request Endpoint for this
        ## client when using the 'client_secret_jwt' or
'private_key_jwt'
        ## pushed_authorization_request_endpoint_auth_method.
        # pushed_authorization_request_endpoint_auth_signing_alg:
'RS256'

        ## Trusted public keys configuration for request object signing
for things such as 'private_key_jwt'.
        ## URL of the HTTPS endpoint which serves the keys. Please note
the 'jwks_uri' and the 'jwks' option below
        ## are mutually exclusive.
        # jwks_uri: 'https://app.example.com/jwks.json'

        ## Trusted public keys configuration for request object signing
for things such as 'private_key_jwt'.
        ## List of JWKs known and registered with this client. It's
recommended to use the 'jwks_uri' option if
        ## available due to key rotation. Please note the 'jwks' and
the 'jwks_uri' option above are mutually exclusive.
        # jwks:
          # -
            ## Key ID used to match the JWT's to an individual
identifier. This option is required if configured.
            # key_id: 'example'

            ## The key algorithm expected with this key.
            # algorithm: 'RS256'

            ## The key use expected with this key. Currently only 'sig'
is supported.
            # use: 'sig'

            ## Required Public Key in PEM DER form.
```

```
            # key: |
              # -----BEGIN RSA PUBLIC KEY-----
              # ...
              # -----END RSA PUBLIC KEY-----

            ## The matching certificate chain in PEM DER form that
matches the key if available.
            # certificate_chain: |
              # -----BEGIN CERTIFICATE-----
              # ...
              # -----END CERTIFICATE-----
              # -----BEGIN CERTIFICATE-----
              # ...
              # -----END CERTIFICATE-----
...
```

Authelia password file

For the user passwords, go to https://argon2.online/ to generate your passwords, use the settings you see below:



Enter your password into the “Plain Text Input”

Click the gear in “Salt” to generate a random string of characters.

Be sure to have “Argon2id” activated.

Other settings:

```
Parallelism: 4
Memory Cost: 65536
Iterations: 3
Hash Length: 32
```

Click “Generate Hash”

Copy the string that starts with $argon2id into the associated user password in the users_database.yml

/opt/authelia/users_database.yml

```
users:
  user1: #username for user 1. change to whatever you'd like
```

```
    displayname: "User Name 1" #whatever you want the display name to
be
    password:
"$argon2i$v=19$m=1024,t=1,p=8$eTQ3MXdqOGFiaDZoMUtMVw$OeHWQSg9zGKslOepe5
t4D1T9BZJjHA1Z+doxZrZYDgI" #generated at https://argon2.online/
    email: youremail1@example.com #whatever your email address is
    groups: #enter the groups you want the user to be part of below
      - admins
      - dev
    disabled: false
  user2: #username for user 2. change to whatever you'd like. Or delete
this section if you only have 1 user
    displayname: "User Name 2" #whatever you want the display name to
be
    password:
"$argon2i$v=19$m=1024,t=1,p=8$eTQ3MXdqOGFiaDZoMUtMVw$OeHWQSg9zGKslOepe5
t4D1T9BZJjHA1Z+doxZrZYDgI" #generated at https://argon2.online/
    email: youremail2@example.com #whatever your email address is
    groups: #enter the groups you want the user to be part of below
      - dev
    disabled: true
```

Config Snippets

authelia-authrequest.conf

```
## Send a subrequest to Authelia to verify if the user is authenticated
and has permission to access the resource.
auth_request /internal/authelia/authz;
## Save the upstream metadata response headers from Authelia to
variables.
auth_request_set $user $upstream_http_remote_user;
auth_request_set $groups $upstream_http_remote_groups;
auth_request_set $name $upstream_http_remote_name;
auth_request_set $email $upstream_http_remote_email;
## Inject the metadata response headers from the variables into the
request made to the backend.
proxy_set_header Remote-User $user;
proxy_set_header Remote-Groups $groups;
proxy_set_header Remote-Email $email;
proxy_set_header Remote-Name $name;
## Configure the redirection when the authz failure occurs. Lines
starting with 'Modern Method' and 'Legacy Method'
## should be commented / uncommented as pairs. The modern method uses
the session cookies configuration's authelia_url
## value to determine the redirection URL here. It's much simpler and
compatible with the mutli-cookie domain easily.
## Modern Method: Set the $redirection_url to the Location header of
```

```
the response to the Authz endpoint.
auth_request_set $redirection_url $upstream_http_location;
## Modern Method: When there is a 401 response code from the authz
endpoint redirect to the $redirection_url.
error_page 401 =302 $redirection_url;
## Legacy Method: Set $target_url to the original requested URL.
## This requires http_set_misc module, replace 'set_escape_uri' with
'set' if you don't have this module.
# set_escape_uri $target_url $scheme://$http_host$request_uri;
## Legacy Method: When there is a 401 response code from the authz
endpoint redirect to the portal with the 'rd'
## URL parameter set to $target_url. This requires users update
'auth.yourdomain.com/' with their external authelia URL.
# error_page 401 =302 https://auth.yourdomain.com/?rd=$target_url;
```

Adjust internal IP and port to your system:

authelia-location.conf

```
set $upstream_authelia http://192.168.1.2:6091/api/authz/auth-request;
location /internal/authelia/authz {
    internal;
    proxy_pass $upstream_authelia;
    proxy_set_header X-Original-Method $request_method;
    proxy_set_header X-Original-URL $scheme://$http_host$request_uri;
    proxy_set_header X-Forwarded-For $remote_addr;
    proxy_set_header Content-Length "";
    proxy_set_header Connection "";
    proxy_pass_request_body off;
    proxy_next_upstream error timeout invalid_header http_500 http_502
http_503;
    proxy_redirect http:// $scheme://;
    proxy_http_version 1.1;
    proxy_cache_bypass $cookie_session;
    proxy_no_cache $cookie_session;
    proxy_buffers 4 32k;
    client_body_buffer_size 128k;
    send_timeout 5m;
    proxy_read_timeout 240;
    proxy_send_timeout 240;
    proxy_connect_timeout 240;
```

proxy.conf

```
proxy_set_header Host $host;
proxy_set_header X-Original-URL $scheme://$http_host$request_uri;
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_set_header X-Forwarded-For $remote_addr;
proxy_set_header X-Real-IP $remote_addr;
client_body_buffer_size 128k;
proxy_next_upstream error timeout invalid_header http_500 http_502
http_503;
proxy_redirect http:// $scheme://;
proxy_http_version 1.1;
proxy_cache_bypass $cookie_session;
proxy_no_cache $cookie_session;
proxy_buffers 64 256k;
real_ip_header X-Forwarded-For;
real_ip_recursive on;
send_timeout 5m;
proxy_read_timeout 360;
proxy_send_timeout 360;
proxy_connect_timeout 360;
```

websocket.conf

```
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection "upgrade";
```

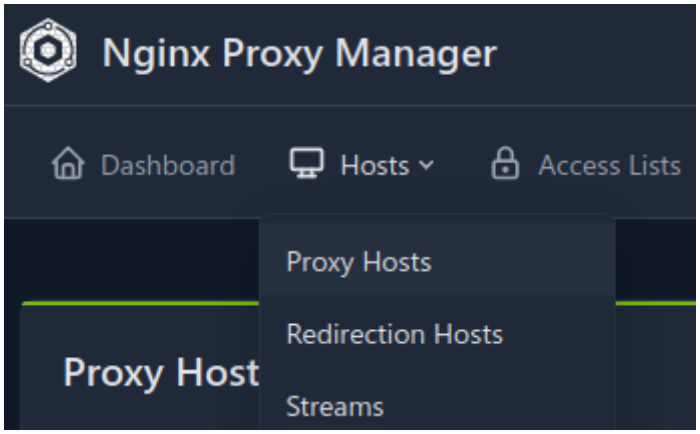
Docker startup

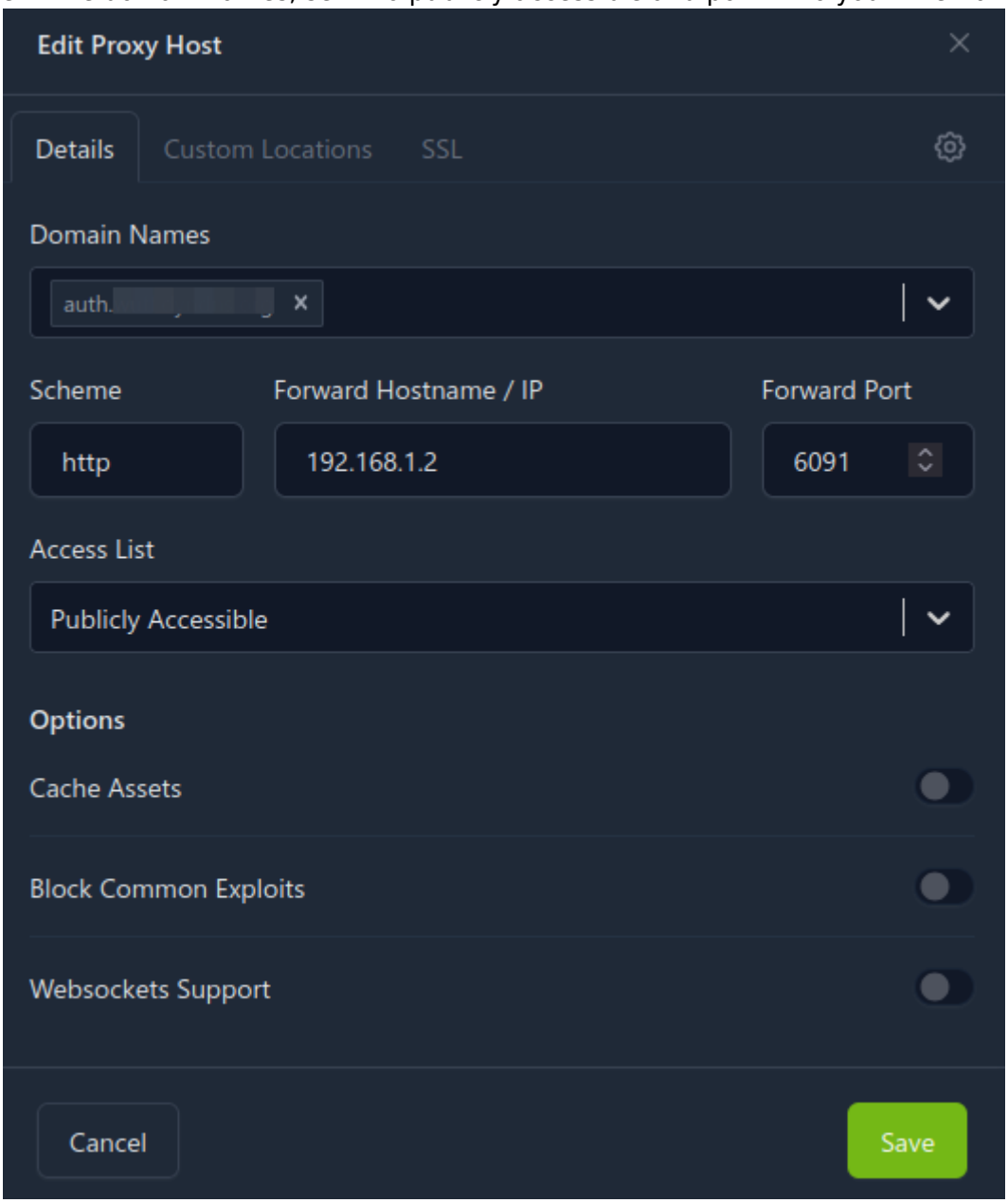
Start the docker containers:

```
docker compose -f docker-npm.yaml up -d
docker compose -f docker-authelia.yaml up -d
```

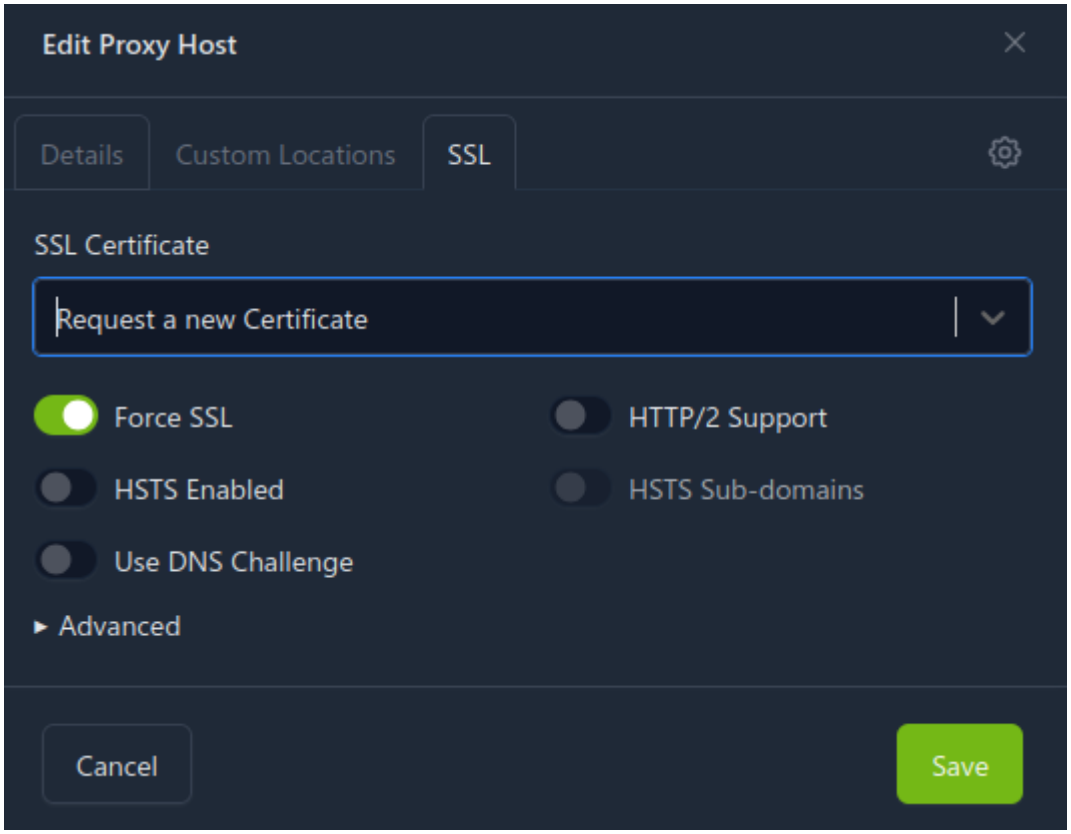
NPM GUI Configuration

The default NPM GUI is internally accessible on http://192.168.1.2:71 (replace with other IP/port if different). Log in using the initial admin email and password as configured in the npm docker yaml file.

Go to Hosts → Proxy Hosts 

Set up the first subdomain 'auth' for Authelia, enter auth.example.com with your domain and click create in the domain names, set it to publicly accessible and point it to your internal IP and Authelia port: 

On the SSL Certificate tab, set it to 'Request a new Certificate' and Force SSL

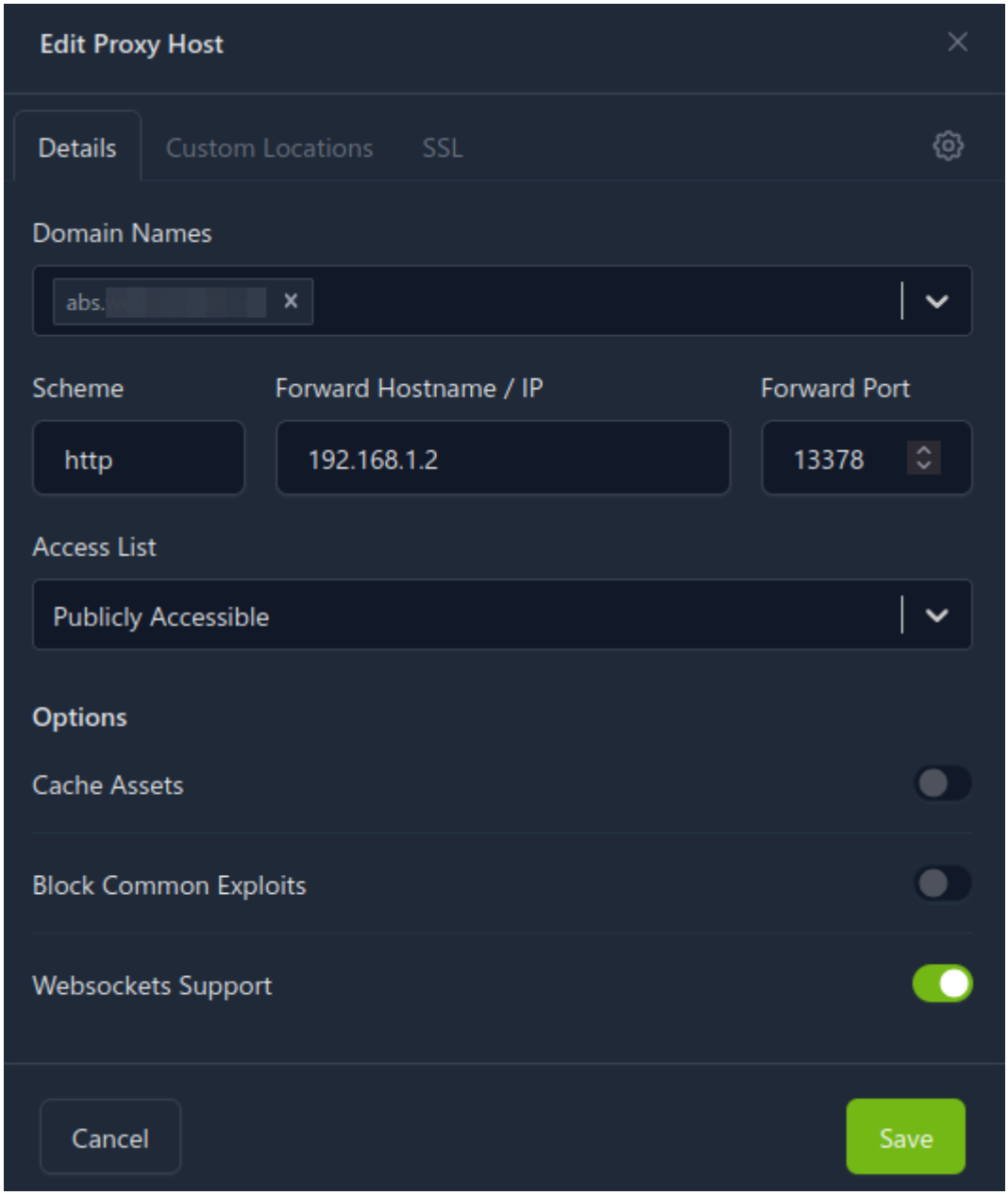


On the cog icon for advanced configuration paste this:

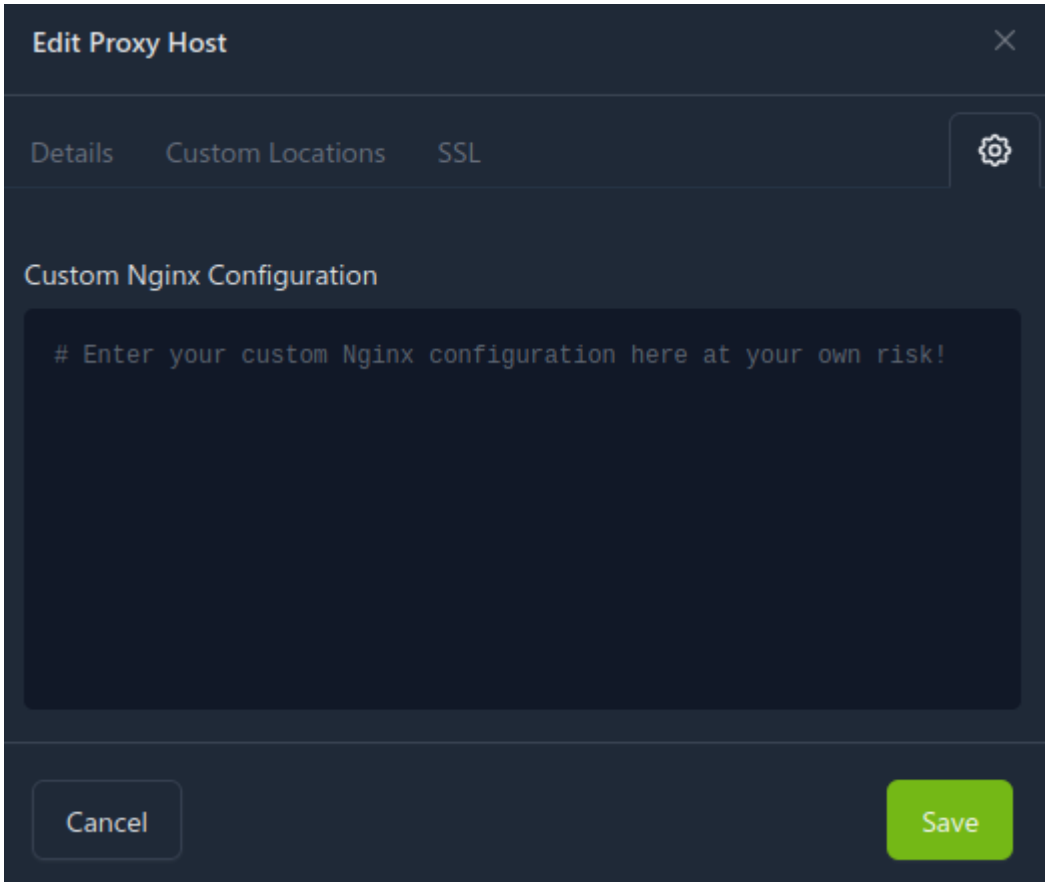
```
location / {
    include /snippets/proxy.conf;
    proxy_pass $forward_scheme://$server:$port;
}
```

Click save.

Then proceed with the first real subdomain.



Request a new ssl certificate as in the auth subdomain. To use Authelia for the main authentication, click the cog icon and paste the below configuration and adjust it to your setup:



Advanced nginx config for Authelia authentication.

```
include /snippets/authelia-location.conf;
location / {
    include /snippets/proxy.conf;
    include /snippets/websocket.conf;
    include /snippets/authelia-authrequest.conf;
    proxy_pass $forward_scheme://$server:$port;
}
```

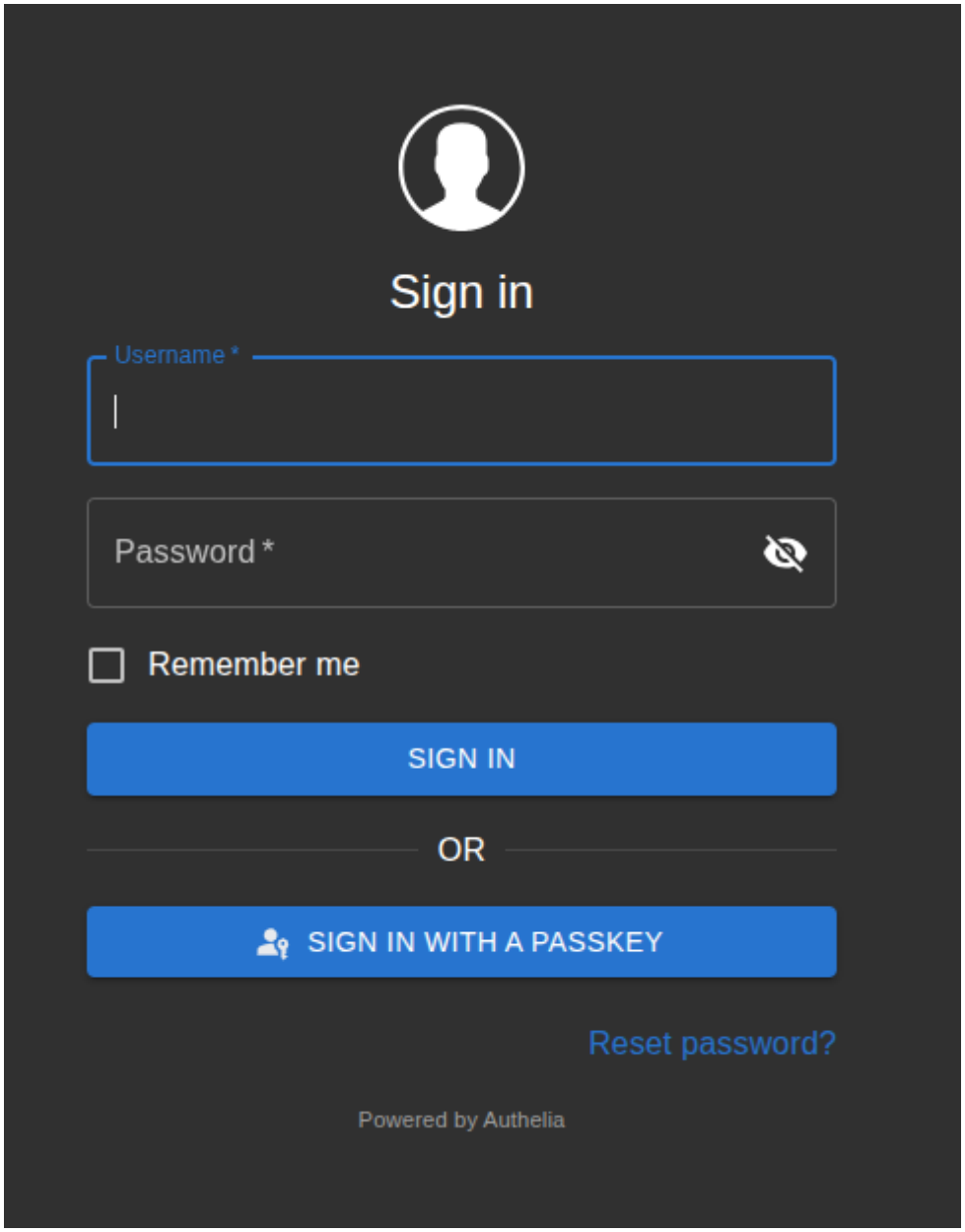
Click save.

Repeat for all additional subdomains as required.

Authelia login

Going to https://auth.example.com (replace with your domain) will show the authelia login prompt now. You can log in and set up 2FA, see your authentication status and change your password and registered devices.

This configuration enables 2FA by default, so logging in to any of the services will prompt 2FA setup. The auth code will be stored in /opt/authelia/notification.txt and not emailed to the user!



By default, authentication is valid for 1 hour or 30 days when the remember me is checked on login. This can be changed in the authelia configuration.yml