Configuring a registry

Estimated reading time: 32 minutes

The Registry configuration is based on a YAML file, detailed below. While it comes with sane default values out of the box, you should review it exhaustively before moving your systems to production.

Override specific configuration options

In a typical setup where you run your Registry from the official image, you can specify a configuration variable from the environment by passing `-e` arguments to your `docker run` stanza or from within a Dockerfile using the `ENV` instruction.

To override a configuration option, create an environment variable named `REGISTRY_variable` where `variable` is the name of the configuration option and the `_` (underscore) represents indention levels. For example, you can configure the `rootdirectory` of the `filesystem` storage backend:

```
storage:
  filesystem:
    rootdirectory: /var/lib/registry
```

To override this value, set an environment variable like this:

```
REGISTRY_STORAGE_FILESYSTEM_ROOTDIRECTORY=/somewhere
```

This variable overrides the `/var/lib/registry` value to the `/somewhere` directory.

> **Note**: Create a base configuration file with environment variables that can be configured to tweak individual values. Overriding configuration sections with environment variables is not recommended.

Overriding the entire configuration file

If the default configuration is not a sound basis for your usage, or if you are having issues overriding keys from the environment, you can specify an alternate YAML configuration file by mounting it as a volume in the container.

Typically, create a new configuration file from scratch,named `config.yml` , then specify it in the `docker run` command:

```
$ docker run -d -p 5000:5000 --restart=always --name registry \
             -v `pwd`/config.yml:/etc/docker/registry/config.yml \
             registry:2
```

Use this example YAML file (https://github.com/docker/distribution/blob/master/cmd/registry/config-example.yml) as a starting point.

List of configuration options

These are all configuration options for the registry. Some options in the list are mutually exclusive. Read the detailed reference information about each option before finalizing your configuration.

```
version: 0.1
log:
  accesslog:
    disabled: true
  level: debug
  formatter: text
  fields:
    service: registry
    environment: staging
  hooks:
    - type: mail
      disabled: true
      levels:
        - panic
      options:
        smtp:
          addr: mail.example.com:25
          username: mailuser
          password: password
          insecure: true
        from: sender@example.com
        to:
          - errors@example.com
loglevel: debug # deprecated: use "log"
storage:
  filesystem:
    rootdirectory: /var/lib/registry
    maxthreads: 100
  azure:
    accountname: accountname
    accountkey: base64encodedaccountkey
    container: containername
  gcs:
    bucket: bucketname
    keyfile: /path/to/keyfile
    rootdirectory: /gcs/object/name/prefix
    chunksize: 5242880
  s3:
    accesskey: awsaccesskey
    secretkey: awssecretkey
    region: us-west-1
    regionendpoint: http://myobjects.local
    bucket: bucketname
    encrypt: true
    keyid: mykeyid
    secure: true
    v4auth: true
    chunksize: 5242880
    multipartcopychunksize: 33554432
    multipartcopymaxconcurrency: 100
    multipartcopythresholdsize: 33554432
    rootdirectory: /s3/object/name/prefix
  swift:
    username: username
```

```
    password: password
    authurl: https://storage.myprovider.com/auth/v1.0 or https://stora
ge.myprovider.com/v2.0 or https://storage.myprovider.com/v3/auth
    tenant: tenantname
    tenantid: tenantid
    domain: domain name for Openstack Identity v3 API
    domainid: domain id for Openstack Identity v3 API
    insecureskipverify: true
    region: fr
    container: containername
    rootdirectory: /swift/object/name/prefix
  oss:
    accesskeyid: accesskeyid
    accesskeysecret: accesskeysecret
    region: OSS region name
    endpoint: optional endpoints
    internal: optional internal endpoint
    bucket: OSS bucket
    encrypt: optional data encryption setting
    secure: optional ssl setting
    chunksize: optional size valye
    rootdirectory: optional root directory
  inmemory:  # This driver takes no parameters
  delete:
    enabled: false
  redirect:
    disable: false
  cache:
    blobdescriptor: redis
  maintenance:
    uploadpurging:
      enabled: true
      age: 168h
      interval: 24h
      dryrun: false
    readonly:
      enabled: false
auth:
  silly:
    realm: silly-realm
    service: silly-service
  token:
    realm: token-realm
    service: token-service
    issuer: registry-token-issuer
    rootcertbundle: /root/certs/bundle
  htpasswd:
    realm: basic-realm
    path: /path/to/htpasswd
middleware:
  registry:
    - name: ARegistryMiddleware
      options:
        foo: bar
  repository:
```

```
    - name: ARepositoryMiddleware
      options:
        foo: bar
  storage:
    - name: cloudfront
      options:
        baseurl: https://my.cloudfronted.domain.com/
        privatekey: /path/to/pem
        keypairid: cloudfrontkeypairid
        duration: 3000s
  storage:
    - name: redirect
      options:
        baseurl: https://example.com/
reporting:
  bugsnag:
    apikey: bugsnagapikey
    releasestage: bugsnagreleasestage
    endpoint: bugsnagendpoint
  newrelic:
    licensekey: newreliclicensekey
    name: newrelicname
    verbose: true
http:
  addr: localhost:5000
  prefix: /my/nested/registry/
  host: https://myregistryaddress.org:5000
  secret: asecretforlocaldevelopment
  relativeurls: false
  tls:
    certificate: /path/to/x509/public
    key: /path/to/x509/private
    clientcas:
      - /path/to/ca.pem
      - /path/to/another/ca.pem
    letsencrypt:
      cachefile: /path/to/cache-file
      email: emailused@letsencrypt.com
  debug:
    addr: localhost:5001
  headers:
    X-Content-Type-Options: [nosniff]
  http2:
    disabled: false
notifications:
  endpoints:
    - name: alistener
      disabled: false
      url: https://my.listener.com/event
      headers: <http.Header>
      timeout: 500
      threshold: 5
      backoff: 1000
      ignoredmediatypes:
        - application/octet-stream
```

```
redis:
  addr: localhost:6379
  password: asecret
  db: 0
  dialtimeout: 10ms
  readtimeout: 10ms
  writetimeout: 10ms
  pool:
    maxidle: 16
    maxactive: 64
    idletimeout: 300s
health:
  storagedriver:
    enabled: true
    interval: 10s
    threshold: 3
  file:
    - file: /path/to/checked/file
      interval: 10s
  http:
    - uri: http://server.to.check/must/return/200
      headers:
        Authorization: [Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==]
      statuscode: 200
      timeout: 3s
      interval: 10s
      threshold: 3
  tcp:
    - addr: redis-server.domain.com:6379
      timeout: 3s
      interval: 10s
      threshold: 3
proxy:
  remoteurl: https://registry-1.docker.io
  username: [username]
  password: [password]
compatibility:
  schema1:
    signingkeyfile: /etc/registry/key.json
validation:
  enabled: true
  manifests:
    urls:
      allow:
        - ^https?://([^/]+\.)*example\.com/
      deny:
        - ^https?://www\.example\.com/
```

In some instances a configuration option is **optional** but it contains child options marked as **required**. In these cases, you can omit the parent with all its children. However, if the parent is included, you must also include all the children marked **required**.

version

```
version: 0.1
```

The `version` option is **required**. It specifies the configuration's version. It is expected to remain a top-level field, to allow for a consistent version check before parsing the remainder of the configuration file.

log

The `log` subsection configures the behavior of the logging system. The logging system outputs everything to stdout. You can adjust the granularity and format with this configuration section.

```
log:
  accesslog:
    disabled: true
  level: debug
  formatter: text
  fields:
    service: registry
    environment: staging
```

Parameter	Required	Description
`level`	no	Sets the sensitivity of logging output. Permitted values are `error`, `warn`, `info`, and `debug`. The default is `info`.
`formatter`	no	This selects the format of logging output. The format primarily affects how keyed attributes for a log line are encoded. Options are `text`, `json`, and `logstash`. The default is `text`.
`fields`	no	A map of field names to values. These are added to every log line for the context. This is useful for identifying log messages source after being mixed in other systems.

accesslog

```
accesslog:
  disabled: true
```

Within `log`, `accesslog` configures the behavior of the access logging system. By default, the access logging system outputs to stdout in Combined Log Format (https://httpd.apache.org/docs/2.4/logs.html#combined). Access logging can be disabled by setting the boolean flag `disabled` to `true`.

hooks

```
hooks:
  - type: mail
    levels:
      - panic
    options:
      smtp:
        addr: smtp.sendhost.com:25
        username: sendername
        password: password
        insecure: true
      from: name@sendhost.com
      to:
        - name@receivehost.com
```

The `hooks` subsection configures the logging hooks' behavior. This subsection includes a sequence handler which you can use for sending mail, for example. Refer to `loglevel` to configure the level of messages printed.

loglevel

> **DEPRECATED:** Please use log (/registry/configuration/#log) instead.

```
loglevel: debug
```

Permitted values are `error`, `warn`, `info` and `debug`. The default is `info`.

storage

```
storage:
  filesystem:
    rootdirectory: /var/lib/registry
  azure:
    accountname: accountname
    accountkey: base64encodedaccountkey
    container: containername
  gcs:
    bucket: bucketname
    keyfile: /path/to/keyfile
    rootdirectory: /gcs/object/name/prefix
  s3:
    accesskey: awsaccesskey
    secretkey: awssecretkey
    region: us-west-1
    regionendpoint: http://myobjects.local
    bucket: bucketname
    encrypt: true
    keyid: mykeyid
    secure: true
    v4auth: true
    chunksize: 5242880
    multipartcopychunksize: 33554432
    multipartcopymaxconcurrency: 100
    multipartcopythresholdsize: 33554432
    rootdirectory: /s3/object/name/prefix
  swift:
    username: username
    password: password
    authurl: https://storage.myprovider.com/auth/v1.0 or https://stora
ge.myprovider.com/v2.0 or https://storage.myprovider.com/v3/auth
    tenant: tenantname
    tenantid: tenantid
    domain: domain name for Openstack Identity v3 API
    domainid: domain id for Openstack Identity v3 API
    insecureskipverify: true
    region: fr
    container: containername
    rootdirectory: /swift/object/name/prefix
  oss:
    accesskeyid: accesskeyid
    accesskeysecret: accesskeysecret
    region: OSS region name
    endpoint: optional endpoints
    internal: optional internal endpoint
    bucket: OSS bucket
    encrypt: optional data encryption setting
    secure: optional ssl setting
    chunksize: optional size valye
    rootdirectory: optional root directory
  inmemory:
  delete:
    enabled: false
  cache:
```

```
    blobdescriptor: inmemory
  maintenance:
    uploadpurging:
      enabled: true
      age: 168h
      interval: 24h
      dryrun: false
    readonly:
      enabled: false
  redirect:
    disable: false
```

The `storage` option is **required** and defines which storage backend is in use. You must configure exactly one backend. If you configure more, the registry returns an error. You can choose any of these backend storage drivers:

Storage driver	Description
`filesystem`	Uses the local disk to store registry files. It is ideal for development and may be appropriate for some small-scale production applications. See the driver's reference documentation (https://github.com/docker/docker.github.io/tree/master/registry/storage-drivers/filesystem.md).
`azure`	Uses Microsoft Azure Blob Storage. See the driver's reference documentation (https://github.com/docker/docker.github.io/tree/master/registry/storage-drivers/azure.md).
`gcs`	Uses Google Cloud Storage. See the driver's reference documentation (https://github.com/docker/docker.github.io/tree/master/registry/storage-drivers/gcs.md).
`s3`	Uses Amazon Simple Storage Service (S3) and compatible Storage Services. See the driver's reference documentation (https://github.com/docker/docker.github.io/tree/master/registry/storage-drivers/s3.md).
`swift`	Uses Openstack Swift object storage. See the driver's reference documentation (https://github.com/docker/docker.github.io/tree/master/registry/storage-drivers/swift.md).
`oss`	Uses Aliyun OSS for object storage. See the driver's reference documentation (https://github.com/docker/docker.github.io/tree/master/registry/storage-drivers/oss.md).

For testing only, you can use the `inmemory` storage driver (https://github.com/docker/docker.github.io/tree/master/registry/storage-drivers/inmemory.md). If you would like to run a registry from volatile memory, use the `filesystem` driver (https://github.com/docker/docker.github.io/tree/master/registry/storage-drivers/filesystem.md) on a ramdisk.

If you are deploying a registry on Windows, a Windows volume mounted from the host is not recommended. Instead, you can use a S3 or Azure backing data-store. If you do use a Windows volume, the length of the `PATH` to the mount point must be within the `MAX_PATH` limits (typically 255 characters), or this error will occur:

```
mkdir /XXX protocol error and your registry will not function properly
.
```

maintenance

Currently, upload purging and read-only mode are the only `maintenance` functions available.

uploadpurging

Upload purging is a background process that periodically removes orphaned files from the upload directories of the registry. Upload purging is enabled by default. To configure upload directory purging, the following parameters must be set.

Parameter	Required	Description
`enabled`	yes	Set to `true` to enable upload purging. Defaults to `true` .
`age`	yes	Upload directories which are older than this age will be deleted.Defaults to `168h` (1 week).
`interval`	yes	The interval between upload directory purging. Defaults to `24h` .
`dryrun`	yes	Set `dryrun` to `true` to obtain a summary of what directories will be deleted. Defaults to `false` .

> **Note**: `age` and `interval` are strings containing a number with optional fraction and a unit suffix. Some examples: `45m` , `2h10m` , `168h` .

readonly

If the `readonly` section under `maintenance` has `enabled` set to `true`, clients will not be allowed to write to the registry. This mode is useful to temporarily prevent writes to the backend storage so a garbage collection pass can be run. Before running garbage collection, the registry should be restarted with readonly's `enabled` set to true. After the garbage collection pass finishes, the registry may be restarted again, this time with `readonly` removed from the configuration (or set to false).

delete

Use the `delete` structure to enable the deletion of image blobs and manifests by digest. It defaults to false, but it can be enabled by writing the following on the configuration file:

```
delete:
  enabled: true
```

cache

Use the `cache` structure to enable caching of data accessed in the storage backend. Currently, the only available cache provides fast access to layer metadata, which uses the `blobdescriptor` field if configured.

You can set `blobdescriptor` field to `redis` or `inmemory`. If set to `redis`,a Redis pool caches layer metadata. If set to `inmemory`, an in-memory map caches layer metadata.

> **NOTE**: Formerly, `blobdescriptor` was known as `layerinfo`. While these are equivalent, `layerinfo` has been deprecated.

redirect

The `redirect` subsection provides configuration for managing redirects from content backends. For backends that support it, redirecting is enabled by default. In certain deployment scenarios, you may decide to route all data through the Registry, rather than redirecting to the backend. This may be more efficient when using a backend that is not co-located or when a registry instance is aggressively caching.

To disable redirects, add a single flag `disable`, set to `true` under the `redirect` section:

```
redirect:
  disable: true
```

auth

```
auth:
  silly:
    realm: silly-realm
    service: silly-service
  token:
    realm: token-realm
    service: token-service
    issuer: registry-token-issuer
    rootcertbundle: /root/certs/bundle
  htpasswd:
    realm: basic-realm
    path: /path/to/htpasswd
```

The `auth` option is **optional**. Possible auth providers include:

- `silly` (/registry/configuration/#silly)
- `token` (/registry/configuration/#token)
- `htpasswd` (/registry/configuration/#token)

You can configure only one authentication provider.

silly

The `silly` authentication provider is only appropriate for development. It simply checks for the existence of the `Authorization` header in the HTTP request. It does not check the header's value. If the header does not exist, the `silly` auth responds with a challenge response, echoing back the realm, service, and scope for which access was denied.

The following values are used to configure the response:

Parameter	Required	Description
`realm`	yes	The realm in which the registry server authenticates.
`service`	yes	The service being authenticated.

token

Token-based authentication allows you to decouple the authentication system from the registry. It is an established authentication paradigm with a high degree of security.

Parameter	Required	Description
`realm`	yes	The realm in which the registry server authenticates.
`service`	yes	The service being authenticated.
`issuer`	yes	The name of the token issuer. The issuer inserts this into the token so it must match the value configured for the issuer.
`rootcertbundle`	yes	The absolute path to the root certificate bundle. This bundle contains the public part of the certificates used to sign authentication tokens.

For more information about Token based authentication configuration, see the specification (https://docs.docker.com/registry/spec/auth/token/).

htpasswd

The *htpasswd* authentication backed allows you to configure basic authentication using an Apache htpasswd file (https://httpd.apache.org/docs/2.4/programs/htpasswd.html). The only supported password format is `bcrypt` (http://en.wikipedia.org/wiki/Bcrypt). Entries with other hash types are ignored. The `htpasswd` file is loaded once, at startup. If the file is invalid, the registry will display an error and will not start.

> **Warning**: Only use the `htpasswd` authentication scheme with TLS configured, since basic authentication sends passwords as part of the HTTP header.

Parameter	Required	Description
`realm`	yes	The realm in which the registry server authenticates.
`path`	yes	The path to the `htpasswd` file to load at startup.

middleware

The `middleware` structure is **optional**. Use this option to inject middleware at named hook points. Each middleware must implement the same interface as the object it is wrapping. For instance, a registry middleware must implement the `distribution.Namespace` interface, while a repository middleware must implement `distribution.Repository`, and a storage middleware must implement `driver.StorageDriver`.

This is an example configuration of the `cloudfront` middleware, a storage middleware:

```
middleware:
  registry:
    - name: ARegistryMiddleware
      options:
        foo: bar
  repository:
    - name: ARepositoryMiddleware
      options:
        foo: bar
  storage:
    - name: cloudfront
      options:
        baseurl: https://my.cloudfronted.domain.com/
        privatekey: /path/to/pem
        keypairid: cloudfrontkeypairid
        duration: 3000s
```

Each middleware entry has `name` and `options` entries. The `name` must correspond to the name under which the middleware registers itself. The `options` field is a map that details custom configuration required to initialize the middleware. It is treated as a `map[string]interface{}`. As such, it supports any interesting structures desired, leaving it up to the middleware initialization function to best determine how to handle the specific interpretation of the options.

cloudfront

Parameter	Required	Description
`baseurl`	yes	The `SCHEME://HOST[/PATH]` at which Cloudfront is served.
`privatekey`	yes	The private key for Cloudfront, provided by AWS.
`keypairid`	yes	The key pair ID provided by AWS.

Parameter	Required	Description
`duration`	no	An integer and unit for the duration of the Cloudfront session. Valid time units are `ns`, `us` (or `µs`), `ms`, `s`, `m`, or `h`. For example, `3000s` is valid, but `3000 s` is not. If you do not specify a `duration` or you specify an integer without a time unit, the duration defaults to `20m` (20 minutes).

redirect

You can use the `redirect` storage middleware to specify a custom URL to a location of a proxy for the layer stored by the S3 storage driver.

Parameter	Required	Description
`baseurl`	yes	`SCHEME://HOST` at which layers are served. Can also contain port. For example, `https://example.com:5443`.

reporting

```
reporting:
  bugsnag:
    apikey: bugsnagapikey
    releasestage: bugsnagreleasestage
    endpoint: bugsnagendpoint
  newrelic:
    licensekey: newreliclicensekey
    name: newrelicname
    verbose: true
```

The `reporting` option is **optional** and configures error and metrics reporting tools. At the moment only two services are supported:

- Bugsnag (/registry/configuration/#bugsnag)
- New Relic (/registry/configuration/#new-relic)

A valid configuration may contain both.

bugsnag

Parameter	Required	Description
`apikey`	yes	The API Key provided by Bugsnag.

Parameter	Required	Description
`releasestage`	no	Tracks where the registry is deployed, using a string like `production`, `staging`, or `development`.
`endpoint`	no	The enterprise Bugsnag endpoint.

newrelic

Parameter	Required	Description
`licensekey`	yes	License key provided by New Relic.
`name`	no	New Relic application name.
`verbose`	no	Set to `true` to enable New Relic debugging output on `stdout`.

http

```
http:
  addr: localhost:5000
  net: tcp
  prefix: /my/nested/registry/
  host: https://myregistryaddress.org:5000
  secret: asecretforlocaldevelopment
  relativeurls: false
  tls:
    certificate: /path/to/x509/public
    key: /path/to/x509/private
    clientcas:
      - /path/to/ca.pem
      - /path/to/another/ca.pem
    letsencrypt:
      cachefile: /path/to/cache-file
      email: emailused@letsencrypt.com
  debug:
    addr: localhost:5001
  headers:
    X-Content-Type-Options: [nosniff]
  http2:
    disabled: false
```

The `http` option details the configuration for the HTTP server that hosts the registry.

Parameter	Required	Description
`addr`	yes	The address for which the server should accept connections. The form depends on a network type (see the `net` option). Use `HOST:PORT` for TCP and `FILE` for a UNIX socket.
`net`	no	The network used to create a listening socket. Known networks are `unix` and `tcp` .
`prefix`	no	If the server does not run at the root path, set this to the value of the prefix. The root path is the section before `v2` . It requires both preceding and trailing slashes, such as in the example `/path/` .
`host`	no	A fully-qualified URL for an externally-reachable address for the registry. If present, it is used when creating generated URLs. Otherwise, these URLs are derived from client requests.
`secret`	no	A random piece of data used to sign state that may be stored with the client to protect against tampering. For production environments you should generate a random piece of data using a cryptographically secure random generator. If you omit the secret, the registry will automatically generate a secret when it starts. **If you are building a cluster of registries behind a load balancer, you MUST ensure the secret is the same for all registries.**
`relativeurls`	no	If `true` , the registry returns relative URLs in Location headers. The client is responsible for resolving the correct URL. **This option is not compatible with Docker 1.7 and earlier.**

`tls`

The `tls` structure within `http` is **optional**. Use this to configure TLS for the server. If you already have a web server running on the same host as the registry, you may prefer to configure TLS on that web server and proxy connections to the registry server.

Parameter	Required	Description
`certificate`	yes	Absolute path to the x509 certificate file.
`key`	yes	Absolute path to the x509 private key file.

Parameter	Required	Description
`clientcas`	no	An array of absolute paths to x509 CA files.

`letsencrypt`

The `letsencrypt` structure within `tls` is **optional**. Use this to configure TLS certificates provided by Let's Encrypt (https://letsencrypt.org/how-it-works/).

> **NOTE**: When using Let's Encrypt, ensure that the outward-facing address is accessible on port `443`. The registry defaults to listening on port `5000`. If you run the registry as a container, consider adding the flag `-p 443:5000` to the `docker run` command or using a similar setting in a cloud configuration.

Parameter	Required	Description
`cachefile`	yes	Absolute path to a file where the Let's Encrypt agent can cache data.
`email`	yes	The email address used to register with Let's Encrypt.

`debug`

The `debug` option is **optional**. Use it to configure a debug server that can be helpful in diagnosing problems. The debug endpoint can be used for monitoring registry metrics and health, as well as profiling. Sensitive information may be available via the debug endpoint. Please be certain that access to the debug endpoint is locked down in a production environment.

The `debug` section takes a single required `addr` parameter, which specifies the `HOST:PORT` on which the debug server should accept connections.

`headers`

The `headers` option is **optional**. Use it to specify headers that the HTTP server should include in responses. This can be used for security headers such as `Strict-Transport-Security`.

The `headers` option should contain an option for each header to include, where the parameter name is the header's name, and the parameter value a list of the header's payload values.

Including `X-Content-Type-Options: [nosniff]` is recommended, so that browsers will not interpret content as HTML if they are directed to load a page from the registry. This header is included in the example configuration file.

http2

The `http2` structure within `http` is **optional**. Use this to control http2 settings for the registry.

Parameter	Required	Description
`disabled`	no	If `true`, then `http2` support is disabled.

notifications

```
notifications:
  endpoints:
    - name: alistener
      disabled: false
      url: https://my.listener.com/event
      headers: <http.Header>
      timeout: 500
      threshold: 5
      backoff: 1000
      ignoredmediatypes:
        - application/octet-stream
```

The notifications option is **optional** and currently may contain a single option, `endpoints`.

endpoints

The `endpoints` structure contains a list of named services (URLs) that can accept event notifications.

Parameter	Required	Description
`name`	yes	A human-readable name for the service.
`disabled`	no	If `true`, notifications are disabled for the service.
`url`	yes	The URL to which events should be published.

Parameter	Required	Description
`headers`	yes	A list of static headers to add to each request. Each header's name is a key beneath `headers`, and each value is a list of payloads for that header name. Values must always be lists.
`timeout`	yes	A value for the HTTP timeout. A positive integer and an optional suffix indicating the unit of time, which may be `ns`, `us`, `ms`, `s`, `m`, or `h`. If you omit the unit of time, `ns` is used.
`threshold`	yes	An integer specifying how long to wait before backing off a failure.
`backoff`	yes	How long the system backs off before retrying after a failure. A positive integer and an optional suffix indicating the unit of time, which may be `ns`, `us`, `ms`, `s`, `m`, or `h`. If you omit the unit of time, `ns` is used.
`ignoredmediatypes`	no	A list of target media types to ignore. Events with these target media types are not published to the endpoint.

redis

```
redis:
  addr: localhost:6379
  password: asecret
  db: 0
  dialtimeout: 10ms
  readtimeout: 10ms
  writetimeout: 10ms
  pool:
    maxidle: 16
    maxactive: 64
    idletimeout: 300s
```

Declare parameters for constructing the `redis` connections. Registry instances may use the Redis instance for several applications. Currently, it caches information about immutable blobs. Most of the `redis` options control how the registry connects to the `redis` instance. You can control the pool's behavior with the pool (/registry/configuration/#pool) subsection.

You should configure Redis with the **allkeys-lru** eviction policy, because the registry does not set an expiration value on keys.

Parameter	Required	Description
`addr`	yes	The address (host and port) of the Redis instance.
`password`	no	A password used to authenticate to the Redis instance.
`db`	no	The name of the database to use for each connection.
`dialtimeout`	no	The timeout for connecting to the Redis instance.
`readtimeout`	no	The timeout for reading from the Redis instance.
`writetimeout`	no	The timeout for writing to the Redis instance.

`pool`

```
pool:
  maxidle: 16
  maxactive: 64
  idletimeout: 300s
```

Use these settings to configure the behavior of the Redis connection pool.

Parameter	Required	Description
`maxidle`	no	The maximum number of idle connections in the pool.
`maxactive`	no	The maximum number of connections which can be open before blocking a connection request.
`idletimeout`	no	How long to wait before closing inactive connections.

health

```
health:
  storagedriver:
    enabled: true
    interval: 10s
    threshold: 3
  file:
    - file: /path/to/checked/file
      interval: 10s
  http:
    - uri: http://server.to.check/must/return/200
      headers:
        Authorization: [Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==]
      statuscode: 200
      timeout: 3s
      interval: 10s
      threshold: 3
  tcp:
    - addr: redis-server.domain.com:6379
      timeout: 3s
      interval: 10s
      threshold: 3
```

The health option is **optional**, and contains preferences for a periodic health check on the storage driver's backend storage, as well as optional periodic checks on local files, HTTP URIs, and/or TCP servers. The results of the health checks are available at the `/debug/health` endpoint on the debug HTTP server if the debug HTTP server is enabled (see http section).

storagedriver

The `storagedriver` structure contains options for a health check on the configured storage driver's backend storage. The health check is only active when `enabled` is set to `true`.

Parameter	Required	Description
`enabled`	yes	Set to `true` to enable storage driver health checks or `false` to disable them.
`interval`	no	How long to wait between repetitions of the storage driver health check. A positive integer and an optional suffix indicating the unit of time. The suffix is one of `ns`, `us`, `ms`, `s`, `m`, or `h`. Defaults to `10s` if the value is omitted. If you specify a value but omit the suffix, the value is interpreted as a number of nanoseconds.

Parameter	Required	Description
`threshold`	no	A positive integer which represents the number of times the check must fail before the state is marked as unhealthy. If not specified, a single failure marks the state as unhealthy.

`file`

The `file` structure includes a list of paths to be periodically checked for the\ existence of a file. If a file exists at the given path, the health check will fail. You can use this mechanism to bring a registry out of rotation by creating a file.

Parameter	Required	Description
`file`	yes	The path to check for existence of a file.
`interval`	no	How long to wait before repeating the check. A positive integer and an optional suffix indicating the unit of time. The suffix is one of `ns` , `us` , `ms` , `s` , `m` , or `h` . Defaults to `10s` if the value is omitted.

`http`

The `http` structure includes a list of HTTP URIs to periodically check with `HEAD` requests. If a `HEAD` request does not complete or returns an unexpected status code, the health check will fail.

Parameter	Required	Description
`uri`	yes	The URI to check.
`headers`	no	Static headers to add to each request. Each header's name is a key beneath `headers` , and each value is a list of payloads for that header name. Values must always be lists.
`statuscode`	no	The expected status code from the HTTP URI. Defaults to `200` .
`timeout`	no	How long to wait before timing out the HTTP request. A positive integer and an optional suffix indicating the unit of time. The suffix is one of `ns` , `us` , `ms` , `s` , `m` , or `h` . If you specify a value but omit the suffix, the value is interpreted as a number of nanoseconds.

Parameter	Required	Description
`interval`	no	How long to wait before repeating the check. A positive integer and an optional suffix indicating the unit of time. The suffix is one of `ns`, `us`, `ms`, `s`, `m`, or `h`. Defaults to `10s` if the value is omitted. If you specify a value but omit the suffix, the value is interpreted as a number of nanoseconds.
`threshold`	no	The number of times the check must fail before the state is marked as unhealthy. If this field is not specified, a single failure marks the state as unhealthy.

`tcp`

The `tcp` structure includes a list of TCP addresses to periodically check using TCP connection attempts. Addresses must include port numbers. If a connection attempt fails, the health check will fail.

Parameter	Required	Description
`addr`	yes	The TCP address and port to connect to.
`timeout`	no	How long to wait before timing out the TCP connection. A positive integer and an optional suffix indicating the unit of time. The suffix is one of `ns`, `us`, `ms`, `s`, `m`, or `h`. If you specify a value but omit the suffix, the value is interpreted as a number of nanoseconds.
`interval`	no	How long to wait between repetitions of the check. A positive integer and an optional suffix indicating the unit of time. The suffix is one of `ns`, `us`, `ms`, `s`, `m`, or `h`. Defaults to `10s` if the value is omitted. If you specify a value but omit the suffix, the value is interpreted as a number of nanoseconds.
`threshold`	no	The number of times the check must fail before the state is marked as unhealthy. If this field is not specified, a single failure marks the state as unhealthy.

proxy

```
proxy:
  remoteurl: https://registry-1.docker.io
  username: [username]
  password: [password]
```

The `proxy` structure allows a registry to be configured as a pull-through cache to Docker Hub. See mirror (https://github.com/docker/docker.github.io/tree/master/registry/recipes/mirror.md) for more information. Pushing to a registry configured as a pull-through cache is unsupported.

Parameter	Required	Description
`remoteurl`	yes	The URL for the repository on Docker Hub.
`username`	no	The username registered with Docker Hub which has access to the repository.
`password`	no	The password used to authenticate to Docker Hub using the username specified in `username` .

To enable pulling private repositories (e.g. `batman/robin`) specify the username (such as `batman`) and the password for that username.

> **Note**: These private repositories are stored in the proxy cache's storage. Take appropriate measures to protect access to the proxy cache.

compatibility

```
compatibility:
  schema1:
    signingkeyfile: /etc/registry/key.json
```

Use the `compatibility` structure to configure handling of older and deprecated features. Each subsection defines such a feature with configurable behavior.

schema1

Parameter	Required	Description

Parameter	Required	Description
`signingkeyfile`	no	The signing private key used to add signatures to `schema1` manifests. If no signing key is provided, a new ECDSA key is generated when the registry starts.

validation

```
validation:
  enabled: true
  manifests:
    urls:
      allow:
        - ^https?://([^/]+\.)*example\.com/
      deny:
        - ^https?://www\.example\.com/
```

enabled

Use the `enabled` flag to enable the other options in the `validation` section. They are disabled by default.

manifests

Use the `manifest` subsection to configure manifest validation.

URLS

The `allow` and `deny` options are each a list of regular expressions (https://godoc.org/regexp/syntax) that restrict the URLs in pushed manifests.

If `allow` is unset, pushing a manifest containing URLs fails.

If `allow` is set, pushing a manifest succeeds only if all URLs match one of the `allow` regular expressions **and** one of the following holds:

1. `deny` is unset.
2. `deny` is set but no URLs within the manifest match any of the `deny` regular expressions.

Example: Development configuration

You can use this simple example for local development:

```
version: 0.1
log:
  level: debug
storage:
    filesystem:
        rootdirectory: /var/lib/registry
http:
    addr: localhost:5000
    secret: asecretforlocaldevelopment
    debug:
        addr: localhost:5001
```

This example configures the registry instance to run on port `5000`, binding to `localhost`, with the `debug` server enabled. Registry data is stored in the `/var/lib/registry` directory. Logging is set to `debug` mode, which is the most verbose.

See config-example.yml (https://github.com/docker/distribution/blob/master/cmd/registry/config-example.yml) for another simple configuration. Both examples are generally useful for local development.

Example: Middleware configuration

This example configures Amazon Cloudfront (http://aws.amazon.com/cloudfront/) as the storage middleware in a registry. Middleware allows the registry to serve layers via a content delivery network (CDN). This reduces requests to the storage layer.

Cloudfront requires the S3 storage driver.

This is the configuration expressed in YAML:

```
middleware:
  storage:
  - name: cloudfront
    disabled: false
    options:
      baseurl: http://d111111abcdef8.cloudfront.net
      privatekey: /path/to/asecret.pem
      keypairid: asecret
      duration: 60
```

See the configuration reference for Cloudfront (/registry/configuration/#cloudfront) for more information about configuration options.

> **Note**: Cloudfront keys exist separately from other AWS keys. See the documentation on AWS credentials (http://docs.aws.amazon.com/general/latest/gr/aws-security-credentials.html) for more information.

registry (https://docs.docker.com/glossary/?term=registry), on-prem (https://docs.docker.com/glossary/?term=on-prem), images (https://docs.docker.com/glossary/?term=images), tags (https://docs.docker.com/glossary/?term=tags), repository (https://docs.docker.com/glossary/?term=repository), distribution (https://docs.docker.com/glossary/?term=distribution), configuration (https://docs.docker.com/glossary/?term=configuration)