Setting up Authentication

Jump to bottom

Nick Sweeting edited this page on Jul 11 · 120 revisions

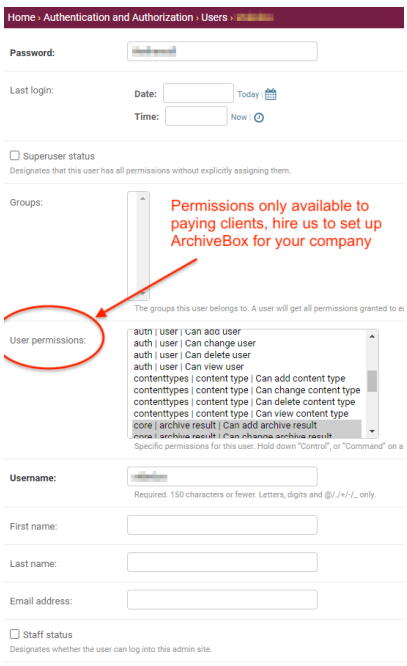
> 💬 *We offer consulting services to set up, integrate, and maintain ArchiveBox with your org's auth & hosting.*
> *If you need support, advanced development to capture difficult sites, audit logging, and more, we can provide it!*
>
> We use this revenue (from corporate clients who can afford to pay) to support open source development and keep ArchiveBox free.

ArchiveBox supports several types of authentication for users logging in via the Admin Web UI or REST API.

Set Up Admin Web UI Permissions

Use these three options to set up your desired permissions for non-admin guest users:

- `PUBLIC_INDEX=True`: Default *allows* non-logged-in users to see Snapshot list
- `PUBLIC_SNAPSHOTS=True`: Default *allows* non-logged-in users to see Snapshot content
- `PUBLIC_ADD_VIEW=False`: Default *doesn't allow* non-logged-in users to submit new URLs



> ℹ Note
>
> **Open source ArchiveBox does not support setting up *non-admin* users** & groups with custom permissions. We do offer this feature, audit logging, and more to paying clients.

- Wiki: Configuration (`PUBLIC_ADD_VIEW` , `PUBLIC_SNAPSHOTS` , `PUBLIC_INDEX`)

- Wiki: Security Overview

Admin Web UI Authentication Methods

Username & Password (the default)

You need a user account to access the Admin UI, you can run the commands below to create/edit a user from the CLI:

```
archivebox manage createsuperuser
archivebox manage changepassword <username>

# equivalent: docker compose run archivebox manage [...]
# equivalent: docker run -v $PWD:/data archivebox/archivebox manage [...]
```

> Tip
>
> If using Docker, you can set `ADMIN_USERNAME` & `ADMIN_PASSWORD` to auto-create an admin account on first run.

Existing users can be managed from the Admin UI here: `/admin/auth/user/`,
and you can change your password in the UI here: `/admin/password_change/`.

Reverse Proxy Authentication

> Can be used with a reverse proxy auth provider like oauth2-proxy, Cloudflare Zero Trust, Authentik, and others.

Set these ArchiveBox configuration values based on your reverse proxy setup and needs:

```
# REQUIRED: the header where your upstream reverse proxy will place the authenti d
# EXAMPLE: Cf-Access-Authenticated-User-Email (if using Cloudflare Access / Zero Trus
REVERSE_PROXY_USER_HEADER=X-Remote-User

# REQUIRED: the IP/CIDR of your upstream reverse proxy server
```

```
# WARNING: make sure this range contains ONLY your reverse proxy server!
# ArchiveBox will completely trust any IP in this range for authentication
REVERSE_PROXY_WHITELIST=192.0.2.3/32

# OPTIONAL: redirect users to an external URL after they log out
LOGOUT_REDIRECT_URL=https://auth.yourcompany.example.com/after/logout
```

- https://github.com/ArchiveBox/ArchiveBox/wiki/Configuration#reverse_proxy_user_header
- https://github.com/ArchiveBox/ArchiveBox/wiki/Configuration#reverse_proxy_whitelist
- https://github.com/ArchiveBox/ArchiveBox/wiki/Configuration#logout_redirect_url
- https://github.com/ArchiveBox/ArchiveBox/pull/866

LDAP Authentication

> Can be used with an SSO provider like Authentik, Authelia, Okta / Auth0, Keycloak, and others.

First, `pip`-install the `ldap` add-on to use this feature (not needed for Docker Archivebox).

```
pip install archivebox[ldap]
```

Then set these configuration values to finish configuring LDAP:

```
LDAP=True
LDAP_SERVER_URI="ldap://ldap.example.com:3389"
LDAP_BIND_DN="ou=archivebox,ou=services,dc=ldap.example.com"
LDAP_BIND_PASSWORD="secret-bind-user-password"
LDAP_USER_BASE="ou=users,ou=archivebox,ou=services,dc=ldap.example.com"
LDAP_USER_FILTER="(objectClass=user)"

LDAP_USERNAME_ATTR="uid"
LDAP_FIRSTNAME_ATTR="givenName"
LDAP_LASTNAME_ATTR="sn"
LDAP_EMAIL_ATTR="mail"
```

- https://github.com/ArchiveBox/ArchiveBox/wiki/Configuration#ldap
- https://github.com/ArchiveBox/ArchiveBox/pull/1214
- https://github.com/django-auth-ldap/django-auth-ldap#example-configuration
- https://jumpcloud.com/blog/what-is-ldap-authentication

Not Yet Supported: SAML / OAuth2 / OpenID Authentication

> *We'd welcome PRs to add support for these using `django-allauth`!*

These methods are not natively supported by ArchiveBox at the moment. However it is still possible to use them with ArchiveBox by running your own IdP (Identity Provider) server to act as a bridge (e.g. Authentik, Authelia, oauth2-proxy).

The IdP server can act as a middleman gateway to authenticate users using an external SAML/OAuth/OpenID/etc. provider (e.g. Google, Microsoft, Github, Facebook, etc.), and then pass on the authenticated user's session info to ArchiveBox using LDAP or reverse proxy headers (as described above).

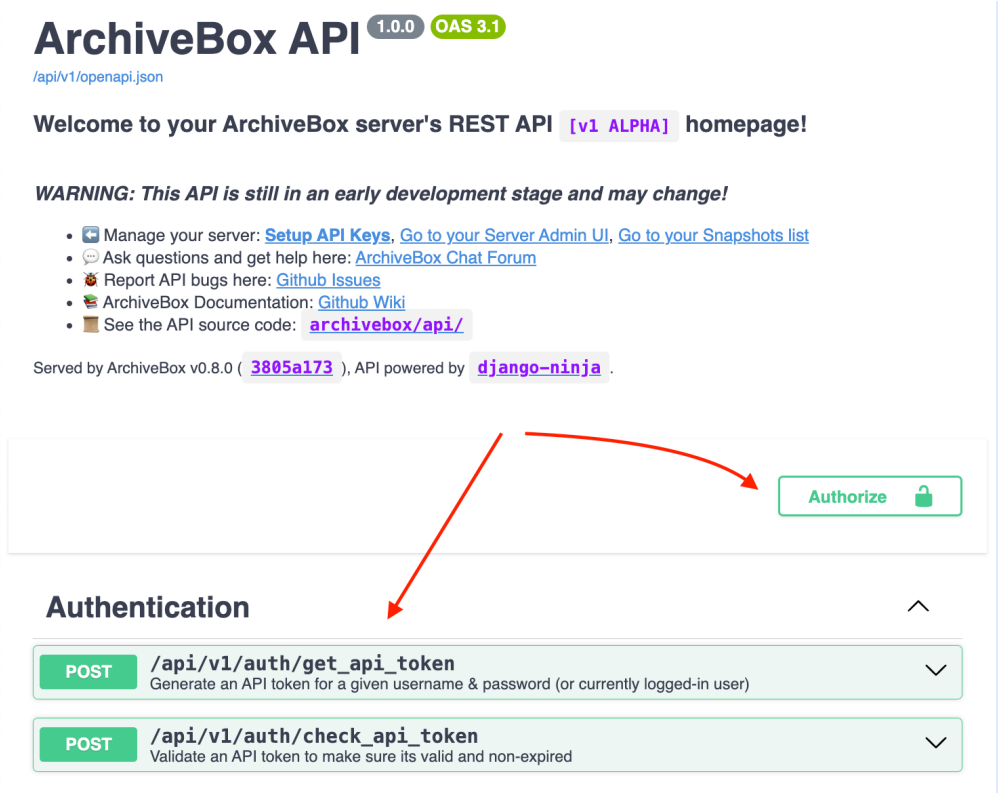
- https://www.cloudflare.com/learning/access-management/what-is-saml/
- https://docs.goauthentik.io/docs/providers/saml/
- https://docs.goauthentik.io/docs/providers/oauth2/
- https://www.authelia.com/configuration/identity-providers/introduction/#openid-connect-10
- https://github.com/oauth2-proxy/oauth2-proxy
- https://oauth2-proxy.github.io/oauth2-proxy/configuration/overview

---

REST API

The REST API (available starting in v0.8.0) supports several methods of authentication for convenience.

To see API docs, try endpoints interactively, and see how auth works, visit this URL on your ArchiveBox server:
`http://127.0.0.1:8000/api/v1/docs`



To get started using the REST API, you can generate an API key for your user in the Admin Web UI: http://127.0.0.1:8000/admin/api/apitoken/add/

or by calling the `http://127.0.0.1:8000/api/v1/auth/get_api_token` endpoint with a username & password:

```
curl -X 'POST' \
  'http://127.0.0.1:8000/api/v1/auth/get_api_token' \
  -H 'Content-Type: application/json'
  -d '{"username": "YOURUSERNAMEHERE", "password": "YOURPASSWORDHERE"}'
```

> Tip
>
> Bearer Tokens are the recommended method for the best balance of security and convenience.

API Bearer Token Authentication

Pass `Authorization=Bearer YOURAPITOKENHERE` as a request header.

```
curl -X 'GET' \
  'http://127.0.0.1:8000/api/v1/core/snapshots?limit=10' \
```

```
  -H 'accept: application/json' \
  -H 'Authorization: Bearer YOURAPITOKENHERE'
```

API Request Header Authentication

> This method is provided in case you have a reverse proxy in front of ArchiveBox that consumes the bearer header.

Pass `X-ArchiveBox-API-Key=YOURAPITOKENHERE` as a request header.

```
curl -X 'GET' \
  'http://127.0.0.1:8000/api/v1/core/snapshots?limit=10' \
  -H 'accept: application/json' \
  -H 'X-ArchiveBox-API-Key: YOURAPITOKENHERE'
```

API Query Parameter Authentication

> ⚠ Warning
>
> This method is sometimes known as "Capability URLs" because anyone in possession of the URL can perform API actions. It comes with important security caveats and is not recommended unless you fully understand the risks.

Pass `api_key=YOURAPITOKENHERE` as a GET/POST query parameter.

```
curl -X 'GET' \
  'http://127.0.0.1:8000/api/v1/core/snapshots?limit=10&api_key=YOURAPITOKENHERE' \
  -H 'accept: application/json'
```

API Session Cookie Authentication

> ⚠ Caution
>
> We recommend sticking to header-based authentication and not using this method unless you deeply understand the CSRF/CORS security risks. This method is mostly useful when accessing the API from external apps where CSRF/CORS is not a concern (e.g. `curl`, mobile apps, other servers, etc.).

> Browsers enforce that requests made to the ArchiveBox API from *other origins* will not include any session cookies by default. This is is a foundational security principle of the web that protects you from API requests being initiated by JS on websites you don't control (aka CSRF/CORS attacks).
>
> To allow incoming POST/PUT/DELETE requests from other domains **that you trust**, you must add them to `CSRF_TRUSTED_ORIGINS` in the `archivebox/core/settings.py` source code on your machine (open an issue and explain your use-case for help).

Log in via the Admin Web UI: `/admin/login/`, you can then re-use your login session id (stored in the `sessionid` cookie) for REST API requests. By default, this only allows you to make requests from the same domain ArchiveBox is being served on (e.g. from browser devtools open on an ArchiveBox page or CLI tools).

```
curl -X 'GET' \
  'http://127.0.0.1:8000/api/v1/core/snapshots?limit=10' \
  -H 'accept: application/json' \
  -H 'Cookie: sessionid=YOURSESSIONIDVALUEHERE'
```

API HTTP Basic Authentication

> Caution
>
> This method is fairly uncommon and is only useful in a few niche situations where the other methods are not available.
> **We will likely remove this method in a future ArchiveBox release if nobody uses it.**
> *If you rely on this method and want us to keep it, please open an issue and explain your use-case!*

Pass your ArchiveBox admin username & password via HTTP Basic Authentication.

```
curl -X 'GET' \
  'http://127.0.0.1:8000/api/v1/core/snapshots?limit=10' \
  -u 'YOURUSERNAMEHERE:YOURPASSWORDHERE'
  -H 'accept: application/json'
```

Further Reading

- The ArchiveBox API auth implementation: `archivebox/api/auth.py` + `archivebox/api/v1_auth.py`
- The `django-ninja` auth documentation (which powers our API)
- The Swagger auth documentation for the interactive API Docs UI

✏️ Help improve our documentation...

▸ **Pages** 37

Getting Started

- 🔢 Quickstart
- 🖥️ Install
- 🐳 Docker
- ➡️ Supported Sources
- ⬅️ Supported Outputs

Usage

- $ Command Line
- 🌐 Web UI
- 🧩 Browser Extension

- 👾 REST API / Webhooks
- 📜 Python API / REPL / SQL API

Reference

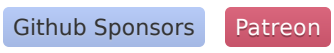
- ⚙️ Configuration
- 📦 Dependencies
- 💿 Disk Layout
- 🔒 Security Overview
- 📝 Developer Documentation

Guides

- Upgrading
- Setting up Storage (NFS/SMB/S3/etc)
- Setting up Authentication (SSO/LDAP/etc)
- Setting up Search (rg/sonic/etc)
- Scheduled Archiving
- Publishing Your Archive
- Chromium Install
- Cookies & Sessions Setup
- Merging Collections
- Troubleshooting

More Info

- ⭐ Web Archiving Community
- Background & Motivation
- Comparison to Other Tools
- Changelog & Roadmap



Clone this wiki locally

```
https://github.com/ArchiveBox/ArchiveBox.wiki.git
```