Bookstack

Wiki software to organize and store information in a structured manner, using a model of shelves, books, chapters, and pages

- About
- 01. Deployments
    - Docker Deployment
    - ZimaOS Deployment
    - TrueNas Deployment
    - YunoHost Deployment
- 02. Configuration
    - Authentik Security
- 03. Docker Deployment
    - 1. Installation
    - 2 Email Verification and Registration
    - 3 Multi Factor Authentication
    - 4. Import & Exporting Books
    - 5. LXC Installation
    - 5. Sorting Rules

About

01. Deployments

Docker Deployment

This installation walks you through installing Bookstack on a docker machine that also has a static IPv4/IPv6 address. If you don't know what this means than please check our Network Basics on the Bookstack Homelab instance.

1. The first thing I like to do is setup my reverse proxy to point to the VM or server that the application will be hosted on. I use the Caddy reverse proxy plugin created for my OPNSense firewall. This is extremely simple for a home lab since it provides a great WebGui and it is all handled on my router instead of on my servers. You can of course choose your own port but in this case I setup the following Caddy entry

- https://docs.example.org --> IPv4-Address-VM:6875
- https://docs.example.org A Record ----> Public IP address of Network

2. The next thing we need to do is create an APP_KEY. So what is that ? Well The APP_KEY in Bookstack is used for session encryption and security purposes, ensuring that sensitive data is protected during user sessions. It is a unique key that helps secure the application against various types of attacks. For this enter your docker machine and generate the key with

```
docker run -it --rm --entrypoint /bin/bash lscr.io/linuxserver/bookstack:latest appkey
```

3. If run correctly it will spit out an APP_KEY in the following format

```
base64:A8PyIIbsR8FZT+5F6QoZYVQesoJvJ0rsP3LXyzGdhHg=
```

4. Copy the APP_KEY including the base64 to a place that you can retrieve it from again later.

5. I assume that you have a dedicated docker folder in which you keep all your docker projects. If not you can create it with the following code below that will create a docker folder and inside it a bookstack folder.

```
mkdir -p docker/bookstack
```

6. Enter the docker/bookstack folder and create a new docker compose file with

```
nano compose.yml
```

7. Then copy paste the boilerplate below inside of it.

Docker Compose File

```
services:
  # The container for BookStack itself
  bookstack:
    env_file: .env
    # You should update the version here to match the latest
    # release of BookStack: https://github.com/BookStackApp/BookStack/releases
    # You'll change this when wanting to update the version of BookStack used.
    image: lscr.io/linuxserver/bookstack:version-v26.03.3
    container_name: bookstack-name # Give this the name of your instance s.t you can distinguish it from multiple instances on the same machine.
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Etc/UTC
      # APP_URL must be set as the base URL you'd expect to access BookStack
      # on via the browser. The default shown here is what you might use if accessing
      # direct from the browser on the docker host, hence the use of the port as configured below.
      - APP_URL=https://docs.example.org
      # APP_KEY must be a unique key. Generate your own by running
      # docker run -it --rm --entrypoint /bin/bash lscr.io/linuxserver/bookstack:latest appkey
      # You should keep the "base64:" part for the option value.
      - APP_KEY=base64:XSuNYw6vfOmXJ3kZfaloEVNE39BEClMrDXOlc2TIGT8=
      # The below database details are purposefully aligned with those
      # configured for the "mariadb" service below:
      - DB_HOST=mariadb
      - DB_PORT=3306
      - DB_DATABASE=sdf092jefjsdlkfj203fjlksdjfdsf
      - DB_USERNAME=9823fj98fj32jf0sjodfijsdf
      - DB_PASSWORD=j2j93jfjsodiifj023jf8349j93jfksjdfdf
    volumes:
      # You generally only ever need to map this one volume.
```

```
      # This maps it to a "bookstack_app_data" folder in the same
      # directory as this compose config file.
      - ./bookstack_app_data:/config
    ports:
      # This exposes port 6875 for general web access.
      # Commonly you'd have a reverse proxy in front of this,
      # redirecting incoming requests to this port.
      - 6875:80        # Check the port! If you have multiple services running on your
dockervm make sure the ports don't clash
    restart: unless-stopped
  # The container for the database which BookStack will use to store
  # most of its core data/content.
  mariadb:
    # You should update the version here to match the latest
    # main version of the linuxserver mariadb container version:
    # https://github.com/linuxserver/docker-
mariadb/pkgs/container/mariadb/versions?filters%5Bversion_type%5D=tagged
    image: lscr.io/linuxserver/mariadb:11.4.9
    container_name: bookstack_name_mariadb
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Etc/UTC
      # You may want to change the credentials used below,
      # but be aware the latter three options need to align
      # with the DB_* options for the BookStack container.
      - MYSQL_ROOT_PASSWORD=sdfjop2fjisodifj203f9j0jfsklfjls
      - MYSQL_DATABASE=sdf092jefjsdlkfj203fjlksdjfdsf
      - MYSQL_USER=9823fj98fj32jf0sjodfijsdf
      - MYSQL_PASSWORD=j2j93jfjsodiifj023jf8349j93jfksjdfdf
    volumes:
      # You generally only ever need to map this one volume.
      # This maps it to a "bookstack_db_data" folder in the same
      # directory as this compose config file.
      # If you're hosting this stack on Windows, you may instead need to
      # use a named docker volume instead of a local path due to
      # filesystem issues causing problems with local mounts on Windows.
      - ./bookstack_db_data:/config
    # These ports are commented out as you don't really need this port
```

```
    # exposed for normal use, mainly only if connecting direct the the
    # database externally. Otherwise, this risks exposing access to the
    # database when not needed.
    # ports:
    #   - 3306:3306
    restart: unless-stopped
```

8. What is important to change in your docker compose file from the boiler plate is the following

- APP_URL=https://docs.example.org
- APP_KEY=base64:Xjslkdjf....
- DB_DATABASE=databaseusername
- DB_USERNAME=bookstackusername
- DB_PASSWORD=bookstackpassword
- MYSQL_ROOT_PASSWORD=randompassword
- MYSQL_DATABASE=databaseusername
- MYSQL_USER=bookstackusername
- MYSQL_PASSWORD=bookstackpassword

If these are all changed then you are ready to run the application inside of the folder where the docker compose file is stored with docker compose up. I like to use **docker compose up** instead of **docker compose up -d** on the first run to see if the logs give any errors. Then you can directly fix what's wrong ;)

```
docker compose up
```

9. If the DNS and reverse proxy are set correctly you should now be able to launch your bookstack instance at https://docs.example.org

10. The first time login requires the following credentials

- Username: admin(@)admin.com
- Password: password

11. After Login I would recommend to directly change your username and password and save the credentials in an open source password manager like Vaultwarden.

ZimaOS Deployment

TrueNas Deployment

YunoHost Deployment

02. Configuration

Authentik Security

For optimum security you can create Captchas at Authentik’s public enrollment flow, since that’s where people create the shared account. Authentik supports CAPTCHA—including hCaptcha and Cloudflare Turnstile—in flows. goauthentik.io

1. In Authentik, create a CAPTCHA stage and configure it with your provider’s site/public key and secret/private key. For Turnstile, use its documented script and verification URLs; for hCaptcha, use its corresponding URLs. goauthentik.io
2. Edit the Enrollment flow used for sign-up and add the CAPTCHA stage before the stage that writes/creates the user.
3. Add email verification to that flow too, so sign-ups must prove access to the address they provide. Authentik enrollment flows can include both CAPTCHA and email verification stages. goauthentik.io
4. Add rate limiting at the reverse proxy in front of Authentik’s enrollment flow as well. CAPTCHA can deter automated sign-ups, while rate limiting slows repeated attempts; neither alone stops a botnet using many IP addresses.
5. Keep newly enrolled users out of the ten BookStack applications by default, or give them only low permissions. Grant access through Authentik groups or policies rather than automatically allowing every registrant into every wiki.

Test registration from a private browser window after changing the flow, including that failed CAPTCHA and unverified email submissions don’t create usable accounts.

03. Docker Deployment

The multi user documentation page that even your grandmother can operate ;)

- **E2EE = NO**
- **Authentication:** Password, 2FA, SSO
- **Admin Metadata Access:** Access to all data
- **Website:** bookstackapp.com
- **Repository:** Codeberg
- **Installation Documentation:** Docker Deployment
- **Installation Videos**: Jim's Garage, Tony Teaches Tech
- **Community:**
- **Donations:** Github-Sponsors, Kofi

1. Installation

This installation walks you through installing Bookstack on a docker machine that also has a static IPv4/IPv6 address. If you don't know what this means than please check our Network Basics on the Bookstack Homelab instance.

1. The first thing I like to do is setup my reverse proxy to point to the VM or server that the application will be hosted on. I use the Caddy reverse proxy plugin created for my OPNSense firewall. This is extremely simple for a home lab since it provides a great WebGui and it is all handled on my router instead of on my servers. You can of course choose your own port but in this case I setup the following Caddy entry

- https://docs.example.org --> IPv4-Address-VM:6875
- https://docs.example.org A Record ----> Public IP address of Network

2. The next thing we need to do is create an APP_KEY. So what is that ? Well The APP_KEY in Bookstack is used for session encryption and security purposes, ensuring that sensitive data is protected during user sessions. It is a unique key that helps secure the application against various types of attacks. For this enter your docker machine and generate the key with

```
docker run -it --rm --entrypoint /bin/bash lscr.io/linuxserver/bookstack:latest appkey
```

3. If run correctly it will spit out an APP_KEY in the following format

> base64:A8PyIIbsR8FZT+5F6QoZYVQesoJvJ0rsP3LXyzGdhHg=

4. Copy the APP_KEY including the base64 to a place that you can retrieve it from again later.

5. I assume that you have a dedicated docker folder in which you keep all your docker projects. If not you can create it with the following code below that will create a docker folder and inside it a bookstack folder.

```
mkdir -p docker/bookstack
```

6. Enter the docker/bookstack folder and create a new docker compose file with

```
nano compose.yml
```

7. Then copy paste the boilerplate below inside of it.

Docker Compose File

```
services:
  # The container for BookStack itself
  bookstack:
    env_file: .env
    # You should update the version here to match the latest
    # release of BookStack: https://github.com/BookStackApp/BookStack/releases
    # You'll change this when wanting to update the version of BookStack used.
    image: lscr.io/linuxserver/bookstack:version-v26.03.3
    container_name: bookstack-name # Give this the name of your instance s.t you can distinguish it from multiple instances on the same machine.
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Etc/UTC
      # APP_URL must be set as the base URL you'd expect to access BookStack
      # on via the browser. The default shown here is what you might use if accessing
      # direct from the browser on the docker host, hence the use of the port as configured below.
      - APP_URL=https://docs.example.org
      # APP_KEY must be a unique key. Generate your own by running
      # docker run -it --rm --entrypoint /bin/bash lscr.io/linuxserver/bookstack:latest appkey
      # You should keep the "base64:" part for the option value.
      - APP_KEY=base64:XSuNYw6vfOmXJ3kZfaloEVNE39BEClMrDXOlc2TIGT8=
      # The below database details are purposefully aligned with those
      # configured for the "mariadb" service below:
      - DB_HOST=mariadb
      - DB_PORT=3306
      - DB_DATABASE=sdf092jefjsdlkfj203fjlksdjfdsf
      - DB_USERNAME=9823fj98fj32jf0sjodfijsdf
      - DB_PASSWORD=j2j93jfjsodiifj023jf8349j93jfksjdfdf
    volumes:
      # You generally only ever need to map this one volume.
```

```
      # This maps it to a "bookstack_app_data" folder in the same
      # directory as this compose config file.
      - ./bookstack_app_data:/config
    ports:
      # This exposes port 6875 for general web access.
      # Commonly you'd have a reverse proxy in front of this,
      # redirecting incoming requests to this port.
      - 6875:80        # Check the port! If you have multiple services running on your
dockervm make sure the ports don't clash
    restart: unless-stopped
  # The container for the database which BookStack will use to store
  # most of its core data/content.
  mariadb:
    # You should update the version here to match the latest
    # main version of the linuxserver mariadb container version:
    # https://github.com/linuxserver/docker-
mariadb/pkgs/container/mariadb/versions?filters%5Bversion_type%5D=tagged
    image: lscr.io/linuxserver/mariadb:11.4.9
    container_name: bookstack_name_mariadb
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Etc/UTC
      # You may want to change the credentials used below,
      # but be aware the latter three options need to align
      # with the DB_* options for the BookStack container.
      - MYSQL_ROOT_PASSWORD=sdfjop2fjisodifj203f9j0jfsklfjls
      - MYSQL_DATABASE=sdf092jefjsdlkfj203fjlksdjfdsf
      - MYSQL_USER=9823fj98fj32jf0sjodfijsdf
      - MYSQL_PASSWORD=j2j93jfjsodiifj023jf8349j93jfksjdfdf
    volumes:
      # You generally only ever need to map this one volume.
      # This maps it to a "bookstack_db_data" folder in the same
      # directory as this compose config file.
      # If you're hosting this stack on Windows, you may instead need to
      # use a named docker volume instead of a local path due to
      # filesystem issues causing problems with local mounts on Windows.
      - ./bookstack_db_data:/config
    # These ports are commented out as you don't really need this port
```

```
    # exposed for normal use, mainly only if connecting direct the the
    # database externally. Otherwise, this risks exposing access to the
    # database when not needed.
    # ports:
    #   - 3306:3306
    restart: unless-stopped
```

8. What is important to change in your docker compose file from the boiler plate is the following

- APP_URL=https://docs.example.org
- APP_KEY=base64:Xjslkdjf....
- DB_DATABASE=databaseusername
- DB_USERNAME=bookstackusername
- DB_PASSWORD=bookstackpassword
- MYSQL_ROOT_PASSWORD=randompassword
- MYSQL_DATABASE=databaseusername
- MYSQL_USER=bookstackusername
- MYSQL_PASSWORD=bookstackpassword

If these are all changed then you are ready to run the application inside of the folder where the docker compose file is stored with docker compose up. I like to use **docker compose up** instead of **docker compose up -d** on the first run to see if the logs give any errors. Then you can directly fix what's wrong ;)

```
docker compose up
```

9. If the DNS and reverse proxy are set correctly you should now be able to launch your bookstack instance at https://docs.example.org

10. The first time login requires the following credentials

- Username: admin(@)admin.com
- Password: password

11. After Login I would recommend to directly change your username and password and save the credentials in an open source password manager like Vaultwarden.

2 Email Verification and Registration

1. After having your Bookstack running it is essential to create email verification for registration and allow users to receive emails for password resets. Email verification is not the only way or safest way but it will definitely get your starting quick. To set this up we need to enter the **.env** file that is located at (if you followed this guide)

2. Also we need to make sure that in the docker compose file the .env file shows up for the bookstack container to use. If not check if it is included like so

```
  bookstack:
    env_file: .env
```

3. Then we can enter in the .env file by going to

```
cd docker/bookstack/bookstack_app_data/www
```

4. After entering the folder we will nano into it with

```
nano .env
```

5. We then need to fill in the email information that sits at the bottom of the file

```
# Mail system to use
# Can be 'smtp' or 'sendmail'
MAIL_DRIVER=smtp

# Mail sender details
MAIL_FROM_NAME="Bookstack Name"
MAIL_FROM=mail(@)example.org

# SMTP mail options
# These settings can be checked using the "Send a Test Email"
# feature found in the "Settings > Maintenance" area of the system.
```

```
# For more detailed documentation on mail options, refer to:
# https://www.bookstackapp.com/docs/admin/email-webhooks/#email-configuration
MAIL_HOST=smtp.mailserver.com
MAIL_PORT="587"
MAIL_USERNAME="info@email.org"
MAIL_PASSWORD="passwordofyouremail"
MAIL_ENCRYPTION=tls
```

> Note the " " for the Mail port, username and password. For some reason I did not get it working without the " ". So if you have the same problem please use " " for these values

6. Then restart your container for the changes to take effect

7. To check if your emails work enter your bookstack instance and go to Settings > Maintenance > Sent a Test Email. If you get a green confirmation message you are all set.

8. If you then want to allow registration by email verification you can set this up at Settings > Registration.

> Require email confirmation

3 Multi Factor Authentication

The best way to protect your Bookstack instance is to force multi factor authentication for all of your users. In this way each new user that signs up needs to have an Authenticator app to be able to create an account. This is a huge security feature, where not even the admin of the Bookstack instance has the capacity to impersonate a user. The most amazing thing is that it is just a click of a button to get this working !

Go to Settings > Roles > Create new Role

In here you can specify the role name TEST and a short description. Straight after that you can hit the box

> Requires Multi-Factor Authentication

- Just hit that box and set the permissions you want new users to have when they enroll for the first time.
- Once that is done you should go back to the registration section. In here you will find the box with
- Default user role after registration
- Then choose the role that includes the Multi-Factor Authentication
- Now, each time you get a new enrollment the user needs to set up Multi-Factor authentication by default! Super useful ;)

4. Import & Exporting Books

Now, one great thing about Bookstack is that it is also extremely simple to export a shelf, book, chapter or page to another Bookstack instance. This is great since you can copy and past basically a whole book from one hub or a society and copy it into your local hub's Bookstack's instance.

1. Simply click on the book you want to export
2. Export it as a zip file
3. Go to the new instance
4. Click on the Books Menu item in the top right corner
5. Click on import
6. Import your exported book
7. Voila! You did it ;) Yes it is that simple and it even keeps it's whole structure.

5. LXC Installation

BookStack is an open-source wiki software designed for organizing and storing information in a book-like structure, featuring a user-friendly interface and support for multiple languages. It allows users to create content using WYSIWYG and Markdown editors, and is available for self-hosting under the MIT License. It is especially simple, lightweight and allows for collaboration at the same time, which makes it a great tool for documentation. Heck you are reading from it right now ;). Now, there are other alternatives like grama.ax,

Installation

Since this is a small project we will host this Bookstack instance as an LXC container inside the production Proxmox PVE1 server. Regarding backups we have one backup to our PBS VM hosted on our PVE2 and an offsite backup also via PBS. The LXC container is installed simply by using the Proxmox helper script bookstack.

1.Enter the Shell of your Proxmox machine

2. Copy paste the following and follow the installation steps

```
bash -c "$(curl -fsSL https://raw.githubusercontent.com/community-scripts/ProxmoxVE/main/ct/bookstack.sh)"
```

3. If you want this documentation page to be viewed publicly it is recommended to host the container on the DMZ VLAN for security purposes.

4. Also make sure you allow root access to your container with a password such that you can configure it to your needs after installation.

5. After installation you can check if your Bookstack instance is working by going to IP-Address-of-container:6875. The first time login uses the following credentials

- Username: admin@admin .com
- Password: password

6. If all good, you can go Inside the container to find the configuration file at

```
nano /opt/bookstack/.env
```

7. Once in there, assuming you want to make the wiki publicly accessible you can set the website name of your wiki by adjusting the value at

```
APP_URL=https://wiki.example.org
```

8. It is assumed that you have set your reverse proxy settings correctly with any kind of proxy server. For ease of use we recommend the Caddy plugin for OPNsense which is discussed at ------> INSERT.

5. Sorting Rules

Now Bookstack by default sorts with