Docker Commands

Build and Run a Container

Build the Docker image

```
docker build -t <image-name> .
```

Run the container:

```
docker run -d --name <container-name> <image-name>
```

You can verify it’s working by checking the logs with:

```
docker logs <container-name>
```

Or by viewing the cron log specifically:

```
docker exec -it <container-name> tail -f /var/log/cron.log
```

Reverse Proxy with Nginx

To install and configure Nginx with HTTPS support (SSL/TLS) on your Linux server, follow these steps. I'll outline the process assuming you're setting up Nginx on a Debian/Ubuntu system. Adjust commands and paths as needed for other distributions.

Step 1: Install Nginx

First, ensure your package lists are up-to-date, then install Nginx:

```
sudo apt update
sudo apt install nginx
```

Step 2: Obtain SSL/TLS Certificates

You can obtain SSL/TLS certificates for your domain using Let's Encrypt, which provides free certificates. Here’s how to set it up with Certbot, a tool for automatically managing Let's Encrypt certificates:

Install Certbot

```
sudo apt install certbot python3-certbot-nginx
```

Step 3: Configure Nginx for HTTPS

1. **Configure Nginx**
   Create a new configuration file for your domain under Nginx's sites-available directory:

   ```
   sudo nano /etc/nginx/sites-available/<domain>
   ```

   Example Nginx configuration for HTTPS:

   ```
   server {
       listen 80;
       server_name <domain>;
   ```

```
    location / {
        proxy_pass http://localhost:<port>;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}
```

   Replace `<domain>` with your actual domain name and adjust `proxy_pass` to point to your BookStack Docker container.

2. **Enable the Site**
   Create a symbolic link to enable the site in Nginx:

   ```
   sudo ln -s /etc/nginx/sites-available/<domain> /etc/nginx/sites-enabled/
   ```

3. **Test Nginx Configuration**
   Verify the Nginx configuration for syntax errors:

   ```
   sudo nginx -t
   ```

4. **Reload Nginx**
   Apply the new configuration:

   ```
   sudo systemctl reload nginx
   ```

Step 4: Obtain SSL/TLS Certificates with Certbot

Run Certbot to obtain SSL certificates for your domain (`<domain>`):

```
sudo certbot --nginx -d <domain>
```

Follow the prompts to set up HTTPS for your domain. Certbot will automatically configure Nginx with SSL/TLS settings and handle certificate renewal.

Step 5: Verify HTTPS Setup

Access `https://<domain>` in your web browser to verify that Nginx is correctly serving your BookStack application over HTTPS.

Notes:

- **Firewall**: Ensure ports 80 (HTTP) and 443 (HTTPS) are open in your firewall.
- **Security**: Regularly update Nginx and renew SSL certificates before expiry.
- **Backup**: Maintain backups of your Nginx configurations and SSL certificates.

This setup ensures secure access to your BookStack application with HTTPS, enhancing data security and user trust.

Enter a Container as Root

```
docker exec -u 0 -it <container_name_or_id> /bin/bash
```

Migrate Managed Volume to Host Volume

Migrating a managed volume into a host volume typically involves these steps, assuming you're working with Docker or a similar containerization platform:

1. **Stop Containers**: First, stop any containers that are currently using the managed volume. This ensures that no data is being actively written to the volume during migration.
2. **Inspect Volume Configuration**: Identify the current configuration of your managed volume. This could involve checking Docker Compose files or Docker commands to understand how the volume is mounted and used.
3. **Copy Data**: Copy the data from the managed volume to the desired host directory where you want to store the volume data permanently. You can use utilities like `cp` or `rsync` depending on your operating system and setup.
4. **Update Docker Configuration**:
   - Modify your Docker configuration (like Docker Compose file or Docker command) to use a bind mount instead of the managed volume. For example, change:

     ```
     volumes:
       - my_managed_volume:/path/in/container
     ```

     to

     ```
     volumes:
       - /host/path:/path/in/container
     ```

     Replace `/host/path` with the path on your host where you copied the data.
5. **Start Containers**: Start your containers again with the updated configuration. They should now use the host volume instead of the managed volume.
6. **Verify**: Ensure that the containers start correctly and that the data is accessible and functioning as expected from the host volume.

By following these steps, you can effectively migrate from a managed volume to a host volume in your Docker environment. If you have specific tools or configurations in use (like Docker Compose or Kubernetes), adjust the steps accordingly to fit your setup.

User Temporary Container for Copying Data

Copying data from a managed volume in Docker depends on where the data is stored and how it's managed by your Docker setup. Here's how you can generally approach it:

1. **Identify the Volume**: Determine the name of the managed volume in Docker. You can list all volumes using `docker volume ls`.
2. **Inspect Volume Details**: Use `docker volume inspect <volume_name>` to get more details about the volume, including its mount point on the host system.
3. **Copy Data**:
   - If you have access to the host system where Docker is running, you can directly access the volume's mount point. Use `docker volume inspect` to find the mount point path on the host.
   - If you don't have direct access or prefer to use Docker commands, you can use a temporary container to mount the volume and copy its contents:

     ```
     docker run --rm -v <volume_name>:/source -v /host/path:/destination busybox cp -a /source/.
     /destination
     ```

     Replace `<volume_name>` with the name of your managed volume, `/host/path` with the path on your host where you want to store the volume data, and `/destination` within the container where you want to copy the data.
4. **Verify**: After copying, ensure that the data has been successfully transferred to the host directory (`/host/path`).
5. **Update Docker Configuration**: Modify your Docker configuration (like Docker Compose file or Docker command) to use the host volume (bind mount) instead of the managed volume, as described in the previous response.

By following these steps, you can effectively copy data from a managed volume in Docker to a host volume, ensuring data continuity and accessibility for your containers.