Supernote-Private-Cloud-Manual-Deployment-Method-Using-Docker-Containers

- **Supernote Private Cloud Container Port Usage Guide**
  - **Container Port 8080:** Port 8080 within the container is listened to by Nginx, which acts as a reverse proxy, forwarding requests to the supernote-service container to provide synchronization and web management interface services. To access the private cloud using a browser or Supernote device, this port must be mapped to the Docker host to allow access from outside the Docker host. For instructions on publishing applications from within Docker containers, see the "Exposing Ports" section (see "Host Port Mapping 19072").
  - **Container Port 443:** Port 443 within the container is listened to by Nginx, and it works in the same way as port 8080. Port 443 provides HTTPS service by default. Please refer to the latest deployment manual for instructions on configuring HTTPS.
  - **Container Port 18072:** This port is primarily used for automatic synchronization with supernote-service
  - **Container Port 9888:** This is a Node.js frontend service used to provide the Supernote Private Cloud web management page. It provides the service through port 8080 within the container.
  - **Host Port Mapping 19072:** This port is primarily used to map port 8080 of the supernote-service container to the Docker host currently deploying the Supernote private cloud. This port mapping allows you to access the private cloud service deployed on the Docker host, enabling browser-based management and configuration, as well as synchronization between your Supernote device and the private cloud. This mapping opens a port (e.g., 19072) on the Docker host and creates a rule that forwards all traffic destined for that host port directly to the specified container's internal port (e.g., 8080). For more information on port exposure, please refer to the official Docker documentation: https://docs.docker.com/engine/containers/run/#exposed-ports
  - **Host Port Mapping 19443:** This port is primarily used to map port 443 of the supernote-service container to the Docker host currently deploying the Supernote private cloud. This port mapping allows you to access the private cloud service deployed on the Docker host via a web page over HTTPS (Supernote's self-signed certificate, browser untrusted), enabling browser-based management and configuration, as well as synchronization between Supernote devices and the private cloud. This mapping opens a port (e.g., 19443) on the Docker host and creates a rule that forwards all traffic destined for that host port directly to the specified container's internal port (e.g., 443). For more information on port exposure, please refer to the Docker official documentation: https://docs.docker.com/engine/containers/run/#exposed-ports

    1. Why does the supernote-service container need to expose port 8080?

        1. Port 8080 within the container is listened to by Nginx, which provides reverse proxy services to forward requests to the supernote-service container for synchronization services and the web management interface. To enable browsers or Supernote devices to access the private cloud, this port must be mapped to the Docker host for external access.

    2. Docker's official documentation explicitly states: “By default, the container runtime does not expose any ports to the host. To access a port listening inside a container from the host, you must publish the port.” Port publishing is typically performed using the format `-p {host_port}:{container_port}`. For reference: https://docs.docker.com/engine/containers/run/#exposed-ports
2. Why is it necessary to map ports 443 and 8080 inside the container to ports 19072 and 19443 on the Docker host?
    1. Supernote's private cloud must meet the needs of users who cannot use reverse proxies. Therefore, we need to provide a default port on the Docker host as the access point for the private cloud container service. Port mapping (e.g., `-p 19072:8080` and `-p 19443:443`) is the standard practice recommended by Docker: it opens the specified ports (e.g., 19072, 19443) on the host and forwards all traffic destined for that port to the container's internal ports (e.g., 8080, 443). This is the standard and necessary method for accessing container services from outside the host. For reference: https://docs.docker.com/engine/containers/run/#exposed-ports
3. Why does Supernote use ports 19072 and 19443 by default instead of ports 80 or 443 when publishing applications within a container?
    1. Supernote's private cloud currently enables HTTPS service by default. Ports 80 and 443 are commonly used service ports. Directly mapping ports 8080 and 443 inside the container to ports 80 or 443 on the Docker host can easily lead to port conflicts, causing service unavailability. Therefore, we specify port 19072 as the default HTTP mapping port and 19443 as the default HTTPS mapping port to reduce the risk of conflicts and ensure service stability.
    2. To customize the port, follow these steps:
        1. **For users deploying using Docker:** directly modify the Docker entries `-p <local http port>:8080` and `-p <local https port>:443`.
        2. **For users deploying using Docker Compose:** Simply modify the `HTTPS_PORT` and `HTTP_PORT` in the `.env` file to the ports you need for external access.
        3. **For users deploying using the install.sh script:** Currently, ports 19072 and 19443 are fixed as the default entry points for external connections. If you need to use HTTPS, please refer to the reverse proxy configuration instructions.
4. Why can't Supernote provide trusted HTTPS service?
    1. The Supernote team cannot issue trusted SSL certificates for every user-deployed private cloud instance. Certificate issuance requires verification from a trusted certificate authority and proof of domain ownership, which is impractical for private cloud services distributed across numerous user-owned servers. Therefore, Supernote provides sample configurations for Nginx and NAS reverse proxies for users requiring HTTPS functionality, and provides HTTPS setup instructions in the deployment manual.
    2. We have already integrated support for untrusted certificates into our service by default and enabled SSL/TLS configuration by default. If you need to configure your trusted certificate for your private cloud service, please refer to the following deployment methods. We provide deployment documentation for both `docker run` and `docker-compose` methods, both of which include instructions on how to configure HTTPS.

5. We will provide further updates and explanations regarding private cloud-related questions in our FAQ section.

- **Host Port Mapping 18072:** This port is primarily used to map the automatic synchronization functionality within the supernote-server container to the current host. Through this port, you can use the automatic synchronization functionality implemented on the Supernote device via the WebSocket protocol; this port cannot currently be customized. If you are certain you will not use the automatic synchronization functionality, you can choose not to open this port. For detailed instructions on port configuration, please refer to the official Docker documentation.: https://docs.docker.com/engine/containers/run/#exposed-ports

    1. Why is port 18072 required?

        1. Port 18072 enables automatic synchronization between Supernote devices and private clouds. If you require this automatic functionality, you must expose this port on your Docker host. If you confirm you will not use automatic synchronization, you may close this port.
        To disable it:

            1. Users who deployed via Docker or Docker Compose can directly remove the `-p 18072:18072` configuration.
            2. For users deploying using the install.sh script: the port is currently fixed at 18072 and is enabled by default.

Container Component Description

- mariadb: Deploy using the officaial Mariadb 10.6.24 image to store user registration,file synchronization,and other information
- redis: Deploy using the official Redis 7.4.7 image to store login information and other data. We recommend persisting the Redis data directory locally. This will be addressed in the Redis deployment section of the subsequent deployment documentation.
- notelib: The officaial Supernote note conversion program,built on Ubuntu 24.04
- supernote-service: Supernote Private Cloud Program,built on Ubuntu 24.04

Create the installation directory

example:

```
    mkdir -pv /data/supernote
```

Download database file

example:

```
    cd /data/supernote && curl -O https://supernote-private-
cloud.supernote.com/cloud/supernotedb.sql
```

create data persistence path

example:

```
    mkdir -pv {sndata/cert,supernote_data}
```

Deploying a Supernote private cloud using Docker Compose

Download link for docker-compose configuration manifest: docker-compose.yml

- After downloading, please do not run this file directly. You can continue to refer to this document to understand what settings you need to make to successfully deploy your private cloud using docker-compose.

Create a variable file

You need to create a variable file in the installation directory to set your private cloud database configuration, SSL configuration, and other related information.

Variable file configuration instructions:

- `DB_HOSTNAME="Mariadb Container Name":` Configure the container name used by the Supernote private cloud application to connect to the database service. This name should be the MariaDB container name so that the container can correctly resolve data. The private cloud application uses this name to connect to the database service.
- `MYSQL_ROOT_PASSWORD="Mariadb ROOT Password":` Set your database administrator password and keep it safe.
- `MYSQL_DATABASE="supernotedb":` Please keep the default value (supernotedb).
- `MYSQL_USER="Username":` Configure the username used to connect the Supernote private cloud application to the database service. The private cloud application will use this username to connect to the database service.
- `MYSQL_PASSWORD="":` This is the database user password you set for the Supernote Private Cloud application during the database service installation. The private cloud uses this password to connect to the database service.
- `REDIS_PASSWORD="Redis Password":` This is the Redis password you set for your Supernote private cloud application during the Redis service installation. The private cloud application uses this password to connect to the Redis service.
- `REDIS_HOST="Redis Container Name":` Configure the container name that the Supernote private cloud application uses to connect to the Redis service. This name should match the Redis container name to ensure correct container name resolution. The private cloud application will use this name to connect to the Redis service.
- `REDIS_PORT="6379":` Please configure the port for your Redis container service. This port will be used by the private cloud application to connect to Redis within the container. This port does not need to be exposed externally.
- `HTTP_PORT="local http port":` Please configure the external connection service port for your private cloud container's Docker host. For details, refer to “Host Port Mapping 19072”.

  How to enable HTTPS on a custom domain

  If you do not need to enable HTTPS using a custom domain, please remove the relevant variable settings in the "Enable HTTPS Settings" option in the following deployment example; and you also need to manually remove the configuration item `-./sndata/cert:/etc/nginx/cert` related to the `supernote-service` container configuration in the `docker-compose.yml` configuration. Otherwise, the private cloud may fail to start.

1. Before enabling HTTPS, please ensure that port 443 on your Docker host is not already in use. HTTPS uses this port by default; using other ports will result in your browser displaying an untrusted certificate warning.
2. Ensure your SSL certificate is valid and matches your domain name; otherwise, your browser will display an untrusted certificate warning.
3. Confirm that your domain name can be resolved correctly by DNS.

4. To enable HTTPS in your private cloud, please add the following settings to your `.env` file.

- `DOMAIN_NAME="test-private-cloud.supernote.com":`
- `HTTPS_PORT="local https port":` Please configure the HTTPS port that your Docker host needs to use.
- `SSL_CERT_NAME="your domain ssl cert name":` Please set the HTTPS certificate public key name (usually ending with `.crt`, `.pem`, etc.). If this configuration is not set, the default name `server.crt` will be used.
- `SSL_KEY_NAME="your domain ssl key name":` Please set the HTTPS certificate private key name (usually ending with `.key`). If this configuration is not set, the default name `server.key` will be used.
- Add the following setting to the supernote-service container configuration section of your docker-compose.yml file: `- ./sndata/cert:/etc/nginx/cert`. This configuration already exists in your docker-compose.yml file if you haven't modified it.

```
# Example
touch $(install_dir)/.env

# Please set your variable configuration information in the .env file, as shown
in the example below:
cat .env
# Dtabases settings
DB_HOSTNAME="supernote-mariadb"
MYSQL_ROOT_PASSWORD="MysqlrootPassword"
MYSQL_DATABASE="supernotedb"
MYSQL_USER="supernote"
MYSQL_PASSWORD="supernotedbpassword"

# Redis setings
REDIS_HOST="supernote-redis"
REDIS_PASSWORD="supernoteredispassword"
REDIS_PORT="6379"

# supernote-service SSL setitings
# If you do not need to enable HTTPS, no additional configuration is required.
# For detailed HTTPS configuration information, please refer to the deployment
documentation.
DOMAIN_NAME="supernote.xxxx.com"
HTTPS_PORT="443"
HTTP_PORT="19072"
SSL_CERT_NAME="xxxx.com.crt"
SSL_KEY_NAME="xxxx.com.key"
```

Modify docker-compose file

1. By default, all images in the downloaded docker-compose file, except for the database and Redis images, are provided by Supernote.
2. We have not set image tags in this file; you will need to modify the tags yourself.

Image acquisition address: Docker Hub

```
# Modify the notelib container image tag to the latest version.
notelib:
  image: docker.io/supernote/notelib:{image_tag}
  ...
  ...
# Modify the supernote-service container image label to the latest version.
supernote-service:
  image: docker.io/supernote/supernote-service:{image_tag}
```

3. After the above custom configuration, you can now deploy private cloud applications using docker-compose.

```
# Check if the configuration is correct.
docker-compose -f docker-compose.yml config -q

# Deploy Supernote Private Cloud
docker-compose up -d
```

How to upgrade Supernote Private Cloud components

1. Before updating, please ensure that you have set up the persistent directory for the relevant containers according to the above settings.
2. To ensure data integrity, we recommend backing up data-related directories, including the local persistent directory for the database service, the local persistent directory for Supernote data files (default: /{install_dir}/supernote_data), and the local persistent directory for the Redis service.
    - For instructions on how to update the database configuration, please scroll to the end of the document to view the section "**How to upgrade MariaDB or update services**".
    - To upgrade your private cloud application, you can directly modify the image tag for the corresponding service in the docker-compose file, and then manually execute the docker-compose file to complete the upgrade.

Deploying a Supernote private cloud using docker run

create a custom network

example:

- Create a custom network named supernote-net and ensure all Supernote Private Cloud services run within this network.

    ```
    docker network create supernote-net
    ```

Install the database service

`--name:` Please set the name of your MariaDB database container.

`--network:` Please configure the custom network name for the container running your private cloud application.

`MYSQL_ROOT_PASSWORD:` Please set the variable value to your database administrator password.

`MYSQL_DATABASE:` Please keep the default value (supernotedb).

`MYSQL_USER`: Please configure the user for the private cloud application to connect to the database.

`MYSQL_PASSWORD:` Please set the user password for the private cloud application to connect to the database.

`-v "local store:/var/lib/mysql":` Set the local persistence directory. Please note that to prevent data loss, we strongly recommend setting a local persistence path for the database service.

`-v $(pwd)/supernotedb.sql:/docker-entrypoint-initdb.d/supernotedb.sql:ro:` Please map the database initialization files required by SuperNote Private Cloud to the database container to enable database initialization.

- Run the following command to quickly build a database service for Supernote Private Cloud.

```
docker run -d \
--name supernote-mariadb \  # Please set the name of your MariaDB database
container.
--network supernote-net \ # Please configure the custom network name for the
container running your private cloud application.
-e MYSQL_ROOT_PASSWORD="Set database administrator password" \
-e MYSQL_DATABASE=supernotedb \
-e MYSQL_USER="Set up the database user for the Supernote Private Cloud program"
\
-e MYSQL_PASSWORD="Set the database user password for the Supernote Private
Cloud program" \
-v "Local persistence directory for database data":/var/lib/mysql \
-v $(pwd)/supernotedb.sql:/docker-entrypoint-initdb.d/supernotedb.sql:ro \
--restart=unless-stopped \
mariadb:10.6.24
```

Install the Redis service

`--name:` Please set your Redis container name.

`--network:` Please configure the custom network name for the container running your private cloud application.

`--requirepass '':` Please set a password for the Supernote Private Cloud application to connect to the Redis service.

`-v:` Set the local persistence directory

- Run the following command to quickly set up a Redis service for the Supernote Private Cloud application. Please note to modify --requirepass ‘Set Redis Password’.

```
docker run -d \
--name "Please set your Redis container name." \
--network supernote-net \
-v "Local Persistence Directory for Redis Data":/data \
--restart=unless-stopped \
redis:7.4.7 \
redis-server --requirepass 'Set Redis Password' --dir /data --dbfilename
dump.rdb
```

Install the Notelib Service

- Run the following command to quickly build a Supernote Private Cloud note conversion program.

  `ImageTags:` Please obtain the latest notelib image from Docker Hub.

```
docker run -d --name notelib --network supernote-net
docker.io/supernote/notelib:{ImageTag}
```

Install the Supernote Private Cloud program

`--name:` Please set the name for the container running your private cloud application.

`--network:` Please configure the custom network name for the container running your private cloud application.

`-p 18072:18072:` If you need to use the automatic synchronization feature, please open this port.

`-p "local http port:8080":` Please configure the external connection service port for your private cloud container's Docker host. For details, refer to “Host Port Mapping 19072”.

`-p local https port:443:` Please configure the HTTPS port that your Docker host needs to use.

`-v $(pwd)/sndata/recycle:/home/supernote/recycle:` Set the local persistent path for your Recycle Bin

`-v $(pwd)/supernote_data:/home/supernote/data:` Set the local persistent path for your notes and synchronized files.

`-v $(pwd)/sndata/logs/cloud:` Set the local persistence path for frontend program logs

`-v $(pwd)/sndata/logs/app:/home/supernote/logs:` Set the local persistence path for private cloud service logs

`-v $(pwd)/sndata/logs/web:/var/log/nginx:` Set the local persistence path for web service logs

`-v $(pwd)/sndata/convert:/home/supernote/convert:` Set the local persistent path for notes after rotation

`-v $(pwd)/sndata/cert:/etc/nginx/cert:` Configure your SSL certificate path. Please rename the SSL certificate to `server.crt` and `server.key`. If this is not set, a Supernote self-signed certificate will be used by default.

`-e DB_HOSTNAME`: Please set your MariaDB container name. Please refer to the name specified in the `--name` field during database service installation.

`-e MYSQL_DATABASE:` Please keep the default value (supernotedb).

`-e MYSQL_USER:` The database user you set for the Supernote Private Cloud program during database service installation. Please refer to the MYSQL_USER variable in the “Install the database service” section.

`-e MYSQL_PASSWORD:` The database user password you set for the Supernote Private Cloud program during database service installation. Please refer to the MYSQL_PASSWORD in the “Install the database service” section.

`-e REDIS_HOST:` Please set your Redis container name. Refer to the name specified in the `--name` field during the Redis service installation process.

`-e REDIS_PORT:` Please configure the port for your Redis container service. This port will be used by the private cloud application to connect to Redis within the container. This port does not need to be exposed externally.

`-e REDIS_PASSWORD:` The Redis password you set for the Supernote Private Cloud program during Redis service installation. Please refer to the --requirepass option in the “Install the Redis service” section.

`-v /etc/localtime:/etc/localtime:ro`: Set the local time zone for the Supernote Private Cloud program.

`-e DOMAIN_NAME="your.domain.com":` sets the domain name used for HTTPS. Replace `your.domain.com` with your actual domain name, which must match the subject name in the SSL certificate.

`-e SSL_CERT_NAME="your domain ssl cert":` Please set the HTTPS certificate public key name (usually ending with `.crt`, `.pem`, etc.). If this configuration is not set, the default name `server.crt` will be used.

`-e SSL_KEY_NAME="your domain ssl key":` Please set the HTTPS certificate private key name (usually ending with `.key`). If this configuration is not set, the default name `server.key` will be used.

`ImageTag:` Please obtain the latest Supernote Private Cloud image from Docker Hub.

How to enable HTTPS on a custom domain

- If you do not need to enable HTTPS using a custom domain, please remove the relevant variable settings in the "Enable HTTPS Settings" option in the following deployment example; otherwise, the private cloud may fail to start.

1. Before enabling HTTPS, please ensure that port 443 on your Docker host is not already in use. HTTPS uses this port by default; using other ports will result in your browser displaying an untrusted certificate warning.
2. Ensure your SSL certificate is valid and matches your domain name; otherwise, your browser will display an untrusted certificate warning.
3. Confirm that your domain name can be resolved correctly by DNS.
4. The settings for enabling HTTPS are as follows::
    1. `-p local https port:443:` Please configure the HTTPS port that your Docker host needs to use.
    2. `-v $(pwd)/sndata/cert:/etc/nginx/cert:` Configure the SSL certificate mount path. Upload your SSL certificate to the `$(pwd)/sndata/cert` directory on your Docker host. If this directory does not exist, you can create it manually using `mkdir -pv $(pwd)/sndata/cert`. Custom certificate directories are not currently supported.

3. `-e DOMAIN_NAME="your.domain.com":` sets the domain name used for HTTPS. Replace `your.domain.com` with your actual domain name, which must match the subject name in the SSL certificate.
4. `-e SSL_CERT_NAME="your domain ssl cert":` Please set the HTTPS certificate public key name (usually ending with `.crt`, `.pem`, etc.). If this configuration is not set, the default name `server.crt` will be used.
5. `-e SSL_KEY_NAME="your domain ssl key":` Please set the HTTPS certificate private key name (usually ending with `.key`). If this configuration is not set, the default name `server.key` will be used.

Use the following commands to quickly run a Supernote private cloud instance with HTTPS capabilities. Note that you will need to modify the persistence settings and variables above according to your own configuration.

```
docker run -d \
--name supernote-service \
--network supernote-net \
-p 18072:18072 \
-p "local http port":8080 \
-p "local https port:443" \
-v $(pwd)/sndata/recycle:/home/supernote/recycle \
-v $(pwd)/supernote_data:/home/supernote/data \
-v $(pwd)/sndata/logs/cloud:/home/supernote/cloud/logs \
-v $(pwd)/sndata/logs/app:/home/supernote/logs \
-v $(pwd)/sndata/logs/web:/var/log/nginx \
-v $(pwd)/sndata/convert:/home/supernote/convert \
-v $(pwd)/sndata/cert:/etc/nginx/cert \
-v /etc/localtime:/etc/localtime:ro \
-e DB_HOSTNAME="Please set your MariaDB container name." \
-e MYSQL_DATABASE=supernotedb \
-e MYSQL_USER="Enter the private cloud database user you set when creating the
database container." \
-e MYSQL_PASSWORD="Enter the private cloud database user password you set when
creating the database container." \
-e REDIS_HOST="Please set your Redis container name." \
-e REDIS_PORT="Please configure the port for your Redis container." \
-e REDIS_PASSWORD="Enter the password you set when creating the Redis
container." \
-e DOMAIN_NAME="your.domain.com" \
-e SSL_CERT_NAME="your domain ssl cert name" \
-e SSL_KEY_NAME="your domain ssl key name" \
docker.io/supernote/supernote-service:{ImageTag}
```

How to upgrade Supernote Private Cloud components

1. Before updating, please ensure that you have set up the persistent directory for the relevant containers according to the above settings.
2. To ensure data integrity, we recommend backing up data-related directories, including the local persistent directory for the database service, the local persistent directory for Supernote data files (default: /{install_dir}/supernote_data), and the local persistent directory for the Redis service.

- **How to upgrade the container image used by Supernote private cloud?**

  Note: This process does not include upgrading the MariaDB database service.

1. Please ensure that the persistence directory used by your latest version of the container is consistent with the path you used during your initial installation.
2. Remove the containers you need to upgrade from the Docker host.
3. Obtain the latest version image from Docker Hub, and then upgrade your container to the latest version by reinstalling. By default, we only provide updates for notelib and supernote-service.

- **How to upgrade MariaDB or update services**

  1.For MariaDB database services, there are two update methods.

  1. **Upgrade MariaDB version:** Directly upgrading the MariaDB database version is generally not recommended for stable systems, as it can easily cause compatibility issues and prevent the private cloud from functioning properly. We are currently testing which MariaDB versions are supported by private cloud providers.
  2. **Update the supernotedb.sql script:** Typically, when updating the MariaDB database, we only need to update the SQL script used by supernotedb (see "Download database file" for instructions on downloading the SQL script). Regarding updating the SQL script, please note that you should ensure the data directory used by the MariaDB container is persisted to a local path on the Docker host.

     1. **The update method is as follows (Execute SQL script to update is recommended):**

        1. **Using the method of rebuilding containers:** You can download the SQL script to the installation directory of your Docker private cloud host to replace the supernotedb.sql script downloaded during the last installation. To do this, you will need to remove the previously running MariaDB container and then reinstall it using the "Install Database Service" command.
        2. **Execute SQL script to update:** You can download the SQL script to the installation directory of your Docker host's private cloud to replace the supernotedb.sql script you downloaded during your last installation. After the replacement is complete, you can manually execute the update directly in the container using the `docker exec` command. The specific steps are as follows:

```
# To avoid file system caching issues, the MariaDB container
needs to be restarted.
docker restart "Mariadb Container Name"

# Enter the container using the following command.
docker exec -ti "Mariadb Container Name" /bin/bash

# Check if the latest SQL script has been successfully mounted.
ls /docker-entrypoint-initdb.d/supernotedb.sql

# Update SQL script:
Method 1: mysql -u "DB Username" -p supernotedb < /docker-
entrypoint-initdb.d/supernotedb.sql

#Enter data user password
Method 2:
mysql -u "DB Username" -p
sql: use supernotedb;
sql: source /docker-entrypoint-initdb.d/supernotedb.sql
```