Migrate From Self-Hosted to EMS

Notes

Before starting with this guide, please contact EMS support from https://ems.element.io/support or by emailing ems-support@element.io

- Except where specified, you should be able to just copy-paste each command in succession.
- Please do not change any file names anywhere.

Preparation

This section outlines what you should do ahead of the migration in order to ensure the migration goes as quickly as possible and without issues.

- At the latest 48 hours before your migration is scheduled, set the TTL on any DNS records that need to be updated to the lowest allowed value.
- Upgrade your Synapse to the same version as EMS is running. Generally this will be the latest stable release. https://element.ems.host/_matrix/federation/v1/version is a good indicator, but confirm version with your EMS contact.
  - This is not required, but if your Synapse version is not the same as the EMS version, your migration will take longer.
- Check the size of your database and report to your EMS contact:
  - PostgreSQL: Connect to your database and issue the command `\l+`
  - SQLite: `ls -lah /path/to/homeserver.db`
- Check the size of your media repository and report to your EMS contact.
  - Synapse Media Store: `du -hs /path/to/synapse/media_store/`
  - Matrix Media Repo: https://github.com/turt2live/matrix-media-repo/blob/master/docs/admin.md#per-server-usage
- If you are using SQLite instead of PostgreSQL, you should port your database to PostgreSQL by following this guide before dumping your database and sending to your EMS contact.
  - This step is not required, but will speed up your migration.

SSH to your matrix server

You might want to run everything in a `tmux` or a `screen` session to avoid disruption in case of a lost SSH connection.

Generate password for gpg encryption

```
pwgen -s 64 1
```

Alternatively, you can use our GPG key. Note, this expires on 2023-04-28, if this is soon, please talk to your EMS contact.

ems-support-public.pgp

GPG

If `gpg` is being uncooperative, use the command `gpgconf --kill gpg-agent`.

Create a folder to store everything

```
mkdir -p /tmp/synapse_export
cd /tmp/synapse_export
```

The guide from here on assumes your current working directory is `/tmp/synapse_export`.

Set restrictive permissions on the folder

If you are working as root: (otherwise set restrictive permissions as needed):

```
chmod 000 /tmp/synapse_export
```

Copy Synapse config

Copy the following files and send to EMS Support:

- Your Synapse configuration file (usually `homeserver.yaml`)
- Your message signing key.
  - This is stored in a separate file. See the Synapse config file for the path. The variable is `signing_key_path` https://element-hq.github.io/synapse/latest/usage/configuration/config_documentation.html#signing_key_path

Stop Synapse

DO NOT START IT AGAIN AFTER THIS
Doing so can cause issues with federation and inconsistent data for your users.

While you wait for the database to export or files to transfer, you should edit or create the well-known files and DNS records to point to your EMS host. This can take a while to update so should be done as soon as possible in order to ensure your server will function properly when the migration is complete.

Database export

PostgreSQL

Dump, compress and encrypt

Replace:

- `<dbhost>` (ip or fqdn for your database server)
- `<dbusername>` (username for your synapse database)
- `<dbname>` (the name of the database for synapse)

```
pg_dump -O -h <dbhost> -U <dbusername> -d <dbname> | gzip > customer_db_export.sql.gz
gpg --symmetric --no-symkey-cache customer_db_export.sql.gz
rm customer_db_export.sql.gz
```

If required, split into smaller files

Please only do this if you have a slow connection and are worried about transferring a single large file.

```
split -b 100m customer_db_export.sql.gz.gpg customer_db_export.sql.gz.gpg.part-
rm customer_db_export.sql.gz.gpg
```

SQLIte

Compress and encrypt

```
tar -zcvf homeserver.db.tar.gz /path/to/homeserver.db
gpg --symmetric --no-symkey-cache homeserver.db.tar.gz
rm homeserver.db.tar.gz
```

If required, split into smaller files

Please only do this if you have a slow connection and are worried about transferring a single large file.

```
split -b 100m homeserver.db.tar.gz homeserver.db.tar.gz.part-
rm homeserver.db.tar.gz
```

Media export

If you are using SQLIte as database

Skip ahead to and follow Backup media export.

Download the export tool

Download the latest version of `export_synapse_for_import-linux-x64` (or `export_synapse_for_import-win-x64.exe`) from https://github.com/turt2live/matrix-media-repo/releases

```
wget https://github.com/turt2live/matrix-media-repo/releases/download/vx.x.x/export_synapse_for_import-linux-x64
chmod +x export_synapse_for_import-linux-x64
```

Run the export

Replace:

- `<dbhost>` (ip or fqdn for your database server)
- `<dbname>` (the name of the database for synapse)
- `<dbusername>` (username for your synapse database)
- `/path/to/synapse/media_store` (the path to where synapse stores your media)
- `<yourdomain.tld>` (the domain for your server. this is the part that is in your usernames)

```
./export_synapse_for_import-linux-x64 -h
./export_synapse_for_import-linux-x64 -dbHost <dbhost> -dbPort 5432 -dbName <dbname> -dbUsername <dbusername> -mediaDirectory /path/to/synapse/media_store -serverName <yourdomain.tld> -destination ./customer_media_export
mv logs customer_media_export
mv media-repo.yaml customer_media_export
rm export_synapse_for_import-linux-x64
```

Compress and encrypt

```
tar -zcvf customer_media_export.tar.gz customer_media_export
gpg --symmetric --no-symkey-cache customer_media_export.tar.gz
rm customer_media_export.tar.gz
rm -r customer_media_export
```

If required, split into smaller files

Please only do this if you have a slow connection and are worried about transferring a single large file.

```
split -b 100m customer_media_export.tar.gz.gpg customer_media_export.tar.gz.gpg.part-
rm customer_media_export.tar.gz.gpg
```

Backup media export

Compress and encrypt

Replace * `/path/to/synapse/media_store` (the path to where synapse stores your media)

```
tar -zcvf customer_backup_media_export.tar.gz /path/to/synapse/media_store
gpg --symmetric --no-symkey-cache customer_backup_media_export.tar.gz
rm customer_backup_media_export.tar.gz
```

If required, split into smaller files

Please only do this if you have a slow connection and are worried about transferring a single large file.

```
split -b 100m customer_backup_media_export.tar.gz.gpg customer_backup_media_export.tar.gz.gpg.part-
rm customer_backup_media_export.tar.gz.gpg
```

Transfer

Download the files, then upload to the Google Drive folder shared by EMS or a location as agreed with your EMS contact.

On your local computer:

```
scp -r -P 1234 -i ~/.ssh/matrix-server youruser@1.2.3.4:/tmp/synapse_export /some/local/folder
```

Cleanup

We strongly recommend that you leave the export and Synapse untouched until the import is finished and everything is verified working.

Note on users and Element

Element does have support for changing the delegated homeserver URL. All your users will have to sign out and sign in again to Element. You should ensure everyone has Key Backup configured and working.

Your users will not be able to decrypt messages send in their encrypted rooms while your server is offline for the migration.

Force logout of old sessions after migration

If you do not log out all sessions for your users before the migration, you can force this later. Below is a sample config file for `nginx` that tells all clients trying to connect to it to sign out.

Note that the headers are important, otherwise this will not work one one or more of the Element clients. Valid HTTPS is required.

This is not tested on any other Matrix clients, but it should work in theory if the client follows the Matrix Spec.

```
server {
    listen [::]:443 ssl http2;
    listen 443 ssl http2;

    server_name old.delegated.url.com;

    location / {
        if ($request_method = 'OPTIONS') {
            add_header 'Access-Control-Allow-Origin' '*';
            add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS';
            add_header 'Access-Control-Allow-Headers' 'authorization,DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type,Range';
            add_header 'Access-Control-Max-Age' 1728000;
            add_header 'Content-Type' 'text/plain; charset=utf-8';
            add_header 'Content-Length' 0;
            return 204;
        }
        if ($request_method = 'POST') {
            add_header 'Access-Control-Allow-Origin' '*' always;
            add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS' always;
            add_header 'Access-Control-Allow-Headers' 'authorization,DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type,Range' always;
            add_header 'Access-Control-Expose-Headers' 'Content-Length,Content-Range' always;
        }
        if ($request_method = 'GET') {
            add_header 'Access-Control-Allow-Origin' '*' always;
            add_header 'Access-Control-Allow-Methods' 'GET, POST, OPTIONS' always;
            add_header 'Access-Control-Allow-Headers' 'authorization,DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type,Range' always;
            add_header 'Access-Control-Expose-Headers' 'Content-Length,Content-Range' always;
        }

        default_type application/json;
        return 401 '{"errcode":"M_UNKNOWN_TOKEN","error":"Server moved, please log in again."}';
    }
    ssl_session_timeout 1d;
    ssl_session_cache shared:MozSSL:10m;  # about 40000 sessions
    ssl_session_tickets off;
    ssl_protocols TLSv1.3;
    ssl_prefer_server_ciphers on;
    add_header Strict-Transport-Security "max-age=63072000" always;
    ssl_stapling on;
    ssl_stapling_verify on;

    error_log /var/log/nginx/old.delegated.url.com.error.log;
    access_log /var/log/nginx/old.delegated.url.com.access.log;

    ssl_certificate /etc/letsencrypt/live/old.delegated.url.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/old.delegated.url.com/privkey.pem;
}

# Redirect HTTP to HTTPS
server {
    listen 80;
    listen [::]:80;

    server_name old.delegated.url.com;

    if ($host = old.delegated.url.com) {
        return 301 https://$host$request_uri;
    }

    return 404;
}
```

Revision #9
Created 18 April 2022 14:19:45 by Karl Abbott
Updated 6 November 2024 12:28:43 by Kieran Mitchell Lane