

Comment installer et créer un serveur de discussion à l'aide de Matrix Synapse et Element sur Debian 12

Matrix est un standard ouvert pour les communications décentralisées et cryptées de bout en bout. Il s'agit d'un ensemble de serveurs et de services qui communiquent entre eux à l'aide d'une API standardisée qui se synchronise en temps réel. Il utilise des serveurs domestiques pour stocker les informations de compte et l'historique des discussions. Si un serveur domestique tombe en panne, les autres serveurs peuvent continuer la communication sans problème en raison de la nature de la décentralisation. Vous pouvez soit utiliser un serveur domestique Matrix hébergé par quelqu'un d'autre, soit héberger le vôtre pour garder le contrôle de vos données.

Dans ce didacticiel, vous apprendrez à installer et créer un serveur de discussion à l'aide de Synapse, une implémentation de serveur domestique de Matrix. Element est un client Web Matrix construit à l'aide du SDK Matrix React. Cela vous permettra de proposer le chat Matrix sur le web. Vous pouvez également utiliser le serveur en utilisant n'importe quel autre client Matrix de votre choix. Nous installerons également le serveur Coturn pour activer la voix et la vidéo appel. Le service Coturn est facultatif au cas où vous ne souhaitez pas l'utiliser.

Conditions préalables

1. Un serveur exécutant Debian 12.
2. Un utilisateur non sudo avec les privilèges root.
3. Le pare-feu simple (UFW) est activé et en cours d'exécution.
4. Noms de domaine complets (FQDN) pour Matrix, Element et Coturn pointant vers votre serveur. Nous utiliserons `matrix.example.com`, `element.example.com`, et `coturn.example.com` respectivement pour les 3 pratiques de gestion technique
5. Assurez-vous que tout est mis à jour.

```
$ sudo apt update && sudo apt upgrade
```

Étape 1 - Configurer le pare-feu

Avant d'installer des packages, la première étape consiste à configurer le pare-feu pour ouvrir les ports pour HTTP, HTTPS et Synapse.

Vérifiez l'état du pare-feu.

```
$ sudo ufw status
```

Vous devriez voir quelque chose comme ce qui suit.

```
Status: active
-----
To Action From
-----
OpenSSH ALLOW Anywhere
OpenSSH (v6) ALLOW Anywhere (v6)
```

Ouvrez les ports HTTP, HTTPS et Synapse dans le pare-feu.

```
$ sudo ufw allow 8448
$ sudo ufw allow http
$ sudo ufw allow https
```

Vérifiez à nouveau l'état pour confirmer.

```
$ sudo ufw status
Status: active
-----
To Action From
-----
OpenSSH ALLOW Anywhere
80/tcp ALLOW Anywhere
443 ALLOW Anywhere
8448 ALLOW Anywhere
OpenSSH (v6) ALLOW Anywhere (v6)
80/tcp (v6) ALLOW Anywhere (v6)
443 (v6) ALLOW Anywhere (v6)
8448 (v6) ALLOW Anywhere (v6)
```

Étape 2 - Installer Matrix Synapse

Ajoutez la clé Matrix GPG.

```
$ sudo wget -O /usr/share/keyrings/matrix-org-archive-keyring.gpg https://packages.matrix.org/debian/matrix-org-archive-keyring.gpg
```

Ajoutez le référentiel Matrix APT.

```
$ echo "deb [signed-by=/usr/share/keyrings/matrix-org-archive-keyring.gpg] https://packages.matrix.org/debian/ $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/matrix-org.list
```

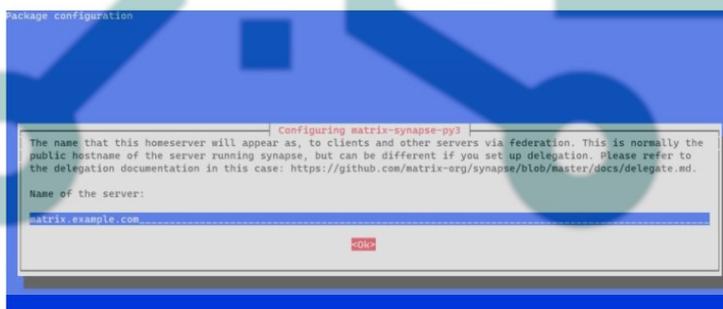
Mettez à jour la liste des référentiels système.

```
$ sudo apt update
```

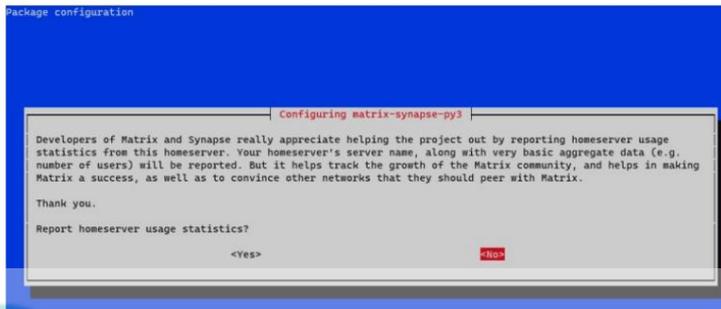
Installez Matrix Synapse.

```
$ sudo apt install matrix-synapse-py3
```

Lors de l'installation, il vous sera demandé le nom du serveur, qui fait partie de votre Matrix ID. Entrez votre nom de domaine Matrix à la place. Cela fera office d'adresse de votre serveur domestique.



Il vous sera également demandé si vous souhaitez ou non renvoyer des statistiques anonymisées sur votre serveur domestique à Matrix. Taper "N" pour refuser.



Vous pouvez modifier ces paramètres ultérieurement dans le fichier `/etc/matrix-synapse/conf.d/server_name.yaml`

Le service de Matrix Synapse est activé et démarré lors de l'installation. Vérifiez l'état du service.

```
$ sudo systemctl status matrix-synapse
? matrix-synapse.service - Synapse Matrix homeserver
Loaded: loaded (/lib/systemd/system/matrix-synapse.service; enabled; preset: enabled)
Active: active (running) since Mon 2023-10-02 07:47:43 UTC; 1min 33s ago
Process: 1558 ExecStartPre=/opt/venvs/matrix-synapse/bin/python -m synapse.app.homeserver --config-path=/etc/matrix-synapse/homeserver.yaml --config-path=/etc/matrix-synapse/conf.d/ --generate-keys (code=> Main PID: 1563 (python))
Tasks: 8 (limit: 2315)
Memory: 102.7M
CPU: 3.609s
CGROUP: /system.slice/matrix-synapse.service
??1563 /opt/venvs/matrix-synapse/bin/python -m synapse.app.homeserver --config-path=/etc/matrix-synapse/homeserver.yaml --config-path=/etc/matrix-synapse/conf.d/
Oct 02 07:47:41 lomp matrix-synapse[1558]: Generating signing key file /etc/matrix-synapse/homeserver.signing.key
Oct 02 07:47:43 lomp matrix-synapse[1563]: This server is configured to use 'matrix.org' as its trusted key server via the
Oct 02 07:47:43 lomp matrix-synapse[1563]: 'trusted_key_servers' config option. 'matrix.org' is a good choice for a key
Oct 02 07:47:43 lomp matrix-synapse[1563]: server since it is long-lived, stable and trusted. However, some admins may
Oct 02 07:47:43 lomp matrix-synapse[1563]: wish to use another server for this purpose.
Oct 02 07:47:43 lomp matrix-synapse[1563]: To suppress this warning and continue using 'matrix.org', admins should set
Oct 02 07:47:43 lomp matrix-synapse[1563]: 'suppress_key_server_warning' to 'true' in homeserver.yaml.
Oct 02 07:47:43 lomp matrix-synapse[1563]: -----
Oct 02 07:47:43 lomp matrix-synapse[1563]: Config is missing macaroon_secret_key
Oct 02 07:47:43 lomp systemd[1]: Started matrix-synapse.service - Synapse Matrix homeserver.
```

Étape 3 - Installer et configurer PostgreSQL

Nous utiliserons le référentiel APT officiel de PostgreSQL pour installer la dernière version de PostgreSQL. Exécutez la commande suivante pour ajouter la clé GPG PostgreSQL.

```
$ curl https://www.postgresql.org/media/keys/ACCC4CF8.asc | gpg --dearmor | sudo tee /usr/share/keyrings/postgresql-key.gpg >/dev/null
```

Ajoutez le référentiel APT à votre liste de sources.

```
$ sudo sh -c "echo \"deb [signed-by=/usr/share/keyrings/postgresql-key.gpg arch=amd64] http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main\" > /etc/apt/sources.list.d/pgdg.list"
```

Mettez à jour le référentiel système.

```
$ sudo apt update
```

Maintenant, vous pouvez installer PostgreSQL à l'aide de la commande suivante.

```
$ sudo apt install postgresql postgresql-contrib
```

Au moment de la rédaction de ce didacticiel, PostgreSQL 16 est la dernière version. Pour installer une version différente, modifiez la commande comme suit qui installera PostgreSQL 14 à la place.

```
$ sudo apt install postgresql-14 postgresql-contrib-14
```

Vérifiez l'état du service PostgreSQL.

```
$ sudo systemctl status postgresql
? postgresql.service - PostgreSQL RDBMS
Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; preset: enabled)
Active: active (exited) since Mon 2023-10-02 07:51:10 UTC; 13s ago
Main PID: 4001 (code=exited, status=0/SUCCESS)
CPU: 1ms
Oct 02 07:51:10 lomp systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
Oct 02 07:51:10 lomp systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
```

Vous pouvez voir que le service est activé et exécuté par défaut

connectez vous au compte postgres

```
$ sudo -su postgres
```

Créez un nouvel utilisateur de base de données et une base de données pour PostgreSQL.

```
$ createuser --prompt synapse
$ createdb --encoding=UTF8 --locale=C --template=template0 --owner=synapse synapsedb
```

Quittez le compte postgres

```
$ exit
```

Étape 4 - Installer Nginx

Pour l'environnement de production, il est recommandé d'exécuter le serveur Synapse à l'aide d'un proxy Nginx.

Debian 12 est livré avec une ancienne version de Nginx. Pour installer la dernière version, vous devez télécharger le référentiel officiel Nginx.

Importez la clé de signature de Nginx.

```
$ curl https://nginx.org/keys/nginx_signing.key | gpg --dearmor \
| sudo tee /usr/share/keyrings/nginx-archive-keyring.gpg >/dev/null
```

Ajoutez le référentiel pour la version stable de Nginx.

```
$ echo \"deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \
http://nginx.org/packages/debian \"lsb_release -cs\" nginx\" \
| sudo tee /etc/apt/sources.list.d/nginx.list
```

Mettez à jour les référentiels système.

```
$ sudo apt update
```

Installez Nginx.

```
$ sudo apt install nginx
```

Vérifiez l'installation. Puisque nous utilisons Debian, la commande suivante doit être exécutée avec

sudo autorisation.

```
$ sudo nginx -v
nginx version: nginx/1.24.0
```

Démarrez le serveur Nginx.

```
$ sudo systemctl start nginx
```

Vérifiez l'état du service.

```
$ sudo systemctl status nginx
? nginx.service - nginx - high performance web server
Loaded: loaded (/lib/systemd/system/nginx.service; enabled; preset: enabled)
Active: active (running) since Mon 2023-10-02 07:59:12 UTC; 1s ago
Docs: https://nginx.org/en/docs/
Process: 5767 ExecStart=/usr/sbin/nginx -c /etc/nginx/nginx.conf (code=exited, status=0/SUCCESS)
Main PID: 5768 (nginx)
Tasks: 2 (Limit: 2315)
Memory: 1.7M
CPU: 7ms
CGroup: /system.slice/nginx.service
        5768 nginx: master process /usr/sbin/nginx -c /etc/nginx/nginx.conf
        5769 nginx: worker process
```

Étape 5 - Installer SSL

Nous devons installer Certbot pour générer les certificats SSL gratuits proposés par Let's Encrypt. Vous pouvez soit installer Certbot à l'aide du référentiel Debian, soit récupérer la dernière version à l'aide de l'outil Snapd. Nous utiliserons la version Snapd.

Debian 12 n'est pas fourni avec Snapd installé. Exécutez la commande suivante pour installer Snapd.

```
$ sudo apt install snapd -y
```

Exécutez les commandes suivantes pour vous assurer que votre version de Snapd est à jour.

```
$ sudo snap install core
```

```
$ sudo snap refresh core
```

Installez Certbot.

```
$ sudo snap install --classic certbot
```

Utilisez la commande suivante pour vous assurer que la commande Certbot peut être exécutée en créant un lien symbolique vers le `/usr/bin` annuaire.

```
$ sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

Vérifiez la version de Certbot.

```
$ certbot --version
certbot 2.6.0
```

Exécutez la commande suivante pour générer un certificat SSL.

```
$ sudo certbot certonly --nginx --agree-tos --no-eff-email --staple-ocsp --preferred-challenges http -m name@example.com -d matrix.example.com
```

La commande ci-dessus téléchargera un certificat dans le répertoire `/etc/letsencrypt/live/matrix.example.com` de votre serveur.

Générez un certificat de groupe Diffie-Hellman.

```
$ sudo openssl dhparam -dsaparam -out /etc/ssl/certs/dhparam.pem 4096
```

Vérifiez le service de planification de renouvellement Certbot.

```
$ sudo systemctl list-timers
```

tu trouveras `snap.certbot.renew.service` comme l'un des services dont l'exécution est planifiée.

| NEXT | LEFT | LAST | PASSED | UNIT | ACTIVATES |
|--------------------------------------|---------------------------------------|------|--------|--------------------------|----------------------------|
| Mon 2023-10-02 16:33:00 UTC 8h left | - | - | - | snap.certbot.renew.timer | snap.certbot.renew.service |
| Tue 2023-10-03 00:00:00 UTC 15h left | - | - | - | dpkg-db-backup.timer | dpkg-db-backup.service |
| Tue 2023-10-03 00:00:00 UTC 15h left | Mon 2023-10-02 07:38:16 UTC 26min ago | - | - | exim4-base.timer | exim4-base.service |

Effectuez un essai à sec du processus pour vérifier si le renouvellement SSL fonctionne correctement.

```
$ sudo certbot renew --dry-run
```

Si vous ne voyez aucune erreur, vous êtes prêt. Votre certificat se renouvellera automatiquement.

Étape 6 - Configurer Matrix Synapse

Vous pouvez configurer le serveur Matrix via le fichier `/etc/matrix-synapse/homeserver.yaml` mais ce n'est pas recommandé car il est écrasé après chaque mise à jour. Pour une utilisation en production, vous devez placer les fichiers de configuration dans le dossier `/etc/matrix-synapse/conf.d`.

L'installation de Synapse a créé deux fichiers de configuration dans le dossier `/etc/matrix-synapse/conf.d`

```
$ ls /etc/matrix-synapse/conf.d
report_stats.yaml server_name.yaml
```

Créez un nouveau fichier de configuration pour la base de données et ouvrez-le pour le modifier.

```
$ sudo nano /etc/matrix-synapse/conf.d/database.yaml
```

Collez les lignes suivantes dans l'éditeur. Remplacez l'hébergement votre mot de passe champ par le mot de passe utilisateur PostgreSQL que vous avez créé à l'étape 3. Remplacez `hôte local` avec l'adresse IP de votre serveur, si vous êtes de la base de données ailleurs.

```
database:
  name: psycopg2
  args:
    user: synapse
    password: 'your-password'
    database: synapsedb
    host: localhost
    cp_min: 5
    cp_max: 10
```

Enregistrez le fichier en appuyant sur `Ctrl + X` et en entrant `Y` lorsque vous y êtes invité.

Créez une clé d'enregistrement secrète. La clé doit être sécurisée car elle permettra à quiconque d'enregistrer un nouvel utilisateur, même si l'enregistrement est désactivé.

```
$ echo "registration_shared_secret: $(cat /dev/urandom | tr -cd '[a-zum:]' | fold -w 256 | head -n 1)" | sudo tee /etc/matrix-synapse/conf.d/registration_shared_secret.yaml
registration_shared_secret: vgd73p26ZDaFEExpX4OPv45DsA2ZMAxiVZR7um9rBoBoFESmg5MSs68xAMUhwQ8Zn3NqzZMRsQxLeIFatppfne7xD2RHL16YfuiKmnNeJ1FCIQszO1SZkUVwOPyDiPe5gCCwD9cHfa3dLTzND5Y0SdH7GBkwYqKjbae0Joc8mKty3HWd6ulga3QewhTX
```

Par défaut, Synapse active des indicateurs de présence qui indiquent si une personne est en ligne. Cela peut entraîner une utilisation élevée du processeur, vous pouvez donc le désactiver. Créez un nouveau fichier de configuration pour le même.

```
$ sudo nano /etc/matrix-synapse/conf.d/presence.yaml
```

Collez la ligne suivante dans l'éditeur.

```
presence:
  enabled: false
```

Enregistrez le fichier en appuyant sur Ctrl + X et en entrant Y lorsque vous y êtes invité.

Redémarrez le service Synapse pour appliquer les modifications.

```
$ sudo systemctl restart matrix-synapse
```

Créez un nouvel utilisateur matriciel. Il vous sera demandé votre nom d'utilisateur et votre mot de passe. Puisqu'il s'agit du premier utilisateur que nous créons, tapez `Oui` lorsqu'on lui a demandé s'il fallait faire de l'utilisateur un administrateur.

```
$ register_new_matrix_user -c /etc/matrix-synapse/conf.d/registration_shared_secret.yaml http://localhost:8008 New user
localpart [navjot]: navjot
Password:
Confirm password:
Make admin [no]: yes
Sending registration request...
Success!
```

Si vous souhaitez ouvrir l'inscription publique, créez un nouveau fichier de configuration.

```
$ sudo nano /etc/matrix-synapse/conf.d/registration.yaml
```

Collez-y les lignes suivantes.

```
enable_registration: true
```

Par défaut, Synapse n'autorise pas les inscriptions sans vérification par e-mail. Pour activer la vérification des e-mails, collez les lignes suivantes.

```
registrations_require_3pid:
  - email

email:
  smtp_host: mail.example.com
  smtp_port: 587

  # If mail server has no authentication, skip these 2 lines
  smtp_user: 'noreply@example.com'
  smtp_pass: 'password'

  # Optional, require encryption with STARTTLS
  require_transport_security: true

  app_name: 'HowtoForge Example Chat' # defines value for %(app)s in notif_from and email subject
  notif_from: "%(app)s <noreply@example.com>"
```

Pour désactiver la vérification des e-mails, collez plutôt la ligne suivante.

```
enable_registration_without_verification: true
```

Enregistrez le fichier en appuyant sur Ctrl + X et en entrant Y lorsque vous y êtes invité.

Redémarrez Synapse pour appliquer la configuration.

```
$ sudo systemctl restart matrix-synapse
```

Étape 7 - Configurer Nginx

Ouvrir le fichier `/etc/nginx/nginx.conf` pour l'édition.

```
$ sudo nano /etc/nginx/nginx.conf
```

Ajoutez la ligne suivante avant la ligne `include /etc/nginx/conf.d/*.conf;`

```
server_names_hash_bucket_size 64;
```

Enregistrez le fichier en appuyant sur Ctrl + X et en entrant Y lorsque vous y

êtes invité. Créez et ouvrez le fichier `/etc/nginx/conf.d/synapse.conf` pour l'édition.

```
$ sudo nano /etc/nginx/conf.d/synapse.conf
```

Collez-y le code suivant.

```
# enforce HTTPS
server {
    # Client port
    listen 80;
    listen [::]:80;
    server_name matrix.example.com;
    return 301 https://$host$request_uri;
}

server {
    server_name matrix.example.com;

    # Client port
    listen 443 ssl http2;
    listen [::]:443 ssl http2;

    # Federation port
    listen 8448 ssl http2 default_server;
    listen [::]:8448 ssl http2 default_server;

    access_log /var/log/nginx/synapse.access.log;
    error_log /var/log/nginx/synapse.error.log;

    tcp_nopush on;
    gzip on;

    location ~ ^(/matrix|/.synapse/client) {
        proxy_pass http://localhost:8008;
        proxy_http_version 1.1;

        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header Host $host;

        # Nginx by default only allows file uploads up to 1M in size
        # Increase client max body size to match max_upload_size defined in homeserver.yaml
        client_max_body_size 50M;
    }
}
```

Enregistrez le fichier en appuyant sur Ctrl + X et en saisissant Y lorsque vous y êtes invité une fois que vous avez terminé. La configuration ci-dessus fonctionne en supposant que les adresses IP des domaines example.com et matrix.example.com pointent vers le même serveur. Si ce n'est pas le cas, utilisez le fichier de configuration suivant pour le serveur example.com

```
server {
    server_name example.com;

    listen 443 ssl http2;
    listen [::]:443 ssl http2;

    # TLS configuration
    ssl_certificate /etc/letsencrypt/live/matrix.example.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/matrix.example.com/privkey.pem;
    ssl_trusted_certificate /etc/letsencrypt/live/matrix.example.com/chain.pem;
    ssl_session_timeout 1d;
    ssl_session_cache shared:MozSSL:10m;
    ssl_session_tickets off;
    ssl_prefer_server_ciphers on;
    ssl_stapling on;
    ssl_stapling_verify on;
    ssl_dhparam /etc/ssl/certs/dhparam.pem;
    resolver 1.1.1.1 1.0.0.1 [2606:4700:4700::1111] [2606:4700:4700::1001] 8.8.8.8 8.8.4.4 [2001:4860:4860::8888] [2001:4860:4860::8844] valid=60s;
    resolver_timeout 2s;

    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384;

    # Redirect
    location ~ ^(/matrix|/synapse/client) {
        return 301 "https://matrix.example.com$request_uri";
    }

    # Client homeserver autodiscovery
    location /.well-known/matrix/client {

    }

    # Domain delegation
    location /.well-known/matrix/server {
        default_type application/json;
        add_header Access-Control-Allow-Origin *;
        return 200 '{"m.server": "matrix.example.com"}';
    }
}
```

Vérifiez la syntaxe du fichier de configuration Nginx.

```
$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Redémarrez le service Nginx.

```
$ sudo systemctl restart nginx
```

Étape 8 - Installer Coturn

Vous devrez installer un Traversal Using Relays autour du serveur NAT (TURN) pour activer les appels vocaux et vidéo. Pour cela, nous installerons le package Coturn. Si tu n'as pas besoin de ça fonctionnalité, vous pouvez ignorer cette étape.

Installez Coturn.

```
$ sudo apt install coturn
```

Ouvrez les ports TURN et UDP.

```
$ sudo ufw allow 3478
$ sudo ufw allow 5349
$ sudo ufw allow 49152:65535/udp
```

Générez un certificat SSL pour Turn (turn.example.com).

```
$ sudo certbot certonly --nginx --agree-tos --no-eff-email --staple-ocsp --preferred-challenges http -m name@example.com -d turn.example.com
```

Générez un secret d'authentification et enregistrez-le dans le fichier de configuration.

```
$ echo "static-auth-secret=$(cat /dev/urandom | tr -cd '[a-zum:]' | fold -w 256 | head -n 1)" | sudo tee /etc/turnserver.conf
static-auth-secret=OckBLuwp6tMoi9mPcqVfaL7PwJRFUuKh5EvGBwcvb7tunevQ3cpP74we8cF4XSN8fNgrqxJeyiIKocoOABWjdTnChmJeB4WmrsLV2JnsPs3U61s9Rtj3OxBpZux0CGft8ClyNDweVLqxxNaYphNesoAT4y51RxLvAP2ros9S3jRR7IYRccJVRmpqTa8USBUbQ A
```

Ouvrez le fichier de configuration pour le modifier.

```
$ sudo nano /etc/turnserver.conf
```

Collez les lignes suivantes sous le secret d'authentification

```
use-auth-secret
realm=turn.example.com
cert=/etc/letsencrypt/live/turn.example.com/fullchain.pem
pkey=/etc/letsencrypt/live/turn.example.com/privkey.pem

# VoIP is UDP, no need for TCP
no-tcp-relay

# Do not allow traffic to private IP ranges
no-multicast-peers
denied-peer-ip=0.0.0.0-0.255.255.255
denied-peer-ip=10.0.0.0-10.255.255.255
denied-peer-ip=100.64.0.0-100.127.255.255
denied-peer-ip=127.0.0.0-127.255.255.255
denied-peer-ip=169.254.0.0-169.254.255.255
denied-peer-ip=172.16.0.0-172.31.255.255
denied-peer-ip=192.0.0.0-192.0.0.255
denied-peer-ip=192.0.2.0-192.0.2.255
denied-peer-ip=192.88.99.0-192.88.99.255
denied-peer-ip=192.168.0.0-192.168.255.255
denied-peer-ip=198.18.0.0-198.19.255.255
denied-peer-ip=198.51.100.0-198.51.100.255
denied-peer-ip=203.0.113.0-203.0.113.255
denied-peer-ip=240.0.0.0-255.255.255.255
denied-peer-ip=:1
denied-peer-ip=64:ff9b::-64:ff9b::ffff:ffff
denied-peer-ip=:ffff:0.0.0.0::ffff:255.255.255.255
denied-peer-ip=100::-100::ffff:ffff:ffff:ffff
denied-peer-ip=2001::-2001:fff:ffff:ffff:ffff:ffff:ffff:ffff
denied-peer-ip=2002::-2002:fff:ffff:ffff:ffff:ffff:ffff:ffff
denied-peer-ip=fc00::-fdfd:ffff:ffff:ffff:ffff:ffff:ffff:ffff
denied-peer-ip=fe80::-febf:ffff:ffff:ffff:ffff:ffff:ffff:ffff

# Limit number of sessions per user
user-quota=12
# Limit total number of sessions
total-quota=1200
```

Enregistrez le fichier en appuyant sur Ctrl + X et en entrant Y lorsque vous y êtes invité.

Redémarrez Coturn pour appliquer la configuration.

```
$ sudo systemctl restart coturn
```

Créez un nouveau fichier de configuration Synapse pour Coturn.

```
$ sudo nano /etc/matrix-synapse/conf.d/turn.yaml
```

Collez-y les lignes suivantes. Remplacez la valeur de turn_shared_secret par la valeur de static-auth-secret du fichier /etc/turnserver.conf.

```
turn_uris: [ "turn:turn.example.com?transport=udp", "turn:turn.example.com?transport=tcp" ]
turn_shared_secret: 'static-auth-secret'
turn_user_lifetime: 86400000
turn_allow_guests: True
```

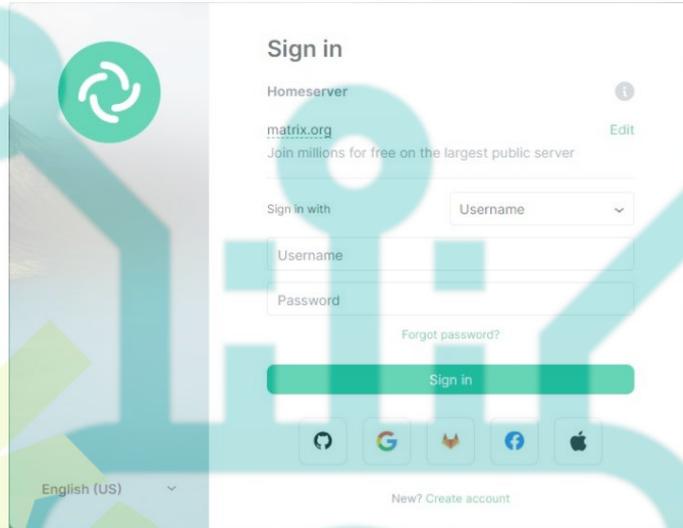
Enregistrez le fichier en appuyant sur Ctrl + X et en entrant Y lorsque vous y êtes invité.

Redémarrez Synapse pour appliquer les modifications.

```
$ sudo systemctl restart matrix-synapse
```

Étape 9 - Matrice d'accès

Vous pouvez accéder à Matrix Chat à l'aide du client Web d'Element à l'adresse <https://app.element.io>. Cliquez sur le bouton Se connecter pour continuer.



Sign in

Homeserver [matrix.org](#) [Edit](#)

Join millions for free on the largest public server

Sign in with

[Forgot password?](#)

English (US) [New? Create account](#)

Cliquez sur le lien Modifier sous Homeserver. Entrer matrix.example.com comme serveur domestique.

Sign into your homeserver

We call the places where you can host your account 'homeservers'. Matrix.org is the biggest public homeserver in the world, so it's a good place for many.

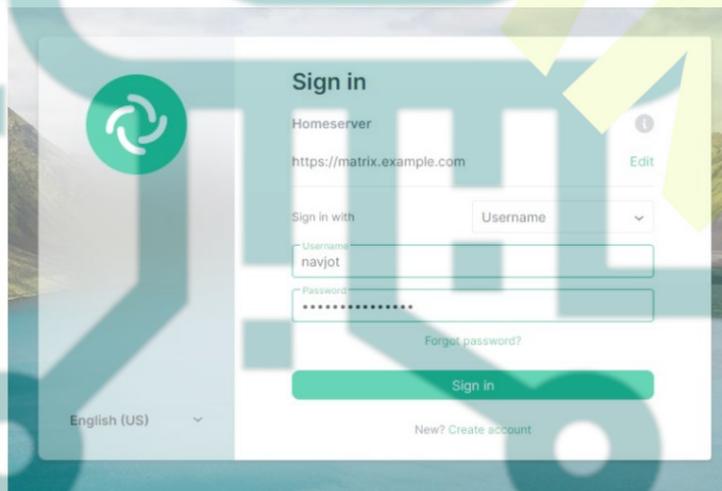
- matrix.org
- Other homeserver
- matrix.example.com

Use your preferred Matrix homeserver if you have one, or host your own.

[Learn more](#)

[About homeservers](#)

Si le client détecte correctement votre serveur domestique, la limite et le texte deviendront verts, sinon ils seront affichés en rouge. Cliquez sur Continuer pour continuer.



Sign in

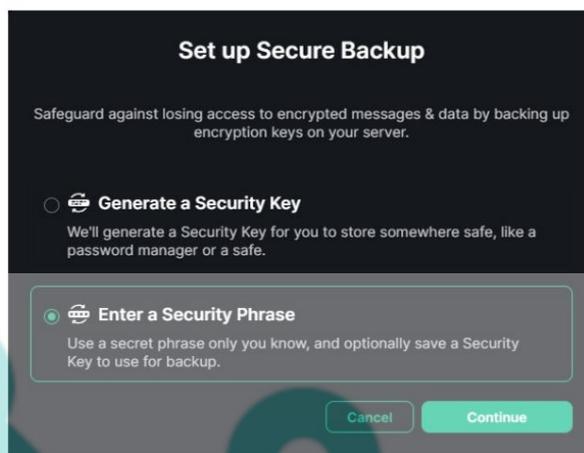
Homeserver <https://matrix.example.com> [Edit](#)

Sign in with

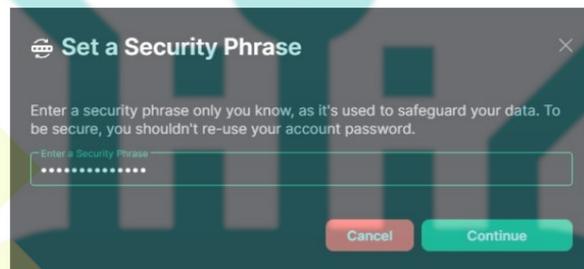
[Forgot password?](#)

English (US) [New? Create account](#)

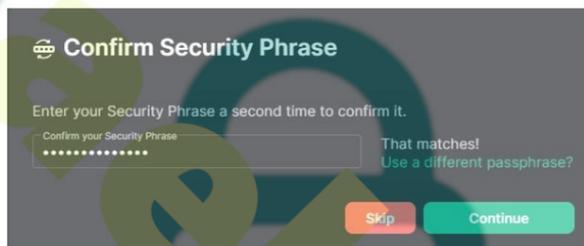
Cliquez sur le bouton Connexion pour vous connecter. Il vous sera demandé de créer une sauvegarde sécurisée et cryptée.



Sélectionnez l'option Entrer une phrase de sécurité pour créer une phrase de sécurité qui sera requise à chaque fois que vous vous connecterez. Cliquez sur Continuer pour continuer.



Entrez une phrase de sécurité et cliquez sur le bouton Continuer pour continuer. Il vous sera demandé de le confirmer à nouveau sur l'écran suivant.

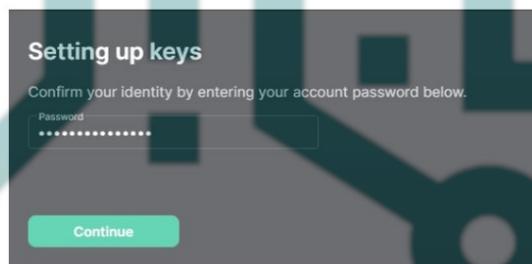


Saisissez à nouveau la phrase et cliquez sur Continuer pour continuer.

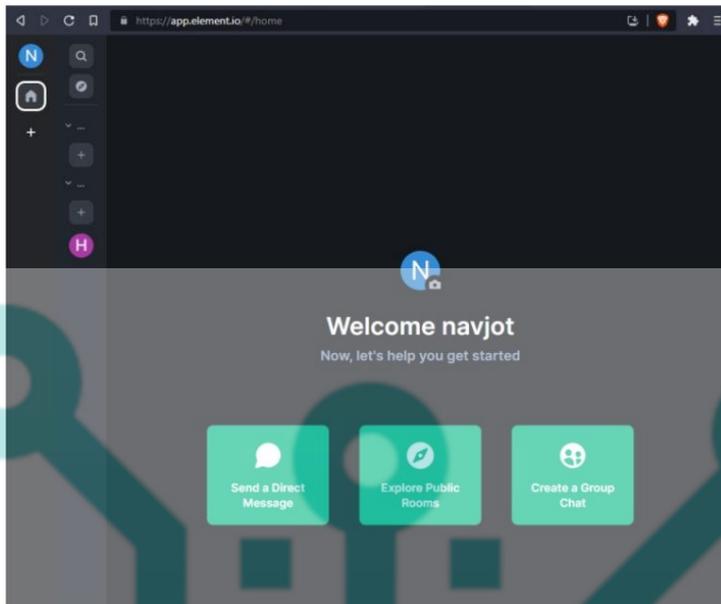


Vous recevrez un ensemble de clés de sécurité que vous pourrez utiliser si vous oubliez votre phrase de sécurité. Cliquez sur le bouton Télécharger pour les enregistrer.

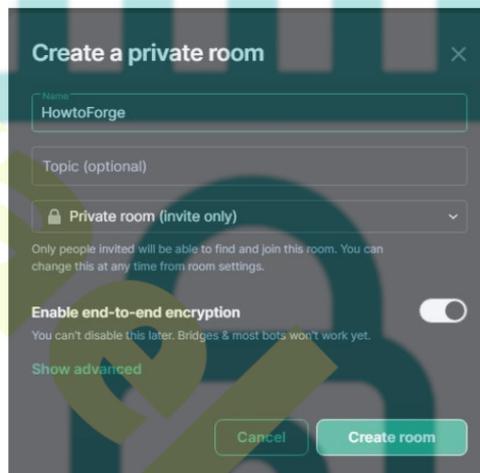
Cliquez sur le bouton Continuer pour continuer.



Il vous sera demandé le mot de passe de votre compte. Entrez le mot de passe et cliquez sur le bouton Continuer pour terminer la configuration de la sauvegarde cryptée.



Nous avons créé une salle de discussion de groupe nommée HowtoForge en utilisant le bouton Créer une discussion de groupe sur la page d'accueil. Vous obtiendrez la fenêtre contextuelle suivante lorsque vous cliquerez sur le bouton.



Vous pouvez restreindre les membres au domaine en développant le menu avancé et en sélectionnant l'option. Terminez la création de la pièce en cliquant sur le bouton Créer une pièce .

Étape 10 - Installer l'élément

jq package pour installer le processeur de texte JSON. Nous l'utiliserons pour récupérer la dernière version d'Element à partir de son référentiel GitHub, ce qui ne nécessite pas que vous ajoutiez les numéros de version manuellement et l'installation vous permet de le mettre à jour à l'aide d'une seule commande.

```
$ sudo apt install jq
```

Créez un répertoire pour Element.

```
$ sudo mkdir -p /var/www/element
```

Créez un nouveau fichier pour récupérer la dernière version d'Element.

```
$ sudo nano /var/www/element/update.sh
```

Ajoutez-y les lignes suivantes.

```
#!/bin/sh
set -e

install_location="/var/www/element"
latest=$(curl -s https://api.github.com/repos/vector-im/element-web/releases/latest | jq -r .tag_name)

cd "$install_location"

[ ! -d "archive" ] && mkdir -p "archive"
[ -d "archive/element-{$latest}" ] && rm -r "archive/element-{$latest}"
[ -f "archive/element-{$latest}.tar.gz" ] && rm "archive/element-{$latest}.tar.gz"

wget "https://github.com/vector-im/element-web/releases/download/{$latest}/element-{$latest}.tar.gz" -P "archive"
tar xf "archive/element-{$latest}.tar.gz" -C "archive"

[ -L "{$install_location}/current" ] && rm "{$install_location}/current"
ln -sf "{$install_location}/archive/element-{$latest}" "{$install_location}/current"
ln -sf "{$install_location}/config.json" "{$install_location}/current/config.json"
```

Enregistrez le fichier en appuyant sur Ctrl + X et en entrant Y lorsque vous y êtes invité.

Rendre le fichier exécutable.

```
$ sudo chmod +x /var/www/element/update.sh
```

Exécutez le script pour télécharger Element.

```
$ sudo /var/www/element/update.sh
```

Étape 11 - Configurer l'élément

Copiez le même fichier de configuration d'élément.

```
$ sudo cp /var/www/element/current/config.sample.json /var/www/element/config.json
```

Ouvrez le fichier de configuration pour le modifier.

