

# serveur chat sécurisé

- [Système d'échange sécurisé : discussion sur les options possibles](#)
- [Chaton sur Element/Matrix](#)

# Systeme d'échange sécurisé : discussion sur les options possibles

<https://claude.ai/chat/64d8e228-2365-4c99-8e91-d108a2091d3d>

Q: Dans le cadre d'échanges politique et confidentiel, je souhaite mettre en place un service de chat amélioré, type mattermost par exemple. Le principe est d'installer une application sur un serveur linux dédié, accessible par le net. Je voudrais avoir un avis sur le niveau de confidentialité réel, les conseils de sécurisation.

Pour un service de chat confidentiel à usage politique, voici les aspects clés à considérer :

## Niveau de confidentialité réel

**Mattermost** offre un bon niveau de base mais nécessite une configuration rigoureuse :

- Chiffrement en transit (HTTPS/TLS) et au repos possible
- Contrôle total des données (hébergement propre)
- Code open-source auditable

**Alternatives à considérer :**

- **Element/Matrix** : protocole décentralisé, chiffrement de bout en bout natif
- **Rocket.Chat** : similaire à Mattermost, bonnes options de sécurité
- **Signal** en version serveur (plus complexe à déployer)

## Conseils de sécurisation essentiels

**Infrastructure :**

- Serveur dédié isolé, pas de services tiers
- VPS chez un hébergeur respectant la confidentialité (Hetzner, OVH, ou mieux : hébergeur local)
- Système à jour, hardening Linux standard
- Firewall strictement configuré
- Monitoring des accès et tentatives d'intrusion

## Application :

- Chiffrement de bout en bout activé
- Authentification forte (2FA obligatoire)
- Rétention des messages limitée dans le temps
- Logs d'audit configurés mais anonymisés
- Pas de plugins tiers non audités

## Opérationnel :

- Accès administrateur minimal et tracé
- Sauvegardes chiffrées, stockage séparé
- Plan de continuité en cas de compromission
- Formation des utilisateurs aux bonnes pratiques

Pour un usage réellement sensible, considérez également l'aspect juridique : localisation du serveur, législation applicable, possibles réquisitions.

Quel est votre niveau technique actuel pour ce type de déploiement ? Q: C'est quasiment tout ce à quoi je pensais. Précision : dans la proposition "Chiffrement en transit (HTTPS/TLS) et au repos possible", que veut dire "au repos possible" ?

"Au repos possible" signifie que le chiffrement des données stockées sur le disque dur du serveur n'est pas activé par défaut dans Mattermost - c'est une option à configurer. Chiffrement au repos = chiffrement des données quand elles sont stockées physiquement sur le serveur (base de données, fichiers uploadés, logs, etc.) Dans Mattermost spécifiquement :

Par défaut : les messages sont stockés en clair dans la base de données PostgreSQL/MySQL Il faut configurer manuellement le chiffrement au niveau :

- De la base de données (encryption-at-rest PostgreSQL/MySQL)
- Du système de fichiers (LUKS, dm-crypt)
- Ou des deux

Pourquoi c'est crucial pour votre usage : Sans chiffrement au repos, si quelqu'un accède physiquement au serveur ou aux sauvegardes, il peut lire tous les messages en clair, même avec des mots de passe forts.

## Solutions recommandées :

- Chiffrement complet du disque (LUKS) lors de l'installation Linux
- Chiffrement au niveau base de données
- Éventuellement chiffrement applicatif des messages sensibles

C'est pourquoi j'évoquais Element/Matrix qui propose du chiffrement de bout en bout natif - dans ce cas, même l'administrateur serveur ne peut pas lire les messages, contrairement à Mattermost

standard.

Cette précision change-t-elle votre orientation ?

Q/R : Mon niveau technique administrateur système linux basé sur 15 ans d'expérience, sans formation de base préalable. Donc, humilité oblige. Mais je sais déployer mattermost et autre application, sous docker en général, et assurer la partie sécurisation avec nginx/certbot par exemple. Je code en python depuis longtemps et j'ai une expertise moyenne avec postgres plus qu'avec mysql ou mariadb ; je dirais celle d'un programmeur utilisant les SGBDR

Parfait, avec votre profil vous avez largement les compétences pour un déploiement sécurisé !

## Stack recommandée pour votre cas

### Déploiement :

- Docker Compose avec Mattermost + PostgreSQL + nginx
- Volumes chiffrés (LUKS au niveau host)
- PostgreSQL avec TDE (Transparent Data Encryption) si version récente, sinon chiffrement au niveau filesystem

### Sécurisation PostgreSQL spécifique :

```
# Dans postgresql.conf
ssl = on
ssl_cert_file = 'server.crt'
ssl_key_file = 'server.key'
# Connexions chiffrées uniquement
hostssl all all 0.0.0.0/0 md5
```

### Points d'attention avec votre expérience :

- Mattermost stocke aussi des fichiers uploadés → volume séparé chiffré
- Logs applicatifs peuvent contenir des données sensibles → rotation + chiffrement
- PostgreSQL WAL et dumps de sauvegarde → chiffrement obligatoire

### Monitoring discret :

- Fail2ban sur les tentatives de connexion
- Monitoring PostgreSQL des requêtes anormales
- Alertes sur les connexions admin

**Question pratique :** Envisagez-vous une authentification LDAP/AD ou comptes locaux ? Pour du politique, l'intégration avec des annuaires existants peut créer des surfaces d'attaque supplémentaires.

Avez-vous déjà une préférence d'hébergeur en tête, ou des contraintes géographiques particulières ?

Q: *pas d'authentification par annuaire. seul l'admin peut ajouter un/des comptes et gérer les accès : c'est mon idée si elle est conforme à une politique de sécurité restrictive.*

*Hébergement possible : un des héberger cités ou alors sur un serveur dédié sur un LAN privé (à domicile) ; sur ce sujet, quel serait l'optimum ?*

## Authentification : excellente approche

Comptes locaux avec admin unique = optimal pour la sécurité. Réduire la surface d'attaque au maximum.

### Configuration Mattermost recommandée :

```
# Dans config.json
"EnableUserCreation": false,
"EnableTeamCreation": false,
"EnableSignInWithEmail": true,
"EnableSignUpWithEmail": false
```

L'admin crée tous les comptes manuellement, pas d'auto-inscription.

## Hébergement : LAN privé vs VPS

### LAN privé (domicile) - AVANTAGES :

- Contrôle physique total
- Pas de tiers impliqué
- Latence minimale pour utilisateurs locaux
- Coût récurrent faible

### LAN privé - INCONVÉNIENTS :

- Votre connexion Internet = point de défaillance unique
- IP résidentielle souvent bloquée par filtres corporate
- Pas de redondance électrique/réseau
- Vous êtes responsable de la maintenance 24/7
- Juridiquement : les données sont chez vous

### VPS dédié - AVANTAGES :

- Infrastructure professionnelle (redondance, bande passante)
- IP propre, réputée
- Maintenance infrastructure déléguée
- Possibilité de changer rapidement d'hébergeur

**RECOMMANDATION pour usage politique** : VPS dédié chez **Hetzner** (Allemagne) ou hébergeur français type **Scaleway** - législation européenne, pas de coopération automatique avec services US.

L'avantage juridique du VPS : en cas de problème, vous pouvez migrer rapidement ailleurs.

Q : **Quelle est la criticité/sensibilité des échanges prévus ?**

Q/R: *échanges de stratégie politique en vue de campagne électorale dans un contexte tendu.*

## Dans ce contexte : VPS dédié fortement recommandé

**Raisons spécifiques à votre usage :**

- **Disponibilité critique** : campagne = timing serré, pas de marge pour pannes domicile
- **Mobilité des utilisateurs** : équipes sur le terrain, déplacements fréquents
- **Pression potentielle** : domicile peut devenir cible (légale ou autre)
- **Déni plausible** : hébergement tiers = séparation claire données/personne

## Hébergeur recommandé : **Hetzner (Allemagne)**

- RGPD strict, pas de Patriot Act
- Historique respectueux de la vie privée
- Infrastructure robuste, prix correct
- Serveur dédié (pas VPS partagé) pour isolation maximum

## Sécurisation renforcée pour ce contexte

**Accès :**

- VPN obligatoire pour tous les utilisateurs (WireGuard)
- IP serveur jamais exposée directement
- Changement de ports standards
- Géoblocking des pays non pertinents

**Juridique/Opérationnel :**

- Nom de domaine via registrar respectueux (Gandi, Namecheap)

- Pas de logs applicatifs détaillés
- Rétention messages courte (7-15 jours max)
- Plan de migration rapide vers autre hébergeur
- Dead man's switch pour destruction automatique

**Question cruciale :** *Avez-vous envisagé Element/Matrix plutôt que Mattermost ? Le chiffrement bout-en-bout natif serait un plus énorme pour ce niveau de sensibilité.*

Q/R: *suite à notre conversation, je vais explorer la solution Element/Matrix que je ne connais que de nom.*

*Une précision supplémentaire est la sécurisation des application mobile. Je trouve la sécurité Mattermost Mobile assez faible. Qu'en est-il pour Element/Matrix ?*

## Element Mobile : bien plus sécurisé

### Chiffrement :

- **Bout-en-bout par défaut** (contrairement à Mattermost)
- Même l'admin serveur ne peut pas lire les messages
- Verification croisée des appareils obligatoire
- Perfect Forward Secrecy (clés changent régulièrement)

### Sécurité mobile spécifique :

- Verrouillage par PIN/biométrie natif
- Pas de stockage en clair sur l'appareil
- Screenshots désactivables par politique
- Session timeout configurable
- Destruction à distance possible

## Comparaison Mattermost vs Element Mobile

### Mattermost Mobile - faiblesses :

- Messages stockés déchiffrés localement
- Push notifications contiennent du contenu
- Sécurisation dépend largement du MDM
- Pas de chiffrement bout-en-bout

### Element Mobile - avantages :

- Stockage local chiffré même si appareil compromis
- Notifications chiffrées (pas de contenu lisible)
- Vérification cryptographique des identités

- Mode "low bandwidth" pour connexions surveillées

# Recommandations pour votre contexte

## Configuration Element serveur :

- Serveur Matrix auto-hébergé (Synapse ou Dendrite)
- Pas de fédération (serveur isolé)
- Registration fermée (admin seul)
- Retention politiques strictes

## Politique mobile :

- Installation uniquement via F-Droid ou APK direct
- Pas de Google Play (tracking réduit)
- Téléphones dédiés si budget le permet

Q : *Element vous semble-t-il maintenant plus adapté ? L'installation est un peu plus technique que Mattermost mais largement dans vos cordes. Q/R: Je vois tout l'intérêt de cette solution. Je vais étudier ces 2 applications. Bien sûr, je compterais sur ton aide pour l'installation le cas échéant.*

Parfait ! C'est une excellente approche d'étudier les deux solutions avant de trancher.

## Pour votre évaluation, je suggère :

### Tests locaux :

- Docker Compose pour tester rapidement les deux
- Element : essayez Synapse (plus mature) vs Dendrite (plus léger)
- Interface utilisateur : testez avec vos futurs utilisateurs non-techniques

### Points de comparaison :

- Facilité d'adoption par l'équipe (crucial en campagne)
- Performance avec votre charge prévue
- Complexité de maintenance au quotidien
- Fonctionnalités spécifiques (partage fichiers, calls, etc.)

### Ressources utiles :

- Documentation Element : très complète
- Matrix.org : spécifications du protocole
- Communauté active sur leurs channels respectifs

N'hésitez absolument pas à revenir quand vous aurez fait votre choix ! Je serai ravi de vous accompagner sur :

- Architecture détaillée
- Docker Compose sécurisé
- Configuration des politiques de sécurité
- Scripts de déploiement/maintenance
- Procédures de sauvegarde/restauration

Bonne exploration des deux solutions ! La décision finale dépendra beaucoup de l'adoption par vos utilisateurs - c'est souvent le facteur déterminant en pratique.

Q: Une question supplémentaire au fait : utiliser une adresse IP directement plutôt qu'une redirection DNS est-ce une solution plus ou moins sécurisée dans ce contexte ?

# IP directe vs DNS : plus sécurisé mais avec des trade-offs

## IP directe - AVANTAGES sécuritaires :

- **Pas de trace DNS** dans les logs des résolveurs publics
- **Anonymat renforcé** : pas d'enregistrement domaine traçable
- **Pas de certificat SSL** associé à un nom (moins d'OSINT possible)
- **Résistance à la censure** : pas de blocage DNS possible

## IP directe - INCONVÉNIENTS :

- **Certificats SSL complexes** : soit auto-signés (alertes navigateur), soit certificats IP (rares)
- **Flexibilité limitée** : changement serveur = nouveau partage IP à tous
- **Mémorisation difficile** pour les utilisateurs
- **Applications mobiles** : certaines n'acceptent que HTTPS valide

# Recommandation hybride pour votre contexte

## Solution optimale :

1. **Domaine neutre** via registrar anonyme (Njalla, 1984.is)
2. **Sous-domaine générique** : `chat.entreprise-consulting.com`
3. **Pas de WHOIS traçable**
4. **TTL DNS très court** (300s) pour migration rapide

## Alternative très sécurisée :

- IP directe + **certificats auto-signés** pré-distribués
- Formation équipe : accepter le certificat manuellement
- Backup sur Tor (.onion) si situation se dégrade

**Pour Element spécifiquement** : Les clients mobiles Element gèrent mieux les certificats non-standard que Mattermost.

Dans votre contexte tendu, je pencherais pour le **domaine neutre** : bon compromis sécurité/usabilité. L'IP directe si la situation devient vraiment critique.

Q: *Votre instinct vous pousse vers quelle approche ?*

R: *Je n'ai pas de compétences sur cette question. Je m'en remet à cette analyse.*

Parfait, alors **domaine neutre** ce sera !

### **Plan d'action concret :**

#### **1. Registrar anonyme :**

- **Njalla.is** (recommandé) : paiement crypto accepté, pas de WHOIS
- Domaine générique type : `consulting-web.com`, `tech-services.eu`

#### **2. Configuration DNS :**

```
chat.votre-domaine.com A xxx.xxx.xxx.xxx (TTL 300)
```

#### **3. Certificat SSL :**

- Let's Encrypt via certbot (gratuit, automatique)
- Renouvellement auto dans Docker

#### **4. Migration d'urgence :**

- Script préparé pour changer l'IP en 5 minutes
- Nouveau serveur déployable rapidement

Cette approche vous donne :

- Sécurité élevée (pas de trace personnelle)
- Facilité d'usage (HTTPS valide, mémorisable)
- Flexibilité (migration rapide possible)

Quand vous serez prêt pour le déploiement, on pourra détailler :

- Le choix exact du registrar
- La config Docker complète

- Les scripts de migration d'urgence

C'est parti pour une solution robuste et discrète ! ☐☐

# Chaton sur Élement/Matrix

[https://wiki.chatons.org/doku.php/services/messagerie\\_instantanee/matrix](https://wiki.chatons.org/doku.php/services/messagerie_instantanee/matrix)

## Matrix/Element

**Element est une application libre de messagerie instantanée et voix sur basée sur le protocole Matrix.**

Matrix fonctionne sur le principe de l'email, c'est à dire qu'il vous faut un compte chez un fournisseur de compte Matrix. Vous pouvez ensuite utiliser Element pour vous y connecter par l'outil de votre choix (smartphone, tablette ou ordinateur).

[La messagerie instantanée de confiance de l'Etat français](#), Tchap, repose sur Matrix/Element.

## Fonctionnalités

- messagerie instantanée entre deux personnes ou pour un groupe
- possibilité d'échanger photos, vidéos ou n'importe quel type de fichiers dans le fil de conversation
- chiffrement des communications de bout en bout pour la messagerie via les protocoles OLM et MEGOLM;
- plusieurs clients peuvent être utilisés simultanément avec la même identité ;
- en installant son propre homeserver, on peut fonctionner complètement en autonome ou bien en rejoignant la fédération. Serveur et client libre et décentralisé (auto-hébergeable)
- Multiplateforme ;
- Connexions possibles avec d'autres messageries instantanées via des passerelles Matrix : envoi de SMS, de courriels, [WhatsApp](#), XMPP,...
- appels audio-vidéo entre deux personnes ou bien audio et visio conférences entre plusieurs personnes;
- Bonne qualité sonore, sans temps de latence (en fonction de la qualité de la connexion internet) ;
- Partage d'écran ;

Complicé du à la fédération :

- Base d'utilisateur. inscription/sauvegarde/restauration facile.
- Appels audio/visio pas vraiment exploitable en réseau mobile

# Découvrir à quoi ça ressemble

Si vous le souhaitez, vous pouvez essayer Element sur le navigateur internet **depuis votre ordinateur** en allant sur le groupe de discussion ouvert d'[Alsace Entraide Numérique](#) . Nota : pour essayer depuis un smartphone il vous faudra l'application Element et un compte.

Important: n'hésitez pas à indiquer que c'est un test pour essayer Element pour que les bénévoles d'ARN ou du Hackstub ne soient pas trop confus :) .

## Procédure pour se connecter

### Etape 1 : choisir votre fournisseur pour le compte Matrix

Ci-dessous une liste d'hébergeur du collectif qui proposent d'ouvrir un compte Matrix en accès libre (ou sur simple inscription) :

[https://www.chatons.org/entraide/list?title=service\\_type\\_target\\_id=All&field\\_software\\_target\\_id=274&field\\_is\\_shared\\_value=All](https://www.chatons.org/entraide/list?title=service_type_target_id=All&field_software_target_id=274&field_is_shared_value=All)

[L'outil de recherche des services CHATONS](#) peut vous permettre de trouver d'autres hébergeurs du collectif qui peuvent accueillir votre compte Matrix sous conditions.

Si vous cherchez à vous interconnecter avec des contacts Signal ou même WhatsApp comme sur le chat [Alsace Entraide Numérique](#), vous pouvez adhérer chez Alsace Réseau Neutre pour bénéficier de leurs passerelles.

### Etape 2 : installer Element sur votre smartphone

Une fois que vous avez votre compte, vous avez probablement envie de pouvoir l'utiliser en dehors d'un navigateur web.

1. Installer l'application Element sur votre smartphone depuis Google Play, le store d'iOS ou encore [F-droid](#) (l'application pèse 140Mo)
2. Ouvrez l'application
3. Cliquez sur "J'ai déjà un compte"
4. Cliquez sur le lien: "Me connecter avec mon identifiant Matrix"
5. Indiquez votre compte Matrix, sous la forme: "@utilisateur:domaine.com"
6. Indiquez votre mot de passe
7. Validez
8. A cette étape: il vous sera demandé une phrase de chiffrement à retenir. Cette phrase est nécessaire en plus du mot de passe du fait des techniques utilisées par Element pour sécuriser votre communication.

## Etape 3 : discuter pour la première fois

Arrivée à cette étape, vous avez accès à l'interface pour discuter, mais il n'y a aucun contact disponible.

Si vous connaissez quelqu'un sur Matrix vous pouvez ajouter son compte en utilisant l'icône + verte.

Si vous ne connaissez personne, vous pouvez chercher et ajouter un salon, par exemple celui des chatons ou celui d'Alsace Entraide Numérique cité plus haut.

Enfin, il est possible en utilisant des passerelles de discuter avec d'autres protocoles notamment IRC. Si vous souhaitez communiquer avec quelqu'un utilisant Signal, Whatsapp ou Facebook messenger, vous devriez vous rapprocher des chatons qui proposent des passerelles pré-installées:

- [Alsace Réseau Neutre](#) - WhatsApp / Signal - pas de compte personnel sur WA ou Signal nécessaire

# Documentation à destination des techniciens des hébergeurs des CHATONS

## Relier Matrix et d'autres réseaux de messagerie via des passerelles

Traduction de <https://matrix.org/docs/guides/types-of-bridging> et adaptation au contexte CHATONS.

Une idée fondamentale qui a motivé le développement de Matrix est la possibilité de communiquer entre différents réseaux de messagerie, libres comme ouverts. L'idée est que deux utilisateur.ice.s de deux réseaux différents discutent dans un salon qui est rendu accessible sur les deux réseaux grâce à une passerelle. Plusieurs solutions techniques existent pour implémenter une passerelle entre deux réseaux (en Anglais "bridge").

## Types de salons "rooms"

Migrer des salons entre portail et salon raccord n'est pas pratique. En effet il n'y a pas de possibilité pour les utilisateurs de retirer les portails une fois créés. Le risque est donc de se retrouver avec un mélange d'utilisateurs de portails et raccords dans un salon.

### Salons Portails "portal rooms"

La passerelle peut s'identifier elle-même sur le serveur du réseau tiers et contrôle des morceaux du domaine d'adressage des alias/identifiants de salons. Ainsi les utilisateur.ice.s Matrix rejoignent un salon du réseau tiers de manière transparente en rejoignant `freenode_#wherever:matrix.org` Le salon Matrix résultant est ponté automatiquement au seul salon cible sur le réseau tiers. Le contrôle d'accès des users Matrix est généralement géré par le réseau tiers. C'est la passerelle la plus puissante, qui permet l'intégration des deux réseaux la plus étroite et transparente. Il est possible pour un user Matrix de rejoindre un salon du réseau tiers, simplement en renseignant l'identifiant, sans manipulation complexe.

### Salons Raccords "plumbed rooms"

Un salon Matrix existant peut être raccordé à un ou plusieurs salons tiers en configurant une passerelle (qui peut être auto-hébergé). Par exemple, `#matrix:matrix.org` est raccordé sur `#matrix` sur Freenode. Le contrôle d'accès des utilisateurs Matrix est nécessairement géré par le salon du côté Matrix. C'est utile pour relier différents réseaux via Matrix, même dans le cas de réseaux privés pour lesquels le protocole n'est pas libre et/ou il n'y a pas d'accès admin aux serveurs pour mettre en place une passerelle de type portail.

## Types de passerelles

Du plus simple au plus avancé

### Passerelle reposant sur un robot

La manière la plus simple d'échanger des messages avec un réseau tiers est que la passerelle se connecte au réseau tiers en utilisant un (ou plusieurs) compte fictif appelé robot-passerelle. Par exemple on crée un compte `WhatsAppBot` ou `WhatsAppBot[2]` dont la passerelle connaît les identifiants. Ce robot relaie les messages entre les deux réseaux pour le compte des autres utilisateur.ice.s. L'expérience utilisateur est très mauvaise car toutes les méta-données concernant les messages sont perdues, en particulier leur expéditeur. C'est une méthode utilisée pour les réseaux privés centralisés des GAFAM tel que Telegram

<https://github.com/SijmenSchoon/telematrix>

## Passerelle reposant sur un robot-API pour la création d'utilisateurs virtuels

Certains réseaux tiers supportent l'injection de messages provenant de comptes fictifs / utilisateurs virtuels. Ce peut être utilisé pour représenter les utilisateurs Matrix avec un compte fictif sur le réseau tiers. Par exemple, Slack permet la création de robots distants sur demande. Ces robots peuvent alors créer des comptes fictifs aux utilisateurs Matrix qui sont alors affichés proprement dans les salons Slack comme des utilisateurs virtuels. Cependant, les comptes fictifs ne sont pas implémentés de la même manière que les comptes réels du réseau tiers, et n'ont pas les mêmes droits/accès aux méta-données et notifications temps-réel du réseau (sémantique). Ainsi, ils n'ont pas de présence, de profil, ne peuvent pas être contactés en message privé (MP), et leur identifiant ne peut pas être auto-complété, par exemple pour être pingé. Ils ne voient pas non plus quand quelqu'un est entrain d'écrire. C'est la méthode utilisée pour la passerelle Slack [matrix-appservice-slack](https://github.com/matrix-org/matrix-appservice-slack) <https://github.com/matrix-org/matrix-appservice-slack>

## Passerelle marionnette simple

C'est une passerelle permettant une intégration plus étroite. La passerelle s'identifie sur le réseau tiers comme étant un logiciel client tiers. L'utilisateur Matrix doit donc posséder un compte sur le réseau tiers. La passerelle crée des "marionnettes" Matrix des utilisateurs du réseau tiers. Autrement dit chaque utilisateur du réseau tiers se voit attribuer un compte fictif Matrix avec lequel l'utilisateur Matrix discute. A chaque salon du réseau tiers est associé un salon miroir Matrix. Les utilisateurs côté réseau tiers ont l'impression de discuter avec le compte de l'utilisateur Matrix du réseau tiers et ne peuvent pas savoir que tout est mis en miroir sur Matrix. Toute la sémantique (méta-données et notifications) du réseau tiers est disponible pour la passerelle via le protocole/API du logiciel client et peut donc être mise en miroir sur Matrix. Cependant, la passerelle doit gérer le processus d'authentification du réseau tiers pour que l'utilisateur Matrix se connecte à son compte sur le réseau tiers. C'est la méthode utilisée par les passerelles IRC (si vous la configurez pour se connecter au serveur IRC avec votre compte IRC réel). <https://github.com/matrix-org/matrix-appservice-irc>, télégram <https://github.com/matrix-org/matrix-appservice-tg> et git <https://github.com/matrix-org/matrix-appservice-gitter>

## Passerelle marionnette double

Dans le cas d'une marionnette simple, la passerelle "tire les ficelles" sur le réseau tiers pour le compte de l'utilisateur Matrix. Donc ce qui se passe sur Matrix est répercuté sur le réseau tiers. Dans l'idéal on aimerait aussi que ce que l'utilisateur fait sur le réseau tiers avec son compte tiers soit également répercuté sur Matrix. Par exemple si l'utilisateur lit un message depuis l'appli WhatsApp, on aimerait que ce message soit marqué comme lu dans la conversation Matrix miroir. Pour cela, la passerelle doit pouvoir tirer les ficelles de la partie Matrix du miroir. Et donc manipuler le compte Matrix de l'utilisateur en fonction de ce que le compte tiers de l'utilisateur fait. C'est la méthode ultime pour une passerelle personnelle, car toute la sémantique (méta-données et notifications) est précisément mise-en-miroir sur chaque réseau. A l'inverse, dans le cas d'un robot-relai, le compte côté réseau tiers et celui côté Matrix ne représentent pas un et un seul même

utilisateur “ne font pas un”. La difficulté côté Matrix est qu'il faut trouver une manière élégante pour que la passerelle se connecte au compte Matrix de l'utilisateur (délégation de jetons d'accès). Les difficultés sont spécifiques aux limitations de chaque réseau tiers. Par exemple, la plupart des réseaux tiers ne permettent pas de représenter un autre utilisateur Matrix que celui fournissant son compte tiers (voir robot-relai hybride). Matrix-puppet-bridge était le premier projet pour de tels passerelles <https://github.com/matrix-hacks/matrix-puppet-bridge#examples> Les passerelles maatrix sont les plus performantes à ce jour pour les réseaux privateurs des GAFAM tels que Telegram, WhatsApp, Facebook (Messenger). <https://docs.mau.fi/bridges/index.html> Voir aussi celles de <https://github.com/Sorunome/mx-puppet-bridge>

## Passerelle marionnette à robot-relai hybride

Il s'agit d'une combinaison entre les passerelles marionnette simple et double. Une telle passerelle essaie de permettre à plusieurs utilisateurs Matrix d'être représenté sur le réseau tiers via un seul compte tiers. Elle exploite la méthode de robot-passerelle.

## Passerelle serveur-à-serveur

Certains réseaux tiers (IRC, XMPP, SIP, SMTP, NNTP, GnuSociql, etc.), tout comme Matrix, supportent la fédération, libre ou contrôlée. C'est-à-dire que ces réseaux sont composés de plusieurs sous-réseaux hébergés sur différents serveurs et noms de domaines. Les utilisateurs ayant un compte sur un sous-réseau peuvent communiquer avec les utilisateurs d'un autre sous-réseau à condition que les deux serveurs soient fédérés, c-à-d qu'ils acceptent de communiquer entre eux. La manière la plus élégante de ponter ces réseaux entre eux est que la passerelle se comporte comme un serveur fédéré du réseau tiers. Ainsi tous les utilisateurs et salons de Matrix sont directement accessibles depuis le réseau tiers et vice-versa. Il suffit pour un utilisateur Matrix de préfixer le pseudo d'un utilisateur ou l'identifiant d'un salon tiers par le nom de son serveur pour lancer la conversation. Voir salons-portail.

## Passerelle à sens-unique

Des passerelles d'un réseau tiers vers Matrix sont utilisés pour suivre des actualités, comme par exemple <https://github.com/turt2live/matrix-appservice-instagram> ou github.

## Passerelle sidecar

La passerelle est un client tiers pour le réseau tiers. Seul un utilisateur Matrix peut voir les conversations du réseau tiers. Les permissions sont gérées par le réseau tiers, pas par Matrix. Voir bitlbee. <https://matrix.org/docs/guides/types-of-bridging/#sidecar-bridge>

# Présentations / tutoriels d'utilisation

## Tutoriels détaillés

- [Concept et première utilisation](#)
- [Première utilisation](#)
- [Passerelle WhatsApp ARN-Messenger](#)

## Tutoriels succincts

[Wiki de Tetaneutral](#)

## Présentations

- [Doc de Tedonum](#)
- [Présentation de Matrix par Grésille](#)
- <https://linuxfr.org/news/matrix-pour-decentraliser-skype-whatsapp-signal-slack-et-discord>

## Configuration / architecture / guides d'installation du service

- [Déployer la passerelle WhatsApp ARN-Messenger](#). Repose sur:
  - la [passerelle Mautrix-WhatsApp installée via Yunohost](#)
  - le [robot de mise en relation ARN-Messenger](#).
  - [Contacter ARN](#) (#arn:sans-nuage.fr) pour plus d'infos et les droits sur le tuto de déploiement.

## Bug de connexion CAS

Si un SSO/CAS est configuré comme avec le paquet `synapse_ynh`, seule la connexion par CAS est possible, la connexion par identifiant+mdp est désactivée. Certains clients comme pidgin ne supportent pas la connexion CAS, ou Element iOS est buggué. Voici comment résoudre ça :

[https://github.com/YunoHost-Apps/synapse\\_ynh/issues/379](https://github.com/YunoHost-Apps/synapse_ynh/issues/379)

## Configuration du Client web Element

Pour permettre aux utilisateurs de trouver facilement les salons publics des autres CHATONS ajouter à `element/config.json` :

```
"embedded_pages": {
  "login_for_welcome": true
},
"roomDirectory": {
  "servers": [
    "deuxfleurs.fr",
    "exarius.org",
    "garbaye.fr",
    "hadoly.fr",
    "libreon.fr",
    "matrix.domainepublic.net",
    "matrix.fdn.fr",
    "matrix.interhop.org",
    "matrix.lavallee.ynh.fr",
    "matrix.nomagic.uk",
    "matrix.org",
    "matrix.parinix.org",
    "matrix.underworld.fr",
    "sans-nuage.fr",
    "tedomum.net"
  ]
},
```

Notez que seuls `exarius`, `garbaye`, `nomagic`, `sans-nuage`, `underworld` et `tedomum` ont autorisé à lire la liste des salons public (dans la config synapse `allow_public_rooms_over_federation: true`). Ou alors pour les autres c'est qu'ils n'ont pas de salons publics ou que l'url renseignée ici est inexacte.

## Gestion quotidienne de synapse

L'implémentation serveur Matrix [synapse](#) propose une API d'administration plutôt complète, accessible aux URL débutant par `/_synapse/`. Cette API est documentée sur la [documentation officielle](#) de synapse. Les requêtes et les réponses sont formatées en JSON pour la plupart et interprétables automatiquement grâce à l'outil [jq](#). Pour employer aisément la commande, nous proposons la fonction bash suivante.

```
function synadm { curl -X "$1" -H "Content-Type: application/json" -H "Authorization: Bearer $SYNAPSE_TOKEN"
# Avant usage, exporter les variables suivantes (le token doit être associé à un compte administrateur)
export SYNAPSE_HS=matrix.monchaton.org
export SYNAPSE_TOKEN=XXX
```

Il existe des outils d'administration alternatifs tels que [synadm en Python](#), présentant les fonctionnalités de l'API Synapse avec plus d'abstraction encore.

A titre d'exemple, la fonction permet de manipuler les utilisateurs et les salons :

```

# Afficher les profils dont le nom contient un mot-clé
synadm GET 'v2/users?name=bot' | jq
# Afficher uniquement les mxid (pour export ou traitement automatique)
synadm GET 'v2/users?name=bot' | jq -r '.users[].name'
# Profil basique d'un compte donné
synadm GET v2/users/@user:monchaton.org | jq
# Informations de connexion du compte
synadm GET v1/whois/@user:monchaton.org | jq
# Toutes les IP connues du compte
synadm GET v1/whois/@user:monchaton.org | jq '.devices[].sessions[].connections | map(.ip) | ur
# Salons rejoints par le compte
synadm GET v1/users/@user:monchaton.org/joined_rooms | jq
# Réinitialiser un mot de passe
synadm POST v1/reset_password/@user:monchaton.org -d '{"new_password": "<mdp aleatoire>", "logou
# Désactiver un compte
synadm POST v1/deactivate/@user:monchaton.org -d '{"erase": true}'
# Bloquer un compte silencieusement (shadow ban)
synadm POST v1/users/@user:monchaton.org/shadow_ban

# Lister les salons sur le serveur
synadm GET v1/rooms | jq
# Lister les salons sans limite
synadm GET 'v1/rooms?limit=9999999' | jq
# Informations générales sur un salon
synadm GET 'v1/rooms/!0hSiGLbNAQMVGdLMaF:monchaton.org' | jq
# Lister les membres d'un salon
synadm GET 'v1/rooms/!0hSiGLbNAQMVGdLMaF:monchaton.org/members' | jq
# Supprimer un salon localement (adapter le block et le purge)
synadm DELETE 'v1/rooms/!0hSiGLbNAQMVGdLMaF:monchaton.org' -d '{"block": false, "purge": true}'
# Vider l'historique de messages d'un salon (cache local)
synadm POST v1/purge_history/!0hSiGLbNAQMVGdLMaF:monchaton.org' -d '{"purge_up_to_ts\":"$(date

```

## Nettoyage d'un serveur synapse

Préférer les solutions basées sur l'API. Modifier directement la BDD est fortement déconseillé, car cela risque de corrompre des choses au niveau de la fédération.

- [recommandé sur le chat synapse](<https://jacksonchen666.com/posts/2022-12-03/14-33-00/>)
- <https://foss-notes.blog.nomagic.uk/2021/03/matrix-database-house-cleaning/>
- <https://levans.fr/shrink-synapse-database.html>
- <https://gitea.prk.st/romain/synapse-tools/raw/branch/master/synapse-purge.sh>
- <https://www.tedomum.net/dev/service/matrix/administration/>

Un serveur synapse est gourmand en ressources, principalement en consommation disque et mémoire à travers sa base de données. Nettoyer régulièrement le serveur permet de conserver l'usage mémoire et la taille des tables à un niveau raisonnable (en 2021, un serveur synapse de

taille moyenne doit consommer moins de 6Go de RAM et 100Go de disque, beaucoup moins s'il ne fédère pas avec de gros salons). Le [guide de Levans](#) est très complet pour le nettoyage d'un serveur Matrix, et quelques extraits adaptés sont proposés à suivre.

Supprimer les salons sans membre local :

```
# Récupérer tous les salons dans un fichier
synadm GET 'v1/rooms?limit=999999' > rooms.json
# Supprimer les salons vides (attention, l'exécution peut prendre plusieurs jours et consommer c
jq -r '.rooms | map(select(.joined_local_members == 0)) | .[].room_id' < rooms.json | xargs -ti
```

Nettoyer l'historique des salons remplis :

```
# D'abord extraire la liste des salons les plus remplis, à lancer sur le serveur PostgreSQL
> \copy (select room_id, count(*) as cnt from events group by room_id order by cnt desc) to '/tmp/rooms_to_clean'
# Puis purger l'historique du top 10 jusqu'à -1 mois, attention les traitements sont lancés en p
head -n 10 /tmp/rooms_to_clean | cut -d, -f1 | xargs -ti $SHELL -ic 'synadm POST \'v1/purge_hist
```

Optimiser la table d'état (nécessite l'installation de [synapse-compress-state](#)) :

```
# D'abord extraire la liste des salons les plus lourds en table d'état, à lancer sur le serveur
> \copy (select room_id, count(*) AS cnt from state_groups_state group by room_id order by cnt c
# Puis compresser l'historique du top 100 (exporter les variables de connexion au serveur postgr
# Cette commande ne modifie pas l'état en base et peut tourner pendant plusieurs jours
head -n 100 /tmp/rooms_to_compress | cut -d, -f1 | xargs -ti synapse-compress-state -p "host=$PG
# Finalement exécuter tous les fichiers SQL
ls *.sql | xargs -ti psql -U$PGUSER -p 5432 -h $PGHOST $PGUSER -f '{}'
```

## Nettoyer les événements orphelins

Problèmes : <https://github.com/element-hq/synapse/issues/9406> <https://github.com/element-hq/synapse/issues/15575>

Solution [trouvée par club1](#) puis [fouine@underworld](#) : Avec le temps, ma base a continué de grossir et ce malgré l'application régulière des préceptes/règles d'hygiène ci-dessus, qui permettaient quand même de limiter les dégâts.

En cherchant... encore... et encore, je suis tombé sur ce blog :

<https://blog.gelesneeuw.nl/technology/2024/04/20/matrix-maintenance/>

Il est fait référence à un "bug" de jeunesse de synapse, qui perdure, dont voici l'issue :

<https://github.com/matrix-org/synapse/issues/3364>

Et dont voici le remède : <https://github.com/erikjohnston/synapse-find-unreferenced-state-groups>

C'est EXACTEMENT ce qui causait l'embonpoint de ma base de donnée. Après application du traitement, celle-ci est passée de 250Go à 18Go... oui, DIX-HUIT Giga.

Pour faire simple, ARRÊTEZ SYNAPSE, ARRÊTEZ SYNAPSE et ARRÊTEZ SYNAPSE, puis générez le fichier csv qui va contenir la liste des “événements orphelins”:

```
rust-synapse-find-unreferenced-state-groups -p "postgresql://$DBUSER:$PGPASSWORD@localhost/$DBNAME"
```

Enfin, dans le cli psql, une fois connecté à votre base de donnée:

```
CREATE TEMPORARY TABLE unreffed(id BIGINT PRIMARY KEY);
COPY unreffed FROM '/home/user/unreferenced.csv' WITH (FORMAT 'csv');
DELETE FROM state_groups_state WHERE state_group IN (SELECT id FROM unreffed);
DELETE FROM state_group_edges WHERE state_group IN (SELECT id FROM unreffed);
DELETE FROM state_groups WHERE id IN (SELECT id FROM unreffed);
```

Ce bug m'aura pourri la vie durant 3 ans, m'imposant des plans de remédiations lourds, des arbitrages de ressources, avec indispo. bref, je suis content :)

Si vous souhaitez automatiser les opérations de nettoyage, vous pouvez reprendre / vous inspirer du [script proposé ici](#) et l'appeler via une tâche cron de façon quotidienne pendant les heures creuses (en anglais).

## Instances publiques déployées

Il ne s'agit pas de proposer une liste exhaustive car il est difficile d'avoir connaissance de toutes les instances déployées. Si vous êtes administrateur d'une instance et qu'elle n'est pas encore recensée, n'hésitez pas à l'ajouter. L'usage de cette liste est à la discrétion de toutes les personnes y ayant accès. Nous ne garantissons pas la qualité du service fourni par les liens ci-dessous.

## Chez les CHATONS :

[https://www.chatons.org/entraide/list?title=service\\_type\\_target\\_id=All&field\\_software\\_target\\_id=274&field\\_is\\_shared\\_value=All](https://www.chatons.org/entraide/list?title=service_type_target_id=All&field_software_target_id=274&field_is_shared_value=All)

## Ailleurs :

## Création de compte libre

\* matrix.org (serveur par défaut hébergé par la fondation New Vector <https://vector.im>)

<https://matrix.org/docs/projects/try-matrix-now/>

# Création de compte sur demande/inscription

\* matrix.fdn.fr (Pack FDN niveau I) [https://www.fdn.fr/fdn-ouvre-son-instance-synapse-matrix-et-la-](https://www.fdn.fr/fdn-ouvre-son-instance-synapse-matrix-et-la-bridge-a-irc-geeknode/)

[bridge-a-irc-geeknode/](https://www.fdn.fr/fdn-ouvre-son-instance-synapse-matrix-et-la-bridge-a-irc-geeknode/)