Référence de Configuration

Guide complet sur tous les paramètres de configuration

Aperçu de l'Architecture

La passerelle OmniMessage SMPP est un **frontend de protocole sans état** qui traduit les messages SMPP vers/depuis OmniMessage. Toute la logique métier, les décisions de routage et le stockage des messages sont gérés par OmniMessage Core - la passerelle se contente de :

1. **Recevoir** des PDUs SMPP des opérateurs et des clients
2. **Les traduire** au format OmniMessage via l'API REST
3. **Interroger** OmniMessage pour les messages à envoyer
4. **Envoyer** des PDUs SMPP aux opérateurs
5. **Rapporter** l'état de livraison à OmniMessage

Ceci est identique à la façon dont d'autres frontends OmniMessage (Diameter, MAP, IMS) fonctionnent - ce sont tous des traducteurs de protocole sans état qui délèguent à OmniMessage Core.

Emplacement du Fichier de Configuration

`/opt/omnimessage-smpp/config/runtime.exs`

Important : Après avoir modifié la configuration, redémarrez la passerelle :

```
sudo systemctl restart omnimessage-smpp
```

Structure de Configuration

Le fichier de configuration utilise la syntaxe Elixir. Structure de base :

```
import Config

# Paramètres globaux
config :omnimessage_smpp,
  setting_name: value

# Liens SMPP
config :omnimessage_smpp, :binds, [
  %{
    name: "bind_name",
    # ... paramètres de liaison
  }
]
```

Paramètres Globaux

API_BASE_URL

URL de la plateforme OmniMessage Core

```
config :omnimessage_smpp,
  api_base_url: "https://omnimessage-core.example.com:8443"
```

Paramètre	Type	Requis	Par défaut
`api_base_url`	Chaîne (URL)	Oui	-

But : URL de la plateforme OmniMessage Core. La passerelle communique avec OmniMessage via l'API REST pour tout le traitement des messages :

- **Soumettre des Messages** : Envoyer les messages SMPP reçus à OmniMessage pour traitement
- **Récupérer des Messages** : Interroger pour les messages destinés aux opérateurs SMPP
- **Rapporter l'État de Livraison** : Mettre à jour l'état de livraison des messages vers OmniMessage
- **Santé du Système** : Vérifications de santé périodiques

Critique : C'est ici que la passerelle obtient tout son "cerveau". OmniMessage gère :

- ✓ Validation des messages et vérification de format
- ✓ Décisions de routage (quel opérateur utiliser)
- ✓ Limitation de débit et régulation
- ✓ Validation de numéro
- ✓ Stockage et persistance des messages
- ✓ Logique de réessai de livraison
- ✓ Suivi d'état

La passerelle se contente de traduire le format SMPP ↔ OmniMessage.

Exemples :

```
# HTTPS avec IP
api_base_url: "https://192.168.1.100:8443"

# HTTPS avec nom d'hôte
api_base_url: "https://omnimessage-core.company.com:8443"

# HTTP (non recommandé pour la production)
api_base_url: "http://192.168.1.100:8080"
```

Exigences Réseau :

- La passerelle doit avoir un accès réseau à OmniMessage Core
- Utilisez HTTPS en production (configurer `verify_ssl_peer`)
- Le pare-feu doit autoriser le HTTPS sortant sur le port spécifié

OAM/Port de l'API de Gestion (`api_ex`)

Port pour l'API de Gestion/OAM de la passerelle

La passerelle sert sa **propre** API de Gestion/OAM (`/status`, `/smpp/client_peers`, `/smpp/server_peers`, `/smpp/binds`, `/smpp/summary`) via `api_ex`, par défaut sur `8443` (`config/config.exs`). C'est un **serveur** que la passerelle héberge - distinct de `api_base_url`, qui est le point de terminaison **client** que la passerelle appelle sur OmniMessage Core.

Ne confondez pas les deux :

Paramètre	Direction	Par défaut	Signification
`api_base_url`	Passerelle → Core (client)	`…:8443`	Où la passerelle envoie/interroge les messages
`api_ex` `port`	Passerelle (serveur)	`8443`	Où l'API OAM de la passerelle écoute

Colocation avec OmniMessage Core : OmniMessage Core héberge également son API de stockage/messages sur le port `api_ex` `8443`. Lorsque la passerelle SMPP fonctionne sur le **même hôte** que Core, les deux sont par défaut sur `8443` et un seul peut le lier - l'autre boucle de crash avec `:eaddrinuse` (et les requêtes `/api/messages` atteignent le mauvais service, retournant 404).

Pour corriger, éditez `config/runtime.exs` (déployé à `/etc/omnimessage_smpp/runtime.exs`) et décommentez la substitution `api_ex`, en définissant un port libre afin que l'API OAM de la passerelle se déplace hors du port de Core. `api_base_url` cible toujours le `8443` de Core :

```
# Core possède :8443 pour l'API de stockage ; déplacez l'API OAM
de la passerelle vers :8444
config :api_ex,
  api:
    Application.get_env(:api_ex, :api, %{})
    |> Map.put(:port, 8444)
    |> Map.put(:listen_ip, "0.0.0.0")
```

Si la substitution est laissée commentée, le port par défaut est `8443`. Les passerelles autonomes (Core sur un hôte différent) n'ont pas besoin de changement.

SMPP_POLL_INTERVAL

Fréquence de vérification de la file d'attente (millisecondes)

```
config :omnimessage_smpp,
  smpp_poll_interval: 100
```

Paramètre	Type	Requis	Par défaut
`smpp_poll_interval`	Entier	Non	100

But : À quelle fréquence (en millisecondes) chaque client vérifie la file d'attente des messages.

Directives :

- **Volume élevé (>100 TPS)** : 100-500ms
- **Volume moyen (10-100 TPS)** : 500-1000ms
- **Volume faible (<10 TPS)** : 1000-2000ms

Variable d'environnement : `SMPP_POLL_INTERVAL`

VERIFY_SSL_PEER

Vérification du certificat SSL

```
config :omnimessage_smpp,
  verify_ssl_peer: false
```

Paramètre	Type	Requis	Par défaut
`verify_ssl_peer`	Booléen	Non	false

But : Vérifier les certificats SSL lors de la connexion à l'API backend.

Valeurs :

- `true` : Vérifier les certificats (production avec certificats valides)
- `false` : Ignorer la vérification (certificats auto-signés, test)

Variable d'environnement : `VERIFY_SSL_PEER`

SMSC_NAME

Identifiant de la passerelle pour l'enregistrement

```
config :omnimessage_smpp,
  smsc_name: "smpp_gateway"
```

Paramètre	Type	Requis	Par défaut
`smsc_name`	Chaîne	Non	"smpp_gateway"

But : Identifie cette instance de passerelle dans le backend de la file d'attente des messages.

Variable d'environnement : `SMSC_NAME`

Configuration de Liaison Client SMPP

Les liaisons clients sont des **connexions sortantes** où la passerelle agit en tant qu'**ESME** (client) se connectant aux serveurs **SMSC** des opérateurs. Dans ce mode, la passerelle initie la connexion pour envoyer et recevoir des messages via des opérateurs externes.

Exemple Complet de Liaison Client

```
config :omnimessage_smpp, :binds, [
  %{
    # Identifiant unique pour cette connexion
    name: "vodafone_uk",

    # Mode de connexion
    mode: :client,

    # Type de liaison SMPP
    bind_type: :transceiver,

    # Adresse du serveur SMPP de l'opérateur
    host: "smpp.vodafone.co.uk",
    port: 2775,

    # Fixer l'adresse IP/port source local de la connexion
sortante (optionnel,
    # à définir si l'opérateur liste blanche par IP/port source)
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5000,

    # Informations d'authentification
    system_id: "your_username",
    password: "your_password",

    # Champs de protocole SMPP (optionnels, à définir si
l'opérateur l'exige)
    system_type: "",
    addr_ton: 0,
    addr_npi: 0,
    address_range: "",

    # Limitation de débit
    tps_limit: 100,

    # Nom de la file d'attente partagée (optionnel, par défaut au
nom de la liaison). Définir la même
    # valeur sur plusieurs liaisons pour qu'elles drainent une
file d'attente commune.
    queue: "vodafone_uk",
```

```
    # Fréquence de vérification de la file d'attente
    queue_check_frequency: 1000,

    # Intervalle de maintien (secondes, 0 pour désactiver)
    enquire_link_interval: 60,

    # Mise en cache des messages (optionnel)
    cache_enabled: false,
    cache_max_size: 10000,
    cache_retry_interval: 60
  }
]
```

Paramètres de Liaison Client

name

Identifiant unique de connexion

Type	Requis	Exemple
Chaîne	Oui	`"vodafone_uk"`

But : Identifie de manière unique cette connexion SMPP.

- Utilisé dans les journaux et les métriques
- Doit être unique parmi toutes les liaisons
- Utilisez des noms descriptifs (opérateur, région, but)

Conventions de nommage :

- `carrier_region` : `"vodafone_uk"`, `"att_us"`
- `purpose_number` : `"marketing_1"`, `"alerts_primary"`

mode

Type de connexion

Type	Requis	Valeur
Atome	Oui	`:client`

But : Définit ceci comme une connexion sortante où la passerelle agit en tant qu'**ESME** se connectant à un **SMSC** externe.

Valeur fixe : Toujours `:client` pour les connexions sortantes.

bind_type

Type de session SMPP

Type	Requis	Valeurs Autorisées
Atome	Oui	`:transmitter`, `:receiver`, `:transceiver`

But : Définit la capacité de direction des messages.

Options :

- `:transmitter` - Envoyer des messages uniquement (submit_sm)
- `:receiver` - Recevoir des messages uniquement (deliver_sm)
- `:transceiver` - Envoyer et recevoir (le plus courant)

Recommandation : Utilisez `:transceiver` sauf si l'opérateur exige un type spécifique.

host

Nom d'hôte ou IP du serveur SMPP de l'opérateur

Type	Requis	Exemple
Chaîne	Oui	`"smpp.carrier.com"` ou `"10.5.1.100"`

But : Adresse du serveur SMPP de l'opérateur.

Exemples :

```
host: "smpp.vodafone.co.uk"
host: "10.20.30.40"
host: "smpp-primary.carrier.net"
```

port

Port du serveur SMPP

Type	Requis	Par défaut	Plage
Entier	Oui	2775	1-65535

But : Port TCP pour la connexion SMPP.

Port standard : 2775

Exemples :

```
port: 2775  # Standard
port: 3000  # Personnalisé
```

bind_source_ip

IP source locale pour la connexion sortante

Type	Requis	Par défaut	Exemple
Chaîne	Non	(par défaut OS)	`"10.20.30.40"`

But : Fixe l'adresse IP source locale utilisée pour la connexion TCP sortante vers l'opérateur. Définissez ceci lorsque l'opérateur liste blanche les connexions par IP source et que l'hôte de la passerelle a plusieurs adresses. Une adresse invalide est enregistrée et ignorée (le système d'exploitation choisit l'IP source).

Remarque : Il s'agit d'un champ uniquement de configuration (lu à partir de la configuration de l'application `:binds`) ; il n'est pas modifiable depuis l'interface Web / le magasin Mnesia.

bind_source_port

Port source local pour la connexion sortante

Type	Requis	Par défaut	Plage
Entier	Non	(éphémère)	1-65535

But : Fixe le port source local utilisé pour la connexion TCP sortante. Définissez ceci lorsque l'opérateur liste blanche par IP source **et** port. Le socket est ouvert avec `SO_REUSEADDR` afin que le port fixé puisse se rebinder immédiatement lors de la reconnexion (plutôt que d'échouer avec `:eaddrinuse` pendant que le socket précédent est en `TIME_WAIT`).

Remarque : Comme `bind_source_ip`, il s'agit d'un champ uniquement de configuration et n'est pas modifiable depuis l'interface Web.

system_id

Nom d'utilisateur d'authentification

Type	Requis	Exemple
Chaîne	Oui	`"company_user"`

But : Nom d'utilisateur fourni par l'opérateur pour l'authentification.

Sécurité : Protégez cette information d'identification - stockée dans le fichier de configuration.

password

Mot de passe d'authentification

Type	Requis	Exemple
Chaîne	Oui	`"secret_password"`

But : Mot de passe fourni par l'opérateur pour l'authentification.

Sécurité :

- Protégez cette information d'identification
- Utilisez des mots de passe forts
- Faites des rotations périodiques

tps_limit

Limite de transactions par seconde

Type	Requis	Par défaut	Plage
Entier	Oui	100	1-10000

But : Nombre maximum de messages par seconde à envoyer via cette connexion.

Directives :

- Réglez à 70-80 % du maximum de l'opérateur
- Emp��che la régulation/déconnexion
- Permet une marge pour les accusés de réception de livraison

Exemples :

```
tps_limit: 10    # Volume faible
tps_limit: 50    # Volume moyen
tps_limit: 100   # Volume élevé (le plus courant)
tps_limit: 1000  # Volume très élevé
```

Calcul :

```
Si max opérateur = 100 TPS
Réglez tps_limit = 70-80
Laisse 20-30 TPS de marge
```

queue

Nom de la file d'attente partagée

Type	Requis	Par défaut	Exemple
Chaîne	Non	nom de la liaison	`"sinch.smpp"`

But : La file d'attente dont cette liaison se sert dans le backend. Par défaut, elle est égale au `name` de la liaison, donc chaque liaison se sert de sa propre file d'attente. Définissez la même valeur `queue` sur plusieurs liaisons pour qu'elles drainent toutes une file d'attente partagée - par exemple, deux connexions Sinch drainant toutes deux `"sinch.smpp"` pour le partage de charge/redondance.

Exemples :

```
# Deux liaisons partageant une file d'attente
%{name: "sinch_1", queue: "sinch.smpp", ...}
%{name: "sinch_2", queue: "sinch.smpp", ...}
```

queue_check_frequency

Intervalle de sondage de la file d'attente de messages (millisecondes)

Type	Requis	Par défaut	Plage
Entier	Oui	1000	100-10000

But : À quelle fréquence vérifier le backend pour de nouveaux messages à envoyer.

Directives :

- **Volume élevé (>100 TPS)** : 500-1000ms
- **Volume moyen (10-100 TPS)** : 1000-2000ms
- **Volume faible (<10 TPS)** : 2000-5000ms

Compromis :

- Valeur plus basse = récupération de message plus rapide, plus de charge API
- Valeur plus élevée = récupération plus lente, moins de charge API

enquire_link_interval

Intervalle de maintien SMPP (secondes)

Type	Requis	Par défaut	Plage
Entier	Non	60	0-3600

But : À quelle fréquence (en secondes) envoyer des PDUs enquire_link SMPP pour vérifier que la connexion est active. Le serveur distant répond avec enquire_link_resp.

Directives :

- **Par défaut (60)** : Convient à la plupart des opérateurs
- **Valeurs plus basses (15-30)** : Détection de défaillance plus rapide, plus de trafic
- **Valeurs plus élevées (120-300)** : Moins de surcharge, détection de défaillance plus lente
- **0** : Désactive complètement enquire_link (non recommandé)

Exemples :

```
enquire_link_interval: 60   # Standard (1 minute)
enquire_link_interval: 30   # Maintien agressif
enquire_link_interval: 0    # D��sactivé
```

system_type

Identifiant de type de système SMPP

Type	Requis	Par défaut	Exemple
Chaîne	Non	`""`	`"0TP"`

But : Champ de protocole SMPP envoyé lors de la liaison. Certains opérateurs exigent une valeur spécifique. Laissez vide à moins que l'opérateur n'en spécifie une.

addr_ton

Type d'Adresse du Numéro

Type	Requis	Par défaut	Plage
Entier	Non	0	0-6

But : Champ de protocole SMPP spécifiant le type de numéro utilisé dans la demande de liaison.

Valeurs courantes :

- `0` - Inconnu
- `1` - International
- `2` - National
- `5` - Alphanumérique

Définir selon les exigences de l'opérateur.

addr_npi

Indicateur de Plan de Numérotation d'Adresse

Type	Requis	Par défaut	Plage
Entier	Non	0	0-18

But : Champ de protocole SMPP spécifiant le plan de numérotation dans la demande de liaison.

Valeurs courantes :

- `0` - Inconnu
- `1` - ISDN/E.164
- `3` - Données/X.121
- `9` - Privé

Définir selon les exigences de l'opérateur.

address_range

Plage d'adresses pour la liaison

Type	Requis	Par défaut	Exemple
Chaîne	Non	`""`	`"614*"`

But : Champ de protocole SMPP spécifiant la plage d'adresses que cette liaison gère. Utilisé par certains opérateurs pour filtrer quels messages sont livrés à cette connexion. Laissez vide à moins que l'opérateur ne spécifie une valeur.

enabled

État de l'activation du pair

Type	Requis	Par défaut
Booléen	Non	`true`

But : Contrôle si ce pair est actif. Les pairs désactivés sont conservés dans la configuration mais n'établissent pas de connexions. Utile pour prendre temporairement une connexion hors ligne sans supprimer sa configuration.

cache_enabled

Activer la mise en cache locale des messages

Type	Requis	Par défaut
Booléen	Non	`false`

But : Lorsqu'il est activé, les messages entrants sont mis en cache localement si l'API backend est inaccessible, puis livrés automatiquement lorsque la connectivité est rétablie. Voir MESSAGE_CACHE.md pour tous les détails.

cache_max_size

Messages mis en cache maximum

Type	Requis	Par défaut	Plage
Entier	Non	10000	1-1000000

But : Nombre maximum de messages à mettre en cache par liaison. Lorsque la limite est atteinte, les messages les plus anciens sont évincés (FIFO). Ne s'applique que lorsque `cache_enabled` est `true`.

cache_retry_interval

Intervalle de réessai de base (secondes)

Type	Requis	Par défaut
Entier	Non	60

But : Intervalle de base en secondes avant de réessayer la livraison d'un message mis en cache. Combiné avec un backoff exponentiel (réessai 0 : 60s,

réessai 1 : 120s, réessai 2 : 240s, etc.). Ne s'applique que lorsque `cache_enabled` est `true`.

Exemple d'Interface Web :

Configuration de Liaison Serveur SMPP

Les liaisons serveur définissent des **connexions entrantes** où la passerelle agit en tant qu'**SMSC** (serveur) acceptant des connexions de **ESMEs** externes (clients). Dans ce mode, les systèmes partenaires se connectent à la passerelle pour envoyer et recevoir des messages.

Exemple Complet de Liaison Serveur

```
config :omnimessage_smpp, :server_binds, [
  %{
    # Identifiant unique pour ce client
    name: "partner_acme",

    # Informations d'identification attendues du client
    system_id: "acme_corp",
    password: "acme_secret",

    # Types de liaison autorisés
    allowed_bind_types: [:transmitter, :receiver, :transceiver],

    # Restrictions IP
    ip_whitelist: ["192.168.1.0/24", "10.50.1.100"],

    # Restrictions d'adresse source (vide = autoriser tout)
    source_address_whitelist: [],

    # Limitation de débit
    tps_limit: 50,

    # Fréquence de vérification de la file d'attente
    queue_check_frequency: 1000,

    # Intervalle de maintien (secondes, 0 pour désactiver)
    enquire_link_interval: 60,

    # Mise en cache des messages (optionnel)
    cache_enabled: false,
    cache_max_size: 10000,
    cache_retry_interval: 60,

    # Accusés de réception de livraison (DLR) vers cette liaison
(optionnel)
    deliver_receipts: false
  }
]
```

Paramètres de Liaison Serveur

name

Identifiant du client

Type	Requis	Exemple
Chaîne	Oui	`"partner_acme"`

But : Identifie le client externe qui se connecte à vous.

Conventions de nommage : Utilisez le nom du partenaire/client pour une identification facile.

system_id

Nom d'utilisateur attendu du client

Type	Requis	Exemple
Chaîne	Oui	`"acme_corp"`

But : Nom d'utilisateur que le client externe doit fournir pour s'authentifier.

À fournir au client : Partagez cette information d'identification avec votre partenaire.

password

Mot de passe attendu du client

Type	Requis	Exemple
Chaîne	Oui	`"secure_password"`

But : Mot de passe que le client externe doit fournir pour s'authentifier.

Sécurité :

- Utilisez des mots de passe forts
- Unique par client
- Partagez en toute sécurité avec le partenaire

allowed_bind_types

Types de session autorisés

Type	Requis	Par défaut
Liste d'Atomes	Oui	-

But : Restreint les types de liaison que le client peut utiliser.

Options :

```
allowed_bind_types: [:transceiver]  # Seulement transceiver
allowed_bind_types: [:transmitter, :receiver]  # TX ou RX
allowed_bind_types: [:transmitter, :receiver, :transceiver]  #
Tout
```

Recommandation : Autorisez les trois à moins que vous n'ayez besoin de restrictions.

ip_whitelist

Adresses IP des clients autorisées

Type	Requis	Par défaut	Format
Liste de Chaînes	Oui	`[]`	IPs ou notation CIDR

But : Sécurité - n'autoriser que les connexions provenant d'IP connues.

Formats :

- IP unique : `"192.168.1.100"` (automatiquement `/32`)
- Sous-réseau CIDR : `"192.168.1.0/24"`, `"10.0.0.0/8"`
- Mélange des deux : `["192.168.1.0/24", "10.50.1.100"]`

Exemples :

```
# Autoriser n'importe quelle IP (non recommandé)
ip_whitelist: []

# IP unique
ip_whitelist: ["203.0.113.50"]

# Plusieurs IPs
ip_whitelist: ["203.0.113.50", "203.0.113.51"]

# Sous-réseau
ip_whitelist: ["192.168.1.0/24"]

# Mixte
ip_whitelist: ["192.168.1.0/24", "10.50.1.100", "10.60.0.0/16"]
```

Sous-réseaux courants :

- `/32` - IP unique (automatique pour les IP sans masque)
- `/24` - 256 adresses (par exemple, 192.168.1.0-255)
- `/16` - 65,536 adresses (par exemple, 10.50.0.0-255.255)
- `/8` - 16,777,216 adresses (par exemple, 10.0.0.0-255.255.255.255)

source_address_whitelist

Adresses d'origine autorisées

Type	Requis	Par défaut	Format
Liste de chaînes	Non	`[]`	Modèles Regex

Objectif : Restreint les adresses d'origine (identifiants d'expéditeur) que les clients connectés peuvent utiliser lors de l'envoi de messages. Une liste vide

permet toutes les adresses. Chaque entrée est un modèle d'expression régulière.

Exemples :

```
# Autoriser n'importe quelle adresse source
source_address_whitelist: []

# Correspondance exacte (ancrée)
source_address_whitelist: ["^MyBrand$", "^AlertService$"]

# Correspondance par préfixe
source_address_whitelist: ["^614", "^\+61"]

# Plage de numéros avec classe de caractères
source_address_whitelist: ["^61[45]\d{8}$"]

# Mixte
source_address_whitelist: ["^MyBrand$", "^614", "^\+61400000001$"]
```

Les messages avec des adresses source non autorisées sont rejetés avec `ESME_RINVSRCADR`. Voir SOURCE_ADDRESS_WHITELIST.md pour tous les détails.

tps_limit

Limite de messages par seconde

Identique à la liaison client `tps_limit` - contrôle le taux de livraison des `deliver_sm` sortants vers les clients connectés.

queue_check_frequency

Intervalle de sondage de la file d'attente

Identique à la liaison client `queue_check_frequency` - fréquence à laquelle vérifier les messages à livrer à ce client.

enquire_link_interval

Intervalle de maintien de connexion SMPP (secondes)

Identique à la liaison client `enquire_link_interval`. Contrôle la fréquence à laquelle le serveur envoie des PDU `enquire_link` aux clients connectés pour vérifier qu'ils sont toujours actifs.

enabled

État d'activation du pair

Identique à la liaison client `enabled`. Les pairs de serveur désactivés n'acceptent pas les connexions entrantes.

cache_enabled

Activer la mise en cache locale des messages

Identique à la liaison client `cache_enabled`. Voir MESSAGE_CACHE.md.

cache_max_size

Nombre maximum de messages mis en cache

Identique à la liaison client `cache_max_size`.

cache_retry_interval

Intervalle de nouvelle tentative de base (secondes)

Identique à la liaison client `cache_retry_interval`.

deliver_receipts

Envoyer des accusés de réception de livraison SMPP (DLR) à cette liaison

Type	Requis	Par défaut
Booléen	Non	`false`

Objectif : La passerelle envoie un accusé de réception de livraison SMPP à cette liaison pour chaque message qu'elle soumet. La passerelle suit chaque

message soumis et envoie l'accusé lorsque le message atteint un état terminal. L'accusé est un PDU `deliver_sm` avec `esm_class` défini sur `0x04` (Accusé de réception de livraison SMSC), conformément à SMPP v3.4 section 5.2.12.

Quand l'utiliser : Définissez `deliver_receipts: true` pour une liaison d'interconnexion dédiée dont l'opérateur a besoin d'une confirmation de livraison pour chaque message. Gardez le défaut `false` pour les liaisons qui n'ont pas besoin d'accusés.

Exigence : La liaison DOIT fonctionner avec `cache_enabled: false`. L'accusé utilise l'identifiant de message que la passerelle a renvoyé dans le `submit_sm_resp`. Le cache renvoie un identifiant temporaire qui ne correspond pas à l'identifiant principal, donc une liaison mise en cache casse la corrélation des accusés. Si une liaison définit `deliver_receipts: true` et `cache_enabled: true`, la passerelle écrit un avertissement au démarrage et l'identifiant de l'accusé ne correspond pas.

Comment cela fonctionne : La liaison soumet un `submit_sm`. La passerelle accepte le message et renvoie un `submit_sm_resp` avec un identifiant de message. La passerelle suit le message et interroge le cœur d'OmniMessage pour son statut. Lorsque le message est livré, expiré ou rejeté, la passerelle envoie un accusé `deliver_sm` à la liaison. L'accusé porte le même identifiant de message et un champ d'état (`stat:DELIVRD`, `stat:EXPIRED`, `stat:UNDELIV`, ou `stat:REJECTD`).

Le tracker conserve l'état uniquement en mémoire. Si la passerelle redémarre, les accusés pour les messages en cours à ce moment-là sont perdus. C'est un mécanisme de meilleur effort.

Mapping des états aux accusés :

État principal	Accusé `stat`
`delivered`	`DELIVRD`
`expired`	`EXPIRED`
`dropped`	`UNDELIV`
`rejected`	`REJECTD`
`balance_rejected`	`REJECTD`

Voir USAGE.md pour le flux de message de bout en bout et MONITORING.md pour les métriques associées.

Exemple d'interface Web :

Configuration d'écoute du serveur

Lorsque les liaisons du serveur sont configurées, la passerelle écoute les connexions entrantes.

Exemple d'écoute complet

```
config :omnimessage_smpp, :listen, %{
  host: "0.0.0.0",
  port: 2775,
  max_connections: 100
}
```

Paramètres d'écoute

host

Adresse IP à laquelle se lier

Type	Requis	Par défaut	Valeurs courantes
Chaîne	Non	`"0.0.0.0"`	`"0.0.0.0"`, `"127.0.0.1"`

Objectif : Quelle interface réseau écouter.

Valeurs :

- `"0.0.0.0"` - Écouter sur toutes les interfaces (recommandé)
- `"127.0.0.1"` - Écouter uniquement sur localhost (test)
- `"192.168.1.10"` - Écouter sur une IP spécifique

port

Port TCP à écouter

Type	Requis	Par défaut	Plage
Entier	Non	2775	1-65535

Objectif : Port pour les connexions SMPP entrantes.

Standard : 2775

max_connections

Nombre maximum de connexions simultanées

Type	Requis	Par défaut	Plage
Entier	Non	100	1-10000

Objectif : Limite le nombre total de connexions clients simultanées.

Directives :

- Définir en fonction des clients attendus
- Des valeurs plus ��levées utilisent plus de mémoire
- Typique : 10-100 connexions

Exemples de configuration complets

Exemple 1 : Connexion à un seul opérateur

```
import Config

config :omnimessage_smpp,
  api_base_url: "https://smsc.company.com:8443",
  verify_ssl_peer: true,
  smsc_name: "smpp_prod"

config :omnimessage_smpp, :binds, [
  %{
    name: "att_primary",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.att.com",
    port: 2775,
    system_id: "company_user",
    password: "secure_pass_123",
    tps_limit: 100,
    queue_check_frequency: 1000
  }
]
```

Exemple 2 : Plusieurs opérateurs

```
import Config

config :omnimessage_smpp,
  api_base_url: "https://smsc.company.com:8443"

config :omnimessage_smpp, :binds, [
  # Amérique du Nord
  %{
    name: "att_us",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.att.com",
    port: 2775,
    system_id: "att_username",
    password: "att_password",
    tps_limit: 100,
    queue_check_frequency: 1000
  },

  # Europe
  %{
    name: "vodafone_uk",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.vodafone.co.uk",
    port: 2775,
    system_id: "voda_username",
    password: "voda_password",
    tps_limit: 50,
    queue_check_frequency: 1000
  }
]
```

Exemple 3 : Passerelle avec liaisons serveur

```
import Config

config :omnimessage_smpp,
  api_base_url: "https://smsc.company.com:8443"

# Connexions sortantes
config :omnimessage_smpp, :binds, [
  %{
    name: "upstream_carrier",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.carrier.com",
    port: 2775,
    system_id: "my_username",
    password: "my_password",
    tps_limit: 100,
    queue_check_frequency: 1000
  }
]

# Définitions des clients entrants
config :omnimessage_smpp, :server_binds, [
  %{
    name: "partner_alpha",
    system_id: "alpha_corp",
    password: "alpha_secret",
    allowed_bind_types: [:transmitter, :receiver, :transceiver],
    ip_whitelist: ["203.0.113.0/24"],
    tps_limit: 50,
    queue_check_frequency: 1000
  },
  %{
    name: "partner_beta",
    system_id: "beta_inc",
    password: "beta_password",
    allowed_bind_types: [:transceiver],
    ip_whitelist: ["198.51.100.50"],
    tps_limit: 25,
    queue_check_frequency: 2000
  }
]
```

```
# Écoute du serveur
config :omnimessage_smpp, :listen, %{
  host: "0.0.0.0",
  port: 2775,
  max_connections: 100
}
```

Exemple 4 : File d'attente partagée avec liaison source

Deux connexions au même opérateur, drainant toutes deux une file d'attente partagée (`sinch.smpp`) pour le partage de charge et la redondance. L'opérateur autorise par adresse IP **et** port source, donc chaque liaison fixe un port source local distinct sur la même IP source.

```
import Config

config :omnimessage_smpp,
  api_base_url: "https://smsc.company.com:8443"

config :omnimessage_smpp, :binds, [
  %{
    name: "sinch_1",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.sinch.com",
    port: 2775,
    system_id: "sinch_user",
    password: "sinch_pass",
    tps_limit: 100,
    queue_check_frequency: 1000,

    # Les deux liaisons drainent la même file d'attente backend
    queue: "sinch.smpp",

    # Adresse/port source fixés (l'opérateur autorise par les
deux)
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5001
  },
  %{
    name: "sinch_2",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.sinch.com",
    port: 2775,
    system_id: "sinch_user",
    password: "sinch_pass",
    tps_limit: 100,
    queue_check_frequency: 1000,

    # Même file d'attente partagée que sinch_1
    queue: "sinch.smpp",

    # Même IP source, port source distinct
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5002
```

```
  }
]
```

Comment cela fonctionne : Les deux liaisons enregistrent un seul frontend nommé `sinch.smpp` (leur clé de routage partagée). Les messages avec `dest_smsc: "sinch.smpp"` sont drainés par la liaison qui les récupère en premier, répartissant la charge entre les deux connexions. Si une connexion tombe, l'autre continue de drainer la file d'attente. Chaque connexion présente une IP et un port source fixes à l'opérateur pour l'autorisation ; `SO_REUSEADDR` permet à un port fixé de se rétablir immédiatement lors de la reconnexion.

Voir `queue`, `bind_source_ip`, et `bind_source_port` pour les détails des paramètres.

Validation de la configuration

Après avoir modifié la configuration, validez avant de redémarrer :

Vérification de la syntaxe

```
# Vérifier la syntaxe Elixir
/opt/omnimessage-smpp/bin/omnimessage-smpp eval "File.read!
('config/runtime.exs')"
```

Si la syntaxe est invalide, vous verrez une erreur. Corrigez avant de redémarrer.

Tester la configuration

```
# Redémarrer au premier plan pour voir les erreurs
sudo -u omnimessage-smpp /opt/omnimessage-smpp/bin/omnimessage-
smpp console
```

Appuyez sur `Ctrl+C` deux fois pour quitter.

Variables d'environnement

Tous les paramètres globaux peuvent être remplacés par des variables d'environnement. Définissez-les dans votre fichier d'unité systemd ou dans l'environnement de votre shell avant de démarrer la passerelle.

Variable d'environnement	Clé de configuration	Par de
`API_BASE_URL`	`api_base_url`	`"https://10.17`
`SMSC_NAME`	`smsc_name`	`"smpp_gateway"`
`SMPP_POLL_INTERVAL`	`smpp_poll_interval`	`100`
`VERIFY_SSL_PEER`	`verify_ssl_peer`	`false`
`CACHE_FLUSH_INTERVAL`	`cache_flush_interval`	`10000`
`CACHE_MAX_RETRY_ATTEMPTS`	`cache_max_retry_attempts`	`10`

Variable d'environnement	Clé de configuration	Par de
`CACHE_BACKOFF_MULTIPLIER`	`cache_backoff_multiplier`	`2`
`MNESIA_STORAGE_TYPE`	`mnesia_storage_type`	`disc_copies`

Exemple de remplacement systemd :

```
sudo systemctl edit omnimessage-smpp
```

```
[Service]
Environment="API_BASE_URL=https://omnimessage-
core.company.com:8443"
Environment="SMSC_NAME=smpp_prod_01"
Environment="VERIFY_SSL_PEER=true"
```

Meilleures pratiques de sécurité

1. **Protéger le fichier de configuration** :

```
sudo chmod 600 /opt/omnimessage-smpp/config/runtime.exs
sudo chown omnimessage-smpp:omnimessage-smpp /opt/omnimessage-smpp/config/runtime.exs
```

2. **Utiliser des mots de passe forts** :
   - Minimum 12 caractères
   - Mélanger lettres, chiffres, symboles
   - Unique par connexion
3. **Utiliser des listes blanches IP** :
   - Toujours configurer `ip_whitelist` pour les liaisons serveur
   - Ne jamais utiliser une liste vide `[]` en production
4. **Activer la vérification SSL** :
   - Définir `verify_ssl_peer: true` avec des certificats valides
5. **Rotation régulière des identifiants** :
   - Changer les mots de passe tous les trimestres
   - Coordonner avec les opérateurs/partenaires

Prochaines étapes

- Consultez MONITORING.md pour la configuration des métriques
- Lisez USAGE.md pour gérer les connexions
- Consultez TROUBLESHOOTING.md pour les problèmes courants
- Retournez à README.md pour un aperçu

Glossaire

Termes et Définitions

A

API (Interface de Programmation d'Application) Interface utilisée pour communiquer avec le système backend de la file de messages.

Défilement Automatique Fonction dans l'onglet Journaux de l'interface web qui fait défiler automatiquement pour afficher les nouvelles entrées de journal.

B

Backend Le système de file de messages auquel la passerelle SMPP se connecte pour récupérer et stocker des messages.

Bind Une connexion SMPP entre deux systèmes. Peut être émetteur, récepteur ou transcepteur.

Type de Bind Le type de session SMPP :

- **Émetteur** : Envoie uniquement des messages
- **Récepteur** : Reçoit uniquement des messages
- **Transcepteur** : Envoie et reçoit des messages

Échec de Bind Lorsque tentatives d'authentification SMPP échouent, généralement en raison de mauvaises informations d'identification ou de restrictions IP.

C

CIDR (Routage Inter-Domaines Sans Classe) Notation pour spécifier des plages d'adresses IP (par exemple, `192.168.1.0/24` représente 256 adresses

IP).

Bind Client Une connexion SMPP sortante où la passerelle agit comme un **ESME** se connectant à un **SMSC** externe (généralement le serveur SMPP d'un opérateur). Dans ce mode, la passerelle est le client.

État de Connexion État actuel d'un bind SMPP :

- **Connecté** : Actif et opérationnel
- **Déconnecté** : Non connecté
- **Reconnecté** : Tentative d'établissement de la connexion

Compteur Une métrique qui n'augmente que (se réinitialise au redémarrage du service), utilisée pour des totaux comme les messages envoyés.

D

Codage de Données Champ SMPP spécifiant l'encodage des caractères du message (GSM-7, UCS-2, etc.).

Deliver_SM PDU SMPP envoyé par un SMSC (serveur) pour livrer un message à un ESME (client) connecté. Utilisé par les binds serveurs pour pousser des messages vers des partenaires connectés.

Échec de Livraison Lorsque qu'un message ne peut pas être livré, indiqué par une réponse d'erreur de l'opérateur.

Accusé de Réception de Livraison (DLR) Confirmation de l'opérateur concernant l'état de livraison du message.

deliver_time Champ dans la file de messages contenant l'horodatage auquel un message a été livré. Défini par OmniMessage Core lorsque la passerelle signale une livraison réussie. C'est le nom du champ que le backend attend lors de la marque d'un message comme livré.

dest_smsc Champ dans la file de messages indiquant quelle connexion SMPP doit gérer le message. Sa valeur doit être égale à la **Clé de Routage** sous laquelle le pair cible s'est enregistré (voir Enregistrement Frontend).

Déconnexion Lorsque une connexion SMPP active est terminée, soit intentionnellement, soit en raison d'une erreur.

E

Enquête de Lien Message keepalive SMPP envoyé périodiquement pour vérifier que la connexion est active.

Classe ESM Champ SMPP indiquant le type de message et les fonctionnalités.

ESME (Entité de Message Court Externe) Dans la terminologie SMPP, l'application cliente qui se connecte à un SMSC pour envoyer ou recevoir des messages. Lorsque la passerelle fonctionne en **Mode Client**, elle agit comme un ESME se connectant aux SMSC des opérateurs. Lorsqu'elle fonctionne en **Mode Serveur**, elle accepte les connexions des ESME externes.

Retard Exponentiel Stratégie de réessai où le temps d'attente double après chaque échec (1min, 2min, 4min, 8min...).

F

Pare-feu Système de sécurité réseau qui contrôle le trafic réseau entrant et sortant.

Enregistrement Frontend Processus par lequel la passerelle SMPP s'enregistre auprès d'OmniMessage Core. Un heartbeat est envoyé toutes les 60 secondes pour maintenir l'enregistrement actif. Si la passerelle s'arrête, l'enregistrement expire après 90 secondes et OmniMessage cesse de router des messages vers elle. Chaque pair est enregistré sous sa **Clé de Routage** donc le nom enregistré est égal au `dest_smsc` sur lequel le backend route : pairs clients sous leur `queue` (ou nom de pair nu), binds serveurs sous leur nom de bind.

G

Passerelle L'application de la passerelle SMPP qui fait le lien entre la file de messages et les réseaux mobiles.

Jauge Une métrique qui peut augmenter ou diminuer, représentant la valeur actuelle (par exemple, état de connexion).

Grafana Outil de visualisation populaire pour afficher les métriques Prometheus dans des tableaux de bord.

GSM-7 Encodage de caractères standard à 7 bits pour les SMS, supportant jusqu'à 160 caractères par message.

H

HTTP/HTTPS Protocoles utilisés pour la communication web. HTTPS est la version cryptée.

I

Liste Blanche IP Liste des adresses IP autorisées à se connecter à la passerelle (fonction de sécurité).

ISDN (Réseau Numérique à Intégration de Services) Plan de numérotation couramment utilisé pour les numéros de téléphone.

J

(Pas de termes)

K

Keepalive Messages périodiques (enquire_link) envoyés pour maintenir la connexion et détecter les échecs.

KPI (Indicateur Clé de Performance) Valeur mesurable indiquant la performance du système (par exemple, taux de réussite de livraison).

L

Étiquette Dans Prometheus, paires clé-valeur attachées aux métriques pour identification (par exemple, `bind_name="vodafone_uk"`).

LiveView Technologie du framework Phoenix utilisée pour les mises à jour en temps réel de l'interface web.

M

File de Messages Système backend qui stocke les messages en attente d'être envoyés ou reçus.

Métriques Mesures quantitatives de la performance du système, exposées au format Prometheus.

MO (Mobile Originaire) Messages envoyés depuis des téléphones mobiles vers la passerelle (entrant).

MT (Mobile Terminé) Messages envoyés de la passerelle vers des téléphones mobiles (sortant).

MSISDN (Numéro de Répertoire International de l'Abonné Mobile) Format standard pour les numéros de téléphone mobile.

N

NPI (Indicateur de Plan de Numérotation) Champ SMPP spécifiant le schéma de numérotation (par exemple, ISDN).

O

Sortant Messages circulant de la passerelle vers les réseaux mobiles.

Entrant Messages circulant des réseaux mobiles vers la passerelle.

P

PDU (Unité de Données de Protocole) Paquet de message SMPP individuel (par exemple, submit_sm, deliver_sm).

Prometheus Système de surveillance open-source qui collecte et stocke des métriques de séries temporelles.

Q

File Liste de messages en attente d'être traités ou envoyés.

Fréquence de Vérification de la File À quelle fréquence (en millisecondes) la passerelle interroge le backend pour de nouveaux messages.

Travailleur de File Composant qui récupère des messages de la file et les envoie via SMPP.

R

Limitation de Débit Contrôle du débit de messages pour se conformer aux restrictions des opérateurs. Voir TPS.

Récepteur Type de bind SMPP qui ne reçoit que des messages (deliver_sm).

Reconnecter Rétablir une connexion SMPP déconnectée.

Réessayer Tenter d'envoyer à nouveau un message échoué, généralement avec un retard exponentiel.

Clé de Routage Le nom qu'un pair enregistre en tant que frontend auprès d'OmniMessage Core, et donc la valeur `dest_smsc` qui route les messages vers lui. Pour un pair client, c'est sa `queue` configurée (ou le nom de pair nu lorsqu'aucune `queue` n'est définie) ; pour un bind serveur, c'est le nom de bind.

S

Numéro de Séquence Identifiant numérique unique attribué à chaque PDU SMPP dans une session. Utilisé pour faire correspondre les demandes avec leurs réponses (par exemple, faire correspondre un submit_sm avec son submit_sm_resp).

Bind Serveur Configuration qui permet aux **ESMEs** externes (clients) de se connecter à la passerelle. Dans ce mode, la passerelle agit comme un **SMSC** (serveur) acceptant les connexions entrantes des systèmes partenaires.

Session Connexion SMPP active entre deux systèmes.

File Partagée Une seule file backend drainée par plus d'un bind client. Les binds optent en définissant la même valeur `queue` dans leur configuration (par exemple, deux connexions d'opérateur drainant toutes deux `"sinch.smpp"`), permettant le partage de charge et la redondance entre les connexions. Par défaut, cela correspond au nom propre du bind, auquel cas chaque bind a sa propre file.

Code Court Une adresse d'origine/destination SMS raccourcie (généralement 5 à 6 chiffres) utilisée pour la messagerie application-à-personne. Les destinations `deliver_sm` à cinq chiffres sont envoyées avec TON **Inconnu** afin que les opérateurs ne les interprètent pas à tort comme des numéros internationaux.

Bind Source Fixation de l'adresse IP source locale et/ou du port TCP d'une connexion sortante (client) via les paramètres `bind_source_ip` et

`bind_source_port`. Utilisé lorsqu'un opérateur met sur liste blanche les connexions par IP et port source. Voir CONFIGURATION.md.

source_smsc Champ dans la file de messages indiquant quel bind serveur doit livrer le message à ses clients connectés via deliver_sm.

SMPP (Peer-à-Peer de Message Court) Protocole standard de l'industrie pour l'échange de messages SMS entre systèmes.

SMSC (Centre de Service de Message Court) Dans la terminologie SMPP, le composant serveur qui accepte les connexions des ESMEs (clients) et gère le routage et la livraison des messages SMS. Lorsque la passerelle fonctionne en **Mode Serveur**, elle agit comme un SMSC acceptant les connexions des ESME externes.

SSL/TLS Protocoles de cryptage pour une communication sécurisée.

Submit_SM PDU SMPP pour soumettre un message pour livraison.

Submit_SM_Resp Réponse SMPP à submit_sm, indiquant le succès ou l'échec.

ID Système Nom d'utilisateur utilisé pour l'authentification SMPP.

T

Télémétrie Collecte et transmission automatisées des métriques du système.

TON (Type de Numéro) Champ SMPP spécifiant le format du numéro (par exemple, international, national).

TPS (Transactions Par Seconde) Limite de débit pour le nombre maximum de messages par seconde à travers une connexion.

Transcepteur Type de bind SMPP qui peut à la fois envoyer et recevoir des messages (le plus courant).

Émetteur Type de bind SMPP qui envoie uniquement des messages (submit_sm).

Débit Taux de traitement des messages, généralement mesuré en messages par seconde.

U

UCS-2 Encodage de caractères Unicode à 16 bits pour les SMS, supportant jusqu'à 70 caractères par message.

Uptime Durée pendant laquelle une connexion ou un service a été opérationnel en continu.

V

Période de Validité Limite de temps pour la tentative de livraison d'un message avant expiration.

W

Tableau de Bord Web Interface utilisateur basée sur un navigateur pour surveiller et gérer la passerelle.

Liste Blanche Voir Liste Blanche IP et Liste Blanche d'Adresse Source.

X

(Pas de termes)

Y

(Pas de termes)

Z

(Pas de termes)

Référence Rapide des Acronymes

Acronyme	Terme Complet
API	Interface de Programmation d'Application
CIDR	Routage Inter-Domaines Sans Classe
DLR	Accusé de Réception de Livraison
ESME	Entité de Message Court Externe
GSM	Système Mondial de Communications Mobiles
HTTP	Protocole de Transfert Hypertexte
HTTPS	Protocole de Transfert Hypertexte Sécurisé
IP	Protocole Internet
ISDN	Réseau Numérique à Intégration de Services
KPI	Indicateur Clé de Performance
MO	Mobile Originaire
MSISDN	Numéro de Répertoire International de l'Abonné Mobile
MT	Mobile Terminé
NPI	Indicateur de Plan de Numérotation
PDU	Unité de Données de Protocole
SMPP	Peer-à-Peer de Message Court
SMSC	Centre de Service de Message Court

Acronyme	Terme Complet
SMS	Service de Message Court
SSL	Couche de Sockets Sécurisée
TLS	Sécurité de la Couche de Transport
TON	Type de Numéro
TPS	Transactions Par Seconde
UCS	Ensemble de Caractères Codés Universel
UI	Interface Utilisateur
URL	Localisateur Uniforme de Ressource

Documentation Connexe

- **README.md** - Vue d'ensemble du système et démarrage
- **CONFIGURATION.md** - Paramètres de configuration expliqués
- **USAGE.md** - Opérations quotidiennes
- **MONITORING.md** - Métriques et surveillance
- **TROUBLESHOOTING.md** - Résolution de problèmes

Cache de Messages SMPP

Aperçu

Le Cache de Messages SMPP est une couche de persistance locale qui permet à la passerelle SMPP de continuer à accepter des messages entrants même lorsque l'API backend est indisponible. Les messages sont mis en cache localement dans Mnesia et livrés automatiquement à l'API lorsque la connectivité est rétablie, en utilisant une logique de réessai intelligente avec un backoff exponentiel.

Fonctionnalités

- **Acceptation Résiliente des Messages** - Continue d'accepter les messages SMPP pendant les pannes de l'API
- **Stockage Persistant** - Utilise Mnesia avec `:disc_copies` pour la durabilité à travers les redémarrages
- **Réessai Automatique** - Les travailleurs en arrière-plan tentent automatiquement la livraison avec un backoff exponentiel
- **Configuration par Bind** - Activer/désactiver le caching indépendamment pour chaque bind SMPP
- **Protection contre le Débordement** - Éviction FIFO lorsque le cache atteint la limite de taille configurée
- **Conservation des Messages Échoués** - Les messages échoués de manière permanente sont conservés pour révision manuelle
- **Surveillance en Temps Réel** - Tableau de bord LiveView avec statistiques et métriques de cache
- **Métriques Prometheus** - Exportation complète des métriques pour la surveillance et les alertes

Architecture

Flux de Messages

Avec Cache Activé

```
Client SMPP → submit_sm → Serveur Passerelle
                                ↓
                        Cache dans Mnesia (réponse immédiate)
                                ↓
                        CacheFlushWorker (arrière-plan)
                                ↓
                          API Backend
```

Avec Cache Désactivé

```
Client SMPP → submit_sm → Serveur Passerelle
                                ↓
                          API Backend (direct)
```

Composants

1. **Module MessageCache**
    - Logique de caching principale
    - Gestion du débordement
    - Fonctions de requête pour LiveView et travailleurs
2. **CacheFlushWorker**
    - GenServer par bind avec caching activé
    - Interroge les messages prêts pour réessai
    - Implémente un backoff exponentiel
    - Marque les messages échoués de manière permanente
3. **Table Mnesia** (`:smpp_message_cache`)

- Stockage persistant avec `:disc_copies`
- Indexé par bind_name, next_retry_at, et status
- Survit aux redémarrages d'application

Configuration

Paramètres Globaux

Modifier `config/runtime.exs` :

```
config :omnimessage_smpp,
  # À quelle fréquence les travailleurs de flush interrogent les
messages (millisecondes)
  cache_flush_interval: 10_000,

  # Nombre maximum de tentatives de réessai avant de marquer comme
échoué_permanent
  cache_max_retry_attempts: 10,

  # Multiplicateur de backoff exponentiel
  cache_backoff_multiplier: 2,

  # Type de stockage Mnesia (:disc_copies ou :ram_copies)
  mnesia_storage_type: :disc_copies
```

Configuration par Bind

Chaque bind SMPP (client ou serveur) peut être configuré indépendamment :

```
config :omnimessage_smpp, :binds, [
  %{
    name: "my_smpp_bind",
    mode: :server,
    system_id: "username",
    password: "password",

    # Configuration du cache
    cache_enabled: true,         # Activer le caching (par défaut
: false)
    cache_max_size: 10_000,      # Max messages à mettre en cache
(par défaut : 10 000)
    cache_retry_interval: 60     # Intervalle de réessai de base
en secondes (par défaut : 60)
  }
]
```

Variables d'Environnement

```
# Paramètres globaux du cache
CACHE_FLUSH_INTERVAL=10000              # Intervalle de flush (ms)
CACHE_MAX_RETRY_ATTEMPTS=10             # Max réessais avant échec
permanent
CACHE_BACKOFF_MULTIPLIER=2              # Multiplicateur de
backoff exponentiel
MNESIA_STORAGE_TYPE=disc_copies         # Type de stockage Mnesia
```

Comportement de Réessai

Backoff Exponentiel

Lorsque la livraison d'un message échoue, l'intervalle de réessai double à chaque tentative :

```
Intervalle de base : 60 secondes
Multiplicateur de backoff : 2

Réessai 0 : 60s   (1 minute)
Réessai 1 : 120s  (2 minutes)
Réessai 2 : 240s  (4 minutes)
Réessai 3 : 480s  (8 minutes)
Réessai 4 : 960s  (16 minutes)
Réessai 5 : 1920s (32 minutes)
...
Réessai 9 : 30 720s (8,5 heures)
```

Échec Permanent

Après 10 tentatives échouées (par défaut), les messages sont marqués comme `failed_permanent` et :

- Restent dans le cache pour révision manuelle
- Cessent d'être réessayés automatiquement
- Apparaissent dans la section "Échoué Permanent" du tableau de bord du cache
- Peuvent être réessayés ou supprimés manuellement

Transitions d'État

```
:pending → :delivering → SUCCESS (supprimé du cache)
                       → FAILURE → :pending (réessai avec backoff)
                                 → :failed_permanent (après max
réessais)
```

Surveillance

Tableau de Bord LiveView

Acc��dez au tableau de bord du cache à `http://your-server:4000/smpp` → onglet "Cache de Messages"

Cartes Résumé :

- Total Mis en Cache - Tous les messages actuellement dans le cache
- Livraison en Attente - Messages en attente de réessai
- Échoué Permanent - Messages ayant dépassé le nombre max de réessais

Tableau par Bind :

- Nom du bind
- Nombre de messages mis en cache
- Répartition en Attente / Échoués
- Taille max du cache
- Pourcentage d'utilisation (avec barre de progression visuelle)

Métriques Prometheus

```
# Taille actuelle du cache par bind
smpp_cache_size{bind_name="my_bind",mode="server"} 42

# Total des messages livrés avec succès
smpp_cache_delivered_total{bind_name="my_bind"} 1234

# Total des tentatives de réessai
smpp_cache_retry_total{bind_name="my_bind"} 56

# Total des échecs permanents
smpp_cache_permanent_failures_total{bind_name="my_bind"} 2

# Total des événements de débordement (messages supprimés)
smpp_cache_overflow_total{bind_name="my_bind"} 0
```

Messages de Journal

```
INFO  Message -123456789 mis en cache pour my_smpp_bind
INFO  Message mis en cache -123456789 livré avec succès, ID API :
99999
WARN  Échec de la livraison du message -123456789 (réessai 3/10),
prochain réessai à 2026-02-01 12:34:56Z
ERROR Message -123456789 a dépassé le nombre max de réessais (10),
marqué comme échoué_permanent
WARN  Débordement de cache pour my_smpp_bind : message le plus
ancien supprimé
```

Opérations

Activer/Désactiver le Caching

Via l'UI LiveView

1. Naviguez vers `http://your-server:4000/smpp`
2. Allez à l'onglet "Pairs Clients" ou "Pairs Serveurs"
3. Éditez le bind
4. Basculez la case "Cache Activé"
5. Enregistrez les modifications

Via la Console IEx

```
# Activer le caching pour un bind
SmsC.SMPPConfig.update_server_peer("my_bind", "username",
"password",
  cache_enabled: true,
  cache_max_size: 10_000,
  cache_retry_interval: 60
)

# Désactiver le caching
SmsC.SMPPConfig.update_server_peer("my_bind", "username",
"password",
  cache_enabled: false
)
```

Surveiller l'État du Cache

```
# Obtenir le résumé global
SmsC.SMPP.MessageCache.get_cache_summary()
# => %{total_cached: 42, pending: 40, failed: 2}

# Obtenir la répartition par bind
SmsC.SMPP.MessageCache.get_cache_by_bind()
# => [
#   %{bind_name: "bind1", total: 30, pending: 28, failed: 2,
max_size: 10_000},
#   %{bind_name: "bind2", total: 12, pending: 12, failed: 0,
max_size: 10_000}
# ]

# Compter les messages pour un bind spécifique
SmsC.SMPP.MessageCache.count_cached_messages("my_bind")
# => 42
```

Interventions Manuelles

Effacer les Messages Échoués

```
# Obtenir tous les messages échoués pour un bind
{:atomic, failed_messages} = :mnesia.transaction(fn ->
  :mnesia.match_object({:smpp_message_cache, :_, "my_bind", :_,
:_, :_, :_, :_, :_, :failed_permanent})
end)

# Les supprimer
Enum.each(failed_messages, fn {_, {bind_name, msg_id}, _, _, _, _,
_, _, _, _} ->
  SmsC.SMPP.MessageCache.delete_cache_record(bind_name, msg_id)
end)
```

Forcer le Réessai d'un Message Échoué

```
# Réinitialiser un message échoué_permanent à en attente
SmsC.SMPP.MessageCache.update_cache_record("my_bind", -123456, %{
  status: :pending,
  retry_count: 0,
  next_retry_at: DateTime.utc_now(),
  last_error: nil
})
```

Dépannage

Cache Plein / Événements de Débordement

Symptôme : La métrique `cache_overflow_total` augmente, les messages les plus anciens sont supprimés

Cause : Limite de taille du cache atteinte

Solutions :

1. Augmenter `cache_max_size` pour le bind
2. Enquêter sur les raisons pour lesquelles la livraison de l'API échoue (vérifier les journaux de l'API, le réseau)
3. Effacer manuellement les anciens messages échoués

4. Vérifier si l'intervalle de flush est trop lent

Messages Non Livrés

Symptôme : Messages bloqués dans l'état `:pending`

Causes Possibles :

1. **L'API est hors service**
   - Vérifiez la disponibilité de l'API
   - Vérifiez les journaux de l'API backend
   - Vérifiez la connectivité réseau
2. **next_retry_at est dans le futur**
   - Les messages seront réessayés lorsque `next_retry_at` sera atteint
   - Vérifiez le calendrier de backoff exponentiel
3. **Le travailleur de flush ne fonctionne pas**

   ```
   # Vérifiez si les travailleurs fonctionnent
   Supervisor.which_children(SmsC.SMPP.Supervisor)
   ```

4. **Cache désactivé**
   - Vérifiez `cache_enabled: true` dans la configuration du bind

Comptes de Réessai Élevés

Symptôme : De nombreux messages avec des valeurs `retry_count` élevées

Enquête :

```
# Trouver des messages avec des comptes de réessai élevés
{:atomic, messages} = :mnesia.transaction(fn ->
  :mnesia.match_object({:smpp_message_cache, :_, "my_bind", :_,
:_, :_, :_, :_, :_, :_})
end)

high_retry = Enum.filter(messages, fn {_, _, _, _, _, _,
retry_count, _, _, _} ->
  retry_count >= 5
end)

# Vérifiez last_error pour chacun
Enum.each(high_retry, fn {_, _, _, msg_id, _, _, retry_count, _,
last_error, _} ->
  IO.puts("Message #{msg_id} : #{retry_count} réessais, erreur : #
{inspect(last_error)}")
end)
```

Espace Disque Mnesia

Symptôme : Espace disque qui se remplit

Vérifiez le répertoire Mnesia :

```
ls -lh Mnesia.*
du -sh Mnesia.*
```

Nettoyage :

1. Effacer les anciens messages échoués (voir Interventions Manuelles ci-dessus)
2. Réduire `cache_max_size` par bind
3. Activer le débordement de cache (assurez-vous d'une éviction FIFO appropriée)

Considérations de Performance

Utilisation de la Mémoire

- Chaque message mis en cache utilise environ 500-1000 octets (selon la taille du message)
- 10 000 messages ≈ 5-10 Mo de mémoire
- Avec `:disc_copies`, les données sont également écrites sur disque

Utilisation du CPU

- Les travailleurs de flush interrogent toutes les 10 secondes par défaut (configurable)
- Le traitement par lots (100 messages par cycle) réduit la surcharge
- Livraison concurrente (max 10 appels API concurrents par travailleur)

I/O Disque

- `:disc_copies` écrit sur disque à chaque transaction
- Pour un débit très élevé (>1000 msg/sec), envisagez :
  - D'utiliser `:ram_copies` (perte de persistance)
  - D'augmenter les intervalles de flush
  - De mettre à l'échelle horizontalement

Limites Recommandées

Scénario	cache_max_size	cache_flush_interval
Faible volume (<100 msg/sec)	10 000	10 000ms
Volume moyen (100-500 msg/sec)	50 000	5 000ms
Volume élevé (>500 msg/sec)	100 000	3 000ms

Scénarios de Récupération

Redémarrage de l'Application

1. Mnesia charge automatiquement les tables `:disc_copies` depuis le disque
2. Les messages mis en cache restent intacts
3. Les travailleurs de flush redémarrent et continuent le traitement

Migration de Base de Données

Lors de la mise à niveau d'une version sans support de cache :

1. La migration ajoute automatiquement des champs de cache aux binds existants
2. Valeurs par défaut : `cache_enabled: false`, `cache_max_size: 10 000`, `cache_retry_interval: 60`
3. Pas de perte de données
4. La table de cache est créée au premier démarrage

Récupération après Panne de l'API

1. Les messages s'accumulent dans le cache pendant la panne
2. Lorsque l'API se rétablit, les travailleurs de flush livrent automatiquement
3. Les messages les plus anciens sont livrés en premier (FIFO)
4. Le backoff exponentiel empêche la surcharge de l'API pendant la récupération

Meilleures Pratiques

1. **Activer le Caching par Défaut** - Empêche la perte de messages pendant les pannes
2. **Surveiller les Métriques** - Configurer des alertes sur `cache_permanent_failures_total` et `cache_overflow_total`

3. **Dimensionner de Manière Appropriée** - Définir `cache_max_size` en fonction de la durée d'attente prévue
4. **Réviser les Messages Échoués** - Vérifier régulièrement les messages échoués_permanent pour détecter des motifs
5. **Tester le Failover** - Simuler des pannes d'API pour vérifier le comportement du cache
6. **Ajuster les Intervalles de Réessai** - Affiner en fonction des modèles de temps de récupération de l'API
7. **Utiliser un Stockage Persistant** - Garder `mnesia_storage_type: disc_copies` en production

Voir Aussi

- Référence de Configuration
- Surveillance et Métriques
- Dépannage

Guide de Surveillance et de Métriques

Référence complète pour la surveillance de la passerelle SMPP

Vue d'ensemble

La passerelle SMPP expose des métriques au format Prometheus pour surveiller la santé des connexions, le débit des messages et la performance du système.

Critique : Étant donné que la passerelle est sans état et dépend d'OmniMessage Core, **la connectivité OmniMessage est la métrique la plus importante à surveiller**. Surveillez les deux :

1. **Métriques de la passerelle SMPP** - Santé au niveau du protocole
2. **Métriques de l'API OmniMessage** - Connectivité et santé en arrière-plan

Point de terminaison des métriques

URL : `http://your-server:4000/metrics`

Format : Format texte Prometheus

Accès : Ouvert à localhost par défaut (configurer le pare-feu pour un accès distant)

Test rapide

```
curl http://localhost:4000/metrics
```

Métriques Disponibles

Toutes les métriques sont préfixées par `smpp_` et incluent des étiquettes pour l'identification.

Métriques de Licence

omnimessage_smpp_license_status

Type : Gauge
Description : État actuel de la licence
Valeurs :

- `1` = Licence valide
- `0` = Licence invalide/expirée

Étiquettes : Aucune

Exemple :

```
omnimessage_smpp_license_status 1
```

Utilisation :

- Alerter lorsque la valeur est 0 (licence invalide)
- Lorsque la licence est invalide, le traitement de la file d'attente sortante s'arrête mais les liaisons SMPP restent connectées
- L'interface Web reste accessible pour le dépannage

Nom du produit : `omnimessage_smpp`

Remarques :

- Lorsque la licence est invalide (`license_status == 0`), la passerelle cesse de traiter les files d'attente sortantes
- Les liaisons SMPP (client et serveur) restent connectées et acceptent les demandes de liaison

- Les messages entrants sont toujours reçus mais non traités
- L'interface utilisateur et la surveillance restent accessibles indépendamment de l'état de la licence

Exemple d'alerte :

```
- alert: SMPP_License_Invalid
  expr: omnimessage_smpp_license_status == 0
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "Licence de la passerelle SMPP invalide ou expirée"
    description: "L'état de la licence est invalide - le
traitement des messages sortants est bloqué"
```

Métriques de Statut de Connexion

smpp_connection_status

Type : Gauge
Description : État actuel de la connexion de liaison SMPP
Valeurs :

- `1` = Connecté. Pour une liaison serveur, au moins une session ESME est liée.
- `0` = Déconnecté. Pour une liaison serveur, aucune session ESME n'est liée.

Étiquettes :

- `bind_name` - Nom de la connexion (par exemple, "vodafone_uk")
- `mode` - Type de connexion ("client" ou "serveur")
- `host` - Hôte distant (mode client uniquement)
- `port` - Port distant (mode client uniquement)
- `bind_type` - Type de liaison SMPP (mode client uniquement)
- `system_id` - ID système utilisé

Exemple :

```
smpp_connection_status{bind_name="vodafone_uk",mode="client",host="sm
1
```

Utilisation :

- Alerter lorsque la valeur est 0 (déconnecté)
- Suivre le pourcentage de disponibilité de la connexion
- Surveiller la fréquence de reconnexion

Compteurs de Messages

smpp_messages_sent_total

Type : Counter
Description : Nombre total de messages envoyés via la liaison SMPP : `deliver_sm` à l'ESME sur une liaison serveur, `submit_sm` à l'opérateur sur une liaison client
Unité : PDU SMPP. Chaque partie d'un message multipart compte une fois.

Étiquettes : Identiques à connection_status

Exemple :

```
smpp_messages_sent_total{bind_name="vodafone_uk",mode="client",...}
150234
```

Utilisation :

- Calculer le taux de messages (messages/seconde)
- Suivre le volume quotidien/mensuel
- Comparer le débit réel au débit attendu

smpp_messages_received_total

Type : Counter
Description : Nombre total de messages reçus via la liaison SMPP : `submit_sm` de l'ESME sur une liaison serveur, `deliver_sm` d'origine mobile de l'opérateur sur une liaison client. Les accusés de réception de livraison ne sont pas comptés.
Unité : PDU SMPP

Étiquettes : Identiques à connection_status

Exemple :

```
smpp_messages_received_total{bind_name="partner_acme",mode="server",..
45123
```

Utilisation :

- Surveiller le volume de messages entrants
- Suivre le trafic d'origine mobile (MO)
- Alerter sur des changements de volume inattendus

Métriques de Livraison

smpp_delivery_failures_total

Type : Counter
Description : Nombre total d'échecs de livraison de messages : un `deliver_sm` ou `submit_sm` qui n'a pas pu être envoyé, ou qui a été répondu avec un statut non nul
Unité : PDU SMPP

Étiquettes : Identiques à connection_status

Exemple :

```
smpp_delivery_failures_total{bind_name="vodafone_uk",mode="client",..
234
```

Utilisation :

- Calculer le taux de succès de livraison
- Alerter sur des taux d'échec élevés
- Identifier les connexions problématiques

Calcul du Taux de Succès :

```
success_rate = (messages_sent - delivery_failures) / messages_sent
* 100
```

Métriques d'Opération de Liaison

smpp_bind_success_total

Type : Counter
Description : Nombre total d'opérations de liaison réussies
Unité : Tentatives de liaison

Exemple :

```
smpp_bind_success_total{bind_name="vodafone_uk",...} 45
```

Utilisation :

- Suivre la stabilité des liaisons
- Surveiller le succès de l'authentification

smpp_bind_failures_total

Type : Counter
Description : Nombre total d'opérations de liaison échouées
Unité : Tentatives de liaison

Exemple :

```
smpp_bind_failures_total{bind_name="vodafone_uk",...} 3
```

Utilisation :

- Alerter sur les échecs d'authentification
- Identifier les problèmes de crédential
- Suivre les problèmes de connexion avec l'opérateur

Métriques d'Événements de Connexion

smpp_connection_attempts_total

Type : Counter
Description : Nombre total de tentatives de connexion. Sur une liaison serveur, le nombre de sessions ESME qui ont été liées.
Unité : Tentatives

Exemple :

```
smpp_connection_attempts_total{bind_name="vodafone_uk",...} 48
```

Utilisation :

- Suivre le renouvellement des connexions
- Surveiller la fréquence de reconnexion

smpp_disconnection_total

Type : Counter
Description : Nombre total de déconnexions. Sur une liaison serveur, le nombre de sessions ESME liées qui ont pris fin.
Unité : Déconnexions

Exemple :

```
smpp_disconnection_total{bind_name="vodafone_uk",...} 3
```

Utilisation :

- Alerter sur des déconnexions fréquentes
- Identifier les problèmes de réseau
- Suivre la stabilité des connexions

smpp_reconnect_attempts_total

Type : Counter
Description : Nombre total de reconnexions programmées pour une liaison client après une connexion échouée, une liaison rejetée ou une session interrompue
Unité : Tentatives de reconnexion

Exemple :

```
smpp_reconnect_attempts_total{bind_name="vodafone_uk",...} 7
```

Utilisation :

- Identifier un opérateur qui accepte la connexion puis la coupe immédiatement (compteur en forte hausse tandis que `smpp_connection_status` reste `0`)
- Mesurer combien de temps une liaison a passé à essayer de revenir après une panne

---

Métriques de Lien de Demande

smpp_enquire_link_sent_total

Type : Counter
Description : Nombre total de PDUs enquire_link envoyés pour vérifier la vivacité de la connexion
Unité : PDUs

Étiquettes : Identiques à connection_status

Exemple :

```
smpp_enquire_link_sent_total{bind_name="vodafone_uk",mode="client",..
1440
```

Utilisation :

- Suivre l'activité de maintien de la connexion
- Comparer avec les reçus pour détecter des échecs unidirectionnels

smpp_enquire_link_received_total

Type : Counter
Description : Nombre total de PDUs enquire_link_resp reçus du pair distant
Unité : PDUs

Étiquettes : Identiques à connection_status

Exemple :

```
smpp_enquire_link_received_total{bind_name="vodafone_uk",mode="client
1438
```

Utilisation :

- Détecter des pairs non réactifs (envoyés >> reçus)
- Surveiller la santé de la connexion au-delà d'un simple statut

---

Métriques de Disponibilité

smpp_uptime_seconds

Type : Gauge
Description : Temps de disponibilité actuel de la liaison SMPP en secondes
Unité : Secondes

Exemple :

```
smpp_uptime_seconds{bind_name="vodafone_uk",...} 86400
```

Utilisation :

- Suivre la stabilité de la connexion
- Calculer le pourcentage de disponibilité
- Alerter sur les redémarrages récents

Métriques de Cache de Messages

Ces métriques sont disponibles lorsque la mise en cache des messages est activée sur une ou plusieurs liaisons. Voir MESSAGE_CACHE.md pour les détails de configuration du cache.

smpp_cache_size

Type : Gauge
Description : Nombre actuel de messages dans le cache local par liaison
Unité : Messages

Étiquettes :

- `bind_name` - Nom de la connexion
- `mode` - Type de connexion ("client" ou "serveur")

Exemple :

```
smpp_cache_size{bind_name="partner_acme",mode="server"} 42
```

Utilisation :

- Surveiller l'utilisation du cache
- Alerter lorsque l'on approche de `cache_max_size`

smpp_cache_delivered_total

Type : Counter
Description : Nombre total de messages mis en cache livrés avec succès à l'API en arrière-plan
Unité : Messages

Exemple :

```
smpp_cache_delivered_total{bind_name="partner_acme"} 1234
```

smpp_cache_retry_total

Type : Counter
Description : Nombre total de tentatives de réessai pour les messages mis en cache
Unité : Tentatives

Exemple :

```
smpp_cache_retry_total{bind_name="partner_acme"} 56
```

smpp_cache_permanent_failures_total

Type : Counter
Description : Nombre total de messages ayant dépassé le maximum de tentatives de réessai et ayant été marqués comme échoués de manière permanente
Unité : Messages

Exemple :

```
smpp_cache_permanent_failures_total{bind_name="partner_acme"} 2
```

Utilisation :

- Alerter lorsque > 0 (nécessite une révision manuelle)

smpp_cache_overflow_total

Type : Counter
Description : Nombre total d'événements de débordement de cache où le message le plus ancien a été évincé pour faire de la place
Unité : Événements

Exemple :

```
smpp_cache_overflow_total{bind_name="partner_acme"} 0
```

Utilisation :

- Alerter lorsque cela augmente (cache trop petit ou panne API trop longue)

Métriques de Réception de Livraison (DLR)

Ces métriques sont disponibles pour les liaisons serveur qui définissent `deliver_receipts: true`. Voir CONFIGURATION.md pour le paramètre et USAGE.md pour le flux de messages.

smpp_dlr_tracked_total

Type : Counter
Description : Nombre total de messages soumis enregistrés pour le suivi des accusés de réception de livraison
Unité : Messages

Étiquettes :

- `bind_name` - Nom de la connexion
- `mode` - Type de connexion ("client" ou "serveur")

Exemple :

```
smpp_dlr_tracked_total{bind_name="masorange",mode="server"} 10234
```

smpp_dlr_sent_total

Type : Counter
Description : Nombre total d'accusés de réception de livraison envoyés au lien soumetteur
Unité : Accusés de réception

Exemple :

```
smpp_dlr_sent_total{bind_name="masorange",mode="server"} 10180
```

Utilisation :

- Comparer avec `smpp_dlr_tracked_total` pour voir le ratio de complétion des reçus

smpp_dlr_dropped_total

Type : Counter
Description : Nombre total de messages suivis abandonnés sans accusé de réception. La passerelle abandonne un message lorsque le cœur ne l'a plus, ou lorsque le message n'atteint pas un état terminal dans les 24 heures.
Unité : Messages

Exemple :

```
smpp_dlr_dropped_total{bind_name="masorange",mode="server"} 12
```

Utilisation :

- Alerter lorsque cela augmente (les messages n'atteignent jamais un état terminal, ou le cœur les perd)

smpp_dlr_pending

Type : Gauge
Description : Nombre actuel de messages en vol qui attendent un statut

terminal et un accusé de réception
Unité : Messages

Exemple :

```
smpp_dlr_pending{bind_name="masorange",mode="server"} 37
```

Utilisation :

- Surveiller l'arriéré du tracker
- Alerter lorsque la valeur reste élevée (le cœur est lent à atteindre des états terminaux, ou la session du pair est en panne)

Exemples de requêtes :

```
# Ratio de complétion des reçus sur 5 minutes
sum(rate(smpp_dlr_sent_total[5m]))
  / sum(rate(smpp_dlr_tracked_total[5m]))

# Reçus abandonnés par seconde
rate(smpp_dlr_dropped_total[5m])

# Arriéré actuel du tracker par liaison
smpp_dlr_pending
```

Métriques de Santé de l'API OmniMessage

Bien que la passerelle elle-même expose des métriques liées à SMPP, **la santé de l'API OmniMessage est critique**. Vous devez également surveiller :

À partir des Métriques OmniMessage (si disponibles)

- `omnimessage_api_requests_total` - Total des requêtes API de la passerelle
- `omnimessage_api_request_duration_seconds` - Temps de réponse de l'API
- `omnimessage_queue_depth` - Messages en attente dans la file d'attente OmniMessage

À partir des Journaux de la Passerelle (si les métriques ne sont pas exposées)

Recherchez ces modèles pour détecter des problèmes d'API :

- "api.*connection refused" - Impossible d'atteindre OmniMessage
- "api.*timeout" - OmniMessage ne répond pas
- "api.*http 503" - OmniMessage temporairement hors service
- "api.*parse error" - Problème de format de réponse

Configuration de Prometheus

Configuration de Scrape de Base

Ajoutez à `/etc/prometheus/prometheus.yml` :

```
scrape_configs:
  - job_name: 'omnimessage-smpp'
    scrape_interval: 15s
    static_configs:
      - targets: ['your-server:4000']
        labels:
          environment: 'production'
          service: 'omnimessage-smpp'
```

Plusieurs Passerelles

```yaml
scrape_configs:
  - job_name: 'omnimessage-smpp-instances'
    scrape_interval: 15s
    static_configs:
      - targets:
        - 'smpp-gw-1:4000'
        - 'smpp-gw-2:4000'
        - 'smpp-gw-3:4000'
        labels:
          environment: 'production'
```

Découverte de Service

Utilisation de la découverte basée sur des fichiers :

```yaml
scrape_configs:
  - job_name: 'omnimessage-smpp-instances'
    file_sd_configs:
      - files:
        - '/etc/prometheus/targets/smpp-*.json'
```

Fichier `/etc/prometheus/targets/smpp-production.json` :

```json
[
  {
    "targets": ["smpp-gw-1:4000", "smpp-gw-2:4000"],
    "labels": {
      "environment": "production",
      "datacenter": "us-east"
    }
  }
]
```

Tableaux de Bord Grafana

Panneaux de Tableau de Bord Exemples

Panneau de Statut de Connexion

Requête :

```
smpp_connection_status{job="omnimessage-smpp"}
```

Visualisation : Stat
Seuils :

- Rouge : valeur < 1 (déconnecté)
- Vert : valeur == 1 (connecté)

Panneau de Taux de Messages

Requête :

```
rate(smpp_messages_sent_total{job="omnimessage-smpp"}[5m])
```

Visualisation : Graph
Unité : messages/seconde
Légende : `{{bind_name}}`

Panneau de Taux de Succès de Livraison

Requête :

```
100 * (1 - (
  rate(smpp_delivery_failures_total{job="omnimessage-smpp"}[5m])
  /
  rate(smpp_messages_sent_total{job="omnimessage-smpp"}[5m])
))
```

Visualisation : Gauge
Unité : Pourcentage (0-100)
Seuils :

- Rouge : < 95%
- Jaune : 95-98%
- Vert : > 98%

Panneau de Disponibilité de Connexion

Requête :

```
smpp_uptime_seconds{job="omnimessage-smpp"} / 3600
```

Visualisation : Stat
Unité : Heures

Règles d'Alerte

Règles d'Alerte Prometheus

Enregistrez dans `/etc/prometheus/rules/smpp-alerts.yml` :

```yaml
groups:
  - name: smpp_gateway
    interval: 30s
    rules:
      # Connexion hors service
      - alert: SMPPConnectionDown
        expr: smpp_connection_status == 0
        for: 2m
        labels:
          severity: critical
        annotations:
          summary: "La connexion SMPP {{ $labels.bind_name }} est
hors service"
          description: "La connexion {{ $labels.bind_name }} a été
déconnectée pendant plus de 2 minutes."

      # Taux d'échec élevé
      - alert: SMPPHighFailureRate
        expr: |
          (
            rate(smpp_delivery_failures_total[5m])
            /
            rate(smpp_messages_sent_total[5m])
          ) > 0.05
        for: 5m
        labels:
          severity: warning
        annotations:
          summary: "Taux d'échec de livraison élevé sur {{
$labels.bind_name }}"
          description: "Le taux d'échec de livraison est {{ $value
| humanizePercentage }} sur {{ $labels.bind_name }}."

      # Échecs de liaison
      - alert: SMPPBindFailures
        expr: increase(smpp_bind_failures_total[10m]) > 3
        labels:
          severity: warning
        annotations:
          summary: "Multiples échecs de liaison sur {{
$labels.bind_name }}"
          description: "{{ $labels.bind_name }} a échoué à se lier
{{ $value }} fois au cours des 10 dernières minutes."
```

```
      # Aucun message envoyé (lorsqu'attendu)
      - alert: SMPPNoTraffic
        expr: rate(smpp_messages_sent_total[10m]) == 0
        for: 30m
        labels:
          severity: warning
        annotations:
          summary: "Aucun message envoyé sur {{ $labels.bind_name
}}"
          description: "{{ $labels.bind_name }} n'a envoyé aucun
message pendant 30 minutes."

      # Déconnexions fréquentes
      - alert: SMPPFrequentDisconnections
        expr: increase(smpp_disconnection_total[1h]) > 5
        labels:
          severity: warning
        annotations:
          summary: "Déconnexions fréquentes sur {{
$labels.bind_name }}"
          description: "{{ $labels.bind_name }} s'est déconnecté
{{ $value }} fois au cours de la dernière heure."

      # API OmniMessage injoignable
      - alert: OmniMessageAPIUnreachable
        expr: |
          count(count_over_time({job="omnimessage-smpp"} |=
"api.*connection refused"[5m])) > 0
        for: 1m
        labels:
          severity: critical
        annotations:
          summary: "L'API OmniMessage est injoignable"
          description: "La passerelle SMPP ne peut pas atteindre
l'API OmniMessage. Vérifiez la configuration de API_BASE_URL et la
connectivité réseau."

      # Timeouts de l'API OmniMessage
      - alert: OmniMessageAPITimeout
        expr: |
          count(count_over_time({job="omnimessage-smpp"} |=
"api.*timeout"[5m])) > 5
        for: 2m
```

```
        labels:
          severity: warning
        annotations:
          summary: "L'API OmniMessage est en timeout"
          description: "Plusieurs timeouts d'API détectés.
OmniMessage peut être lent ou surchargé."

      # Aucun flux de message (problème d'API)
      - alert: NoMessageFlow
        expr: rate(smpp_messages_sent_total[10m]) == 0 and
rate(smpp_messages_received_total[10m]) == 0
        for: 30m
        labels:
          severity: warning
        annotations:
          summary: "Aucun flux de message détecté - vérifiez la
connectivité OmniMessage"
          description: "Aucun message envoyé ou reçu pendant 30
minutes. Vérifiez la connectivité de l'API OmniMessage et l'état
de la file d'attente."
```

Chargez les règles dans `prometheus.yml` :

```
rule_files:
  - '/etc/prometheus/rules/smpp-alerts.yml'
```

Surveillance du Tableau de Bord Web

L'interface Web intégrée fournit une surveillance en temps réel sans Prometheus.

Accès

URL : `https://your-server:8087`

Page de Statut en Direct

Navigation : SMPP → Statut en Direct

Fonctionnalités :

- Statut de connexion en temps réel
- Compteurs de messages
- Disponibilité de la connexion
- Contrôles de reconnexion/déconnexion manuels
- Actualisation automatique toutes les 5 secondes

Utilisation :

- Vérification rapide de l'état
- Intervention manuelle
- Dépannage en temps réel

Le tableau de bord affiche :

- **Total des Liaisons** : Compte combiné de toutes les connexions client et serveur
- **Liaisons Client** : Connexions sortantes vers les opérateurs (affichant le nombre connecté/déconnecté)

- **Liaisons Serveur** : Connexions entrantes des partenaires (affichant le nombre actif/en attente)
- **Serveur Écoute** : Configuration de la socket serveur entrant (hôte, port, connexions maximales)

Surveillance des Journaux

Journaux Système

Voir les journaux :

```
# Suivre les journaux en temps réel
sudo journalctl -u omnimessage-smpp -f

# Dernières 100 lignes
sudo journalctl -u omnimessage-smpp -n 100

# Depuis un temps spécifique
sudo journalctl -u omnimessage-smpp --since "1 hour ago"

# Filtrer par niveau
sudo journalctl -u omnimessage-smpp -p err
```

Journaux de l'Interface Web

Navigation : Onglet Journaux dans l'interface Web

Fonctionnalités :

- Streaming de journaux en temps réel
- Filtrer par niveau (debug, info, avertissement, erreur)
- Rechercher dans les journaux
- Pause/reprise
- Effacer les journaux

La vue des journaux vous permet de :

- **Filtrer par Niveau** : Sélectionner le niveau de journal (Tous, Debug, Info, Avertissement, Erreur)
- **Rechercher** : Trouver des entrées de journal spécifiques par contenu textuel
- **Défilement Automatique** : Activer/désactiver le défilement automatique à l'arrivée de nouveaux journaux
- **Pause/Reprise** : Mettre en pause les mises à jour des journaux pour examiner des entrées spécifiques
- **Effacer** : Effacer tous les journaux affichés

Indicateurs Clés de Performance (KPI)

Santé de la Connexion

Métrique : Pourcentage de disponibilité de la connexion

```
avg_over_time(smpp_connection_status[24h]) * 100
```

Cible : > 99.9%

Taux de Livraison des Messages

Métrique : Messages livrés par seconde

```
rate(smpp_messages_sent_total[5m])
```

Cible : Correspond au volume attendu

Taux de Succès de Livraison

Métrique : Pourcentage de livraisons réussies

```
100 * (1 - rate(smpp_delivery_failures_total[5m]) /
rate(smpp_messages_sent_total[5m]))
```

Cible: > 98%

Stabilité de la liaison

Métrique: Tentatives de liaison par heure

```
rate(smpp_bind_success_total[1h]) * 3600
```

Cible: < 10 par heure (indique une connexion stable)

Meilleures Pratiques de Surveillance

1. Configurer des Alertes

- Configurer des alertes Prometheus pour les métriques critiques

- Utiliser PagerDuty/OpsGenie pour des alertes 24/7
- Tester les alertes régulièrement

2. Créer des Tableaux de Bord

- Construire des tableaux de bord Grafana pour chaque passerelle
- Inclure toutes les connexions sur un tableau de bord
- Ajouter des panneaux de planification de capacité

3. Revues Régulières

- Examiner les métriques chaque semaine
- Identifier les tendances et les modèles
- Planifier des ajustements de capacité

4. Documenter les Baselines

- Enregistrer les volumes de messages normaux
- Documenter les taux TPS attendus
- Noter les heures/jours de pointe

5. Corréler avec le Backend

- Surveiller les métriques de l'API backend
- Suivre le flux de messages de bout en bout
- Identifier les goulets d'étranglement

Dépannage avec les Métriques

Problèmes de Connexion

Vérifier: `smpp_connection_status`

- Valeur 0 = Examiner les journaux, vérifier le réseau, vérifier les identifiants

- Changements fréquents = Instabilité du réseau

Mauvaises Taux de Livraison

Vérifier: `smpp_delivery_failures_total`

- Taux élevé = Vérifier l'état du transporteur, examiner le format du message
- Comparer entre les connexions = Identifier le transporteur problématique

Faible Débit

Vérifier: `smpp_messages_sent_total` taux

- En dessous des attentes = Vérifier les limites TPS, disponibilité de la file d'attente
- Vérifier les métriques de l'API backend

Problèmes de Liaison

Vérifier: `smpp_bind_failures_total`

- En augmentation = Problèmes d'authentification, problèmes d'identifiants
- Vérifier system_id et mot de passe dans la configuration

Documentation Connexe

- **CONFIGURATION.md** - Configurer les paramètres de surveillance
- **USAGE.md** - Procédures opérationnelles
- **TROUBLESHOOTING.md** - Résoudre les problèmes
- **README.md** - Vue d'ensemble et démarrage rapide

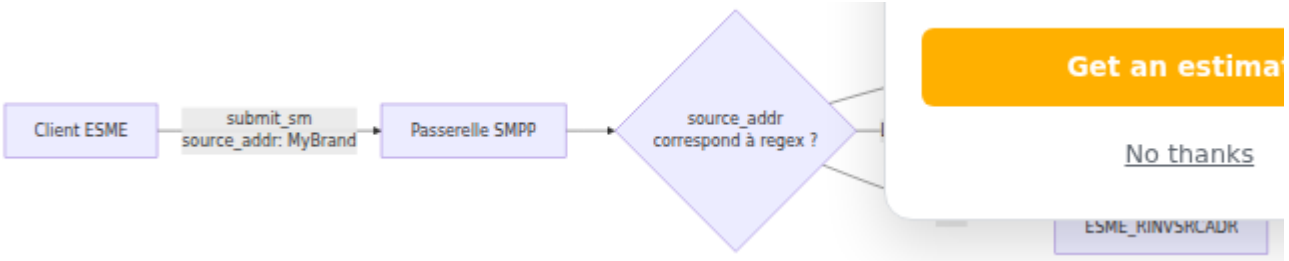
Liste blanche des adresses source

Contrôle par paire sur les adresses source (source_addr) qu'un client SMPP peut utiliser lors de l'envoi de messages.

Aperçu

Lorsqu'un ESME externe (client) envoie un PDU `submit_sm` via la passerelle SMPP, le PDU inclut un champ `source_addr` représentant l'adresse d'origine (CLI / ID de l'expéditeur). Par défaut, les clients authentifiés peuvent utiliser n'importe quelle adresse source. La fonctionnalité de liste blanche des adresses source permet aux opérateurs de restreindre les adresses source que chaque paire de serveur est autorisée à utiliser.

Cela suit le même modèle que la liste blanche IP existante : lorsque la liste blanche est vide, toutes les valeurs sont autorisées. Lorsqu'elle est remplie, seules les adresses source correspondantes sont acceptées.



Règles de correspondance

Chaque entrée de la liste blanche est un **modèle d'expression régulière**. L'adresse source est testée contre chaque modèle. Si un modèle correspond, l'adresse est autorisée.

Correspondance exacte

Ancrez le modèle avec `^` et `$` pour exiger une correspondance exacte. La correspondance est **sensible à la casse** par défaut.

Entrée de la liste blanche	Adresse source	Résultat
`^MyBrand$`	`MyBrand`	Autorisé
`^MyBrand$`	`mybrand`	Rejeté
`^MyBrand$`	`MyBrands`	Rejeté
`^\+61400000001$`	`+61400000001`	Autorisé

Correspondance par préfixe

Utilisez `^` sans `$` final pour correspondre aux adresses commençant par un préfixe.

Entrée de la liste blanche	Adresse source	Résultat
`^614`	`61400000001`	Autorisé
`^614`	`61412345678`	Autorisé
`^614`	`61500000001`	Rejeté
`^\+614`	`+61400000001`	Autorisé

Modèles avancés

La syntaxe complète des expressions régulières est prise en charge, permettant des règles de correspondance puissantes.

Entrée de la liste blanche	Description	Exemples
`^61[45]\d{8}$`	Numéros de mobile australiens 614/615 (11 chiffres)	`61400000001`, `61512345678`
`^(MyBrand|AlertSvc)$`	Correspondance exacte sur l'un ou l'autre nom	`MyBrand`, `AlertSvc`
`(?i)^mybrand$`	Correspondance exacte insensible à la casse	`MyBrand`, `mybrand`, `MYBRAND`
`^(?!Test).+`	Toute adresse non vide ne commençant pas par "Test"	`MyBrand` (autorisé), `TestBrand` (rejeté)

Entrées multiples

Lorsque plusieurs entrées sont configurées, l'adresse source est autorisée si elle correspond à **n'importe quelle** entrée de la liste blanche.

Exemple de liste blanche : `^MyBrand$, ^614, ^\+61400000001$`

Adresse source	Correspondances	Résultat
`MyBrand`	`^MyBrand$`	Autorisé
`61412345678`	`^614`	Autorisé
`+61400000001`	`^\+61400000001$`	Autorisé
`OtherBrand`	Aucune	Rejeté
`61500000001`	Aucune	Rejeté

Gestion des erreurs

Lorsqu'un `submit_sm` est rejeté en raison d'une violation de la liste blanche des adresses source, la passerelle répond avec :

Champ	Valeur
PDU	`submit_sm_resp`
Statut de commande	`0x0000000A`
Nom de l'erreur	`ESME_RINVSRCADR` (Adresse source invalide)
ID de message	Vide

Un avertissement est enregistré avec l'adresse source rejetée et le nom de la paire :

```
Serveur SMPP : Rejet de submit_sm de partner_acme - source_addr
'UnauthorisedBrand' non dans la liste blanche
```

Les modèles regex invalides sont traités comme non correspondants (l'adresse est rejetée) et un avertissement est enregistré.

Configuration

Via l'interface Web

1. Accédez à **SMPP > Pairs de serveurs**
2. Cliquez sur **Modifier** sur la paire cible (ou **Ajouter un nouveau pair de serveur**)
3. Localisez le champ **Liste blanche des adresses source** (sous la liste blanche IP)
4. Entrez des modèles regex séparés par des virgules :

```
^MyBrand$,^614,^\+61400000001$
```

5. Cliquez sur **Enregistrer**

Les modifications prennent effet immédiatement pour les nouveaux PDU `submit_sm` sur les connexions existantes.

Via le fichier de configuration

Ajoutez `source_address_whitelist` à la configuration de liaison du serveur dans `runtime.exs` :

```
config :omnimessage_smpp, :server_binds, [
  %{
    name: "partner_acme",
    system_id: "acme_corp",
    password: "secure_password",
    allowed_bind_types: [:transmitter, :receiver, :transceiver],
    ip_whitelist: ["203.0.113.0/24"],
    source_address_whitelist: ["^MyBrand$", "^614",
"^\+61400000001$"],
    tps_limit: 50,
    queue_check_frequency: 1000
  }
]
```

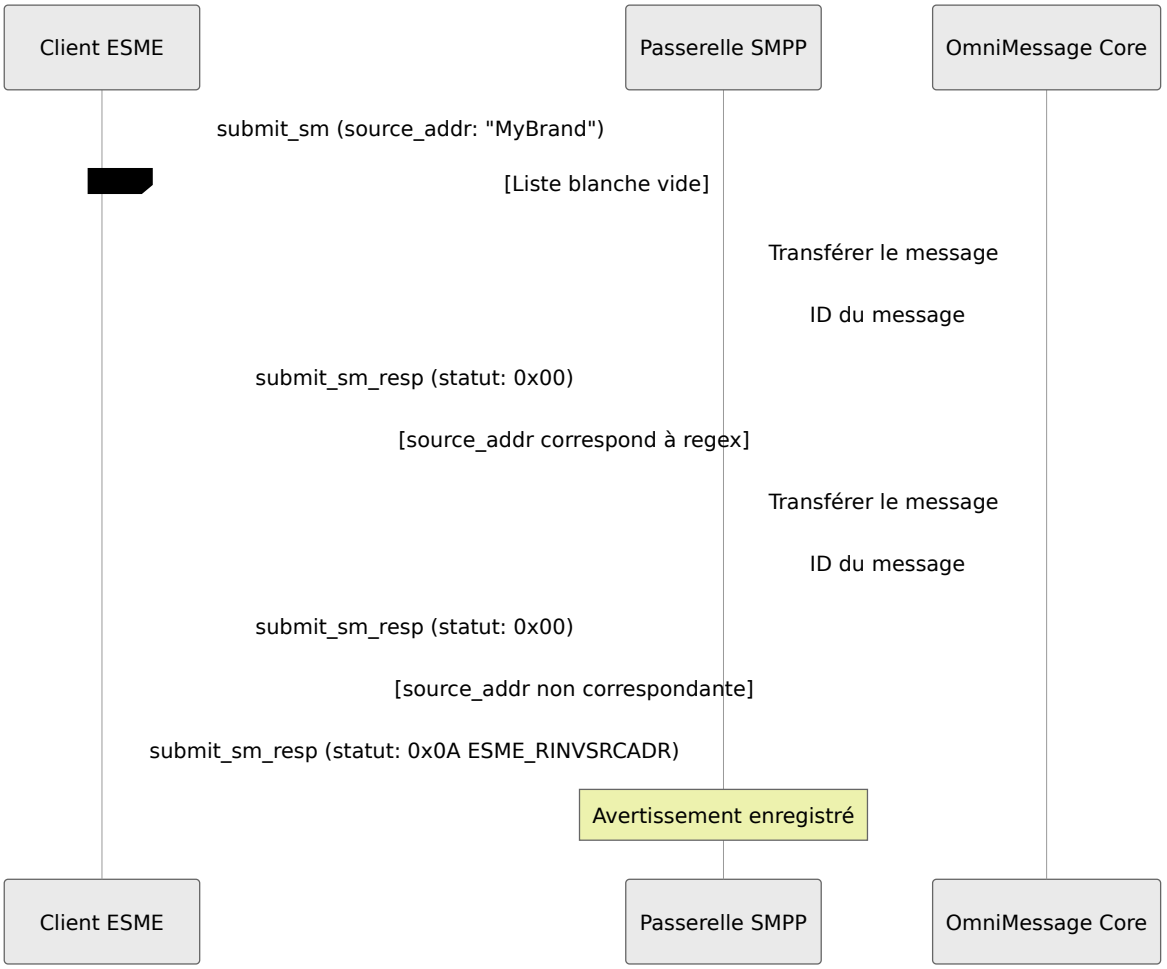
Paramètres

Paramètre	Type	Requis	Par défaut	Description
`source_address_whitelist`	Liste de chaînes	Non	`[]` (autoriser tout)	Liste des modèles regex d'adresses source autorisées. Une liste vide permet toutes les adresses source.

Migration

Les pairs de serveurs existants sont automatiquement migrés lorsque la passerelle démarre. Les pairs créés avant l'ajout de cette fonctionnalité reçoivent une liste blanche vide (toutes les adresses source autorisées), préservant le comportement existant.

Flux de validation



Exemples

Restreindre à une seule marque

Autoriser uniquement les messages de l'ID d'expéditeur `AcmeCorp` :

```
^AcmeCorp$
```

Autoriser une plage de numéros australiens

Autoriser tout numéro de mobile australien (commençant par `614`) :

```
^614
```

Autoriser des plages de numéros spécifiques

Autoriser les numéros de mobile australiens 614 et 615 (exactement 11 chiffres) :

```
^61[45]\d{8}$
```

Alphanumérique et numérique mélangés

Autoriser un nom de marque et une plage de numéros :

```
^AcmeCorp$,^614,^\+61290000001$
```

Correspondance de marque insensible à la casse

Autoriser un nom de marque quelle que soit la casse :

```
(?i)^acmecorp$
```

Autoriser tout (par défaut)

Laissez le champ vide pour autoriser toute adresse source. C'est le comportement par défaut.

Dépannage

Messages rejetés avec ESME_RINVSRCADR

Symptômes : Le partenaire signale `submit_sm_resp` avec le statut de commande `0x0A`.

Causes possibles :

- L'adresse source ne correspond à aucun modèle regex dans la liste blanche
- Le modèle regex a une erreur de syntaxe (traité comme non correspondant)
- Ancrages manquants (`^`/`$`) causant des correspondances partielles inattendues ou des rejets
- Mismatch de casse (la correspondance est sensible à la casse par défaut ; utilisez `(?i)` pour une correspondance insensible à la casse)

Résolution :

1. Vérifiez la liste blanche des adresses source du pair de serveur dans l'interface Web
2. Comparez l'adresse source rejetée avec chaque modèle regex
3. Testez les modèles à l'aide d'un testeur regex Elixir : `Regex.match? (~r/^pattern$/, "address")`
4. Ajoutez le modèle manquant ou ajustez les modèles existants
5. N'oubliez pas d'échapper les caractères spéciaux des regex (par exemple, `\+` pour le `+` littéral)

La liste blanche ne prend pas effet

Symptômes : Messages acceptés malgré le fait que l'adresse source ne corresponde pas à la liste blanche.

Causes possibles :

- La liste blanche est vide (autorise tout par défaut)

- Le modèle n'est pas ancré et correspond comme une sous-chaîne (par exemple, `614` correspond à `X614Y`)
- L'ESME est connecté à un autre pair de serveur
- Le changement de fichier de configuration n'a pas encore été appliqué (nécessite un redémarrage)

Résolution :

1. Vérifiez que la liste blanche est peuplée (non vide) dans l'interface Web
2. Assurez-vous que les modèles utilisent les ancrages `^` et `$` là où une correspondance exacte est prévue
3. Vérifiez à quel pair de serveur l'ESME est lié dans l'état en direct
4. Si vous utilisez un fichier de configuration, redémarrez le service

---

Documentation connexe

- **Référence de configuration** - Documentation complète des paramètres des pairs de serveurs
- **Guide d'utilisation** - Gestion des connexions SMPP
- **Dépannage** - Procédures générales de dépannage

Guide de dépannage

Problèmes courants et solutions

Le service ne démarre pas / Conflits de port

Le Gateway ou OmniMessage Core redémarre en boucle avec `:eaddrinuse` sur le port 8443

Symptômes:

- L'un des services redémarre continuellement (`systemctl show <unit> -p NRestarts` augmente rapidement).
- Les journaux montrent :

  ```
  Running ApiEx.Endpoint with Bandit ... at https failed, port
  8443 already in use
  ... failed to start child: ApiEx.Supervisor ... ** (EXIT)
  :eaddrinuse
  ```

- `curl -sk https://<host>:8443/api/messages` retourne `404` même si le magasin devrait répondre.

Cause: Le gateway SMPP et OmniMessage Core sont colocés sur le même hôte et les deux lient leur API `api_ex` sur le port par défaut `8443`. Celui qui démarre en premier gagne le port ; l'autre redémarre en boucle. Si le gateway gagne, `/api/messages` est servi par le gateway (qui n'a pas de magasin) et retourne 404.

Solution: Déplacez l'API OAM/Gestion du gateway hors du port de Core (Core garde `8443`). Éditez `/etc/omnimessage_smpp/runtime.exs` et décommentez la substitution `api_ex`, en définissant un port libre. Le gateway reste un client de l'API de magasin de Core via `api_base_url`.

```
# /etc/omnimessage_smpp/runtime.exs
config :api_ex,
  api:
    Application.get_env(:api_ex, :api, %{})
    |> Map.put(:port, 8444)
    |> Map.put(:listen_ip, "0.0.0.0")
```

Ensuite, redémarrez les deux services :

```
sudo systemctl restart omnimessage-smpp
sudo systemctl restart omnimessage        # laisser Core récupérer
:8443
```

Après le redémarrage, vérifiez que Core possède 8443 et que le gateway possède son nouveau port :

```
ss -lntp | grep -E ':8443|:8444'
curl -sk -o /dev/null -w '%{http_code}\n'
https://127.0.0.1:8443/api/messages   # le magasin répond (pas
404)
```

Voir CONFIGURATION.md → Port de l'API OAM/Gestion pour plus de détails.

Problèmes de connectivité OmniMessage

Étant donné que le Gateway SMPP est sans état et dépend entièrement d'OmniMessage Core, les problèmes de connectivité avec OmniMessage sont les plus critiques.

Symptômes de déconnexion d'OmniMessage

- **Aucun message sortant** : La file d'attente s'accumule, les messages ne sont pas envoyés

- **Aucun message entrant** : Les partenaires ne peuvent pas soumettre de messages
- **Timeouts** : Les appels API expirent ou se bloquent
- **Les journaux montrent** : "Connection refused", "Timeout", "HTTP 503", "Connection reset"

Diagnostic

1. Vérifiez la disponibilité d'OmniMessage :

```
# Tester la connectivité
curl -k -v https://omnimessage-
core.example.com:8443/api/system/health

# Tester depuis l'hôte du gateway spécifiquement
ssh gateway-server 'curl -k https://omnimessage-
core.example.com:8443/api/system/health'
```

2. Vérifiez l'URL API configurée :

```
# Examiner la configuration
grep -A1 'api_base_url' /opt/omnimessage-smpp/config/runtime.exs

# Vérifiez la connectivité réseau
ping omnimessage-core.example.com
nc -zv omnimessage-core.example.com 8443
```

3. Vérifiez les journaux du Gateway pour les erreurs API :

```
# Recherchez les erreurs liées à l'API
sudo journalctl -u omnimessage-smpp -f | grep -i
'api\|omnimessage\|connect'

# Recherchez les journaux pour les erreurs récentes
sudo journalctl -u omnimessage-smpp -n 200 | grep -i error
```

Solutions

Si OmniMessage est hors service :

1. Contactez l'équipe des opérations d'OmniMessage
2. Les messages en attente s'accumuleront dans la file d'attente
3. Le gateway continuera d'essayer (voir `SMPP_POLL_INTERVAL`)
4. Vérifiez la page d'état d'OmniMessage ou la surveillance

Si OmniMessage est en ligne mais que le gateway ne peut pas y accéder :

1. Vérifiez que les règles de pare-feu autorisent HTTPS sortant
2. Vérifiez la résolution DNS : `nslookup omnimessage-core.example.com`
3. Vérifiez le routage réseau : `traceroute omnimessage-core.example.com`
4. Vérifiez les certificats SSL si vous utilisez HTTPS

Si l'URL API est mal configurée :

1. Éditez `/opt/omnimessage-smpp/config/runtime.exs`
2. Vérifiez que `api_base_url` est correct (doit être HTTPS pour la production)
3. Redémarrez le gateway : `sudo systemctl restart omnimessage-smpp`

Problèmes de connexion

La connexion ne s'établit pas

Symptômes :

- Le statut indique "Déconnecté" (rouge)
- Pas de liaison réussie dans les journaux
- Tentatives de connexion répétées

Causes possibles et solutions :

1. Problèmes de connectivité réseau

Vérifiez :

```
# Tester la résolution DNS
nslookup smpp.carrier.com

# Tester la connectivité
ping -c 3 smpp.carrier.com

# Tester le port
telnet smpp.carrier.com 2775
# ou
nc -zv smpp.carrier.com 2775
```

Solutions :

- Si DNS échoue : Utilisez l'adresse IP au lieu du nom d'hôte dans la configuration
- Si le ping échoue : Vérifiez les règles de pare-feu, contactez le transporteur
- Si le port échoue : Vérifiez le numéro de port correct, vérifiez le pare-feu

2. Identifiants incorrects

Vérifiez :

- Les journaux montrent "bind failed" ou "authentication error"
- Interface Web : SMPP → Pairs clients → vérifiez system_id et mot de passe

Solutions :

- Confirmez les identifiants avec le transporteur
- Vérifiez les fautes de frappe (sensible à la casse)
- Mettez à jour la configuration et reconnectez

3. IP non autorisée

Vérifiez :

- La connexion est immédiatement rejetée
- Les journaux du transporteur montrent une IP non autorisée

Solutions :

- Confirmez l'IP publique de votre gateway :

```
curl ifconfig.me
```

- Demandez au transporteur d'ajouter l'IP à la liste blanche
- Vérifiez que l'IP n'a pas changé (IP dynamique)

4. Pare-feu bloquant

Vérifiez :

```
# Vérifiez si le port est ouvert
sudo iptables -L -n | grep 2775

# Vérifiez UFW (Ubuntu/Debian)
sudo ufw status | grep 2775

# Vérifiez firewalld (RHEL/CentOS)
sudo firewall-cmd --list-ports | grep 2775
```

Solutions :

```
# Ubuntu/Debian
sudo ufw allow out 2775/tcp

# RHEL/CentOS
sudo firewall-cmd --permanent --add-port=2775/tcp
sudo firewall-cmd --reload
```

5. Adresse/port source fixé indisponible

S'applique uniquement lorsque le client bind définit `bind_source_ip` et/ou `bind_source_port` (voir CONFIGURATION.md).

Vérifiez :

- Les journaux montrent que la connexion échoue à ouvrir le socket local

- `bind_source_ip` n'est pas une adresse détenue par cet hôte
- Un autre processus détient déjà le `bind_source_port` configuré

Solutions :

- Confirmez que `bind_source_ip` est une adresse locale : `ip addr show`
- Une `bind_source_ip` invalide est enregistrée et ignorée — le système d'exploitation choisit alors l'IP source, qui peut ne pas correspondre à la liste blanche du transporteur. Corrigez l'adresse si la livraison échoue.
- Le port fixé est ouvert avec `SO_REUSEADDR`, donc une reconnexion se rebinde immédiatement même si l'ancien socket est en `TIME_WAIT`. Si le port est toujours indisponible, vérifiez si un autre service l'utilise : `sudo ss -ltnp | grep <port>`

La connexion se déconnecte fréquemment

Symptômes :

- Connexion établie mais se déconnecte fréquemment
- La métrique `smpp_disconnection_total` augmente
- Les journaux montrent des reconnexions répétées

Causes possibles et solutions :

1. Instabilité du réseau

Vérifiez :

```
# Surveillez la perte de paquets
ping -c 100 smpp.carrier.com | grep loss

# Vérifiez les erreurs réseau
netstat -s | grep -i error
```

Solutions :

- Contactez le transporteur au sujet des problèmes de réseau

- Vérifiez avec votre FAI si c'est de votre côté
- Envisagez une connexion/routage de secours

2. Timeout d'enquête de lien

Vérifiez :

- Les journaux montrent "enquire_link timeout"
- La connexion se déconnecte après des périodes d'inactivité

Solutions :

- Le timeout par défaut est de 30 secondes
- Vérifiez que le réseau autorise les paquets keepalive
- Vérifiez les pare-feu agressifs qui expirent les connexions inactives

3. Limite TPS dépassée

Vérifiez :

- Taux de message élevé au moment de la déconnexion
- Le transporteur limite les messages

Solutions :

- Révisez le paramètre `tps_limit`
- Réduisez le TPS à 70-80% du maximum du transporteur
- Répartissez le trafic sur plusieurs binds

4. Problèmes de serveur du transporteur

Vérifiez :

- Vérifiez l'état du service du transporteur
- Contactez le support du transporteur

Solutions :

- Attendez que le transporteur résolve

- Configurez un transporteur de secours si disponible

Problèmes de livraison de messages

Les messages ne sont pas envoyés

Symptômes :

- Messages bloqués dans la file d'attente
- `smpp_messages_sent_total` n'augmente pas
- La connexion montre qu'elle est connectée

Causes possibles et solutions :

1. Routage `dest_smsc` incorrect

Vérifiez :

- Interface Web → File d'attente → Vérifiez le champ `dest_smsc` du message
- Comparez avec le nom de connexion dans SMPP → État en direct

Solutions :

- Les messages sont routés en fonction du champ `dest_smsc`
- Vérifiez que le backend définit le bon `dest_smsc`
- Si `dest_smsc` est NULL, vérifiez le routage par défaut

2. Messages programmés pour l'avenir

Vérifiez :

- Interface Web → File d'attente → Vérifiez le champ `deliver_after`
- Les messages avec un horodatage futur ne seront pas envoyés encore

Explication :

- Le système de réessai définit `deliver_after` pour les messages échoués
- Les messages attendent jusqu'à ce moment avant de réessayer

Solutions :

- Attendez l'heure programmée
- Si urgent, contactez l'équipe backend pour réinitialiser l'horodatage

3. Limite TPS trop basse

Vérifiez :

- Grande accumulation de files d'attente
- Messages envoyés très lentement

Solutions :

- Augmentez `tps_limit` dans la configuration
- Vérifiez que le transporteur peut gérer un taux plus élevé
- Voir CONFIGURATION.md

4. Le travailleur de file d'attente ne fonctionne pas

Vérifiez :

- État du service
- Journaux pour les erreurs

Solutions :

```
# Redémarrer le service
sudo systemctl restart omnimessage-smpp

# Vérifiez les journaux
sudo journalctl -u omnimessage-smpp -f
```

5. Frontend non enregistré sous le nom attendu

OmniMessage Core ne route un message vers ce gateway que si un pair est enregistré sous un nom de frontend égal à `dest_smsc` du message. Chaque pair s'enregistre sous sa **clé de routage** (voir Enregistrement de frontend):

- **Pairs clients** : la `queue` configurée, ou le nom de pair nu si aucune `queue` n'est définie.
- **Binds de serveur** : le nom de liaison.

Vérifiez :

- Les journaux pour les entrées `Successfully registered SMPP peer frontend '<name>'` — confirmez que `<name>` correspond à `dest_smsc` sur les messages bloqués
- Pour les binds partageant une `queue`, attendez une seule inscription sous le nom de file d'attente partagé, pas une par pair

Solutions :

- Assurez-vous que le backend définit `dest_smsc` sur la clé de routage (la `queue` du pair, sinon son nom), pas sur `{hostname}_{peer}` (un ancien schéma de nommage qui ne s'applique plus)
- Si vous utilisez une `queue` partagée, routez les messages vers le nom de la file d'attente plutôt que vers un nom de pair individuel
- Confirmez que le pair est activé — les pairs désactivés ne sont pas enregistrés

---

Taux d'échec de livraison élevé

Symptômes :

- `smpp_delivery_failures_total` augmente
- Les journaux montrent "submit_sm_resp" avec un statut d'erreur
- Les messages n'atteignent pas les destinataires

Causes possibles et solutions :

1. Numéros de destination invalides

Vérifiez :

- Les journaux pour des codes d'erreur spécifiques
- Examinez le format de destination des messages

Codes d'erreur courants :

- `0x0000000B` - Destination invalide
- `0x00000001` - Longueur de message invalide
- `0x00000003` - Commande invalide

Solutions :

- Validez le format du numéro (E.164 recommandé)
- Vérifiez que le numéro inclut le code pays
- Vérifiez les exigences auprès du transporteur

2. Contenu de message invalide

Vérifiez :

- Longueur du message
- Caractères spéciaux
- Encodage

Solutions :

- GSM-7 : Max 160 caractères
- UCS-2 : Max 70 caractères
- Supprimez les caractères non pris en charge
- Vérifiez les paramètres d'encodage

3. Rejet par le transporteur

Vérifiez :

- Codes d'erreur spécifiques du transporteur
- Modèles dans les messages rejetés

Solutions :

- Contactez le transporteur pour connaître la raison du rejet
- Peut nécessiter un filtrage de contenu
- Vérifiez les modèles de spam/abus

4. Messages expirés

Vérifiez :

- Horodatage `expires` du message
- Timing des tentatives de livraison

Solutions :

- Augmentez la période de validité des messages
- Réduisez le délai de réessai pour les messages sensibles au temps

---

Problèmes de l'interface Web

Impossible d'accéder au tableau de bord Web

Symptômes :

- Le navigateur ne peut pas se connecter à https://your-server:8087
- Timeout ou connexion refusée

Causes possibles et solutions :

1. Service non en cours d'exécution

Vérifiez :

```
sudo systemctl status omnimessage-smpp
```

Solutions :

```
# Si arrêté, démarrez-le
sudo systemctl start omnimessage-smpp

# Vérifiez les journaux pour les erreurs
sudo journalctl -u omnimessage-smpp -n 50
```

2. Pare-feu bloquant le port 8087

Vérifiez :

```
sudo ufw status | grep 8087
# ou
sudo firewall-cmd --list-ports | grep 8087
```

Solutions :

```
# Ubuntu/Debian
sudo ufw allow 8087/tcp

# RHEL/CentOS
sudo firewall-cmd --permanent --add-port=8087/tcp
sudo firewall-cmd --reload
```

3. Problèmes de certificat SSL

Vérifiez :

- Le navigateur affiche un avertissement de sécurité
- Certificat expiré ou invalide

Solutions :

- Acceptez l'exception de sécurité (si auto-signé)
- Installez un certificat SSL valide
- Vérifiez que les fichiers de certificat existent :

  ```
  ls -l /opt/omnimessage-smpp/priv/cert/
  ```

4. URL incorrecte

Vérifiez :

- Vérifiez en utilisant HTTPS (pas HTTP)
- Vérifiez l'IP/hostname du serveur correct
- Vérifiez le port 8087

L'interface Web affiche des erreurs

Symptômes :

- La page se charge mais affiche des erreurs
- Les fonctions ne fonctionnent pas
- Les données ne s'affichent pas

Solutions :

1. **Vider le cache du navigateur** :
    - Ctrl+F5 (rafraîchissement forcé)
    - Vider le cache et les cookies du navigateur
2. **Vérifiez la console du navigateur** :
    - Appuyez sur F12
    - Vérifiez l'onglet Console pour les erreurs JavaScript
    - Signalez au support si des erreurs sont trouvées
3. **Essayez un autre navigateur** :
    - Testez dans Chrome, Firefox, Edge
    - Isolez les problèmes spécifiques au navigateur
4. **Vérifiez les journaux du service** :

```
sudo journalctl -u omnimessage-smpp -f
```

Problèmes de métriques

Les métriques Prometheus ne sont pas disponibles

Symptômes :

- `curl http://localhost:4000/metrics` échoue
- Prometheus ne peut pas récupérer les métriques
- Réponse vide ou erreur

Causes possibles et solutions :

1. Service non en cours d'exécution

Vérifiez :

```
sudo systemctl status omnimessage-smpp
```

Solutions :

```
sudo systemctl start omnimessage-smpp
```

2. Port non accessible

Vérifiez :

```
# Tester localement
curl http://localhost:4000/metrics

# Tester à distance
curl http://your-server-ip:4000/metrics
```

Solutions :

- Si local fonctionne mais distant ne fonctionne pas : Vérifiez le pare-feu
- Ouvrez le port 4000 dans le pare-feu pour le serveur Prometheus

3. Mauvais point de terminaison

Vérifiez :

- Le point de terminaison est `/metrics` (pas `/prometheus` ou `/stats`)
- Le port est 4000 (pas 8087)

Les métriques affichent des valeurs inattendues

Symptômes :

- Les compteurs se réinitialisent à zéro
- Les jauges affichent des valeurs incorrectes
- Des métriques manquent pour certains binds

Solutions :

1. **Le redémarrage du service réinitialise les compteurs** :
    - Les compteurs se réinitialisent lors du redémarrage du service
    - C'est un comportement normal
    - Utilisez `increase()` ou `rate()` dans les requêtes Prometheus
2. **Les nouveaux binds ne s'affichent pas** :
    - Les métriques n'apparaissent qu'après le premier événement
    - Envoyez un message test pour peupler les métriques
    - Vérifiez que le bind est activé et connecté
3. **Métriques obsolètes** :
    - Les anciens binds peuvent encore apparaître dans les métriques
    - Redémarrez le service pour effacer les anciennes entrées

- Ou utilisez le renommage de Prometheus pour filtrer

Problèmes de performance

Utilisation élevée du CPU

Vérifiez :

```
top -p $(pgrep -f omnimessage-smpp)
```

Causes possibles :

- Volume de messages très élevé
- Trop de connexions
- Problème de configuration

Solutions :

- Vérifiez que le taux de messages est dans la capacité
- Révisez les limites TPS
- Contactez le support si l'utilisation élevée du CPU se maintient

Utilisation élevée de la mémoire

Vérifiez :

```
ps aux | grep omnimessage-smpp
```

Causes possibles :

- Grande file d'attente de messages en mémoire
- Fuite de mémoire (rare)

Solutions :

- Redémarrez le service pour libérer de la mémoire
- Vérifiez la taille de la file d'attente de messages
- Contactez le support si la mémoire augmente continuellement

Traitement des messages lent

Symptômes :

- Les messages prennent du temps à être envoyés
- Accumulation de la file d'attente
- Faible taux de messages

Vérifiez :

1. Limites TPS - peuvent être trop restrictives
2. `queue_check_frequency` - peut être trop élevé
3. Temps de réponse de l'API backend - peut être lent
4. Latence réseau vers le transporteur

Solutions :

- Augmentez le TPS si le transporteur le permet
- Diminuez `queue_check_frequency` pour un sondage plus rapide
- Optimisez l'API backend
- Vérifiez la latence réseau

Problèmes de configuration

Erreurs de syntaxe dans le fichier de configuration

Symptômes :

- Le service ne démarre pas après un changement de configuration
- Les journaux montrent "syntax error" ou "parse error"

Vérifiez :

```
# Valider la syntaxe Elixir
/opt/omnimessage-smpp/bin/omnimessage-smpp eval "File.read!
('config/runtime.exs')"
```

Erreurs courantes :

- Virgule manquante entre les entrées de la carte
- Guillemets non appariés (" vs ')
- Crochets ou accolades non appariés
- Manque `import Config` en haut

Solutions :

- Restaurer à partir d'une sauvegarde
- Révisez soigneusement la syntaxe
- Utilisez un éditeur de texte avec la coloration syntaxique Elixir

Changements non pris en compte

Symptômes :

- Configuration modifiée mais aucun changement de comportement
- Anciens paramètres toujours actifs

Solutions :

```
# Les changements de configuration nécessitent un redémarrage
sudo systemctl restart omnimessage-smpp

# Vérifiez que le redémarrage a réussi
sudo systemctl status omnimessage-smpp

# Vérifiez les journaux pour les erreurs
sudo journalctl -u omnimessage-smpp -n 50
```

Récupération d'urgence

Échec complet du système

Étapes :

1. **Vérifiez la santé de base du système** :

```
# Espace disque
df -h

# Mémoire
free -h

# Charge CPU
uptime
```

2. **Vérifiez l'état du service** :

```
sudo systemctl status omnimessage-smpp
```

3. **Examinez les journaux récents** :

```
sudo journalctl -u omnimessage-smpp -n 200
```

4. **Essayez de redémarrer le service** :

```
sudo systemctl restart omnimessage-smpp
```

5. **Si le redémarrage échoue** :

   - Vérifiez la syntaxe de la configuration
   - Vérifiez que les certificats SSL existent
   - Vérifiez les permissions des fichiers
   - Examinez les journaux pour des erreurs spécifiques

6. **Restaurer à partir d'une sauvegarde** (si nécessaire) :

```
# Restaurer la configuration
sudo cp /opt/omnimessage-smpp/config/runtime.exs.backup \
  /opt/omnimessage-smpp/config/runtime.exs

# Redémarrer
sudo systemctl restart omnimessage-smpp
```

7. **Contactez le support** si non résolu

Obtenir de l'aide

Informations à rassembler

Avant de contacter le support, collectez :

1. **Version** : `cat /opt/omnimessage-smpp/VERSION`
2. **Journaux récents** :

```
sudo journalctl -u omnimessage-smpp -n 200 > /tmp/smpp-logs.txt
```

3. **Configuration** (sanitisez les mots de passe) :

```
sudo cp /opt/omnimessage-smpp/config/runtime.exs
/tmp/config.exs
# Éditez /tmp/config.exs pour supprimer les mots de passe avant
l'envoi
```

4. **Sortie des métriques** :

```
curl http://localhost:4000/metrics > /tmp/metrics.txt
```

5. **Informations système** :

```
uname -a > /tmp/system-info.txt
free -h >> /tmp/system-info.txt
df -h >> /tmp/system-info.txt
```

Contacter le support

- **Email** : support@omnitouch.com
- **Téléphone** : +61 XXXX XXXX (24/7)
- **Inclure** : Toutes les informations ci-dessus

Documentation connexe

- **USAGE.md** - Procédures opérationnelles normales
- **CONFIGURATION.md** - Référence de configuration
- **MONITORING.md** - Surveillance et métriques
- **README.md** - Vue d'ensemble du système

Guide des opérations

Procédures opérationnelles quotidiennes

Dépendance critique : OmniMessage Core

IMPORTANT : La passerelle SMPP d'OmniMessage ne peut pas fonctionner sans accès à OmniMessage Core. Tout le traitement des messages se fait dans OmniMessage - la passerelle n'est qu'un traducteur de protocole.

Si OmniMessage devient indisponible :

- □ De nouveaux messages ne peuvent pas être soumis
- □ Les messages en attente ne peuvent pas être récupérés
- □ Le statut de livraison ne peut pas être signalé
- □ Le système semble se bloquer ou expirer

Vérifiez la santé d'OmniMessage :

```
# Tester la connectivité API
curl -k https://omnimessage-
core.example.com:8443/api/system/health

# Vérifier l'URL API configurée dans les journaux
grep api_base_url /opt/omnimessage-smpp/config/runtime.exs
```

Opérations quotidiennes

Vérification de santé matinale

Effectuez ces vérifications au début de chaque journée :

1. **Accéder au tableau de bord Web**

- URL : `https://your-server:8087`
- Vérifiez si le tableau de bord se charge correctement

2. **Vérifier l'état de la connexion**

- Naviguez vers : SMPP → État en direct
- Vérifiez que toutes les connexions affichent "Connecté" (vert)
- Notez les liaisons déconnectées

3. **Examiner les métriques des messages**

- Naviguez vers : Onglet Queue
- Vérifiez que les comptes de messages sont raisonnables
- Vérifiez qu'il n'y a pas d'accumulation inattendue dans la file d'attente

4. **Vérifier les journaux système**

- Naviguez vers : Onglet Logs
- Recherchez les messages d'erreur (rouge)
- Notez les motifs d'avertissement

5. **Examiner les métriques Prometheus**

- `curl http://localhost:4000/metrics`
- Ou vérifiez les tableaux de bord Grafana
- Vérifiez que les taux de message sont normaux

Surveillance continue

Configurez des alertes pour :

- Échecs de connexion (> 2 minutes d'arrêt)
- Taux d'échec de livraison élevé (> 5 %)
- Pas de trafic pendant de longues périodes
- Déconnexions fréquentes

Voir MONITORING.md pour la configuration des alertes.

Comprendre le routage des messages

La passerelle achemine les messages entre OmniMessage Core et les connexions SMPP en utilisant deux champs clés :

- `dest_smsc` - Achemine les messages sortants vers **les liaisons clients**. Lorsque OmniMessage met en file d'attente un message avec `dest_smsc: "vodafone_uk"`, la liaison client de la passerelle nommée `vodafone_uk` le récupère et l'envoie via SMPP `submit_sm`.
- `source_smsc` - Achemine les messages entrants vers **les liaisons serveur**. Lorsque OmniMessage met en file d'attente un message avec `source_smsc: "partner_acme"`, la passerelle le livre aux clients connectés à la liaison serveur nommée `partner_acme` via SMPP `deliver_sm`.

Distinction clé : Les liaisons clients envoient des PDU `submit_sm` (la passerelle est l'ESME soumettant à un opérateur). Les liaisons serveur envoient des PDU `deliver_sm` (la passerelle est le SMSC livrant à un ESME connecté).

Enregistrement Frontend

La passerelle s'enregistre automatiquement auprès d'OmniMessage Core afin que le backend sache quelles connexions SMPP sont disponibles pour le routage des messages.

- **Nom d'enregistrement** : Contrôlé par la configuration `smsc_name` (par défaut : `"smpp_gateway"`, env : `SMSC_NAME`)
- **Heartbeat** : Envoyé toutes les 60 secondes pour maintenir l'enregistrement actif
- **Expiration** : L'enregistrement expire sur le backend après 90 secondes sans heartbeat
- **Enregistrement par liaison** : Chaque pair activé est enregistré individuellement sous sa **clé de routage**, de sorte que le nom frontend enregistré soit égal au `dest_smsc` sur lequel le backend route :
  - **Pairs clients** : enregistrés sous leur `queue` configurée (si définie), sinon sous le nom de pair brut. Plusieurs pairs clients partageant une `queue` s'enregistrent donc sous ce nom partagé unique.
  - **Liaisons serveur** : enregistrées sous leur nom de liaison.

Si la passerelle s'arrête ou perd la connectivité avec OmniMessage Core, ses enregistrements expirent et le backend cesse de router des messages vers elle.

Dépannage : Si les messages ne sont pas routés vers la passerelle, vérifiez :

1. Les journaux pour les entrées "frontend_register"
2. Que le `smsc_name` correspond à ce qu'OmniMessage attend
3. La connectivité réseau avec OmniMessage Core (`api_base_url`)

Gestion des connexions SMPP

Comment les pairs SMPP sont configurés

Les connexions SMPP (pairs) peuvent être configurées en utilisant **deux méthodes** :

Méthode 1 : Interface Web ou API OAM (Recommandée)

- **Avantage** : Les modifications prennent effet immédiatement, aucun redémarrage requis
- **Emplacement** : Onglets SMPP → Pairs clients / Pairs serveurs, ou `/api/smpp/client_peers` et `/api/smpp/server_peers` sur l'API OAM
- **Opérations** : Ajouter, modifier, supprimer des pairs
- **Ce qu'une sauvegarde fait** :
  - **Pair serveur** : son worker de file d'attente est démarré ou redémarré avec les nouveaux paramètres. L'écouteur SMPP démarre si c'est le premier pair serveur. Les sessions déjà liées continuent de fonctionner.
  - **Pair client** : la connexion est déliée et rétablie avec les nouveaux paramètres.
  - **Désactiver ou supprimer un pair** : ses workers sont arrêtés. Un pair client est d'abord délié. Les sessions liées d'un pair serveur reçoivent un `unbind` (fermé après 2 secondes si l'ESME ne répond pas), et de nouvelles liaisons sont refusées.
- **Persistance** : Stockée dans la base de données Mnesia
- **Meilleur pour** : Opérations quotidiennes, tests, modifications rapides

Méthode 2 : Fichier de configuration

- **Avantage** : Configuration en tant que code, contrôle de version
- **Emplacement** : `/opt/omnimessage-smpp/config/runtime.exs`
- **Opérations** : Définir des pairs dans la configuration Elixir
- **Persistance** : Basée sur des fichiers, survit aux redémarrages
- **Nécessite** : Redémarrage du service après modifications
- **Meilleur pour** : Configuration initiale, infrastructure en tant que code

Remarque : Les modifications de l'interface Web sont stockées séparément et remplacent les paramètres du fichier de configuration.

Voir CONFIGURATION.md pour la référence du fichier de configuration.

Ajouter une nouvelle connexion client

Objectif : Configurer la passerelle pour agir en tant qu'**ESME** (client) se connectant au **SMSC** (serveur) d'un opérateur

Préparation : Rassembler les informations de l'opérateur :

- Nom d'hôte/IP du serveur SMPP
- Numéro de port (généralement 2775)
- ID système (nom d'utilisateur)
- Mot de passe
- Type de liaison (généralement transceiver)
- Limite TPS

Choisissez l'une des méthodes suivantes :

Option A : Via l'interface Web (Recommandée)

Avantages : Effet immédiat, aucun redémarrage requis

Étapes :

1. **Naviguer vers les pairs clients** :
    - Ouvrir l'interface Web : `https://your-server:8087`
    - Naviguer vers : SMPP → Pairs clients
2. **Ajouter un nouveau pair** :
    - Cliquez sur "Ajouter un nouveau pair client"
    - Remplissez le formulaire :
        - **Nom** : `vodafone_uk` (identifiant unique)
        - **Hôte** : `smpp.vodafone.co.uk`
        - **Port** : `2775`
        - **ID système** : `your_username`
        - **Mot de passe** : `your_password`
        - **Type de liaison** : `Transceiver`
        - **Limite TPS** : `100`

        - **Fréquence de vérification de la file d'attente** : `1000`
    - Cliquez sur "Sauvegarder"

3. **La connexion s'établit automatiquement** :
    - La passerelle tente immédiatement de se connecter
    - Naviguez vers : SMPP → État en direct
    - Le statut devrait changer en "Connecté" (vert) dans les 10 à 30 secondes
    - Vérifiez l'onglet Logs pour un message de liaison réussi
4. **Tester le flux de messages** :
    - Naviguez vers : Onglet Queue
    - Soumettez un message test avec `dest_smsc` correspondant au nom de liaison
    - Surveillez dans l'état en direct pour la transmission
    - Vérifiez la confirmation de livraison

Option B : Via le fichier de configuration

Avantages : Infrastructure en tant que code, contrôle de version

Étapes :

1. **Modifier le fichier de configuration** :

```
sudo nano /opt/omnimessage-smpp/config/runtime.exs
```

2. **Ajouter une nouvelle liaison à la configuration** :

```
config :omnimessage_smpp, :binds, [
  # Liaisons existantes...

  # Ajouter une nouvelle liaison
  %{
    name: "vodafone_uk",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.vodafone.co.uk",
    port: 2775,
    system_id: "your_username",
    password: "your_password",
    tps_limit: 100,
    queue_check_frequency: 1000,

    # Optionnel : épingler l'IP/port source local si
l'opérateur les liste
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5000,

    # Optionnel : partager une file d'attente backend entre
plusieurs liaisons (par défaut au nom)
    queue: "vodafone_uk"
  }
]
```

Voir CONFIGURATION.md pour la liste complète des paramètres de liaison client, y compris `bind_source_ip`, `bind_source_port`, et `queue`.

3. **Sauvegarder et redémarrer le service** :

```
# Sauvegarder le fichier (Ctrl+X, Y, Entrée dans nano)

# Redémarrer le service
sudo systemctl restart omnimessage-smpp
```

4. **Vérifier la connexion** :
    - Naviguez vers : SMPP → État en direct
    - Trouvez la nouvelle connexion
    - Le statut devrait être "Connecté" (vert)
    - Vérifiez les journaux pour une liaison réussie
5. **Tester le flux de messages** :
    - Naviguez vers : Onglet Queue
    - Soumettez un message test avec `dest_smsc` correspondant au nouveau nom de liaison
    - Surveillez dans l'état en direct pour la transmission
    - Vérifiez la confirmation de livraison

Ajouter une liaison serveur

Objectif : Configurer la passerelle pour agir en tant que **SMSC** (serveur) acceptant des connexions de **ESMEs** externes (clients partenaires)

Préparation :

1. **Générer des identifiants** :
    - Créer un ID système unique : `partner_name`
    - Créer un mot de passe fort
    - Documenter et partager en toute sécurité avec le partenaire
2. **Obtenir les informations du partenaire** :
    - Adresses IP sources du partenaire
    - Volume de messages attendu (pour la limite TPS)
    - Types de liaison requis

Choisissez l'une des méthodes suivantes :

Option A : Via l'interface Web (Recommandée)

Avantages : Effet immédiat, aucun redémarrage requis

Étapes :

1. **Naviguer vers les pairs serveurs** :
   - Ouvrir l'interface Web : `https://your-server:8087`
   - Naviguer vers : SMPP → Pairs serveurs
2. **Ajouter un nouveau pair serveur** :
   - Cliquez sur "Ajouter un nouveau pair serveur"
   - Remplissez le formulaire :
     - **Nom** : `partner_acme` (identifiant unique)
     - **ID système** : `acme_corp`
     - **Mot de passe** : `secure_password_123`
     - **Types de liaison autorisés** : Sélectionnez tous (Transmitter, Receiver, Transceiver)
     - **Liste blanche IP** : `203.0.113.0/24` (séparé par des virgules pour plusieurs)
     - **Limite TPS** : `50`
     - **Fréquence de vérification de la file d'attente** : `1000`
   - Cliquez sur "Sauvegarder"

3. **Passerelle prête pour la connexion** :

- Le pair serveur est maintenant actif et attend la connexion du partenaire
- Aucun redémarrage requis

4. **Partager les informations avec le partenaire** :
   - Adresse IP de la passerelle
   - Port : `2775`
   - ID système : `acme_corp`
   - Mot de passe : `secure_password_123`
   - Type de liaison : Comme configuré
5. **Attendre la connexion du partenaire** :
   - Naviguez vers : SMPP → État en direct
   - Surveillez la connexion entrante
   - Vérifiez le succès de l'authentification
   - Vérifiez que l'IP correspond à la liste blanche

Option B : Via le fichier de configuration

Avantages : Infrastructure en tant que code, contrôle de version

Étapes :

1. **Modifier le fichier de configuration** :

```
sudo nano /opt/omnimessage-smpp/config/runtime.exs
```

2. **Ajouter la liaison serveur et la configuration d'écoute** :

```
# Ajouter à la liste des liaisons serveur
config :omnimessage_smpp, :server_binds, [
  # Liaisons serveur existantes...

  # Ajouter une nouvelle liaison serveur
  %{
    name: "partner_acme",
    system_id: "acme_corp",
    password: "secure_password_123",
    allowed_bind_types: [:transmitter, :receiver,
:transceiver],
    ip_whitelist: ["203.0.113.0/24"],
    tps_limit: 50,
    queue_check_frequency: 1000
  }
]

# Assurez-vous que la configuration d'écoute existe (nécessaire
une seule fois)
config :omnimessage_smpp, :listen, %{
  host: "0.0.0.0",
  port: 2775,
  max_connections: 100
}
```

3. **Sauvegarder et redémarrer le service** :

```
sudo systemctl restart omnimessage-smpp
```

4. **Partager les informations avec le partenaire** :
   - Adresse IP de la passerelle
   - Port : `2775`
   - ID système : `acme_corp`
   - Mot de passe : `secure_password_123`
   - Type de liaison : Comme configuré
5. **Attendre la connexion du partenaire** :
   - Naviguez vers : SMPP → État en direct

- Surveillez la connexion entrante
- Vérifiez le succès de l'authentification
- Vérifiez que l'IP correspond à la liste blanche

Modifier une connexion existante

Objectif : Mettre à jour les paramètres de connexion (limites TPS, mots de passe, liste blanche IP, etc.)

Choisissez l'une des méthodes suivantes :

Option A : Via l'interface Web (Recommandée)

Avantages : Effet immédiat, aucun redémarrage requis

Étapes :

1. **Naviguer vers les pairs** :
   - Ouvrir l'interface Web : `https://your-server:8087`
   - Pour les connexions clients : SMPP → Pairs clients
   - Pour les connexions serveur : SMPP → Pairs serveurs
2. **Modifier le pair** :
   - Trouvez le pair à modifier
   - Cliquez sur le bouton "Modifier"
   - Mettez à jour les paramètres souhaités :
     - Modifications courantes : Limite TPS, mot de passe, liste blanche IP, hôte/port
   - Cliquez sur "Sauvegarder"
3. **Les modifications s'appliquent immédiatement** :
   - La connexion se reconnecte automatiquement avec les nouveaux paramètres
   - Aucun redémarrage de service requis
   - Naviguez vers : SMPP → État en direct pour vérifier

4. **Vérifier les modifications** :
    - Vérifiez que la connexion s'établit avec succès
    - Surveillez l'onglet Logs pour les erreurs
    - Testez le flux de messages si applicable

Option B : Via le fichier de configuration

Avantages : Infrastructure en tant que code, contrôle de version

Étapes :

1. **Modifier le fichier de configuration** :

```
sudo nano /opt/omnimessage-smpp/config/runtime.exs
```

2. **Modifier les paramètres de liaison** :
    - Trouvez la liaison dans la liste `:binds` ou `:server_binds`
    - Mettez à jour les paramètres souhaités :
        - Modifications courantes : Limite TPS, mots de passe, liste blanche IP, hôte/port
    - Exemple :

```
%{
  name: "vodafone_uk",
  # ... autres params
  tps_limit: 150,  # Changé de 100
  password: "new_password"  # Mot de passe mis à jour
}
```

3. **Sauvegarder et redémarrer le service** :

```
sudo systemctl restart omnimessage-smpp
```

4. **Vérifier les modifications** :
    - Naviguez vers : SMPP → État en direct

- Vérifiez que la connexion s'établit avec succès
- Surveillez les journaux pour les erreurs
- Testez le flux de messages

Supprimer une connexion

Objectif : Décommissionner une connexion SMPP

Étapes :

1. **Notifier les parties prenantes** :
   - Informer l'opérateur/partenaire
   - Coordonner une fenêtre d'arrêt
2. **Déconnecter via l'interface Web** :
   - Naviguez vers : SMPP → État en direct
   - Trouvez la connexion
   - Cliquez sur "Drop Connection"
   - Confirmez l'action
3. **Supprimer la configuration** :
   - Naviguez vers : SMPP → Pairs clients/Pairs serveurs
   - Trouvez la connexion
   - Cliquez sur "Supprimer"
   - Confirmez la suppression
4. **Vérifier la suppression** :
   - Vérifiez l'état en direct - la connexion devrait avoir disparu
   - Consultez les journaux pour un arrêt propre

Activer et désactiver des connexions

Objectif : Prendre temporairement une connexion hors ligne sans supprimer sa configuration

Les pairs ont un champ `enabled` qui contrôle s'ils sont actifs. Les pairs désactivés conservent toute leur configuration mais n'établissent ni n'acceptent de connexions.

Via l'interface Web :

1. Naviguez vers : SMPP → Pairs clients ou Pairs serveurs
2. Trouvez le pair à désactiver
3. Cliquez sur "Modifier"
4. Décochez la case "Enabled"
5. Cliquez sur "Sauvegarder"

La connexion sera immédiatement interrompue. Pour réactiver, répétez les étapes et cochez à nouveau la case.

Cas d'utilisation :

- Fenêtres de maintenance planifiées de l'opérateur
- Mettre temporairement en pause une connexion partenaire pendant une enquête
- Désactiver une connexion en attendant de nouveaux identifiants

---

Comportement de connexion

Logique de reconnexion

Lorsqu'une liaison client ne peut pas se connecter, a son lien rejeté, ou perd une session établie pour une raison quelconque, la passerelle se reconnecte automatiquement :

- **Backoff** : La première tentative de reconnexion a lieu après 1 seconde, puis l'attente double à chaque échec consécutif (2s, 4s, 8s ...) jusqu'à un maximum de 30 secondes
- **Réinitialisation du backoff** : L'attente revient à 1 seconde une fois qu'une session est restée liée pendant 30 secondes. Un opérateur qui

accepte la connexion et la rejette immédiatement est donc réessayé à un rythme plus lent, pas chaque seconde

- **Connecté signifie lié** : Un lien n'est affiché comme "Connecté", et n'envoie du trafic, qu'une fois que l'opérateur a accepté le lien. Pendant que la connexion TCP est active mais que la réponse de liaison n'est pas encore arrivée, le lien est affiché comme en cours de connexion
- **Délai d'attente de connexion** : Une connexion TCP qui ne reçoit pas de réponse est abandonnée après 3 secondes et réessayée avec le backoff ci-dessus
- **Isolation** : Les reconnexions pour un lien ne redémarrent jamais d'autres liens ou le service, peu importe combien de fois l'opérateur interrompt la connexion
- **Démarrage échelonné** : Lorsque plusieurs liaisons démarrent simultanément (par exemple, après un redémarrage du service), les connexions sont échelonnées avec des délais de 500 ms entre chaque liaison pour éviter de surcharger le réseau
- **Démarrage résilient** : Si un opérateur est inaccessible au démarrage de la passerelle, celle-ci démarre avec succès et réessaie la connexion en arrière-plan

Suivez les reconnexions avec la métrique `smpp_reconnect_attempts_total` (voir MONITORING.md).

Déconnexion propre

Lorsqu'une liaison client est arrêtée par "Drop Connection" ou en arrêtant ou redémarrant le service, la passerelle envoie un `unbind` SMPP à l'opérateur et attend jusqu'à 2 secondes pour le `unbind_resp` avant de fermer la connexion TCP. L'opérateur libère le lien immédiatement plutôt que d'attendre que son propre keepalive expire. Cela est important pour les opérateurs qui n'autorisent qu'un seul lien par compte, qui refuseraient sinon la reconnexion après un redémarrage.

Un lien qui est encore en cours de connexion (TCP actif, liaison pas encore acceptée) est fermé sans un `unbind`.

Enquête de lien (Keepalive)

La passerelle envoie périodiquement des PDU SMPP `enquire_link` pour vérifier que les connexions sont actives :

- **Intervalle par défaut** : 60 secondes (configurable par liaison via `enquire_link_interval`)
- **Désactiver** : Définir `enquire_link_interval: 0` (non recommandé)
- **Détection des échecs** : Si le pair distant cesse de répondre à l'enquête de lien, la connexion est considérée comme morte et la reconnexion commence

Surveillez la santé de l'enquête de lien via les métriques Prometheus `smpp_enquire_link_sent_total` et `smpp_enquire_link_received_total`. Un écart croissant entre les envoyés et les reçus indique des problèmes de connexion.

Limitation de taux TPS

Chaque liaison applique sa `tps_limit` en utilisant une fenêtre glissante par seconde :

- Les messages sont comptés dans chaque fenêtre de 1 seconde
- Lorsque la limite est atteinte, le worker de la file d'attente fait une pause jusqu'à la seconde suivante
- Un maximum de 100 messages peut être en cours (en attente de réponse) par liaison à tout moment
- La fenêtre se réinitialise automatiquement au début de chaque nouvelle seconde

Si vous constatez un faible débit, vérifiez que :

1. `tps_limit` est suffisamment élevé pour votre trafic
2. `queue_check_frequency` est suffisamment bas pour maintenir le pipeline alimenté
3. L'opérateur répond aux messages rapidement (des réponses lentes réduisent le débit effectif)

Gestion du flux de messages

Vérification de la file d'attente des messages

Objectif : Surveiller les messages en attente

Étapes :

1. **Accéder à la file d'attente** :
    - Naviguez vers : Onglet Queue
    - Voir la liste des messages en attente

2. **Vérifier les détails du message** :
    - Cliquez sur la ligne du message
    - Examiner :
        - Numéro de destination
        - Corps du message
        - SMSC cible (dest_smsc)
        - Tentatives de livraison
        - Statut

3. **Rechercher un message spécifique** :

- Utilisez le filtre de recherche
- Filtrer par destination, contenu ou SMSC

Dépannage des messages bloqués

Symptômes : Les messages ne sont pas livrés

Étapes :

1. **Vérifier l'état de la connexion** :
   - Naviguez vers : SMPP → État en direct
   - Vérifiez que la connexion cible est connectée
   - Si déconnecté, voir Reconnecter
2. **Vérifier les détails du message** :
   - Naviguez vers : Onglet Queue
   - Trouvez le message bloqué
   - Vérifiez que le champ `dest_smsc` correspond au nom de la connexion
   - Vérifiez l'horodatage `deliver_after` (planification de la nouvelle tentative)
3. **Vérifier les tentatives de livraison** :
   - Tentatives élevées = échecs répétés
   - Vérifiez les journaux pour les messages d'erreur
   - Peut indiquer un format invalide ou un rejet de l'opérateur
4. **Intervention manuelle** (si nécessaire) :
   - Contacter l'opérateur pour vérifier le problème
   - Peut nécessiter d'annuler et de soumettre à nouveau le message
   - Vérifiez avec l'équipe backend pour des problèmes de file d'attente

---

Passerelle des accusés de réception de livraison (DLR)

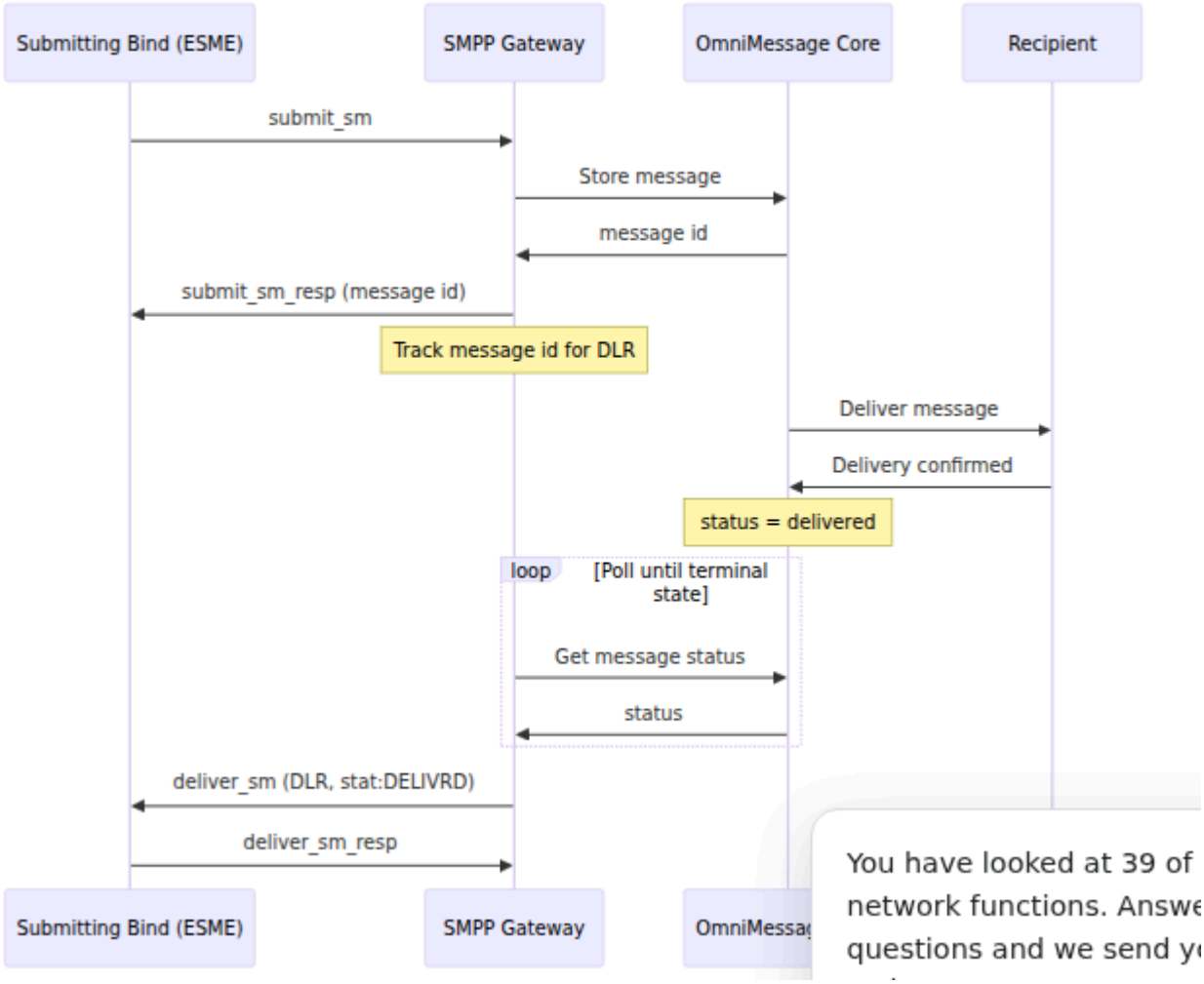
Un accusé de réception de livraison (DLR) est un message SMPP qui signale l'état final d'un message soumis. La passerelle envoie un DLR à une liaison soumise lorsque la liaison définit `deliver_receipts: true`. Utilisez cette fonctionnalité pour une liaison interconnectée dont l'opérateur a besoin d'une confirmation de livraison pour chaque message.

La passerelle envoie le reçu sous forme de PDU `deliver_sm` avec `esm_class` défini sur `0x04` (Accusé de réception de livraison SMSC). Le reçu porte le même identifiant de message que celui que la passerelle a renvoyé dans le `submit_sm_resp`, afin que l'opérateur puisse corréler le reçu avec le soumission originale.

Exigences

- Définir `deliver_receipts: true` sur le serveur bind. Voir CONFIGURATION.md.
- Définir `cache_enabled: false` sur le même bind. Le cache renvoie un identifiant de message temporaire qui ne correspond pas à l'identifiant principal, donc un bind mis en cache casse la corrélation des reçus. La passerelle écrit un avertissement au démarrage si un bind définit les deux.

Flux de messages



Valeurs de statut de reçu

Le champ `stat` du reçu rapporte l'état final du message.

Statut du Core	Reçu `stat`	Signification
`delivered`	`DELIVRD`	Le destinataire a reçu le message.
`expired`	`EXPIRED`	La période de validité du message a pris fin avant la livraison.
`dropped`	`UNDELIV`	La passerelle n'a pas pu livrer le message.
`rejected`	`REJECTD`	Le core a rejeté le message.
`balance_rejected`	`REJECTD`	Le core a rejeté le message pour une raison de solde de compte.

Remarques opérationnelles

- La passerelle suit les messages en cours dans la mémoire. Si le service redémarre, les reçus pour les messages en cours à ce moment-là sont perdus. C'est un mécanisme de meilleur effort.
- Si la session de bind est hors ligne lorsque le message atteint un état terminal, la passerelle conserve le reçu et l'envoie lorsque la session se reconnecte. La passerelle abandonne le reçu après 24 heures.
- Surveillez les métriques DLR pour confirmer que la fonctionnalité fonctionne. Voir MONITORING.md.

Dépannage de connexion

Reconnexion d'un Bind

Symptômes : La connexion affiche "Déconnecté" (rouge)

Étapes :

1. **Vérifier la connectivité réseau** :

```
ping -c 3 carrier-smpp-server.com
telnet carrier-smpp-server.com 2775
```

2. **Vérifier les journaux pour des erreurs** :

   - Naviguer vers : Onglet Journaux
   - Filtrer : Niveau d'erreur
   - Rechercher des échecs d'authentification, des délais d'attente réseau

3. **Vérifier les identifiants** :

   - Naviguer vers : SMPP → Pairs Client/Serveur
   - Vérifier que system_id et mot de passe sont corrects
   - Contacter le transporteur si incertain

4. **Reconnecter manuellement** :

   - Naviguer vers : SMPP → État en direct
   - Trouver le bind déconnecté
   - Cliquer sur le bouton "Reconnecter"
   - Attendre 10-30 secondes
   - Vérifier si le statut change en "Connecté"
   - La reconnexion ramène également un bind qui a été arrêté avec "Drop"

5. **Si la reconnexion échoue** :
    - Vérifier les règles de pare-feu
    - Vérifier que le serveur du transporteur est opérationnel
    - Contacter le support du transporteur
    - Voir TROUBLESHOOTING.md

Gestion des échecs d'authentification

Symptômes : Échecs de bind répétés dans les journaux

Causes :

- Nom d'utilisateur/mot de passe incorrect
- IP non autorisée chez le transporteur
- Compte suspendu/expiré

Étapes :

1. **Vérifier les identifiants** :
    - Naviguer vers : SMPP → Pairs Clients
    - Vérifier à nouveau system_id et mot de passe
    - Confirmer avec le transporteur
2. **Vérifier la liste blanche des IP** :
    - Confirmer votre IP de passerelle avec le transporteur
    - Demander au transporteur de vérifier la liste blanche des IP
3. **Vérifier l'état du compte** :
    - Vérifier que le compte est actif
    - Vérifier les contrats expirés
    - Contacter la facturation du transporteur
4. **Mettre à jour la configuration** :
    - Si les identifiants ont changé, mettre à jour dans l'interface Web

- Cliquer sur "Reconnecter" pour réessayer avec les nouveaux identifiants

---

Surveillance et alertes

Vérification des métriques Prometheus

Vérification rapide :

```
curl http://localhost:4000/metrics | grep smpp_connection_status
```

Sortie attendue :

```
smpp_connection_status{bind_name="vodafone_uk",...} 1
smpp_connection_status{bind_name="att_us",...} 1
```

Toutes les valeurs devraient être `1` (connecté).

Répondre aux alertes

Alerte de connexion hors ligne :

1. Vérifier l'interface Web → SMPP → État en direct
2. Tenter une reconnexion manuelle
3. Vérifier les journaux pour des erreurs
4. Contacter le transporteur en cas de panne prolongée
5. Voir TROUBLESHOOTING.md

Alerte de taux d'échec élevé :

1. Vérifier les journaux pour des motifs d'erreur
2. Revoir les changements de configuration récents
3. Contacter le transporteur au sujet des rejets
4. Vérifier la conformité du format des messages

Alerte de trafic nul :

1. Vérifier que la file d'attente en arrière-plan a des messages
2. Vérifier que le routage `dest_smsc` est correct
3. Vérifier que les limites TPS ne sont pas trop restrictives
4. Revoir le paramètre `queue_check_frequency`

Procédures de maintenance

Maintenance de routine

Effectuer mensuellement :

1. **Revoir les métriques** :
    - Analyser les tendances du volume de messages
    - Vérifier les taux de succès de livraison
    - Identifier les opportunités d'optimisation
2. **Mettre à jour la documentation** :
    - Documenter tout changement de configuration
    - Mettre à jour les informations de contact
    - Noter les fenêtres de maintenance du transporteur
3. **Audit des identifiants** :
    - Revoir tous les mots de passe SMPP
    - Planifier la rotation des identifiants
    - Vérifier que les listes blanches des IP sont à jour
4. **Planification de capacité** :
    - Revoir les taux de messages de pointe
    - Vérifier par rapport aux limites TPS
    - Planifier la croissance

Redémarrage du service

Lorsque nécessaire :

- Après des changements de fichier de configuration
- Après des mises à jour du système
- Lors du dépannage

Étapes :

```
# Vérifier l'état actuel
sudo systemctl status omnimessage-smpp

# Redémarrer le service
sudo systemctl restart omnimessage-smpp

# Vérifier le redémarrage
sudo systemctl status omnimessage-smpp

# Vérifier les journaux
sudo journalctl -u omnimessage-smpp -n 50
```

Vérifier via l'interface Web :

1. Accéder au tableau de bord (peut prendre 30-60 secondes pour se mettre en ligne)
2. Naviguer vers : SMPP → État en direct
3. Attendre que toutes les connexions s'établissent (1-2 minutes)
4. Vérifier les journaux pour des erreurs

Sauvegarde de la configuration

Sauvegarder les fichiers critiques avant les changements :

```
# Sauvegarder la configuration
sudo cp /opt/omnimessage-smpp/config/runtime.exs \
  /opt/omnimessage-smpp/config/runtime.exs.backup.$(date +%Y%m%d)

# Sauvegarder les certificats
sudo tar -czf /tmp/smpp-certs-$(date +%Y%m%d).tar.gz \
  /opt/omnimessage-smpp/priv/cert/
```

Restaurer si nécessaire :

```
# Restaurer la configuration
sudo cp /opt/omnimessage-smpp/config/runtime.exs.backup.YYYYMMDD \
  /opt/omnimessage-smpp/config/runtime.exs

# Redémarrer le service
sudo systemctl restart omnimessage-smpp
```

Procédures d'urgence

Panne de service complète

Étapes :

1. **Vérifier l'état du service** :

   ```
   sudo systemctl status omnimessage-smpp
   ```

2. **Si le service est arrêté, démarrez-le** :

   ```
   sudo systemctl start omnimessage-smpp
   ```

3. **Vérifier les journaux pour la raison du crash** :

```
sudo journalctl -u omnimessage-smpp -n 100
```

4. **Si ne démarre pas** :
   - Vérifier les erreurs de syntaxe de configuration
   - Vérifier que les certificats SSL existent
   - Vérifier l'espace disque : `df -h`
   - Vérifier la mémoire : `free -h`
5. **Contacter le support** si non résolu

Demandes d'urgence de déconnexion du transporteur

Étapes :

1. **Couper la connexion immédiatement** :
   - Naviguer vers : SMPP → État en direct
   - Trouver la connexion affectée
   - Cliquer sur "Couper la connexion"
2. **Documenter la raison** :
   - Noter le nom du transporteur
   - Enregistrer l'heure et la raison
   - Sauvegarder la correspondance
3. **Enquêter sur le problème** :
   - Vérifier les motifs de messages récents
   - Revoir les journaux pour des erreurs
   - Identifier la cause profonde
4. **Coordonner la résolution** :
   - Travailler avec le transporteur
   - Mettre en œuvre des corrections

- Tester avant de se reconnecter

Pic de volume élevé

Symptômes : Trafic de messages anormalement élevé

Étapes :

1. **Vérifier les limites TPS** :
   - Naviguer vers : SMPP → État en direct
   - Vérifier que les connexions ne sont pas throttling
   - Peut nécessiter d'augmenter temporairement les limites TPS
2. **Surveiller la stabilité du transporteur** :
   - Surveiller les déconnexions
   - Vérifier les taux de succès de livraison
3. **Coordonner avec l'arrière-plan** :
   - Vérifier que la source de message est légitime
   - Peut nécessiter de mettre en œuvre une limitation de débit en amont
4. **Élargir si nécessaire** :
   - Peut nécessiter des instances supplémentaires de passerelle
   - Contacter le support pour des conseils d'élargissement

Meilleures pratiques

Liste de contrôle quotidienne

- [ ] Vérifier que toutes les connexions SMPP sont connectées
- [ ] Revoir les journaux d'erreurs pour tout problème
- [ ] Surveiller la file d'attente des messages pour l'accumulation

- [ ] Vérifier les tableaux de bord Prometheus/Grafana
- [ ] Vérifier que les taux de succès de livraison > 98%

Tâches hebdomadaires

- [ ] Revoir les tendances des métriques
- [ ] Vérifier les anomalies de motifs
- [ ] Tester les procédures de récupération après sinistre
- [ ] Mettre à jour la documentation si nécessaire
- [ ] Revoir et reconnaître les alertes

Tâches mensuelles

- [ ] Audit des identifiants
- [ ] Revue de la planification de capacité
- [ ] Mettre à jour les contacts du transporteur
- [ ] Revoir et optimiser les paramètres TPS
- [ ] Sauvegarder les fichiers de configuration

Documentation connexe

- **CONFIGURATION.md** - Configurer les connexions et les paramètres
- **SOURCE_ADDRESS_WHITELIST.md** - Restreindre les adresses d'origine par pair de serveur
- **MONITORING.md** - Configurer l'alerte Prometheus
- **TROUBLESHOOTING.md** - Résoudre les problèmes courants
- **README.md** - Vue d'ensemble du système
