Referencia de Configuración

Guía completa de todos los parámetros de configuración

Visión General de la Arquitectura

El Gateway SMPP de OmniMessage es un **frontend de protocolo sin estado** que traduce mensajes SMPP hacia/desde OmniMessage. Toda la lógica de negocio, decisiones de enrutamiento y almacenamiento de mensajes son manejados por OmniMessage Core - el gateway simplemente:

1. **Recibe** PDUs SMPP de transportistas y clientes
2. **Traduce** estos al formato de OmniMessage a través de la API REST
3. **Consulta** a OmniMessage por mensajes para enviar
4. **Envía** PDUs SMPP a los transportistas
5. **Reporta** el estado de entrega de vuelta a OmniMessage

Esto es idéntico a cómo funcionan otros frontends de OmniMessage (Diameter, MAP, IMS) - todos son traductores de protocolo sin estado que delegan en OmniMessage Core.

Ubicación del Archivo de Configuración

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

Importante: Después de cambiar la configuración, reinicie el gateway:

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

Estructura de Configuración

El archivo de configuración utiliza la sintaxis de Elixir. Estructura básica:

```
import Config

# Configuraciones globales
config :omnimessage_smpp,
  setting_name: value

# Vínculos SMPP
config :omnimessage_smpp, :binds, [
  %{
    name: "bind_name",
    # ... configuraciones de vínculo
  }
]
```

Configuraciones Globales

API_BASE_URL

URL de la plataforma OmniMessage Core

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

Parámetro	Tipo	Requerido	Predeterminado
`api_base_url`	String (URL)	Sí	-

Propósito: URL de la plataforma OmniMessage Core. El gateway se comunica con OmniMessage a través de la API REST para todo el procesamiento de mensajes:

- **Enviar Mensajes**: Enviar mensajes SMPP recibidos a OmniMessage para su procesamiento
- **Recuperar Mensajes**: Consultar mensajes destinados a transportistas SMPP
- **Reportar Estado de Entrega**: Actualizar el estado de entrega de mensajes de vuelta a OmniMessage
- **Salud del Sistema**: Verificaciones de salud periódicas

Crítico: Aquí es donde el gateway obtiene toda su "inteligencia". OmniMessage maneja:

- ✓ Validación de mensajes y verificación de formato
- ✓ Decisiones de enrutamiento (qué transportista usar)
- ✓ Limitación de tasa y control de flujo
- ✓ Validación de números
- ✓ Almacenamiento y persistencia de mensajes
- ✓ Lógica de reintento de entrega
- ✓ Seguimiento de estado

El gateway simplemente traduce el formato SMPP ↔ OmniMessage.

Ejemplos:

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

# HTTPS con nombre de host
api_base_url: "https://omnimessage-core.company.com:8443"

# HTTP (no recomendado para producción)
api_base_url: "http://192.168.1.100:8080"
```

Requisitos de Red:

- El gateway debe tener acceso a la red de OmniMessage Core
- Usar HTTPS en producción (configurar `verify_ssl_peer`)
- El firewall debe permitir HTTPS saliente en el puerto especificado

OAM/Management API port (`api_ex`)

Puerto para la API OAM/Management del gateway

El gateway sirve su **propia** API OAM/Management (`/status`, `/smpp/client_peers`, `/smpp/server_peers`, `/smpp/binds`, `/smpp/summary`) a través de `api_ex`, que por defecto es `8443` (`config/config.exs`). Este es un **servidor** que el gateway hospeda - distinto de `api_base_url`, que es el **endpoint** del cliente que el gateway llama en OmniMessage Core.

No confundir los dos:

Configuración	Dirección	Predeterminado	Significado
`api_base_url`	Gateway → Core (cliente)	`…:8443`	Donde el gateway envía/consulta mensajes
`api_ex` `port`	Gateway (servidor)	`8443`	Donde escucha la propia API OAM del gateway

Colocación con OmniMessage Core: OmniMessage Core también hospeda su API de almacenamiento/mensajes en el puerto `api_ex` `8443`. Cuando el gateway SMPP se ejecuta en el **mismo host** que Core, ambos usan por defecto `8443` y solo uno puede enlazarlo - el otro entra en bucle de fallos con `:eaddrinuse` (y las solicitudes a `/api/messages` llegan al servicio incorrecto, devolviendo 404).

Para solucionarlo, edite `config/runtime.exs` (desplegado en `/etc/omnimessage_smpp/runtime.exs`) y descomente la anulación de `api_ex`, configurando un puerto libre para que la API OAM del gateway se mueva del puerto de Core. `api_base_url` aún apunta al `8443` de Core:

```
# Core posee :8443 para la API de almacenamiento; mover la propia
API OAM del gateway a :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 anulación se deja comentada, el puerto predeterminado es `8443`. Los gateways independientes (Core en un host diferente) no necesitan cambios.

SMPP_POLL_INTERVAL

Frecuencia de verificación de la cola (milisegundos)

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

Parámetro	Tipo	Requerido	Predeterminado
`smpp_poll_interval`	Entero	No	100

Propósito: Con qué frecuencia (en milisegundos) cada cliente verifica la cola de mensajes.

Directrices:

- **Alto volumen (>100 TPS)**: 100-500ms
- **Volumen medio (10-100 TPS)**: 500-1000ms
- **Bajo volumen (<10 TPS)**: 1000-2000ms

Variable de entorno: `SMPP_POLL_INTERVAL`

VERIFY_SSL_PEER

Verificación de certificado SSL

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

Parámetro	Tipo	Requerido	Predeterminado
`verify_ssl_peer`	Booleano	No	false

Propósito: Si verificar certificados SSL al conectarse a la API de backend.

Valores:

- `true`: Verificar certificados (producción con certificados válidos)
- `false`: Saltar verificación (certificados autofirmados, pruebas)

Variable de entorno: `VERIFY_SSL_PEER`

SMSC_NAME

Identificador del gateway para registro

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

Parámetro	Tipo	Requerido	Predeterminado
`smsc_name`	String	No	"smpp_gateway"

Propósito: Identifica esta instancia de gateway en el backend de la cola de mensajes.

Variable de entorno: `SMSC_NAME`

Configuración de Vínculo del Cliente SMPP

Los vínculos del cliente son **conexiones salientes** donde el gateway actúa como un **ESME** (cliente) conectándose a servidores **SMSC** de transportistas. En este modo, el gateway inicia la conexión para enviar y recibir mensajes a través de transportistas externos.

Ejemplo Completo de Vínculo del Cliente

```
config :omnimessage_smpp, :binds, [
  %{
    # Identificador único para esta conexión
    name: "vodafone_uk",

    # Modo de conexión
    mode: :client,

    # Tipo de vínculo SMPP
    bind_type: :transceiver,

    # Dirección del servidor SMPP del transportista
    host: "smpp.vodafone.co.uk",
    port: 2775,

    # Fijar la dirección/puerto de origen local de la conexión
saliente (opcional,
    # establecer si el transportista blinda por IP/puerto de
origen)
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5000,

    # Credenciales de autenticación
    system_id: "your_username",
    password: "your_password",

    # Campos del protocolo SMPP (opcional, establecer si el
transportista lo requiere)
    system_type: "",
    addr_ton: 0,
    addr_npi: 0,
    address_range: "",

    # Limitación de tasa
    tps_limit: 100,

    # Nombre de cola compartida (opcional, predeterminado al
nombre del vínculo). Establecer el mismo
    # valor en varios vínculos para que drenen una cola común.
    queue: "vodafone_uk",
```

```
    # Frecuencia de verificación de la cola
    queue_check_frequency: 1000,

    # Intervalo de keepalive (segundos, 0 para deshabilitar)
    enquire_link_interval: 60,

    # Caché de mensajes (opcional)
    cache_enabled: false,
    cache_max_size: 10000,
    cache_retry_interval: 60
  }
]
```

Parámetros del Vínculo del Cliente

name

Identificador único de conexión

Tipo	Requerido	Ejemplo
String	Sí	`"vodafone_uk"`

Propósito: Identifica de manera única esta conexión SMPP.

- Usado en registros y métricas
- Debe ser único entre todos los vínculos
- Usar nombres descriptivos (transportista, región, propósito)

Convenciones de nomenclatura:

- `transportista_región`: `"vodafone_uk"`, `"att_us"`
- `propósito_número`: `"marketing_1"`, `"alerts_primary"`

mode

Tipo de conexión

Tipo	Requerido	Valor
Átomo	Sí	`:client`

Propósito: Define esto como una conexión saliente donde el gateway actúa como un **ESME** conectándose a un **SMSC** externo.

Valor fijo: Siempre `:client` para conexiones salientes.

bind_type

Tipo de sesión SMPP

Tipo	Requerido	Valores Permitidos
Átomo	Sí	`:transmitter`, `:receiver`, `:transceiver`

Propósito: Define la capacidad de dirección del mensaje.

Opciones:

- `:transmitter` - Solo enviar mensajes (submit_sm)
- `:receiver` - Solo recibir mensajes (deliver_sm)
- `:transceiver` - Enviar y recibir (más común)

Recomendación: Usar `:transceiver` a menos que el transportista requiera un tipo específico.

host

Nombre de host o IP del servidor SMPP del transportista

Tipo	Requerido	Ejemplo
String	Sí	`"smpp.carrier.com"` o `"10.5.1.100"`

Propósito: Dirección del servidor SMPP del transportista.

Ejemplos:

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

port

Puerto del servidor SMPP

Tipo	Requerido	Predeterminado	Rango
Entero	Sí	2775	1-65535

Propósito: Puerto TCP para la conexión SMPP.

Puerto estándar: 2775

Ejemplos:

```
port: 2775  # Estándar
port: 3000  # Personalizado
```

bind_source_ip

IP de origen local para la conexión saliente

Tipo	Requerido	Predeterminado	Ejemplo
String	No	(predeterminado del SO)	`"10.20.30.40"`

Propósito: Fija la dirección IP de origen local utilizada para la conexión TCP saliente al transportista. Establecer esto cuando el transportista blinda conexiones por IP de origen y el host del gateway tiene múltiples direcciones. Una dirección no válida se registra y se ignora (el SO elige la IP de origen).

Nota: Este es un campo solo de configuración (leído desde la configuración de la aplicación `:binds`); no es editable desde la interfaz web / almacenamiento Mnesia.

bind_source_port

Puerto de origen local para la conexión saliente

Tipo	Requerido	Predeterminado	Rango
Entero	No	(efímero)	1-65535

Propósito: Fija el puerto de origen local utilizado para la conexión TCP saliente. Establecer esto cuando el transportista blinda por IP de origen **y** puerto. El socket se abre con `SO_REUSEADDR` para que el puerto fijado pueda volver a enlazarse inmediatamente en la reconexión (en lugar de fallar con `:eaddrinuse` mientras el socket anterior está en `TIME_WAIT`).

Nota: Al igual que `bind_source_ip`, este es un campo solo de configuración y no es editable desde la interfaz web.

system_id

Nombre de usuario de autenticación

Tipo	Requerido	Ejemplo
String	Sí	`"company_user"`

Propósito: Nombre de usuario proporcionado por el transportista para la autenticación.

Seguridad: Proteger esta credencial - almacenada en el archivo de configuración.

password

Contraseña de autenticación

Tipo	Requerido	Ejemplo
String	Sí	`"secret_password"`

Propósito: Contraseña proporcionada por el transportista para la autenticación.

Seguridad:

- Proteger esta credencial
- Usar contraseñas fuertes
- Rotar periódicamente

tps_limit

Límite de transacciones por segundo

Tipo	Requerido	Predeterminado	Rango
Entero	Sí	100	1-10000

Propósito: Máximo de mensajes por segundo a enviar a través de esta conexión.

Directrices:

- Establecer en 70-80% del máximo del transportista
- Previene la limitación/desconexión
- Permite margen para recibos de entrega

Ejemplos:

```
tps_limit: 10    # Bajo volumen
tps_limit: 50    # Volumen medio
tps_limit: 100   # Alto volumen (más común)
tps_limit: 1000  # Muy alto volumen
```

Cálculo:

```
Si el máximo del transportista = 100 TPS
Establecer tps_limit = 70-80
Deja 20-30 TPS de margen
```

queue

Nombre de cola compartida

Tipo	Requerido	Predeterminado	Ejemplo
String	No	nombre del `bind`	`"sinch.smpp"`

Propósito: La cola de la que este vínculo drena del backend. Predeterminado al propio `name` del vínculo, por lo que cada vínculo drena su propia cola. Establecer el mismo valor de `queue` en varios vínculos para que todos drenen una cola compartida - por ejemplo, dos conexiones de Sinch que drenan `"sinch.smpp"` para compartir carga/redundancia.

Ejemplos:

```
# Dos vínculos compartiendo una cola
%{name: "sinch_1", queue: "sinch.smpp", ...}
%{name: "sinch_2", queue: "sinch.smpp", ...}
```

queue_check_frequency

Intervalo de sondeo de la cola de mensajes (milisegundos)

Tipo	Requerido	Predeterminado	Rango
Entero	Sí	1000	100-10000

Propósito: Con qué frecuencia consultar al backend por nuevos mensajes para enviar.

Directrices:

- **Alto volumen (>100 TPS)**: 500-1000ms
- **Volumen medio (10-100 TPS)**: 1000-2000ms
- **Bajo volumen (<10 TPS)**: 2000-5000ms

Compensaciones:

- Valor más bajo = recogida de mensajes más rápida, más carga en la API
- Valor más alto = recogida más lenta, menos carga en la API

enquire_link_interval

Intervalo de keepalive SMPP (segundos)

Tipo	Requerido	Predeterminado	Rango
Entero	No	60	0-3600

Propósito: Con qué frecuencia (en segundos) enviar PDUs enquire_link SMPP para verificar que la conexión está viva. El servidor remoto responde con enquire_link_resp.

Directrices:

- **Predeterminado (60)**: Adecuado para la mayoría de los transportistas
- **Valores más bajos (15-30)**: Detección de fallos más rápida, más tráfico
- **Valores más altos (120-300)**: Menos sobrecarga, detección de fallos más lenta
- **0**: Desactiva enquire_link por completo (no recomendado)

Ejemplos:

```
enquire_link_interval: 60   # Estándar (1 minuto)
enquire_link_interval: 30   # Keepalive agresivo
enquire_link_interval: 0    # Deshabilitado
```

system_type

Identificador de tipo de sistema SMPP

Tipo	Requerido	Predeterminado	Ejemplo
String	No	`""`	`"0TP"`

Propósito: Campo del protocolo SMPP enviado durante el vínculo. Algunos transportistas requieren un valor específico. Dejar en blanco a menos que el transportista especifique uno.

addr_ton

Tipo de Número de Dirección

Tipo	Requerido	Predeterminado	Rango
Entero	No	0	0-6

Propósito: Campo del protocolo SMPP que especifica el tipo de número utilizado en la solicitud de vínculo.

Valores comunes:

- `0` - Desconocido
- `1` - Internacional
- `2` - Nacional
- `5` - Alfanumérico

Establecer según lo requerido por el transportista.

addr_npi

Indicador del Plan de Numeración de Dirección

Tipo	Requerido	Predeterminado	Rango
Entero	No	0	0-18

Propósito: Campo del protocolo SMPP que especifica el plan de numeración en la solicitud de vínculo.

Valores comunes:

- `0` - Desconocido
- `1` - ISDN/E.164
- `3` - Datos/X.121
- `9` - Privado

Establecer según lo requerido por el transportista.

address_range

Rango de direcciones para el vínculo

Tipo	Requerido	Predeterminado	Ejemplo
String	No	`""`	`"614*"`

Propósito: Campo del protocolo SMPP que especifica el rango de direcciones que este vínculo maneja. Usado por algunos transportistas para filtrar qué mensajes se entregan a esta conexión. Dejar en blanco a menos que el transportista especifique un valor.

enabled

Estado habilitado del par

Tipo	Requerido	Predeterminado
Booleano	No	`true`

Propósito: Controla si este par está activo. Los pares deshabilitados se mantienen en la configuración pero no establecen conexiones. Útil para tomar temporalmente una conexión fuera de línea sin eliminar su configuración.

cache_enabled

Habilitar caché local de mensajes

Tipo	Requerido	Predeterminado
Booleano	No	`false`

Propósito: Cuando está habilitado, los mensajes entrantes se almacenan en caché localmente si la API de backend no está disponible, y luego se entregan automáticamente cuando se restaura la conectividad. Consulte MESSAGE_CACHE.md para obtener todos los detalles.

cache_max_size

Máximo de mensajes en caché

Tipo	Requerido	Predeterminado	Rango
Entero	No	10000	1-1000000

Propósito: Número máximo de mensajes para almacenar en caché por vínculo. Cuando se alcanza el límite, los mensajes más antiguos se eliminan (FIFO). Solo se aplica cuando `cache_enabled` es `true`.

cache_retry_interval

Intervalo base de reintento (segundos)

Tipo	Requerido	Predeterminado
Entero	No	60

Propósito: Intervalo base en segundos antes de reintentar la entrega de un mensaje en caché. Combinado con retroceso exponencial (reintento 0: 60s, reintento 1: 120s, reintento 2: 240s, etc.). Solo se aplica cuando `cache_enabled` es `true`.

Ejemplo de Interfaz Web:

Configuración de Vínculo del Servidor SMPP

Los vínculos del servidor definen **conexiones entrantes** donde el gateway actúa como un **SMSC** (servidor) aceptando conexiones de **ESMEs** externos (clientes). En este modo, los sistemas asociados se conectan al gateway para enviar y recibir mensajes.

Ejemplo Completo de Vínculo del Servidor

```
config :omnimessage_smpp, :server_binds, [
  %{
    # Identificador único para este cliente
    name: "partner_acme",

    # Credenciales esperadas del cliente
    system_id: "acme_corp",
    password: "acme_secret",

    # Tipos de vínculo permitidos
    allowed_bind_types: [:transmitter, :receiver, :transceiver],

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

    # Restricciones de dirección de origen (vacío = permitir
todos)
    source_address_whitelist: [],

    # Limitación de tasa
    tps_limit: 50,

    # Frecuencia de verificación de la cola
    queue_check_frequency: 1000,

    # Intervalo de keepalive (segundos, 0 para deshabilitar)
    enquire_link_interval: 60,

    # Caché de mensajes (opcional)
    cache_enabled: false,
    cache_max_size: 10000,
    cache_retry_interval: 60,

    # Recibos de entrega (DLR) de vuelta a este vínculo (opcional)
    deliver_receipts: false
  }
]
```

Parámetros del Vínculo del Servidor

name

Identificador del cliente

Tipo	Requerido	Ejemplo
String	Sí	`"partner_acme"`

Propósito: Identifica al cliente externo que se conecta a usted.

Convenciones de nomenclatura: Usar el nombre del socio/cliente para fácil identificación.

system_id

Nombre de usuario esperado del cliente

Tipo	Requerido	Ejemplo
String	Sí	`"acme_corp"`

Propósito: Nombre de usuario que el cliente externo debe proporcionar para autenticarse.

Proporcionar al cliente: Compartir esta credencial con su socio.

password

Contraseña esperada del cliente

Tipo	Requerido	Ejemplo
String	Sí	`"secure_password"`

Propósito: Contraseña que el cliente externo debe proporcionar para autenticarse.

Seguridad:

- Usar contraseñas fuertes
- Única por cliente
- Compartir de forma segura con el socio

allowed_bind_types

Tipos de sesión permitidos

Tipo	Requerido	Predeterminado
Lista de Átomos	Sí	-

Propósito: Restringe qué tipos de vínculo puede usar el cliente.

Opciones:

```
allowed_bind_types: [:transceiver]  # Solo transceptor
allowed_bind_types: [:transmitter, :receiver]  # TX o RX
allowed_bind_types: [:transmitter, :receiver, :transceiver]  #
Cualquiera
```

Recomendación: Permitir los tres a menos que necesite restricciones.

ip_whitelist

Direcciones IP del cliente permitidas

Tipo	Requerido	Predeterminado	Formato
Lista de Strings	Sí	`[]`	IPs o notación CIDR

Propósito: Seguridad - permitir solo conexiones desde IPs conocidas.

Formatos:

- IP única: `"192.168.1.100"` (automáticamente `/32`)

- Subred CIDR: `"192.168.1.0/24"`, `"10.0.0.0/8"`
- Mezcla ambos: `["192.168.1.0/24", "10.50.1.100"]`

Ejemplos:

```
# Permitir cualquier IP (no recomendado)
ip_whitelist: []

# IP única
ip_whitelist: ["203.0.113.50"]

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

# Subred
ip_whitelist: ["192.168.1.0/24"]

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

Subredes comunes:

- `/32` - IP única (automático para IPs sin máscara)
- `/24` - 256 direcciones (por ejemplo, 192.168.1.0-255)
- `/16` - 65,536 direcciones (por ejemplo, 10.50.0.0-255.255)
- `/8` - 16,777,216 direcciones (por ejemplo, 10.0.0.0-255.255.255.255)

source_address_whitelist

Direcciones de origen permitidas

Tipo	Requerido	Predeterminado	Formato
Lista de Cadenas	No	`[]`	Patrones Regex

Propósito: Restringe qué direcciones de origen (ID de remitente) pueden usar los clientes conectados al enviar mensajes. Una lista vacía permite todas las direcciones. Cada entrada es un patrón de expresión regular.

Ejemplos:

```
# Permitir cualquier dirección de origen
source_address_whitelist: []

# Coincidencia exacta (anclada)
source_address_whitelist: ["^MyBrand$", "^AlertService$"]

# Coincidencia de prefijo
source_address_whitelist: ["^614", "^\+61"]

# Rango de números con clase de caracteres
source_address_whitelist: ["^61[45]\d{8}$"]

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

Los mensajes con direcciones de origen no permitidas son rechazados con `ESME_RINVSRCADR`. Consulte SOURCE_ADDRESS_WHITELIST.md para obtener todos los detalles.

tps_limit

Límite de mensajes por segundo

Igual que el `tps_limit` de enlace de cliente - controla la tasa de entrega de deliver_sm a los clientes conectados.

queue_check_frequency

Intervalo de sondeo de cola

Igual que el `queue_check_frequency` de enlace de cliente - con qué frecuencia verificar si hay mensajes para entregar a este cliente.

enquire_link_interval

Intervalo de mantenimiento de conexión SMPP (segundos)

Igual que el `enquire_link_interval` de enlace de cliente. Controla con qué frecuencia el servidor envía PDUs de enquire_link a los clientes conectados

para verificar que aún están activos.

enabled

Estado habilitado del par

Igual que el `enabled` de enlace de cliente. Los pares de servidor deshabilitados no aceptan conexiones entrantes.

cache_enabled

Habilitar almacenamiento en caché de mensajes local

Igual que el `cache_enabled` de enlace de cliente. Consulte MESSAGE_CACHE.md.

cache_max_size

Máximo de mensajes en caché

Igual que el `cache_max_size` de enlace de cliente.

cache_retry_interval

Intervalo base de reintento (segundos)

Igual que el `cache_retry_interval` de enlace de cliente.

deliver_receipts

Enviar recibos de entrega SMPP (DLRs) a este enlace

Tipo	Requerido	Predeterminado
Booleano	No	`false`

Propósito: La puerta de enlace envía un recibo de entrega SMPP de vuelta a este enlace por cada mensaje que envía. La puerta de enlace rastrea cada mensaje enviado y envía el recibo cuando el mensaje alcanza un estado

terminal. El recibo es un PDU `deliver_sm` con el `esm_class` configurado en `0x04` (Recibo de Entrega de SMSC), según SMPP v3.4 sección 5.2.12.

Cuándo usar: Establezca `deliver_receipts: true` para un enlace de interconexión dedicado cuyo operador necesita confirmación de entrega para cada mensaje. Mantenga el predeterminado `false` para enlaces que no necesitan recibos.

Requisito: El enlace DEBE ejecutarse con `cache_enabled: false`. El recibo utiliza el ID del mensaje que la puerta de enlace devolvió en el `submit_sm_resp`. La caché devuelve un ID temporal que no coincide con el ID principal, por lo que un enlace en caché rompe la correlación del recibo. Si un enlace establece `deliver_receipts: true` y `cache_enabled: true`, la puerta de enlace escribe una advertencia al inicio y el ID del recibo no coincide.

Cómo funciona: El enlace que envía envía un `submit_sm`. La puerta de enlace acepta el mensaje y devuelve un `submit_sm_resp` con un ID de mensaje. La puerta de enlace rastrea el mensaje y sondea el núcleo de OmniMessage para su estado. Cuando el mensaje es entregado, caducado o rechazado, la puerta de enlace envía un recibo `deliver_sm` de vuelta al enlace. El recibo lleva el mismo ID de mensaje y un campo de estado (`stat:DELIVRD`, `stat:EXPIRED`, `stat:UNDELIV` o `stat:REJECTD`).

El rastreador mantiene el estado en memoria solamente. Si la puerta de enlace se reinicia, los recibos de los mensajes que están en tránsito en ese momento se pierden. Este es un mecanismo de mejor esfuerzo.

Mapeo de estado a recibo:

Estado del núcleo	Recibo `stat`
`delivered`	`DELIVRD`
`expired`	`EXPIRED`
`dropped`	`UNDELIV`
`rejected`	`REJECTD`
`balance_rejected`	`REJECTD`

Consulte USAGE.md para el flujo de mensajes de extremo a extremo y MONITORING.md para las métricas relacionadas.

Ejemplo de interfaz web:

Configuración de Escucha del Servidor

Cuando se configuran los enlaces del servidor, la puerta de enlace escucha conexiones entrantes.

Ejemplo Completo de Escucha

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

Parámetros de Escucha

host

Dirección IP a la que enlazar

Tipo	Requerido	Predeterminado	Valores Comunes
Cadena	No	`"0.0.0.0"`	`"0.0.0.0"`, `"127.0.0.1"`

Propósito: Qué interfaz de red escuchar.

Valores:

- `"0.0.0.0"` - Escuchar en todas las interfaces (recomendado)
- `"127.0.0.1"` - Escuchar solo en localhost (pruebas)
- `"192.168.1.10"` - Escuchar en IP específica

port

Puerto TCP a escuchar

Tipo	Requerido	Predeterminado	Rango
Entero	No	2775	1-65535

Propósito: Puerto para conexiones SMPP entrantes.

Estándar: 2775

max_connections

Máximo de conexiones concurrentes

Tipo	Requerido	Predeterminado	Rango
Entero	No	100	1-10000

Propósito: Limita el número total de conexiones de cliente simultáneas.

Directrices:

- Establecer según los clientes esperados
- Valores más altos utilizan más memoria
- Típico: 10-100 conexiones

Ejemplos Completos de Configuración

Ejemplo 1: Conexión de un Solo Operador

```
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
  }
]
```

Ejemplo 2: Múltiples Operadores

```
import Config

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

config :omnimessage_smpp, :binds, [
  # América del Norte
  %{
    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
  },

  # Europa
  %{
    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
  }
]
```

Ejemplo 3: Puerta de Enlace con Enlaces de

Servidor

```
import Config

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

# Conexiones salientes
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
  }
]

# Definiciones de clientes entrantes
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
  }
]
```

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

Ejemplo 4: Cola Compartida con Vinculación de Origen

Dos conexiones al mismo operador, ambas drenando una cola compartida (`sinch.smpp`) para compartir carga y redundancia. El operador permite listas blancas por IP de origen **y** puerto, por lo que cada enlace fija un puerto de origen local distinto en la misma IP de origen.

```
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,

    # Ambos enlaces drenan la misma cola de backend
    queue: "sinch.smpp",

    # Dirección/puerto de origen fijados (el operador permite
listas blancas por ambos)
    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,

    # Misma cola compartida que sinch_1
    queue: "sinch.smpp",

    # Misma IP de origen, puerto de origen distinto
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5002
```

```
  }
]
```

Cómo funciona: Ambos enlaces registran un único frontend llamado `sinch.smpp` (su clave de enrutamiento compartida). Los mensajes con `dest_smsc: "sinch.smpp"` son drenados por el enlace que los recoja primero, distribuyendo la carga entre las dos conexiones. Si una conexión se cae, la otra continúa drenando la cola. Cada conexión presenta una IP y un puerto de origen fijos al operador para la lista blanca; `SO_REUSEADDR` permite que un puerto fijado se vuelva a enlazar inmediatamente al reconectar.

Consulte `queue`, `bind_source_ip` y `bind_source_port` para obtener detalles de los parámetros.

Validación de Configuración

Después de editar la configuración, valide antes de reiniciar:

Verificación de Sintaxis

```
# Verificar sintaxis de Elixir
/opt/omnimessage-smpp/bin/omnimessage-smpp eval "File.read!
('config/runtime.exs')"
```

Si la sintaxis es inválida, verá un error. Corrija antes de reiniciar.

Probar Configuración

```
# Reiniciar en primer plano para ver errores
sudo -u omnimessage-smpp /opt/omnimessage-smpp/bin/omnimessage-
smpp console
```

Presione `Ctrl+C` dos veces para salir.

Variables de Entorno

Todos los ajustes globales pueden ser sobrescritos con variables de entorno. Establezca estos en su archivo de unidad systemd o en el entorno de shell antes de iniciar la puerta de enlace.

Variable de Entorno	Clave de Configuración	Predeter
`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`
`CACHE_BACKOFF_MULTIPLIER`	`cache_backoff_multiplier`	`2`

Variable de Entorno	Clave de Configuración	Predeter
`MNESIA_STORAGE_TYPE`	`mnesia_storage_type`	`disc_copies`

Ejemplo de anulación de 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"
```

Mejores Prácticas de Seguridad

1. **Proteger el archivo de configuración**:

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

2. **Usar contraseñas fuertes**:

   - Mínimo 12 caracteres
   - Mezclar letras, números, símbolos

    - Único por conexión

3. **Usar listas blancas de IP**:
    - Siempre configurar `ip_whitelist` para enlaces de servidor
    - Nunca usar lista vacía `[]` en producción
4. **Habilitar verificación SSL**:
    - Establecer `verify_ssl_peer: true` con certificados válidos
5. **Rotación regular de credenciales**:
    - Cambiar contraseñas trimestralmente
    - Coordinar con operadores/socios

Próximos Pasos

- Revisar MONITORING.md para la configuración de métricas
- Leer USAGE.md para gestionar conexiones
- Ver TROUBLESHOOTING.md para problemas comunes
- Volver a README.md para una visión general

Glosario

Términos y Definiciones

A

API (Interfaz de Programación de Aplicaciones) Interfaz utilizada para comunicarse con el sistema backend de la cola de mensajes.

Auto-Scroll Función en la pestaña de Registros de la interfaz web que se desplaza automáticamente para mostrar las entradas de registro más recientes.

B

Backend El sistema de cola de mensajes al que se conecta la puerta de enlace SMPP para recuperar y almacenar mensajes.

Bind Una conexión SMPP entre dos sistemas. Puede ser transmisor, receptor o transceptor.

Bind Type El tipo de sesión SMPP:

- **Transmisor**: Solo envía mensajes
- **Receptor**: Solo recibe mensajes
- **Transceptor**: Envía y recibe mensajes

Bind Failure Cuando un intento de autenticación SMPP falla, generalmente debido a credenciales incorrectas o restricciones de IP.

C

CIDR (Enrutamiento Inter-Dominio Sin Clase) Notación para especificar rangos de direcciones IP (por ejemplo, `192.168.1.0/24` representa 256

direcciones IP).

Client Bind Una conexión SMPP saliente donde la puerta de enlace actúa como un **ESME** conectándose a un **SMSC** externo (típicamente un servidor SMPP de un operador). En este modo, la puerta de enlace es el cliente.

Connection Status Estado actual de un bind SMPP:

- **Conectado**: Activo y operativo
- **Desconectado**: No conectado
- **Reconectando**: Intentando establecer conexión

Counter Una métrica que solo aumenta (se restablece al reiniciar el servicio), utilizada para totales como mensajes enviados.

D

Data Coding Campo SMPP que especifica la codificación de caracteres del mensaje (GSM-7, UCS-2, etc.).

Deliver_SM PDU SMPP enviado por un SMSC (servidor) para entregar un mensaje a un ESME (cliente) conectado. Utilizado por los binds de servidor para enviar mensajes a socios conectados.

Delivery Failure Cuando un mensaje no puede ser entregado, indicado por una respuesta de error del operador.

Delivery Receipt (DLR) Confirmación del operador sobre el estado de entrega del mensaje.

deliver_time Campo en la cola de mensajes que contiene la marca de tiempo en la que se entregó un mensaje. Establecido por OmniMessage Core cuando la puerta de enlace informa una entrega exitosa. Este es el nombre del campo que el backend espera al marcar un mensaje como entregado.

dest_smsc Campo en la cola de mensajes que indica qué conexión SMPP debe manejar el mensaje. Su valor debe ser igual a la **Routing Key** bajo la cual el par objetivo se registró (ver Frontend Registration).

Disconnection Cuando una conexión SMPP activa se termina, ya sea intencionalmente o debido a un error.

E

Enquire Link Mensaje de mantenimiento de conexión SMPP enviado periódicamente para verificar que la conexión esté activa.

ESM Class Campo SMPP que indica el tipo y las características del mensaje.

ESME (Entidad de Mensaje Corto Externa) En la terminología SMPP, la aplicación cliente que se conecta a un SMSC para enviar o recibir mensajes. Cuando la puerta de enlace opera en **modo Cliente**, actúa como un ESME conectándose a SMSCs de operadores. Cuando opera en **modo Servidor**, acepta conexiones de ESMEs externos.

Exponential Backoff Estrategia de reintento donde el tiempo de espera se duplica después de cada fallo (1min, 2min, 4min, 8min...).

F

Firewall Sistema de seguridad de red que controla el tráfico de red entrante y saliente.

Frontend Registration Proceso mediante el cual la puerta de enlace SMPP se registra con OmniMessage Core. Se envía un latido cada 60 segundos para mantener viva la inscripción. Si la puerta de enlace se detiene, la inscripción expira después de 90 segundos y OmniMessage deja de enrutar mensajes hacia ella. Cada par se registra bajo su **Routing Key** de modo que el nombre registrado sea igual al `dest_smsc` que el backend enruta: pares de clientes bajo su `queue` (o nombre de par simple), binds de servidor bajo su nombre de bind.

G

Gateway La aplicación de la puerta de enlace SMPP que conecta la cola de mensajes con las redes móviles.

Gauge Una métrica que puede aumentar o disminuir, representando el valor actual (por ejemplo, estado de conexión).

Grafana Herramienta de visualización popular para mostrar métricas de Prometheus en paneles.

GSM-7 Codificación de caracteres estándar de 7 bits para SMS, que admite hasta 160 caracteres por mensaje.

H

HTTP/HTTPS Protocolos utilizados para la comunicación web. HTTPS es la versión encriptada.

I

IP Whitelist Lista de direcciones IP permitidas que pueden conectarse a la puerta de enlace (característica de seguridad).

ISDN (Red Digital de Servicios Integrados) Plan de numeración comúnmente utilizado para números de teléfono.

J

(Sin términos)

K

Keepalive Mensajes periódicos (enquire_link) enviados para mantener la conexión y detectar fallos.

KPI (Indicador Clave de Rendimiento) Valor medible que indica el rendimiento del sistema (por ejemplo, tasa de éxito de entrega).

L

Label En Prometheus, pares clave-valor adjuntos a métricas para identificación (por ejemplo, `bind_name="vodafone_uk"`).

LiveView Tecnología del marco Phoenix utilizada para actualizaciones en tiempo real de la interfaz web.

M

Message Queue Sistema backend que almacena mensajes esperando ser enviados o recibidos.

Metrics Mediciones cuantitativas del rendimiento del sistema, expuestas en formato Prometheus.

MO (Origen Móvil) Mensajes enviados desde teléfonos móviles a la puerta de enlace (entrantes).

MT (Terminado Móvil) Mensajes enviados desde la puerta de enlace a teléfonos móviles (salientes).

MSISDN (Número de Directorio Internacional de Suscriptor de Estación Móvil) Formato estándar para números de teléfono móvil.

N

NPI (Indicador de Plan de Numeración) Campo SMPP que especifica el esquema de numeración (por ejemplo, ISDN).

O

Outbound Mensajes que fluyen de la puerta de enlace a las redes móviles.

Inbound Mensajes que fluyen de las redes móviles a la puerta de enlace.

P

PDU (Unidad de Datos de Protocolo) Paquete de mensaje SMPP individual (por ejemplo, submit_sm, deliver_sm).

Prometheus Sistema de monitoreo de código abierto que recopila y almacena métricas de series temporales.

Q

Queue Lista de mensajes esperando ser procesados o enviados.

Queue Check Frequency Con qué frecuencia (en milisegundos) la puerta de enlace sondea el backend en busca de nuevos mensajes.

Queue Worker Componente que recupera mensajes de la cola y los envía a través de SMPP.

R

Rate Limiting Control del rendimiento de mensajes para cumplir con las restricciones del operador. Ver TPS.

Receiver Tipo de bind SMPP que solo recibe mensajes (deliver_sm).

Reconnect Restablecer una conexión SMPP desconectada.

Retry Intentar enviar un mensaje fallido nuevamente, generalmente con retroceso exponencial.

Routing Key El nombre que un par registra como frontend con OmniMessage Core, y por lo tanto el valor `dest_smsc` que enruta mensajes hacia él. Para un par cliente, este es su `queue` configurado (o el nombre de par simple cuando no se establece `queue`); para un bind de servidor, es el nombre del bind.

S

Sequence Number Identificador numérico único asignado a cada PDU SMPP dentro de una sesión. Se utiliza para emparejar solicitudes con sus respuestas (por ejemplo, emparejando un submit_sm con su submit_sm_resp).

Server Bind Configuración que permite a **ESMEs** externos (clientes) conectarse a la puerta de enlace. En este modo, la puerta de enlace actúa como un **SMSC** (servidor) aceptando conexiones entrantes de sistemas socios.

Session Conexión SMPP activa entre dos sistemas.

Shared Queue Una única cola backend drenada por más de un bind de cliente. Los binds optan por ello configurando el mismo valor de `queue` en su configuración (por ejemplo, dos conexiones de operador que drenan `"sinch.smpp"`), permitiendo compartir carga y redundancia a través de conexiones. Por defecto, se establece en el propio nombre del bind, en cuyo caso cada bind tiene su propia cola.

Short Code Una dirección de SMS de origen/destino acortada (típicamente de 5 a 6 dígitos) utilizada para mensajería de aplicación a persona. Los destinos `deliver_sm` de cinco dígitos se envían con TON **Desconocido** para que los operadores no los malinterpreten como números internacionales.

Source Binding Fijar la dirección IP de origen local y/o el puerto TCP de una conexión saliente (cliente) a través de los parámetros `bind_source_ip` y `bind_source_port`. Se utiliza cuando un operador blinda conexiones por IP y puerto de origen. Ver CONFIGURATION.md.

source_smsc Campo en la cola de mensajes que indica qué bind de servidor debe entregar el mensaje a sus clientes conectados a través de deliver_sm.

SMPP (Protocolo de Mensajes Cortos de Par a Par) Protocolo estándar de la industria para el intercambio de mensajes SMS entre sistemas.

SMSC (Centro de Servicio de Mensajes Cortos) En la terminología SMPP, el componente del servidor que acepta conexiones de ESMEs (clientes) y maneja el enrutamiento y la entrega de mensajes SMS. Cuando la puerta de enlace

opera en **modo Servidor**, actúa como un SMSC aceptando conexiones de ESMEs externos.

SSL/TLS Protocolos de encriptación para comunicación segura.

Submit_SM PDU SMPP para enviar un mensaje para entrega.

Submit_SM_Resp Respuesta SMPP a submit_sm, indicando éxito o fallo.

System ID Nombre de usuario utilizado para la autenticación SMPP.

T

Telemetry Recolección y transmisión automatizada de métricas del sistema.

TON (Tipo de Número) Campo SMPP que especifica el formato del número (por ejemplo, internacional, nacional).

TPS (Transacciones Por Segundo) Límite de tasa para el máximo de mensajes por segundo a través de una conexión.

Transceiver Tipo de bind SMPP que puede enviar y recibir mensajes (el más común).

Transmitter Tipo de bind SMPP que solo envía mensajes (submit_sm).

Throughput Tasa de procesamiento de mensajes, típicamente medida en mensajes por segundo.

U

UCS-2 Codificación de caracteres Unicode de 16 bits para SMS, que admite hasta 70 caracteres por mensaje.

Uptime Duración durante la cual una conexión o servicio ha estado operativo continuamente.

V

Validity Period Límite de tiempo para el intento de entrega del mensaje antes de la expiración.

W

Web Dashboard Interfaz de usuario basada en navegador para monitorear y gestionar la puerta de enlace.

Whitelist Ver IP Whitelist y Source Address Whitelist.

X

(Sin términos)

Y

(Sin términos)

Z

(Sin términos)

Referencia Rápida de Acrónimos

Acrónimo	Término Completo
API	Interfaz de Programación de Aplicaciones
CIDR	Enrutamiento Inter-Dominio Sin Clase
DLR	Recibo de Entrega
ESME	Entidad de Mensaje Corto Externa
GSM	Sistema Global para Comunicaciones Móviles
HTTP	Protocolo de Transferencia de Hipertexto
HTTPS	Protocolo de Transferencia de Hipertexto Seguro
IP	Protocolo de Internet
ISDN	Red Digital de Servicios Integrados
KPI	Indicador Clave de Rendimiento
MO	Origen Móvil
MSISDN	Número de Directorio Internacional de Suscriptor de Estación Móvil
MT	Terminado M��vil
NPI	Indicador de Plan de Numeración
PDU	Unidad de Datos de Protocolo
SMPP	Protocolo de Mensajes Cortos de Par a Par

Acrónimo	Término Completo
SMSC	Centro de Servicio de Mensajes Cortos
SMS	Servicio de Mensajes Cortos
SSL	Capa de Sockets Segura
TLS	Seguridad de la Capa de Transporte
TON	Tipo de Número
TPS	Transacciones Por Segundo
UCS	Conjunto de Caracteres Codificados Universalmente
UI	Interfaz de Usuario
URL	Localizador Uniforme de Recursos

Documentación Relacionada

- **README.md** - Visión general del sistema y cómo empezar
- **CONFIGURATION.md** - Parámetros de configuración explicados
- **USAGE.md** - Operaciones diarias
- **MONITORING.md** - Métricas y monitoreo
- **TROUBLESHOOTING.md** - Resolución de problemas

Caché de Mensajes SMPP

Visión General

La Caché de Mensajes SMPP es una capa de persistencia local que permite al gateway SMPP seguir aceptando mensajes entrantes incluso cuando la API de backend no está disponible. Los mensajes se almacenan localmente en Mnesia y se entregan automáticamente a la API cuando se restablece la conectividad, utilizando una lógica de reintento inteligente con retroceso exponencial.

Características

- **Aceptación Resiliente de Mensajes** - Continuar aceptando mensajes SMPP durante interrupciones de la API
- **Almacenamiento Persistente** - Utiliza Mnesia con `:disc_copies` para durabilidad a través de reinicios
- **Reintento Automático** - Trabajadores en segundo plano intentan automáticamente la entrega con retroceso exponencial
- **Configuración por Bind** - Habilitar/deshabilitar la caché de forma independiente para cada bind SMPP
- **Protección contra Desbordamiento** - Evicción FIFO cuando la caché alcanza el límite de tamaño configurado
- **Retención de Mensajes Fallidos** - Mensajes que fallan permanentemente se mantienen para revisión manual
- **Monitoreo en Tiempo Real** - Panel de LiveView con estadísticas y métricas de la caché
- **Métricas de Prometheus** - Exportación completa de métricas para monitoreo y alertas

Arquitectura

Flujo de Mensajes

Con Caché Habilitada

```
SMPP Client → submit_sm → Gateway Server
                                ↓
                        Cache in Mnesia (respuesta inmediata)
                                ↓
                        CacheFlushWorker (en segundo plano)
                                ↓
                          Backend API
```

Con Caché Deshabilitada

```
SMPP Client → submit_sm → Gateway Server
                                ↓
                          Backend API (directo)
```

Componentes

1. **Módulo MessageCache**
    - Lógica central de caché
    - Manejo de desbordamiento
    - Funciones de consulta para LiveView y trabajadores
2. **CacheFlushWorker**
    - GenServer por bind con caché habilitada
    - Consulta mensajes listos para reintentar
    - Implementa retroceso exponencial
    - Marca mensajes que fallan permanentemente
3. **Tabla Mnesia** (`:smpp_message_cache`)

- Almacenamiento persistente con `:disc_copies`
- Indexada por bind_name, next_retry_at y estado
- Sobrevive a reinicios de la aplicación

Configuración

Configuraciones Globales

Edita `config/runtime.exs`:

```
config :omnimessage_smpp,
  # Cada cuánto los trabajadores de vaciado consultan mensajes
(milisegundos)
  cache_flush_interval: 10_000,

  # Máximo de intentos de reintento antes de marcar como
failed_permanent
  cache_max_retry_attempts: 10,

  # Multiplicador de retroceso exponencial
  cache_backoff_multiplier: 2,

  # Tipo de almacenamiento Mnesia (:disc_copies o :ram_copies)
  mnesia_storage_type: :disc_copies
```

Configuración por Bind

Cada bind SMPP (cliente o servidor) puede configurarse de forma independiente:

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

    # Configuración de caché
    cache_enabled: true,         # Habilitar caché
(predeterminado: false)
    cache_max_size: 10_000,      # Máx. mensajes a almacenar en
caché (predeterminado: 10,000)
    cache_retry_interval: 60     # Intervalo base de reintento en
segundos (predeterminado: 60)
  }
]
```

Variables de Entorno

```
# Configuraciones globales de caché
CACHE_FLUSH_INTERVAL=10000              # Intervalo de consulta de
vaciado (ms)
CACHE_MAX_RETRY_ATTEMPTS=10             # Máx. reintentos antes de
fallo permanente
CACHE_BACKOFF_MULTIPLIER=2              # Multiplicador de
retroceso exponencial
MNESIA_STORAGE_TYPE=disc_copies         # Tipo de almacenamiento
Mnesia
```

Comportamiento de Reintento

Retroceso Exponencial

Cuando la entrega de un mensaje falla, el intervalo de reintento se duplica con cada intento:

```
Intervalo base: 60 segundos
Multiplicador de retroceso: 2

Reintento 0: 60s   (1 minuto)
Reintento 1: 120s  (2 minutos)
Reintento 2: 240s  (4 minutos)
Reintento 3: 480s  (8 minutos)
Reintento 4: 960s  (16 minutos)
Reintento 5: 1920s (32 minutos)
...
Reintento 9: 30,720s (8.5 horas)
```

Fallo Permanente

Después de 10 intentos fallidos (por defecto), los mensajes se marcan como `failed_permanent` y:

- Permanecen en la caché para revisión manual
- Dejan de ser reintentados automáticamente
- Aparecen en la sección "Fallo Permanente" del panel de caché
- Pueden ser reintentados o eliminados manualmente

Transiciones de Estado

```
:pending → :delivering → SUCCESS (eliminado de la caché)
                      → FAILURE → :pending (reintento con
retroceso)
                                → :failed_permanent (después de los
máximos reintentos)
```

Monitoreo

Panel de LiveView

Accede al panel de caché en `http://your-server:4000/smpp` → pestaña "Caché de Mensajes"

Tarjetas de Resumen:

- Total en Caché - Todos los mensajes actualmente en caché
- Entrega Pendiente - Mensajes esperando reintento
- Fallo Permanente - Mensajes que superaron los máximos reintentos

Tabla por Bind:

- Nombre del bind
- Conteo de mensajes en caché
- Desglose Pendiente / Fallido
- Tamaño máximo de caché
- Porcentaje de utilización (con barra de progreso visual)

Métricas de Prometheus

```
# Tamaño actual de la caché por bind
smpp_cache_size{bind_name="my_bind",mode="server"} 42

# Total de mensajes entregados con éxito
smpp_cache_delivered_total{bind_name="my_bind"} 1234

# Total de intentos de reintento
smpp_cache_retry_total{bind_name="my_bind"} 56

# Total de fallos permanentes
smpp_cache_permanent_failures_total{bind_name="my_bind"} 2

# Total de eventos de desbordamiento (mensajes descartados)
smpp_cache_overflow_total{bind_name="my_bind"} 0
```

Mensajes de Registro

```
INFO  Mensaje -123456789 almacenado en caché para my_smpp_bind
INFO  Mensaje en caché -123456789 entregado con éxito, ID de API:
99999
WARN  Falló al entregar el mensaje -123456789 (reintento 3/10),
próximo reintento a 2026-02-01 12:34:56Z
ERROR Mensaje -123456789 superó los máximos reintentos (10),
marcando como failed_permanent
WARN  Desbordamiento de caché para my_smpp_bind: mensaje más
antiguo eliminado
```

Operaciones

Habilitar/Deshabilitar la Caché

A través de la Interfaz de Usuario de LiveView

1. Navega a `http://your-server:4000/smpp`
2. Ve a la pestaña "Client Peers" o "Server Peers"
3. Edita el bind
4. Activa/desactiva la casilla "Cache Enabled"
5. Guarda los cambios

A través de la Consola IEx

```
# Habilitar caché para un bind
SmsC.SMPPConfig.update_server_peer("my_bind", "username",
"password",
  cache_enabled: true,
  cache_max_size: 10_000,
  cache_retry_interval: 60
)

# Deshabilitar caché
SmsC.SMPPConfig.update_server_peer("my_bind", "username",
"password",
  cache_enabled: false
)
```

Monitorear el Estado de la Caché

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

# Obtener desglose por 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}
# ]

# Contar mensajes para un bind específico
SmsC.SMPP.MessageCache.count_cached_messages("my_bind")
# => 42
```

Intervenciones Manuales

Limpiar Mensajes Fallidos

```
# Obtener todos los mensajes fallidos para un bind
{:atomic, failed_messages} = :mnesia.transaction(fn ->
  :mnesia.match_object({:smpp_message_cache, :_, "my_bind", :_,
:_, :_, :_, :_, :_, :failed_permanent})
end)

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

Forzar Reintento de un Mensaje Fallido

```
# Restablecer un mensaje failed_permanent a pending
SmsC.SMPP.MessageCache.update_cache_record("my_bind", -123456, %{
  status: :pending,
  retry_count: 0,
  next_retry_at: DateTime.utc_now(),
  last_error: nil
})
```

Solución de Problemas

Caché Llena / Eventos de Desbordamiento

Síntoma: métrica `cache_overflow_total` en aumento, mensajes más antiguos siendo descartados

Causa: Límite de tamaño de caché alcanzado

Soluciones:

1. Aumentar `cache_max_size` para el bind
2. Investigar por qué la entrega de la API está fallando (verificar registros de la API, red)
3. Limpiar manualmente los mensajes fallidos antiguos

4. Verificar si el intervalo de vaciado es demasiado lento

Mensajes No Entregados

Síntoma: Mensajes atascados en estado `:pending`

Posibles Causas:

1. **API fuera de servicio**
   - Verificar disponibilidad de la API
   - Verificar registros de la API de backend
   - Comprobar conectividad de red
2. **next_retry_at está en el futuro**
   - Los mensajes se reintentará cuando se alcance `next_retry_at`
   - Verificar el cronograma de retroceso exponencial
3. **Trabajador de vaciado no está en ejecución**

```
# Verificar si los trabajadores están en ejecución
Supervisor.which_children(SmsC.SMPP.Supervisor)
```

4. **Caché deshabilitada**
   - Verificar `cache_enabled: true` en la configuración del bind

Altas Cuentas de Reintento

Síntoma: Muchos mensajes con altos valores de `retry_count`

Investigación:

```
# Encontrar mensajes con altas cuentas de reintento
{: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)

# Verificar last_error para cada uno
Enum.each(high_retry, fn {_, _, _, msg_id, _, _, retry_count, _,
last_error, _} ->
  IO.puts("Mensaje #{msg_id}: #{retry_count} reintentos, error: #
{inspect(last_error)}")
end)
```

Espacio en Disco de Mnesia

Síntoma: Espacio en disco llenándose

Verificar directorio de Mnesia:

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

Limpiar:

1. Limpiar mensajes fallidos antiguos (ver Intervenciones Manuales arriba)
2. Reducir `cache_max_size` por bind
3. Habilitar desbordamiento de caché (asegurar correcta evicción FIFO)

Consideraciones de Rendimiento

Uso de Memoria

- Cada mensaje almacenado en caché utiliza aproximadamente 500-1000 bytes (dependiendo del tamaño del mensaje)
- 10,000 mensajes ≈ 5-10 MB de memoria
- Con `:disc_copies`, los datos también se escriben en disco

Uso de CPU

- Los trabajadores de vaciado consultan cada 10 segundos por defecto (configurable)
- El procesamiento por lotes (100 mensajes por ciclo) reduce la sobrecarga
- Entrega concurrente (máx. 10 llamadas API concurrentes por trabajador)

I/O de Disco

- `:disc_copies` escribe en disco en cada transacción
- Para un rendimiento muy alto (>1000 msg/sec), considerar:
  - Usar `:ram_copies` (pierde persistencia)
  - Aumentar intervalos de vaciado
  - Escalar horizontalmente

Límites Recomendados

Escenario	cache_max_size	cache_flush_interval
Bajo volumen (<100 msg/sec)	10,000	10,000ms
Volumen medio (100-500 msg/sec)	50,000	5,000ms
Alto volumen (>500 msg/sec)	100,000	3,000ms

Escenarios de Recuperación

Reinicio de la Aplicación

1. Mnesia carga automáticamente las tablas `:disc_copies` desde el disco
2. Los mensajes almacenados en caché permanecen intactos
3. Los trabajadores de vaciado se reinician y continúan procesando

Migración de Base de Datos

Al actualizar desde una versión sin soporte de caché:

1. La migración agrega automáticamente campos de caché a los binds existentes
2. Valores predeterminados: `cache_enabled: false`, `cache_max_size: 10_000`, `cache_retry_interval: 60`
3. Sin pérdida de datos
4. Tabla de caché creada en la primera ejecución

Recuperación de Interrupción de API

1. Los mensajes se acumulan en la caché durante la interrupción
2. Cuando la API se recupera, los trabajadores de vaciado entregan automáticamente
3. Los mensajes más antiguos se entregan primero (FIFO)
4. El retroceso exponencial evita la sobrecarga de la API durante la recuperación

Mejores Prácticas

1. **Habilitar Caché por Defecto** - Previene la pérdida de mensajes durante interrupciones
2. **Monitorear Métricas** - Configurar alertas en `cache_permanent_failures_total` y `cache_overflow_total`

3. **Dimensionar Apropiadamente** - Establecer `cache_max_size` basado en la duración esperada de la interrupción
4. **Revisar Mensajes Fallidos** - Verificar regularmente los mensajes failed_permanent en busca de patrones
5. **Probar Failover** - Simular interrupciones de API para verificar el comportamiento de la caché
6. **Ajustar Intervalos de Reintento** - Afinar según los patrones de tiempo de recuperación de la API
7. **Usar Almacenamiento Persistente** - Mantener `mnesia_storage_type: disc_copies` en producción

Véase También

- Referencia de Configuración
- Monitoreo y Métricas
- Solución de Problemas

Guía de Monitoreo y Métricas

Referencia completa para monitorear el Gateway SMPP

Descripción General

El Gateway SMPP expone métricas en formato Prometheus para monitorear la salud de la conexión, el rendimiento de los mensajes y el rendimiento del sistema.

Crítico: Dado que el gateway es sin estado y depende de OmniMessage Core, **la conectividad de OmniMessage es la métrica más importante a monitorear**. Monitorea ambos:

1. **Métricas del Gateway SMPP** - Salud a nivel de protocolo
2. **Métricas de la API de OmniMessage** - Conectividad y salud del backend

Endpoint de Métricas

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

Formato: Formato de texto de Prometheus

Acceso: Abierto a localhost por defecto (configura el firewall para acceso remoto)

Prueba Rápida

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

Métricas Disponibles

Todas las métricas están prefijadas con `smpp_` e incluyen etiquetas para identificación.

Métricas de Licencia

omnimessage_smpp_license_status

Tipo: Gauge
Descripción: Estado actual de la licencia
Valores:

- `1` = Licencia válida
- `0` = Licencia inválida/expirada

Etiquetas: Ninguna

Ejemplo:

```
omnimessage_smpp_license_status 1
```

Uso:

- Alertar cuando el valor sea 0 (licencia inválida)
- Cuando la licencia es inválida, el procesamiento de la cola de salida se detiene, pero los enlaces SMPP permanecen conectados
- La interfaz web sigue siendo accesible para solución de problemas

Nombre del Producto: `omnimessage_smpp`

Notas:

- Cuando la licencia es inválida (`license_status == 0`), el gateway deja de procesar las colas de salida
- Los enlaces SMPP (tanto cliente como servidor) permanecen conectados y aceptan solicitudes de enlace

- Los mensajes entrantes aún se reciben pero no se procesan
- La interfaz de usuario y el monitoreo siguen siendo accesibles independientemente del estado de la licencia

Ejemplo de Alerta:

```
- alert: SMPP_License_Invalid
  expr: omnimessage_smpp_license_status == 0
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "Licencia del Gateway SMPP inválida o expirada"
    description: "El estado de la licencia es inválido - el
procesamiento de mensajes salientes está bloqueado"
```

Métricas de Estado de Conexión

smpp_connection_status

Tipo: Gauge
Descripción: Estado actual de la conexión del enlace SMPP
Valores:

- `1` = Conectado. Para un enlace de servidor, al menos una sesión ESME está vinculada.
- `0` = Desconectado. Para un enlace de servidor, ninguna sesión ESME está vinculada.

Etiquetas:

- `bind_name` - Nombre de la conexión (por ejemplo, "vodafone_uk")
- `mode` - Tipo de conexión ("client" o "server")
- `host` - Host remoto (solo en modo cliente)
- `port` - Puerto remoto (solo en modo cliente)
- `bind_type` - Tipo de enlace SMPP (solo en modo cliente)
- `system_id` - ID del sistema utilizado

Ejemplo:

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

Uso:

- Alertar cuando el valor sea 0 (desconectado)
- Rastrear el porcentaje de tiempo de actividad de la conexión
- Monitorear la frecuencia de reconexiones

Contadores de Mensajes

smpp_messages_sent_total

Tipo: Counter
Descripción: Número total de mensajes enviados a través del enlace SMPP: `deliver_sm` al ESME en un enlace de servidor, `submit_sm` al operador en un enlace de cliente
Unidad: PDUs SMPP. Cada parte de un mensaje multipartido cuenta una vez.

Etiquetas: Igual que connection_status

Ejemplo:

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

Uso:

- Calcular la tasa de mensajes (mensajes/segundo)
- Rastrear el volumen diario/mensual
- Comparar el rendimiento real con el esperado

smpp_messages_received_total

Tipo: Counter
Descripción: Número total de mensajes recibidos a través del enlace SMPP: `submit_sm` del ESME en un enlace de servidor, `deliver_sm` de origen móvil del operador en un enlace de cliente. Los recibos de entrega no se cuentan.
Unidad: PDUs SMPP

Etiquetas: Igual que connection_status

Ejemplo:

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

Uso:

- Monitorear el volumen de mensajes entrantes
- Rastrear el tráfico de origen móvil (MO)
- Alertar sobre cambios inesperados en el volumen

Métricas de Entrega

smpp_delivery_failures_total

Tipo: Counter
Descripción: Número total de fallos en la entrega de mensajes: un `deliver_sm` o `submit_sm` que no pudo ser enviado, o que fue respondido con un estado distinto de cero
Unidad: PDUs SMPP

Etiquetas: Igual que connection_status

Ejemplo:

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

Uso:

- Calcular la tasa de éxito de entrega
- Alertar sobre altas tasas de fallos
- Identificar conexiones problemáticas

Cálculo de la Tasa de Éxito:

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

Métricas de Operación de Enlace

smpp_bind_success_total

Tipo: Counter
Descripción: Número total de operaciones de enlace exitosas
Unidad: Intentos de enlace

Ejemplo:

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

Uso:

- Rastrear la estabilidad del enlace
- Monitorear el éxito de la autenticación

smpp_bind_failures_total

Tipo: Counter
Descripción: Número total de operaciones de enlace fallidas
Unidad: Intentos de enlace

Ejemplo:

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

Uso:

- Alertar sobre fallos de autenticación
- Identificar problemas de credenciales
- Rastrear problemas de conexión con el operador

Métricas de Eventos de Conexión

smpp_connection_attempts_total

Tipo: Counter
Descripción: Número total de intentos de conexión. En un enlace de servidor, el número de sesiones ESME que se han vinculado.
Unidad: Intentos

Ejemplo:

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

Uso:

- Rastrear la rotación de conexiones
- Monitorear la frecuencia de reconexiones

smpp_disconnection_total

Tipo: Counter
Descripción: Número total de desconexiones. En un enlace de servidor, el número de sesiones ESME vinculadas que han terminado.
Unidad: Desconexiones

Ejemplo:

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

Uso:

- Alertar sobre desconexiones frecuentes
- Identificar problemas de red
- Rastrear la estabilidad de la conexión

smpp_reconnect_attempts_total

Tipo: Counter
Descripción: Número total de reconexiones programadas para un enlace de cliente después de un fallo de conexión, un enlace rechazado o una sesión caída
Unidad: Intentos de reconexión

Ejemplo:

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

Uso:

- Detectar un operador que acepta la conexión y luego la corta de inmediato (contador en rápido aumento mientras `smpp_connection_status` permanece en `0`)
- Medir cuánto tiempo un enlace pasó intentando volver después de una interrupción

---

Métricas de Enlace de Consulta

smpp_enquire_link_sent_total

Tipo: Counter
Descripción: Número total de PDUs enquire_link enviados para verificar la vida de la conexión
Unidad: PDUs

Etiquetas: Igual que connection_status

Ejemplo:

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

Uso:

- Rastrear la actividad de keepalive
- Comparar con los recibidos para detectar fallos unidireccionales

smpp_enquire_link_received_total

Tipo: Counter
Descripción: Número total de PDUs enquire_link_resp recibidos del par remoto
Unidad: PDUs

Etiquetas: Igual que connection_status

Ejemplo:

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

Uso:

- Detectar pares no responsivos (enviados >> recibidos)
- Monitorear la salud de la conexión más allá del estado simple

---

Métricas de Tiempo de Actividad

smpp_uptime_seconds

Tipo: Gauge
Descripción: Tiempo de actividad actual del enlace SMPP en segundos
Unidad: Segundos

Ejemplo:

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

Uso:

- Rastrear la estabilidad de la conexión
- Calcular el porcentaje de tiempo de actividad
- Alertar sobre reinicios recientes

Métricas de Caché de Mensajes

Estas métricas están disponibles cuando la caché de mensajes está habilitada en uno o más enlaces. Consulta MESSAGE_CACHE.md para detalles de configuración de caché.

smpp_cache_size

Tipo: Gauge
Descripción: Número actual de mensajes en la caché local por enlace
Unidad: Mensajes

Etiquetas:

- `bind_name` - Nombre de la conexión
- `mode` - Tipo de conexión ("client" o "server")

Ejemplo:

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

Uso:

- Monitorear la utilización de la caché
- Alertar cuando se acerque a `cache_max_size`

smpp_cache_delivered_total

Tipo: Counter
Descripción: Número total de mensajes en caché entregados exitosamente a la API del backend
Unidad: Mensajes

Ejemplo:

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

smpp_cache_retry_total

Tipo: Counter
Descripción: Número total de intentos de reintento para mensajes en caché
Unidad: Intentos

Ejemplo:

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

smpp_cache_permanent_failures_total

Tipo: Counter
Descripción: Número total de mensajes que excedieron los intentos máximos de reintento y fueron marcados como fallidos permanentemente
Unidad: Mensajes

Ejemplo:

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

Uso:

- Alertar cuando > 0 (requiere revisión manual)

smpp_cache_overflow_total

Tipo: Counter
Descripción: Número total de eventos de desbordamiento de caché donde el mensaje más antiguo fue expulsado para hacer espacio
Unidad: Eventos

Ejemplo:

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

Uso:

- Alertar cuando aumente (caché demasiado pequeña o interrupción de API demasiado larga)

Métricas de Recibo de Entrega (DLR)

Estas métricas están disponibles para enlaces de servidor que configuran `deliver_receipts: true`. Consulta CONFIGURATION.md para la configuración y USAGE.md para el flujo de mensajes.

smpp_dlr_tracked_total

Tipo: Counter
Descripción: Número total de mensajes enviados registrados para el seguimiento de recibos de entrega
Unidad: Mensajes

Etiquetas:

- `bind_name` - Nombre de la conexión
- `mode` - Tipo de conexión ("client" o "server")

Ejemplo:

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

smpp_dlr_sent_total

Tipo: Counter
Descripción: Número total de recibos de entrega enviados de vuelta al enlace que envió
Unidad: Recibos

Ejemplo:

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

Uso:

- Comparar con `smpp_dlr_tracked_total` para ver la tasa de finalización de recibos

smpp_dlr_dropped_total

Tipo: Counter
Descripción: Número total de mensajes rastreados descartados sin un recibo. El gateway descarta un mensaje cuando el núcleo ya no lo tiene, o cuando el mensaje no alcanza un estado terminal dentro de las 24 horas.
Unidad: Mensajes

Ejemplo:

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

Uso:

- Alertar cuando aumente (los mensajes nunca alcanzan un estado terminal, o el núcleo los pierde)

smpp_dlr_pending

Tipo: Gauge
Descripción: Número actual de mensajes en vuelo que esperan un estado terminal y un recibo
Unidad: Mensajes

Ejemplo:

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

Uso:

- Monitorear la acumulación del rastreador
- Alertar cuando el valor se mantenga alto (el núcleo es lento para alcanzar estados terminales, o la sesión del par está caída)

Ejemplos de consultas:

```
# Tasa de finalización de recibos durante 5 minutos
sum(rate(smpp_dlr_sent_total[5m]))
  / sum(rate(smpp_dlr_tracked_total[5m]))

# Recibos descartados por segundo
rate(smpp_dlr_dropped_total[5m])

# Acumulación actual del rastreador por enlace
smpp_dlr_pending
```

Métricas de Salud de la API de OmniMessage

Mientras que el gateway en sí expone métricas relacionadas con SMPP, **la salud de la API de OmniMessage es crítica**. También debes monitorear:

Desde Métricas de OmniMessage (si están disponibles)

- `omnimessage_api_requests_total` - Total de solicitudes de API desde el gateway
- `omnimessage_api_request_duration_seconds` - Tiempos de respuesta de la API
- `omnimessage_queue_depth` - Mensajes pendientes en la cola de OmniMessage

Desde Registros del Gateway (si no se exponen métricas)

Busca estos patrones para detectar problemas de API:

- "api.*connection refused" - No se puede alcanzar OmniMessage
- "api.*timeout" - OmniMessage no responde
- "api.*http 503" - OmniMessage temporalmente fuera de servicio
- "api.*parse error" - Problema de formato de respuesta

Configuración de Prometheus

Configuración Básica de Scrape

Agrega a `/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'
```

Múltiples Gateways

```
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'
```

Descubrimiento de Servicios

Usando descubrimiento basado en archivos:

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

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

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

Dashboards de Grafana

Paneles de Dashboard de Ejemplo

Panel de Estado de Conexión

Consulta:

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

Visualización: Stat
Umbrales:

- Rojo: valor < 1 (desconectado)
- Verde: valor == 1 (conectado)

Panel de Tasa de Mensajes

Consulta:

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

Visualización: Gráfico
Unidad: mensajes/segundo
Leyenda: `{{bind_name}}`

Panel de Tasa de Éxito de Entrega

Consulta:

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

Visualización: Gauge
Unidad: Porcentaje (0-100)
Umbrales:

- Rojo: < 95%
- Amarillo: 95-98%
- Verde: > 98%

Panel de Tiempo de Actividad de Conexión

Consulta:

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

Visualización: Stat
Unidad: Horas

Reglas de Alerta

Reglas de Alerta de Prometheus

Guarda en `/etc/prometheus/rules/smpp-alerts.yml`:

```yaml
groups:
  - name: smpp_gateway
    interval: 30s
    rules:
      # Conexión caída
      - alert: SMPPConnectionDown
        expr: smpp_connection_status == 0
        for: 2m
        labels:
          severity: critical
        annotations:
          summary: "La conexión SMPP {{ $labels.bind_name }} está
caída"
          description: "La conexión {{ $labels.bind_name }} ha
estado desconectada por más de 2 minutos."

      # Alta tasa de fallos
      - alert: SMPPHighFailureRate
        expr: |
          (
            rate(smpp_delivery_failures_total[5m])
            /
            rate(smpp_messages_sent_total[5m])
          ) > 0.05
        for: 5m
        labels:
          severity: warning
        annotations:
          summary: "Alta tasa de fallos de entrega en {{
$labels.bind_name }}"
          description: "La tasa de fallos de entrega es {{ $value
| humanizePercentage }} en {{ $labels.bind_name }}."

      # Fallos de enlace
      - alert: SMPPBindFailures
        expr: increase(smpp_bind_failures_total[10m]) > 3
        labels:
          severity: warning
        annotations:
          summary: "Múltiples fallos de enlace en {{
$labels.bind_name }}"
          description: "{{ $labels.bind_name }} ha fallado en
enlazarse {{ $value }} veces en los últimos 10 minutos."
```

```
      # No se enviaron mensajes (cuando se esperaba)
      - alert: SMPPNoTraffic
        expr: rate(smpp_messages_sent_total[10m]) == 0
        for: 30m
        labels:
          severity: warning
        annotations:
          summary: "No se enviaron mensajes en {{
$labels.bind_name }}"
          description: "{{ $labels.bind_name }} no ha enviado
ningún mensaje durante 30 minutos."

      # Desconexiones frecuentes
      - alert: SMPPFrequentDisconnections
        expr: increase(smpp_disconnection_total[1h]) > 5
        labels:
          severity: warning
        annotations:
          summary: "Desconexiones frecuentes en {{
$labels.bind_name }}"
          description: "{{ $labels.bind_name }} se ha desconectado
{{ $value }} veces en la última hora."

      # API de OmniMessage inalcanzable
      - alert: OmniMessageAPIUnreachable
        expr: |
          count(count_over_time({job="omnimessage-smpp"} |=
"api.*connection refused"[5m])) > 0
        for: 1m
        labels:
          severity: critical
        annotations:
          summary: "API de OmniMessage es inalcanzable"
          description: "El Gateway SMPP no puede alcanzar la API
de OmniMessage. Verifica la configuración de API_BASE_URL y la
conectividad de red."

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

```
        labels:
          severity: warning
        annotations:
          summary: "La API de OmniMessage está teniendo timeouts"
          description: "Se detectaron múltiples timeouts de API.
OmniMessage puede estar lento o sobrecargado."

      # Sin flujo de mensajes (problema de 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: "No se detectó flujo de mensajes - verifica la
conectividad de OmniMessage"
          description: "No se enviaron ni recibieron mensajes
durante 30 minutos. Verifica la conectividad de la API de
OmniMessage y el estado de la cola."
```

Carga las reglas en `prometheus.yml`:

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

Monitoreo del Dashboard Web

La interfaz web incorporada proporciona monitoreo en tiempo real sin Prometheus.

Acceso

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

Página de Estado en Vivo

Navegación: SMPP → Estado en Vivo

Características:

- Estado de conexión en tiempo real
- Contadores de mensajes
- Tiempo de actividad de la conexión
- Controles manuales de reconexión/desconexión
- Actualización automática cada 5 segundos

Uso:

- Verificación rápida del estado
- Intervención manual
- Solución de problemas en tiempo real

El dashboard muestra:

- **Total de Enlaces**: Conteo combinado de todas las conexiones de cliente y servidor
- **Enlaces de Cliente**: Conexiones salientes a operadores (mostrando conteo de conectados/desconectados)

- **Enlaces de Servidor**: Conexiones entrantes de socios (mostrando conteo de activos/en espera)
- **Servidor Escuchando**: Configuración del socket de servidor entrante (host, puerto, conexiones máximas)

Monitoreo de Registros

Registros del Sistema

Ver registros:

```
# Seguir registros en tiempo real
sudo journalctl -u omnimessage-smpp -f

# Últimas 100 líneas
sudo journalctl -u omnimessage-smpp -n 100

# Desde un tiempo específico
sudo journalctl -u omnimessage-smpp --since "1 hour ago"

# Filtrar por nivel
sudo journalctl -u omnimessage-smpp -p err
```

Registros de la Interfaz Web

Navegación: Pestaña de registros en la interfaz web

Características:

- Transmisión de registros en tiempo real
- Filtrar por nivel (debug, info, warning, error)
- Buscar registros
- Pausar/reanudar
- Borrar registros

La vista de registros te permite:

- **Filtrar por Nivel**: Seleccionar nivel de registro (Todo, Debug, Info, Advertencia, Error)
- **Buscar**: Encontrar entradas de registro específicas por contenido de texto
- **Desplazamiento Automático**: Habilitar/deshabilitar el desplazamiento automático a medida que llegan nuevos registros
- **Pausar/Reanudar**: Pausar actualizaciones de registros para revisar entradas específicas
- **Borrar**: Borrar todos los registros mostrados

Indicadores Clave de Rendimiento (KPI)

Salud de la Conexión

Métrica: Porcentaje de tiempo de actividad de la conexión

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

Objetivo: > 99.9%

Tasa de Entrega de Mensajes

Métrica: Mensajes entregados por segundo

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

Objetivo: Coincide con el volumen esperado

Tasa de Éxito de Entrega

Métrica: Porcentaje de entregas exitosas

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

Objetivo: > 98%

Estabilidad de Vinculación

Métrica: Intentos de vinculación por hora

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

Objetivo: < 10 por hora (indica conexión estable)

Mejores Prácticas de Monitoreo

1. Configurar Alertas

- Configurar alertas de Prometheus para métricas críticas
- Usar PagerDuty/OpsGenie para alertas 24/7
- Probar alertas regularmente

2. Crear Tableros

- Construir tableros de Grafana para cada puerta de enlace
- Incluir todas las conexiones en un solo tablero
- Agregar paneles de planificación de capacidad

3. Revisiones Regulares

- Revisar métricas semanalmente
- Identificar tendencias y patrones
- Planificar ajustes de capacidad

4. Documentar Líneas Base

- Registrar volúmenes normales de mensajes
- Documentar tasas de TPS esperadas
- Anotar tiempos/días pico

5. Correlacionar con Backend

- Monitorear métricas de API de backend
- Rastrear flujo de mensajes de extremo a extremo
- Identificar cuellos de botella

Solución de Problemas con Métricas

Problemas de Conexión

Verificar: `smpp_connection_status`

- Valor 0 = Revisar registros, verificar red, validar credenciales
- Cambios frecuentes = Inestabilidad de la red

Bajas Tasas de Entrega

Verificar: `smpp_delivery_failures_total`

- Alta tasa = Verificar estado del operador, revisar formato del mensaje
- Comparar entre conexiones = Identificar operador problemático

Bajo Rendimiento

Verificar: tasa de `smpp_messages_sent_total`

- Por debajo de lo esperado = Verificar límites de TPS, disponibilidad de cola
- Verificar métricas de API de backend

Problemas de Vinculación

Verificar: `smpp_bind_failures_total`

- En aumento = Problemas de autenticación, problemas con credenciales
- Verificar system_id y contraseña en la configuración

Documentación Relacionada

- **CONFIGURATION.md** - Configurar ajustes de monitoreo
- **USAGE.md** - Procedimientos operativos
- **TROUBLESHOOTING.md** - Resolver problemas
- **README.md** - Visión general y guía rápida

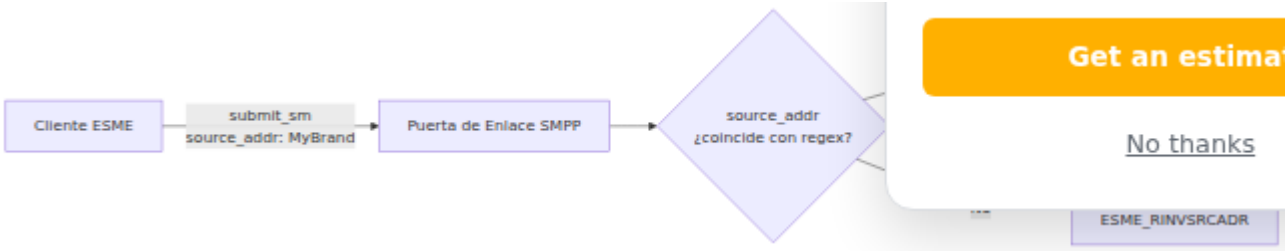
Lista Blanca de Direcciones de Origen

Control por par sobre qué direcciones de origen (source_addr) puede usar un cliente SMPP al enviar mensajes.

Descripción General

Cuando un ESME externo (cliente) envía un PDU `submit_sm` a través de la Puerta de Enlace SMPP, el PDU incluye un campo `source_addr` que representa la dirección de origen (CLI / ID del remitente). Por defecto, los clientes autenticados pueden usar cualquier dirección de origen. La función de Lista Blanca de Direcciones de Origen permite a los operadores restringir qué direcciones de origen puede usar cada par de servidor.

Esto sigue el mismo patrón que la Lista Blanca de IP existente: cuando la lista blanca está vacía, se permiten todos los valores. Cuando está poblada, solo se aceptan las direcciones de origen que coincidan.



Reglas de Coincidencia

Cada entrada de la lista blanca es un **patrón de expresión regular**. La dirección de origen se prueba contra cada patrón. Si algún patrón coincide, se permite la dirección.

Coincidencia Exacta

Ancla el patrón con `^` y `$` para requerir una coincidencia exacta. La coincidencia es **sensible a mayúsculas** por defecto.

Entrada de la Lista Blanca	Dirección de Origen	Resultado
`^MyBrand$`	`MyBrand`	Permitido
`^MyBrand$`	`mybrand`	Rechazado
`^MyBrand$`	`MyBrands`	Rechazado
`^\+61400000001$`	`+61400000001`	Permitido

Coincidencia por Prefijo

Usa `^` sin un `$` final para coincidir con direcciones que comienzan con un prefijo.

Entrada de la Lista Blanca	Dirección de Origen	Resultado
`^614`	`61400000001`	Permitido
`^614`	`61412345678`	Permitido
`^614`	`61500000001`	Rechazado
`^\+614`	`+61400000001`	Permitido

Patrones Avanzados

Se admite la sintaxis completa de expresiones regulares, lo que permite reglas de coincidencia poderosas.

Entrada de la Lista Blanca	Descripción	Ejemplos
`^61[45]\d{8}$`	Números móviles australianos 614/615 (11 dígitos)	`61400000001`, `61512345678`
`^(MyBrand|AlertSvc)$`	Coincidencia exacta en cualquiera de los nombres	`MyBrand`, `AlertSvc`
`(?i)^mybrand$`	Coincidencia exacta sin distinción de mayúsculas	`MyBrand`, `mybrand`, `MYBRAND`
`^(?!Test).+`	Cualquier dirección no vacía que no comience con "Test"	`MyBrand` (permitido), `TestBrand` (rechazado)

Múltiples Entradas

Cuando se configuran múltiples entradas, se permite la dirección de origen si coincide con **cualquiera** de las entradas en la lista blanca.

Ejemplo de lista blanca: `^MyBrand$, ^614, ^\+61400000001$`

Dirección de Origen	Coincidencias	Resultado
`MyBrand`	`^MyBrand$`	Permitido
`61412345678`	`^614`	Permitido
`+61400000001`	`^\+61400000001$`	Permitido
`OtherBrand`	Ninguna	Rechazado
`61500000001`	Ninguna	Rechazado

Manejo de Errores

Cuando un `submit_sm` es rechazado debido a una violación de la lista blanca de direcciones de origen, la puerta de enlace responde con:

Campo	Valor
PDU	`submit_sm_resp`
Estado del Comando	`0x0000000A`
Nombre del Error	`ESME_RINVSRCADR` (Dirección de Origen Inválida)
ID del Mensaje	Vacío

Se registra una advertencia con la dirección de origen rechazada y el nombre del par:

```
Servidor SMPP: Rechazado submit_sm de partner_acme - source_addr
'UnauthorisedBrand' no en la lista blanca
```

Los patrones regex inválidos se tratan como no coincidentes (la dirección es rechazada) y se registra una advertencia.

Configuración

A través de la Interfaz Web

1. Navega a **SMPP > Servidores Peers**
2. Haz clic en **Editar** en el par objetivo (o **Agregar Nuevo Servidor Peer**)
3. Localiza el campo **Lista Blanca de Direcciones de Origen** (debajo de la Lista Blanca de IP)
4. Ingresa patrones regex separados por comas:

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

5. Haz clic en **Guardar**

Los cambios entran en vigor de inmediato para nuevos PDUs `submit_sm` en conexiones existentes.

A través del Archivo de Configuración

Agrega `source_address_whitelist` a la configuración de enlace del servidor en `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
  }
]
```

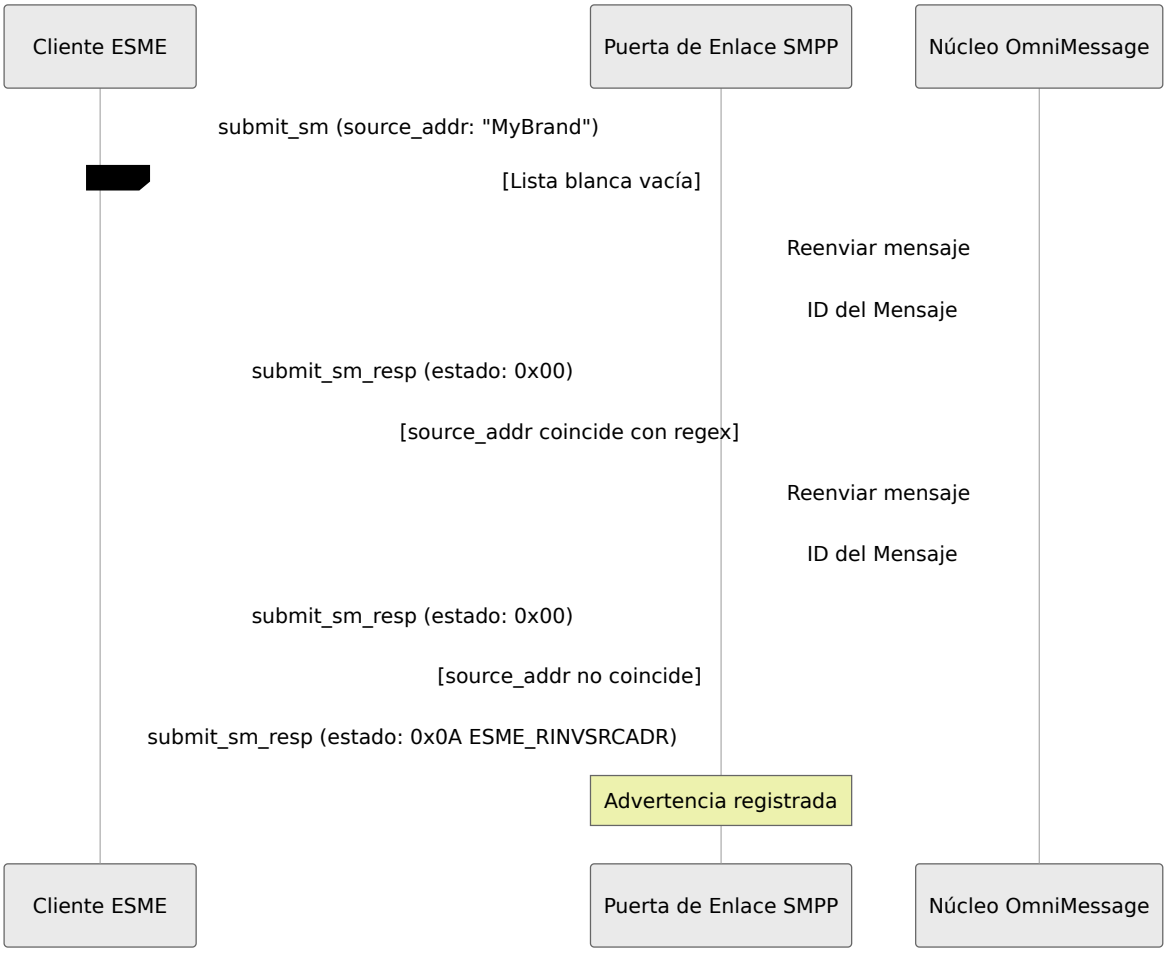
Parámetros

Parámetro	Tipo	Requerido	Predeterminado
`source_address_whitelist`	Lista de cadenas	No	`[]` (permitir todos)

Migración

Los pares de servidor existentes se migran automáticamente cuando se inicia la puerta de enlace. Los pares creados antes de que se agregara esta función reciben una lista blanca vacía (todas las direcciones de origen permitidas), preservando el comportamiento existente.

Flujo de Validación



Ejemplos

Restringir a una Sola Marca

Solo permitir mensajes del ID de remitente `AcmeCorp`:

```
^AcmeCorp$
```

Permitir un Rango de Números Australianos

Permitir cualquier número móvil australiano (que comience con `614`):

```
^614
```

Permitir Rangos de Números Específicos

Permitir números móviles australianos 614 y 615 (exactamente 11 dígitos):

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

Alfanumérico y Numérico Mixto

Permitir un nombre de marca y un rango de números:

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

Coincidencia de Marca Sin Distinción de Mayúsculas

Permitir un nombre de marca sin importar las mayúsculas:

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

Permitir Todos (Predeterminado)

Dejar el campo vacío para permitir cualquier dirección de origen. Este es el comportamiento predeterminado.

Solución de Problemas

Mensajes Rechazados con ESME_RINVSRCADR

Síntomas: El socio informa `submit_sm_resp` con estado de comando `0x0A`.

Causas posibles:

- La dirección de origen no coincide con ningún patrón regex en la lista blanca
- El patrón regex tiene un error de sintaxis (tratado como no coincidente)
- Falta de anclajes (`^`/`$`) que causan coincidencias parciales inesperadas o rechazos
- Desajuste de mayúsculas (la coincidencia es sensible a mayúsculas por defecto; usa `(?i)` para coincidencias sin distinción de mayúsculas)

Resolución:

1. Verifica la Lista Blanca de Direcciones de Origen del par del servidor en la Interfaz Web
2. Compara la dirección de origen rechazada contra cada patrón regex
3. Prueba los patrones usando un probador de regex de Elixir: `Regex.match?(~r/^pattern$/, "address")`
4. Agrega el patrón faltante o ajusta los patrones existentes
5. Recuerda escapar los caracteres especiales de regex (por ejemplo, `\+` para el `+` literal)

La Lista Blanca No Toma Efecto

Síntomas: Mensajes aceptados a pesar de que la dirección de origen no coincide con la lista blanca.

Causas posibles:

- La lista blanca está vacía (permite todos por defecto)
- El patrón no está anclado y coincide como una subcadena (por ejemplo, `614` coincide con `X614Y`)
- El ESME está conectado a un par de servidor diferente
- El cambio en el archivo de configuración aún no se ha aplicado (requiere reinicio)

Resolución:

1. Verifica que la lista blanca esté poblada (no vacía) en la Interfaz Web
2. Asegúrate de que los patrones usen anclajes `^` y `$` donde se pretende una coincidencia exacta
3. Verifica a qué par de servidor está vinculado el ESME en el Estado en Vivo
4. Si usas un archivo de configuración, reinicia el servicio

Documentación Relacionada

- **Referencia de Configuración** - Documentación completa de parámetros de pares de servidores
- **Guía de Uso** - Gestión de conexiones SMPP
- **Solución de Problemas** - Procedimientos generales de solución de problemas

Guía de Solución de Problemas

Problemas comunes y soluciones

El servicio no inicia / Conflictos de puerto

El Gateway o el núcleo de OmniMessage se reinician continuamente con `:eaddrinuse` en el puerto 8443

Síntomas:

- Uno de los servicios se reinicia continuamente (`systemctl show <unit> -p NRestarts` aumenta rápidamente).
- Los registros muestran:

```
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` devuelve `404` aunque la tienda debería responder.

Causa: El gateway SMPP y el núcleo de OmniMessage están colocados en el mismo host y ambos vinculan su API `api_ex` en el puerto predeterminado `8443`. Quien inicie primero gana el puerto; el otro entra en un bucle de reinicio. Si el gateway gana, `/api/messages` es servido por el gateway (que no tiene tienda) y devuelve 404.

Solución: Mover la propia API OAM/Gestión del gateway fuera del puerto del núcleo (el núcleo mantiene `8443`). Editar `/etc/omnimessage_smpp/runtime.exs`

y descomentar la anulación de `api_ex`, estableciendo un puerto libre. El gateway sigue siendo un cliente de la API de tienda del núcleo a través de `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")
```

Luego reinicie ambos servicios:

```
sudo systemctl restart omnimessage-smpp
sudo systemctl restart omnimessage        # deja que el núcleo
reclame :8443
```

Después del reinicio, verifique que el núcleo posee 8443 y el gateway posee su nuevo puerto:

```
ss -lntp | grep -E ':8443|:8444'
curl -sk -o /dev/null -w '%{http_code}\n'
https://127.0.0.1:8443/api/messages   # la tienda responde (no
404)
```

Consulte CONFIGURATION.md → puerto de API OAM/Gestión para más detalles.

Problemas de Conectividad de OmniMessage

Dado que el Gateway SMPP es sin estado y depende completamente del núcleo de OmniMessage, los problemas de conectividad con OmniMessage son los problemas más críticos.

Síntomas de Desconexión de OmniMessage

- **No hay mensajes salientes**: La cola se acumula, los mensajes no se envían
- **No hay mensajes entrantes**: Los socios no pueden enviar mensajes
- **Tiempos de espera**: Las llamadas a la API se agotan o se cuelgan
- **Los registros muestran**: "Conexión rechazada", "Tiempo de espera", "HTTP 503", "Conexión restablecida"

Diagnóstico

1. Verifique la Disponibilidad de OmniMessage:

```
# Probar conectividad
curl -k -v https://omnimessage-
core.example.com:8443/api/system/health

# Probar desde el host del gateway específicamente
ssh gateway-server 'curl -k https://omnimessage-
core.example.com:8443/api/system/health'
```

2. Verifique la URL de API Configurada:

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

# Verificar conectividad de red
ping omnimessage-core.example.com
nc -zv omnimessage-core.example.com 8443
```

3. Verifique los Registros del Gateway para Errores de API:

```
# Buscar errores relacionados con la API
sudo journalctl -u omnimessage-smpp -f | grep -i
'api\|omnimessage\|connect'

# Buscar en los registros errores recientes
sudo journalctl -u omnimessage-smpp -n 200 | grep -i error
```

Soluciones

Si OmniMessage está inactivo:

1. Contactar al equipo de operaciones de OmniMessage
2. Los mensajes pendientes se acumularán en la cola
3. El gateway seguirá reintentando (ver `SMPP_POLL_INTERVAL`)
4. Verificar la página de estado de OmniMessage o el monitoreo

Si OmniMessage está activo pero el gateway no puede alcanzarlo:

1. Verificar que las reglas del firewall permitan HTTPS saliente
2. Verificar la resolución DNS: `nslookup omnimessage-core.example.com`
3. Verificar el enrutamiento de red: `traceroute omnimessage-core.example.com`
4. Verificar los certificados SSL si se usa HTTPS

Si la URL de la API está mal configurada:

1. Editar `/opt/omnimessage-smpp/config/runtime.exs`
2. Verificar que `api_base_url` sea correcto (debe ser HTTPS para producción)
3. Reiniciar el gateway: `sudo systemctl restart omnimessage-smpp`

Problemas de Conexión

La Conexión No Se Establece

Síntomas:

- El estado muestra "Desconectado" (rojo)
- No hay enlace exitoso en los registros
- Intentos de conexión repetidos

Causas Posibles y Soluciones:

1. Problemas de Conectividad de Red

Verificar:

```
# Probar resolución DNS
nslookup smpp.carrier.com

# Probar conectividad
ping -c 3 smpp.carrier.com

# Probar puerto
telnet smpp.carrier.com 2775
# o
nc -zv smpp.carrier.com 2775
```

Soluciones:

- Si DNS falla: Usar la dirección IP en lugar del nombre de host en la configuración
- Si el ping falla: Verificar reglas del firewall, contactar al operador
- Si el puerto falla: Verificar número de puerto correcto, verificar firewall

2. Credenciales Incorrectas

Verificar:

- Los registros muestran "bind failed" o "authentication error"
- Interfaz web: SMPP → Clientes pares → verificar system_id y contraseña

Soluciones:

- Confirmar credenciales con el operador
- Verificar errores tipográficos (sensible a mayúsculas)

- Actualizar la configuración y reconectar

3. IP No Autorizada

Verificar:

- Conexión rechazada inmediatamente
- Los registros del operador muestran IP no autorizada

Soluciones:

- Confirmar la IP pública de su gateway:

```
curl ifconfig.me
```

- Solicitar al operador que agregue la IP a la lista blanca
- Verificar que la IP no haya cambiado (IP dinámica)

4. Firewall Bloqueando

Verificar:

```
# Verificar si el puerto está abierto
sudo iptables -L -n | grep 2775

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

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

Soluciones:

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

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

5. Dirección/ Puerto de Origen Fijado No Disponible

Solo se aplica cuando el cliente bind establece `bind_source_ip` y/o `bind_source_port` (ver CONFIGURATION.md).

Verificar:

- Los registros muestran que la conexión no puede abrir el socket local
- `bind_source_ip` no es una dirección mantenida por este host
- Otro proceso ya mantiene el `bind_source_port` configurado

Soluciones:

- Confirmar que `bind_source_ip` sea una dirección local: `ip addr show`
- Una `bind_source_ip` no válida se registra y se ignora; el SO luego elige la IP de origen, que puede no coincidir con la lista blanca del operador. Corrija la dirección si la entrega está fallando.
- El puerto fijado se abre con `SO_REUSEADDR`, por lo que una reconexión vuelve a enlazar inmediatamente incluso mientras el viejo socket está en `TIME_WAIT`. Si el puerto sigue no disponible, verifique si otro servicio lo utiliza: `sudo ss -ltnp | grep <port>`

La Conexión Sigue Cayendo

Síntomas:

- Conexión establecida pero se desconecta frecuentemente
- Métrica `smpp_disconnection_total` en aumento
- Los registros muestran reconexiones repetidas

Causas Posibles y Soluciones:

1. Inestabilidad de la Red

Verificar:

```
# Monitorear pérdida de paquetes
ping -c 100 smpp.carrier.com | grep loss

# Verificar errores de red
netstat -s | grep -i error
```

Soluciones:

- Contactar al operador sobre problemas de red
- Verificar con el ISP si es de su lado
- Considerar conexión/ruta de respaldo

2. Tiempo de Espera de Enquiry Link

Verificar:

- Los registros muestran "enquire_link timeout"
- La conexión se cae después de períodos de inactividad

Soluciones:

- El tiempo de espera predeterminado es de 30 segundos
- Verificar que la red permita paquetes keepalive
- Verificar si hay firewalls agresivos que agotan conexiones inactivas

3. Límite de TPS Excedido

Verificar:

- Alta tasa de mensajes en el momento de la desconexión
- El operador está limitando los mensajes

Soluciones:

- Revisar la configuración de `tps_limit`
- Reducir TPS al 70-80% del máximo del operador
- Distribuir el tráfico entre múltiples enlaces

4. Problemas del Servidor del Operador

Verificar:

- Verificar el estado del servicio del operador
- Contactar al soporte del operador

Soluciones:

- Esperar a que el operador resuelva
- Configurar un operador de respaldo si está disponible

Problemas de Entrega de Mensajes

Los Mensajes No Se Están Enviando

Síntomas:

- Mensajes atascados en la cola
- `smpp_messages_sent_total` no aumenta
- La conexión muestra conectada

Causas Posibles y Soluciones:

1. Enrutamiento de dest_smsc Incorrecto

Verificar:

- Interfaz web → Cola → Verificar el campo `dest_smsc` del mensaje
- Comparar con el nombre de conexión en SMPP → Estado en Vivo

Soluciones:

- Los mensajes se enrutan según el campo `dest_smsc`
- Verificar que el backend esté configurando el `dest_smsc` correcto
- Si `dest_smsc` es NULL, verificar el enrutamiento predeterminado

2. Mensajes Programados para el Futuro

Verificar:

- Interfaz web → Cola → Verificar el campo `deliver_after`
- Los mensajes con marca de tiempo futura no se enviarán aún

Explicación:

- El sistema de reintentos establece `deliver_after` para mensajes fallidos
- Los mensajes esperan hasta ese momento antes de reintentar

Soluciones:

- Esperar el tiempo programado
- Si es urgente, contactar al equipo de backend para restablecer la marca de tiempo

3. Límite de TPS Demasiado Bajo

Verificar:

- Gran acumulación en la cola
- Mensajes enviándose muy lentamente

Soluciones:

- Aumentar `tps_limit` en la configuración
- Verificar que el operador pueda manejar una tasa más alta
- Ver ver CONFIGURATION.md

4. Trabajador de Cola No Está Ejecutándose

Verificar:

- Estado del servicio
- Registros para errores

Soluciones:

```
# Reiniciar servicio
sudo systemctl restart omnimessage-smpp

# Verificar registros
sudo journalctl -u omnimessage-smpp -f
```

5. Frontend No Registrado Bajo el Nombre Esperado

El núcleo de OmniMessage solo enruta un mensaje a este gateway si un par está registrado bajo un nombre de frontend igual al `dest_smsc` del mensaje. Cada par se registra bajo su **clave de enrutamiento** (ver Registro de Frontend):

- **Pares de clientes**: la `queue` configurada, o el nombre del par simple si no se establece `queue`.
- **Vínculos de servidor**: el nombre de enlace.

Verificar:

- Registros para entradas de `Successfully registered SMPP peer frontend '<name>'` — confirmar que `<name>` coincide con el `dest_smsc` en los mensajes atascados
- Para enlaces que comparten una `queue`, esperar un solo registro bajo el nombre de la cola compartida, no uno por par

Soluciones:

- Asegurarse de que el backend establezca `dest_smsc` a la clave de enrutamiento (la `queue` del par, de lo contrario su nombre), no a `{hostname}_{peer}` (un esquema de nomenclatura anterior que ya no se aplica)
- Si se usa una `queue` compartida, enrutar mensajes al nombre de la cola en lugar de a un nombre de par individual
- Confirmar que el par esté habilitado — los pares deshabilitados no están registrados

---

Alta Tasa de Fallos de Entrega

Síntomas:

- `smpp_delivery_failures_total` en aumento
- Los registros muestran "submit_sm_resp" con estado de error
- Los mensajes no llegan a los destinatarios

Causas Posibles y Soluciones:

1. Números de Destino Inválidos

Verificar:

- Registros para códigos de error específicos
- Revisar el formato de destino del mensaje

Códigos de Error Comunes:

- `0x0000000B` - Destino inválido
- `0x00000001` - Longitud de mensaje inválida
- `0x00000003` - Comando inválido

Soluciones:

- Validar formato de número (se recomienda E.164)
- Verificar que el número incluya el código de país
- Confirmar con los requisitos del operador

2. Contenido del Mensaje Inválido

Verificar:

- Longitud del mensaje
- Caracteres especiales
- Codificación

Soluciones:

- GSM-7: Máx 160 caracteres
- UCS-2: Máx 70 caracteres
- Eliminar caracteres no soportados
- Verificar configuraciones de codificación

3. Rechazo del Operador

Verificar:

- Códigos de error específicos del operador
- Patrones en mensajes rechazados

Soluciones:

- Contactar al operador para conocer la razón del rechazo
- Puede ser necesario filtrar contenido
- Verificar patrones de spam/abuso

4. Mensajes Expirados

Verificar:

- Marca de tiempo `expires` del mensaje
- Tiempo de intento de entrega

Soluciones:

- Aumentar el período de validez del mensaje
- Reducir el retraso de reintento para mensajes sensibles al tiempo

---

Problemas con la Interfaz Web

No se Puede Acceder al Panel Web

Síntomas:

- El navegador no puede conectarse a https://your-server:8087

- Tiempo de espera o conexión rechazada

Causas Posibles y Soluciones:

1. Servicio No Ejecutándose

Verificar:

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

Soluciones:

```
# Si está detenido, inícielo
sudo systemctl start omnimessage-smpp

# Verificar registros para errores
sudo journalctl -u omnimessage-smpp -n 50
```

2. Firewall Bloqueando el Puerto 8087

Verificar:

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

Soluciones:

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

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

3. Problemas con el Certificado SSL

Verificar:

- El navegador muestra advertencia de seguridad
- Certificado expirado o inválido

Soluciones:

- Aceptar excepción de seguridad (si es autofirmado)
- Instalar certificado SSL válido
- Verificar que los archivos del certificado existan:

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

4. URL Incorrecta

Verificar:

- Verificar usando HTTPS (no HTTP)
- Verificar IP/hostname del servidor correcto
- Verificar puerto 8087

La Interfaz Web Muestra Errores

Síntomas:

- La página se carga pero muestra errores
- Las funciones no funcionan
- Los datos no se muestran

Soluciones:

1. **Limpiar Caché del Navegador**:
    - Ctrl+F5 (actualización forzada)
    - Limpiar caché y cookies del navegador
2. **Verificar Consola del Navegador**:
    - Presionar F12
    - Verificar la pestaña de Consola para errores de JavaScript

- Informar al soporte si se encuentran errores

3. **Probar Diferente Navegador**:
   - Probar en Chrome, Firefox, Edge
   - Aislar problemas específicos del navegador
4. **Verificar Registros del Servicio**:

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

Problemas de Métricas

Métricas de Prometheus No Disponibles

Síntomas:

- `curl http://localhost:4000/metrics` falla
- Prometheus no puede raspar métricas
- Respuesta vacía o de error

Causas Posibles y Soluciones:

1. Servicio No Ejecutándose

Verificar:

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

Soluciones:

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

2. Puerto No Accesible

Verificar:

```
# Probar localmente
curl http://localhost:4000/metrics

# Probar remotamente
curl http://your-server-ip:4000/metrics
```

Soluciones:

- Si local funciona pero remoto no: Verificar firewall
- Abrir puerto 4000 en el firewall para el servidor de Prometheus

3. Endpoint Incorrecto

Verificar:

- El endpoint es `/metrics` (no `/prometheus` o `/stats`)
- El puerto es 4000 (no 8087)

Métricas Muestran Valores Inesperados

Síntomas:

- Contadores se restablecen a cero
- Los medidores muestran valores incorrectos
- Faltan métricas para algunos enlaces

Soluciones:

1. **Reinicio del Servicio Restablece Contadores**:
    - Los contadores se restablecen al reiniciar el servicio
    - Este es un comportamiento normal
    - Usar `increase()` o `rate()` en las consultas de Prometheus
2. **Nuevos Enlaces No Aparecen**:

- Las métricas solo aparecen después del primer evento
- Enviar un mensaje de prueba para poblar métricas
- Verificar que el enlace esté habilitado y conectado

3. **Métricas Obsoletas**:
   - Los enlaces antiguos pueden seguir apareciendo en las métricas
   - Reiniciar el servicio para limpiar entradas obsoletas
   - O usar reetiquetado de Prometheus para filtrar

Problemas de Rendimiento

Alto Uso de CPU

Verificar:

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

Causas Posibles:

- Volumen de mensajes muy alto
- Demasiadas conexiones
- Problema de configuración

Soluciones:

- Verificar que la tasa de mensajes esté dentro de la capacidad
- Revisar límites de TPS
- Contactar soporte si el uso de CPU es sostenido

Alto Uso de Memoria

Verificar:

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

Causas Posibles:

- Gran cola de mensajes en memoria
- Fuga de memoria (rara)

Soluciones:

- Reiniciar el servicio para limpiar memoria
- Verificar tamaño de la cola de mensajes
- Contactar soporte si la memoria crece continuamente

Procesamiento Lento de Mensajes

Síntomas:

- Los mensajes tardan mucho en enviarse
- La cola se acumula
- Baja tasa de mensajes

Verificar:

1. Límites de TPS - pueden ser demasiado restrictivos
2. `queue_check_frequency` - puede ser demasiado alto
3. Tiempo de respuesta de la API del backend - puede ser lento
4. Latencia de red hacia el operador

Soluciones:

- Aumentar TPS si el operador lo permite
- Disminuir `queue_check_frequency` para un sondeo más rápido
- Optimizar la API del backend
- Verificar latencia de red

Problemas de Configuración

Errores de Sintaxis en el Archivo de Configuración

Síntomas:

- El servicio no inicia después de un cambio de configuración
- Los registros muestran "error de sintaxis" o "error de análisis"

Verificar:

```
# Validar sintaxis de Elixir
/opt/omnimessage-smpp/bin/omnimessage-smpp eval "File.read!
('config/runtime.exs')"
```

Errores Comunes:

- Falta una coma entre entradas del mapa
- Comillas desajustadas (" vs ')
- Corchetes o llaves no coincidentes
- Falta `import Config` en la parte superior

Soluciones:

- Restaurar desde una copia de seguridad
- Revisar cuidadosamente la sintaxis
- Usar un editor de texto con resaltado de sintaxis de Elixir

Cambios No Tienen Efecto

Síntomas:

- Configuración modificada pero sin cambio en el comportamiento
- Configuraciones antiguas aún activas

Soluciones:

```
# Los cambios de configuración requieren reinicio
sudo systemctl restart omnimessage-smpp

# Verificar que el reinicio haya tenido éxito
sudo systemctl status omnimessage-smpp

# Verificar registros para errores
sudo journalctl -u omnimessage-smpp -n 50
```

Recuperación de Emergencia

Fallo Completo del Sistema

Pasos:

1. **Verificar la salud básica del sistema**:

   ```
   # Espacio en disco
   df -h

   # Memoria
   free -h

   # Carga de CPU
   uptime
   ```

2. **Verificar estado del servicio**:

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

3. **Revisar registros recientes**:

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

4. **Intentar reiniciar el servicio**:

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

5. **Si el reinicio falla**:
   - Verificar sintaxis de configuración
   - Verificar que existan certificados SSL
   - Verificar permisos de archivos
   - Revisar registros para errores específicos
6. **Restaurar desde una copia de seguridad** (si es necesario):

```
# Restaurar configuración
sudo cp /opt/omnimessage-smpp/config/runtime.exs.backup \
  /opt/omnimessage-smpp/config/runtime.exs

# Reiniciar
sudo systemctl restart omnimessage-smpp
```

7. **Contactar soporte** si no se resuelve

Obtener Ayuda

Información a Reunir

Antes de contactar al soporte, recopile:

1. **Versión**: `cat /opt/omnimessage-smpp/VERSION`
2. **Registros Recientes**:

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

3. **Configuración** (sanitizar contraseñas):

```
sudo cp /opt/omnimessage-smpp/config/runtime.exs
/tmp/config.exs
# Editar /tmp/config.exs para eliminar contraseñas antes de
enviar
```

4. **Salida de Métricas**:

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

5. **Información del Sistema**:

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

Contactar Soporte

- **Correo Electrónico**: support@omnitouch.com
- **Teléfono**: +61 XXXX XXXX (24/7)
- **Incluir**: Toda la información de arriba

Documentación Relacionada

- **USAGE.md** - Procedimientos operativos normales
- **CONFIGURATION.md** - Referencia de configuración
- **MONITORING.md** - Monitoreo y métricas
- **README.md** - Visión general del sistema

Guía de Operaciones

Procedimientos operativos diarios

Dependencia Crítica: OmniMessage Core

IMPORTANTE: El Gateway SMPP de OmniMessage no puede funcionar sin acceso a OmniMessage Core. Todo el procesamiento de mensajes ocurre en OmniMessage - el gateway es solo un traductor de protocolo.

Si OmniMessage se vuelve no disponible:

- □ No se pueden enviar nuevos mensajes
- □ No se pueden recuperar mensajes pendientes
- □ No se puede informar el estado de entrega
- □ El sistema parece colgarse o agotar el tiempo

Verificar la Salud de OmniMessage:

```
# Probar conectividad de API
curl -k https://omnimessage-
core.example.com:8443/api/system/health

# Verificar URL de API configurada en los registros
grep api_base_url /opt/omnimessage-smpp/config/runtime.exs
```

Operaciones Diarias

Verificación de Salud Matutina

Realice estas verificaciones al inicio de cada día:

1. **Acceder al Panel Web**

- URL: `https://your-server:8087`
- Verifique si el panel se carga correctamente

2. **Verificar Estado de Conexión**

- Navegar a: SMPP → Estado en Vivo
- Verifique que todas las conexiones muestren "Conectado" (verde)
- Anote cualquier enlace desconectado

3. **Revisar Métricas de Mensajes**

- Navegar a: pestaña de Cola
- Verifique que los conteos de mensajes sean razonables
- Verifique que no haya acumulación inesperada en la cola

4. **Verificar Registros del Sistema**

- Navegar a: pestaña de Registros
- Busque mensajes de error (rojo)
- Anote cualquier patrón de advertencia

5. **Revisar Métricas de Prometheus**

- `curl http://localhost:4000/metrics`
- O verifique los paneles de Grafana
- Verifique que las tasas de mensajes sean normales

Monitoreo Continuo

Configure alertas para:

- Fallos de conexión (> 2 minutos fuera de servicio)
- Altas tasas de fallos de entrega (> 5%)
- Sin tráfico durante períodos prolongados
- Desconexiones frecuentes

Consulte MONITORING.md para la configuración de alertas.

Comprendiendo el Enrutamiento de Mensajes

El gateway enruta mensajes entre OmniMessage Core y conexiones SMPP utilizando dos campos clave:

- `dest_smsc` - Enruta mensajes salientes a **enlaces de cliente**. Cuando OmniMessage coloca un mensaje en la cola con `dest_smsc: "vodafone_uk"`, el enlace de cliente del gateway llamado `vodafone_uk` lo recoge y lo envía a través de SMPP `submit_sm`.
- `source_smsc` - Enruta mensajes entrantes a **enlaces de servidor**. Cuando OmniMessage coloca un mensaje en la cola con `source_smsc: "partner_acme"`, el gateway lo entrega a los clientes conectados al enlace de servidor llamado `partner_acme` a través de SMPP `deliver_sm`.

Distinción clave: Los enlaces de cliente envían PDUs `submit_sm` (el gateway es el ESME que se presenta a un operador). Los enlaces de servidor envían PDUs `deliver_sm` (el gateway es el SMSC que entrega a un ESME conectado).

Registro en el Frontend

El gateway se registra automáticamente con OmniMessage Core para que el backend sepa qué conexiones SMPP están disponibles para el enrutamiento de mensajes.

- **Nombre de registro**: Controlado por la configuración `smsc_name` (predeterminado: `"smpp_gateway"`, env: `SMSC_NAME`)
- **Heartbeat**: Enviado cada 60 segundos para mantener el registro activo
- **Expiración**: El registro expira en el backend después de 90 segundos sin un heartbeat
- **Registro por enlace**: Cada par habilitado se registra individualmente bajo su **clave de enrutamiento**, por lo que el nombre del frontend registrado es igual al `dest_smsc` que el backend enruta:
  - **Pares de cliente**: registrados bajo su `queue` configurada (si está configurada), de lo contrario, el nombre del par en sí. Varios pares de cliente que comparten una `queue` se registran, por lo tanto, bajo ese único nombre compartido.
  - **Enlaces de servidor**: registrados bajo su nombre de enlace.

Si el gateway se detiene o pierde conectividad con OmniMessage Core, sus registros expiran y el backend deja de enrutar mensajes hacia él.

Solución de problemas: Si los mensajes no se están enrutando al gateway, verifique:

1. Registros para entradas "frontend_register"
2. Que el `smsc_name` coincida con lo que OmniMessage espera
3. Conectividad de red con OmniMessage Core (`api_base_url`)

Gestión de Conexiones SMPP

Cómo se Configuran los Pares SMPP

Las conexiones SMPP (pares) se pueden configurar utilizando **dos métodos**:

Método 1: Interfaz Web o API OAM (Recomendado)

- **Ventaja**: Los cambios tienen efecto inmediato, no se requiere reinicio
- **Ubicación**: SMPP → Pares de Cliente / Pares de Servidor, o `/api/smpp/client_peers` y `/api/smpp/server_peers` en la API OAM
- **Operaciones**: Agregar, editar, eliminar pares
- **Lo que guarda un cambio**:
  - **Par de servidor**: su trabajador de cola se inicia o reinicia con la nueva configuración. El oyente SMPP se inicia si este es el primer par de servidor. Las sesiones ya vinculadas siguen funcionando.
  - **Par de cliente**: la conexión se desvincula y se restablece con la nueva configuración.
  - **Deshabilitar o eliminar un par**: sus trabajadores se detienen. Un par de cliente se desvincula primero. Las sesiones vinculadas de un par de servidor reciben un `unbind` (cerradas después de 2 segundos si el ESME no responde), y se rechazan nuevos enlaces.
- **Persistencia**: Almacenada en la base de datos Mnesia
- **Mejor para**: Operaciones diarias, pruebas, cambios rápidos

Método 2: Archivo de Configuración

- **Ventaja**: Configuración como código, control de versiones
- **Ubicación**: `/opt/omnimessage-smpp/config/runtime.exs`
- **Operaciones**: Definir pares en la configuración de Elixir
- **Persistencia**: Basada en archivos, sobrevive a los reinicios
- **Requiere**: Reinicio del servicio después de cambios
- **Mejor para**: Configuración inicial, infraestructura como código

Nota: Los cambios en la interfaz web se almacenan por separado y sobrescriben la configuración del archivo.

Consulte CONFIGURATION.md para referencia del archivo de configuración.

Agregar una Nueva Conexión de Cliente

Propósito: Configurar el gateway para actuar como un **ESME** (cliente) conectándose al **SMSC** (servidor) de un operador

Preparación: Reúna información del operador:

- Nombre de host/IP del servidor SMPP
- Número de puerto (generalmente 2775)
- ID del sistema (nombre de usuario)
- Contraseña
- Tipo de enlace (generalmente transceptor)
- Límite de TPS

Elija uno de los siguientes métodos:

Opción A: A través de la Interfaz Web (Recomendado)

Ventajas: Efecto inmediato, no se requiere reinicio

Pasos:

1. **Navegar a Pares de Cliente**:
    - Abrir Interfaz Web: `https://your-server:8087`
    - Navegar a: SMPP → Pares de Cliente
2. **Agregar Nuevo Par**:
    - Hacer clic en "Agregar Nuevo Par de Cliente"
    - Completar el formulario:
        - **Nombre**: `vodafone_uk` (identificador único)
        - **Host**: `smpp.vodafone.co.uk`
        - **Puerto**: `2775`
        - **ID del Sistema**: `your_username`
        - **Contraseña**: `your_password`
        - **Tipo de Enlace**: `Transceptor`
        - **Límite de TPS**: `100`
        - **Frecuencia de Verificación de Cola**: `1000`
    - Hacer clic en "Guardar"

3. **Conexión Establecida Automáticamente**:
    - El gateway intenta inmediatamente la conexión
    - Navegar a: SMPP → Estado en Vivo
    - El estado debería cambiar a "Conectado" (verde) dentro de 10-30 segundos
    - Verifique la pestaña de Registros para un mensaje de enlace exitoso
4. **Probar Flujo de Mensajes**:
    - Navegar a: pestaña de Cola
    - Enviar un mensaje de prueba con dest_smsc coincidiendo con el nombre de enlace
    - Monitorear en Estado en Vivo para la transmisión
    - Verificar la confirmación de entrega

Opción B: A través del Archivo de Configuración

Ventajas: Infraestructura como código, control de versiones

Pasos:

1. **Editar Archivo de Configuración**:

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

2. **Agregar Nuevo Enlace a la Configuración**:

```
config :omnimessage_smpp, :binds, [
  # Enlaces existentes...

  # Agregar nuevo enlace
  %{
    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,

    # Opcional: fijar la IP/puerto de origen local si el
operador permite listas blancas por ellos
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5000,

    # Opcional: compartir una cola de backend entre varios
enlaces (por defecto se usa el nombre)
    queue: "vodafone_uk"
  }
]
```

Consulte CONFIGURATION.md para la lista completa de parámetros de enlace de cliente, incluyendo `bind_source_ip`, `bind_source_port`, y `queue`.

3. **Guardar y Reiniciar Servicio**:

```
# Guardar archivo (Ctrl+X, Y, Enter en nano)

# Reiniciar servicio
sudo systemctl restart omnimessage-smpp
```

4. **Verificar Conexión**:

- Navegar a: SMPP → Estado en Vivo
- Encontrar nueva conexión
- El estado debería ser "Conectado" (verde)
- Verificar registros para enlace exitoso

5. **Probar Flujo de Mensajes**:

   - Navegar a: pestaña de Cola
   - Enviar un mensaje de prueba con dest_smsc coincidiendo con el nuevo nombre de enlace
   - Monitorear en Estado en Vivo para la transmisión
   - Verificar la confirmación de entrega

Agregar un Enlace de Servidor

Propósito: Configurar el gateway para actuar como un **SMSC** (servidor) aceptando conexiones de **ESMEs** externos (clientes asociados)

Preparación:

1. **Generar Credenciales**:

   - Crear un ID de sistema único: `partner_name`
   - Crear una contraseña fuerte
   - Documentar y compartir de manera segura con el socio

2. **Obtener Información del Socio**:

   - Direcciones IP de origen del socio
   - Volumen de mensajes esperado (para límite de TPS)
   - Tipos de enlace requeridos

Elija uno de los siguientes métodos:

Opción A: A través de la Interfaz Web (Recomendado)

Ventajas: Efecto inmediato, no se requiere reinicio

Pasos:

1. **Navegar a Pares de Servidor**:
    - Abrir Interfaz Web: `https://your-server:8087`
    - Navegar a: SMPP → Pares de Servidor
2. **Agregar Nuevo Par de Servidor**:
    - Hacer clic en "Agregar Nuevo Par de Servidor"
    - Completar el formulario:
        - **Nombre**: `partner_acme` (identificador único)
        - **ID del Sistema**: `acme_corp`
        - **Contraseña**: `secure_password_123`
        - **Tipos de Enlace Permitidos**: Seleccionar todos (Transmisor, Receptor, Transceptor)
        - **Lista Blanca de IP**: `203.0.113.0/24` (separadas por comas para múltiples)
        - **Límite de TPS**: `50`
        - **Frecuencia de Verificación de Cola**: `1000`
    - Hacer clic en "Guardar"
3. **Gateway Listo para la Conexión**:
    - El par de servidor ahora está activo y esperando la conexión del socio
    - No se requiere reinicio

4. **Compartir Información con el Socio**:
    - Direcci��n IP del gateway
    - Puerto: `2775`
    - ID del Sistema: `acme_corp`
    - Contraseña: `secure_password_123`
    - Tipo de Enlace: Según lo configurado
5. **Esperar la Conexión del Socio**:
    - Navegar a: SMPP → Estado en Vivo
    - Observar la conexión entrante
    - Verificar el éxito de la autenticación
    - Comprobar que la IP coincida con la lista blanca

Opción B: A través del Archivo de Configuración

Ventajas: Infraestructura como código, control de versiones

Pasos:

1. **Editar Archivo de Configuración**:

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

2. **Agregar Enlace de Servidor y Configuración de Escucha**:

```
# Agregar a la lista de server_binds
config :omnimessage_smpp, :server_binds, [
  # Enlaces de servidor existentes...

  # Agregar nuevo enlace de servidor
  %{
    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
  }
]

# Asegurar que la configuración de escucha exista (solo se
necesita una vez)
config :omnimessage_smpp, :listen, %{
  host: "0.0.0.0",
  port: 2775,
  max_connections: 100
}
```

3. **Guardar y Reiniciar Servicio**:

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

4. **Compartir Información con el Socio**:
   - Dirección IP del gateway
   - Puerto: `2775`
   - ID del Sistema: `acme_corp`
   - Contraseña: `secure_password_123`
   - Tipo de Enlace: Según lo configurado
5. **Esperar la Conexión del Socio**:
   - Navegar a: SMPP → Estado en Vivo

- Observar la conexión entrante
- Verificar el éxito de la autenticación
- Comprobar que la IP coincida con la lista blanca

Modificar Conexión Existente

Propósito: Actualizar parámetros de conexión (límites de TPS, contraseñas, lista blanca de IP, etc.)

Elija uno de los siguientes métodos:

Opción A: A través de la Interfaz Web (Recomendado)

Ventajas: Efecto inmediato, no se requiere reinicio

Pasos:

1. **Navegar a Pares**:
   - Abrir Interfaz Web: `https://your-server:8087`
   - Para conexiones de cliente: SMPP → Pares de Cliente
   - Para conexiones de servidor: SMPP → Pares de Servidor
2. **Editar Par**:
   - Encontrar el par a modificar
   - Hacer clic en el botón "Editar"
   - Actualizar los parámetros deseados:
     - Cambios comunes: límite de TPS, contraseña, lista blanca de IP, host/puerto
   - Hacer clic en "Guardar"
3. **Los Cambios Se Aplican Inmediatamente**:
   - La conexión se reconecta automáticamente con la nueva configuración
   - No se requiere reinicio del servicio
   - Navegar a: SMPP → Estado en Vivo para verificar
4. **Verificar Cambios**:

- Comprobar que la conexión se establece con éxito
- Monitorear la pestaña de Registros en busca de errores
- Probar el flujo de mensajes si es aplicable

Opción B: A través del Archivo de Configuración

Ventajas: Infraestructura como código, control de versiones

Pasos:

1. **Editar Archivo de Configuración**:

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

2. **Modificar Parámetros de Enlace**:
   - Encontrar el enlace en la lista `:binds` o `:server_binds`
   - Actualizar los parámetros deseados:
     - Cambios comunes: límite de TPS, contraseñas, lista blanca de IP, host/puerto
   - Ejemplo:

```
%{
  name: "vodafone_uk",
  # ... otros parámetros
  tps_limit: 150,  # Cambiado de 100
  password: "new_password"  # Contraseña actualizada
}
```

3. **Guardar y Reiniciar Servicio**:

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

4. **Verificar Cambios**:
   - Navegar a: SMPP → Estado en Vivo
   - Comprobar que la conexión se establece con éxito

- Monitorear registros en busca de errores
- Probar el flujo de mensajes

Eliminar una Conexión

Propósito: Descontinuar una conexión SMPP

Pasos:

1. **Notificar a las Partes Interesadas**:
   - Informar al operador/socio
   - Coordinar ventana de inactividad
2. **Desconectar a través de la Interfaz Web**:
   - Navegar a: SMPP → Estado en Vivo
   - Encontrar conexión
   - Hacer clic en "Eliminar Conexión"
   - Confirmar acción
3. **Eliminar Configuración**:
   - Navegar a: SMPP → Pares de Cliente/Servidor
   - Encontrar conexión
   - Hacer clic en "Eliminar"
   - Confirmar eliminación
4. **Verificar Eliminación**:
   - Comprobar Estado en Vivo - la conexión debería haber desaparecido
   - Revisar registros para un cierre limpio

Habilitar y Deshabilitar Conexiones

Propósito: Tomar temporalmente una conexión fuera de línea sin eliminar su configuración

Los pares tienen un campo `enabled` que controla si están activos. Los pares deshabilitados retienen toda su configuración pero no establecen ni aceptan conexiones.

A través de la Interfaz Web:

1. Navegar a: SMPP → Pares de Cliente o Pares de Servidor
2. Encontrar el par a deshabilitar
3. Hacer clic en "Editar"
4. Desmarcar la casilla "Habilitado"
5. Hacer clic en "Guardar"

La conexión se eliminará de inmediato. Para volver a habilitar, repita los pasos y marque la casilla nuevamente.

Casos de uso:

- Ventanas de mantenimiento planificadas del operador
- Pausar temporalmente una conexión de socio durante una investigación
- Deshabilitar una conexión mientras se esperan nuevas credenciales

---

Comportamiento de Conexión

Lógica de Reconexión

Cuando un enlace de cliente no puede conectarse, tiene su enlace rechazado, o pierde una sesión establecida por cualquier motivo, el gateway lo reconecta automáticamente:

- **Retroceso**: El primer reintento es después de 1 segundo, luego la espera se duplica en cada fallo consecutivo (2s, 4s, 8s ...) hasta un máximo de 30 segundos
- **Reinicio del retroceso**: La espera vuelve a 1 segundo una vez que una sesión ha estado vinculada durante 30 segundos. Un operador que acepta la conexión y luego la cierra de inmediato se reintenta a una tasa más lenta, no cada segundo

- **Conectado significa vinculado**: Un enlace solo se muestra como "Conectado", y solo envía tráfico, una vez que el operador ha aceptado el enlace. Mientras la conexión TCP esté activa pero la respuesta de enlace no haya llegado, el enlace se muestra como conectando
- **Tiempo de espera de conexión**: Una conexión TCP que no recibe respuesta se abandona después de 3 segundos y se reintenta con el retroceso anterior
- **Aislamiento**: Las reconexiones para un enlace nunca reinician otros enlaces o el servicio, sin importar cuántas veces el operador cierre la conexión
- **Inicio escalonado**: Cuando varios enlaces se inician simultáneamente (por ejemplo, después de un reinicio del servicio), las conexiones se escalonan con retrasos de 500 ms entre cada enlace para evitar abrumar la red
- **Inicio resiliente**: Si un operador no es accesible al inicio del gateway, el gateway se inicia con éxito y reintenta la conexión en segundo plano

Realice un seguimiento de las reconexiones con la métrica `smpp_reconnect_attempts_total` (consulte MONITORING.md).

Desconexión Limpia

Cuando un enlace de cliente se cierra mediante "Eliminar Conexión" o al detener o reiniciar el servicio, el gateway envía un `unbind` SMPP al operador y espera hasta 2 segundos por el `unbind_resp` antes de cerrar la conexión TCP. El operador libera el enlace de inmediato en lugar de esperar a que su propio keepalive se agote. Esto es importante para los operadores que permiten solo un enlace por cuenta, que de otro modo rechazarían la reconexión después de un reinicio.

Un enlace que aún se está conectando (TCP activo, enlace aún no aceptado) se cierra sin un `unbind`.

Enlace de Consulta (Keepalive)

El gateway envía periódicamente PDUs SMPP `enquire_link` para verificar que las conexiones estén vivas:

- **Intervalo predeterminado**: 60 segundos (configurable por enlace a través de `enquire_link_interval`)
- **Deshabilitar**: Establecer `enquire_link_interval: 0` (no recomendado)
- **Detección de fallos**: Si el par remoto deja de responder a enquire_link, la conexión se considera muerta y comienza la reconexión

Monitoree la salud del enlace de consulta a través de las métricas de Prometheus `smpp_enquire_link_sent_total` y `smpp_enquire_link_received_total`. Una brecha creciente entre enviados y recibidos indica problemas de conexión.

Limitación de Tasa TPS

Cada enlace aplica su `tps_limit` utilizando una ventana deslizante por segundo:

- Los mensajes se cuentan dentro de cada ventana de 1 segundo
- Cuando se alcanza el límite, el trabajador de la cola se pausa hasta el siguiente segundo
- Un máximo de 100 mensajes puede estar en vuelo (esperando respuesta) por enlace en cualquier momento
- La ventana se restablece automáticamente al inicio de cada nuevo segundo

Si observa un rendimiento lento, verifique que:

1. `tps_limit` esté configurado lo suficientemente alto para su tráfico
2. `queue_check_frequency` sea lo suficientemente bajo para mantener el canal alimentado
3. El operador esté respondiendo a los mensajes de manera oportuna (respuestas lentas reducen el rendimiento efectivo)

Gestión del Flujo de Mensajes

Verificación de Cola de Mensajes

Propósito: Monitorear mensajes pendientes

Pasos:

1. **Acceder a la Cola**:
    - Navegar a: pestaña de Cola
    - Ver lista de mensajes pendientes

2. **Verificar Detalles del Mensaje**:
    - Hacer clic en la fila del mensaje
    - Revisar:
        - Número de destino
        - Cuerpo del mensaje
        - SMSC objetivo (dest_smsc)
        - Intentos de entrega
        - Estado

3. **Buscar Mensaje Específico**:

- Usar filtro de búsqueda
- Filtrar por destino, contenido o SMSC

Solución de Problemas de Mensajes Atascados

Síntomas: Mensajes no entregados

Pasos:

1. **Verificar Estado de Conexión**:
   - Navegar a: SMPP → Estado en Vivo
   - Verificar que la conexión objetivo esté conectada
   - Si está desconectada, consulte Reconectando
2. **Verificar Detalles del Mensaje**:
   - Navegar a: pestaña de Cola
   - Encontrar mensaje atascado
   - Verificar que el campo `dest_smsc` coincida con el nombre de la conexión
   - Verificar la marca de tiempo `deliver_after` (programación de reintentos)
3. **Verificar Intentos de Entrega**:
   - Altos intentos = fallos repetidos
   - Verificar registros para mensajes de error
   - Puede indicar formato inválido o rechazo del operador
4. **Intervención Manual** (si es necesario):
   - Contactar al operador para verificar el problema
   - Puede ser necesario cancelar y reenviar el mensaje
   - Consultar con el equipo de backend sobre problemas de cola

---

Passthrough de Recibos de Entrega (DLR)

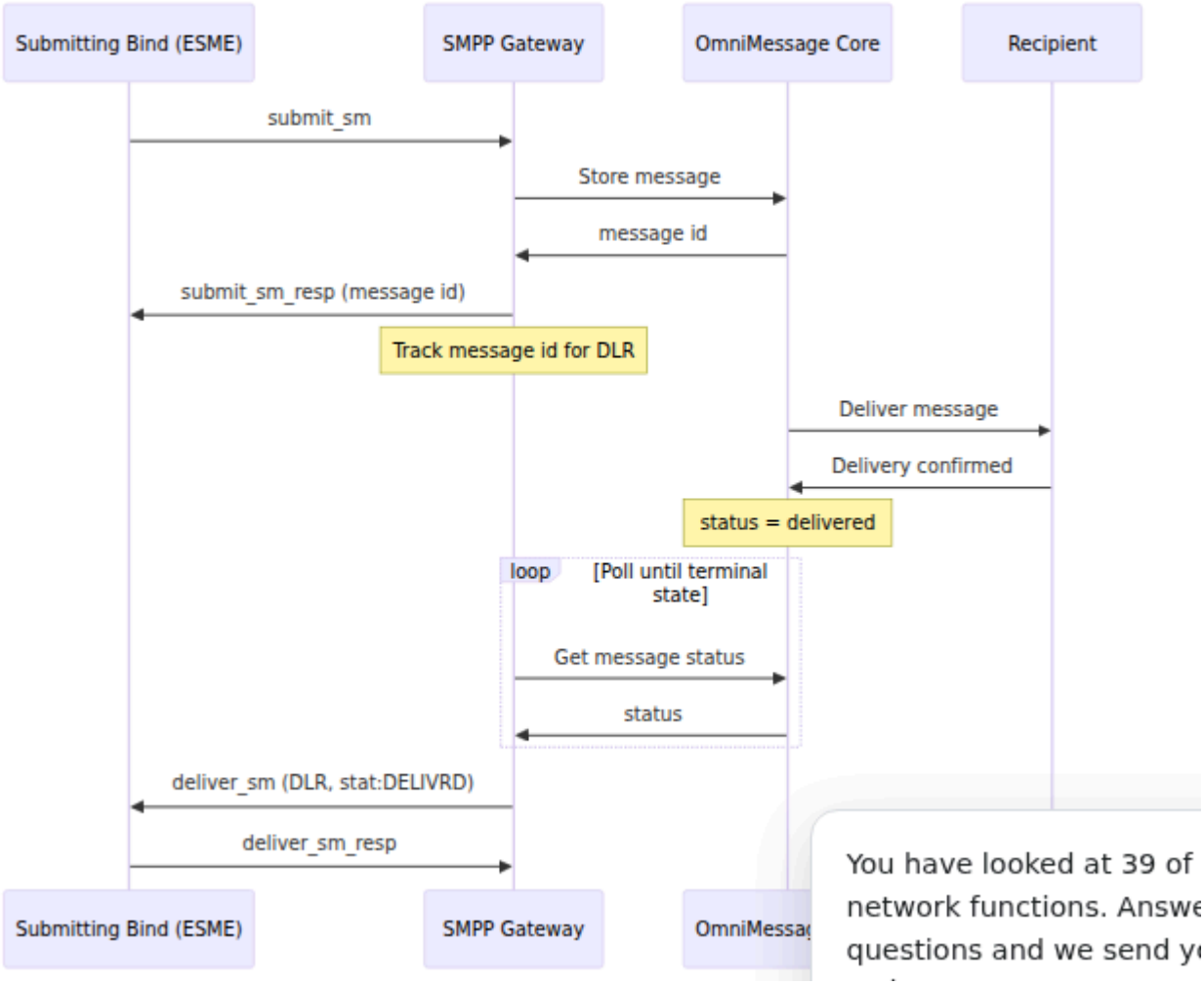
Un recibo de entrega (DLR) es un mensaje SMPP que informa el estado final de un mensaje enviado. El gateway envía un DLR de vuelta a un enlace de envío cuando el enlace establece `deliver_receipts: true`. Utilice esta función para un enlace interconectado cuyo operador necesita confirmación de entrega para cada mensaje.

El gateway envía el recibo como un PDU `deliver_sm` con el `esm_class` establecido en `0x04` (Recibo de Entrega del SMSC). El recibo lleva el mismo ID de mensaje que el gateway devolvió en el `submit_sm_resp`, por lo que el operador puede correlacionar el recibo con el envío original.

Requisitos

- Establecer `deliver_receipts: true` en el enlace del servidor. Ver CONFIGURATION.md.
- Establecer `cache_enabled: false` en el mismo enlace. El almacenamiento en caché devuelve un id de mensaje temporal que no coincide con el id del núcleo, por lo que un enlace en caché rompe la correlación de recibos. La puerta de enlace escribe una advertencia al inicio si un enlace establece ambos.

Flujo de mensajes



Valores de estado de recibo

El campo `stat` del recibo informa el estado final del mensaje.

Estado del núcleo	Recibo `stat`	Significado
`delivered`	`DELIVRD`	El destinatario recibió el mensaje.
`expired`	`EXPIRED`	El período de validez del mensaje terminó antes de la entrega.
`dropped`	`UNDELIV`	La puerta de enlace no pudo entregar el mensaje.
`rejected`	`REJECTD`	El núcleo rechazó el mensaje.
`balance_rejected`	`REJECTD`	El núcleo rechazó el mensaje por una razón de saldo de cuenta.

Notas operativas

- La puerta de enlace rastrea los mensajes en vuelo en memoria. Si el servicio se reinicia, se pierden los recibos de los mensajes que están en vuelo en ese momento. Este es un mecanismo de mejor esfuerzo.
- Si la sesión de enlace está caída cuando un mensaje alcanza un estado terminal, la puerta de enlace mantiene el recibo y lo envía cuando la sesión se reconecta. La puerta de enlace descarta el recibo después de 24 horas.
- Monitoree las métricas de DLR para confirmar que la función funciona. Ver MONITORING.md.

Solución de problemas de conexión

Reconectar un enlace

Síntomas: La conexión muestra "Desconectado" (rojo)

Pasos:

1. **Verificar la conectividad de red**:

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

2. **Verificar registros en busca de errores**:

   - Navegar a: Pestaña de registros
   - Filtrar: Nivel de error
   - Buscar fallos de autenticación, tiempos de espera de red

3. **Verificar credenciales**:

   - Navegar a: SMPP → Clientes/Servidores
   - Comprobar que system_id y contraseña son correctos
   - Contactar al operador si no está seguro

4. **Reconexión manual**:

   - Navegar a: SMPP → Estado en vivo
   - Encontrar el enlace desconectado
   - Hacer clic en el botón "Reconectar"
   - Esperar 10-30 segundos
   - Verificar si el estado cambia a "Conectado"

- La reconexión también recupera un enlace que fue desconectado con "Drop"

5. **Si la reconexión falla**:
   - Verificar reglas del firewall
   - Verificar que el servidor del operador esté operativo
   - Contactar al soporte del operador
   - Ver TROUBLESHOOTING.md

Manejo de fallos de autenticación

Síntomas: Fallos de enlace repetidos en los registros

Causas:

- Nombre de usuario/contraseña incorrectos
- IP no incluida en la lista blanca en el operador
- Cuenta suspendida/expirada

Pasos:

1. **Verificar credenciales**:
   - Navegar a: SMPP → Clientes
   - Verificar nuevamente system_id y contraseña
   - Confirmar con el operador
2. **Verificar la lista blanca de IP**:
   - Confirmar su IP de puerta de enlace con el operador
   - Solicitar al operador que verifique la lista blanca de IP
3. **Verificar el estado de la cuenta**:
   - Verificar que la cuenta esté activa
   - Comprobar si hay contratos expirados
   - Contactar al departamento de facturación del operador

4. **Actualizar configuración**:
    - Si las credenciales cambiaron, actualizarlas en la interfaz web
    - Hacer clic en "Reconectar" para intentar con las nuevas credenciales

---

Monitoreo y alertas

Verificación de métricas de Prometheus

Verificación rápida:

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

Salida esperada:

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

Todos los valores deben ser `1` (conectado).

Respondiendo a alertas

Alerta de conexión caída:

1. Verificar Interfaz Web → SMPP → Estado en vivo
2. Intentar reconexión manual
3. Verificar registros en busca de errores
4. Contactar al operador si la interrupción es prolongada
5. Ver TROUBLESHOOTING.md

Alerta de alta tasa de fallos:

1. Verificar registros en busca de patrones de error
2. Revisar cambios recientes en la configuración

3. Contactar al operador sobre rechazos
4. Verificar cumplimiento del formato del mensaje

Alerta de sin tráfico:

1. Verificar que la cola de backend tenga mensajes
2. Verificar que el enrutamiento `dest_smsc` sea correcto
3. Verificar que los límites de TPS no sean demasiado restrictivos
4. Revisar la configuración de `queue_check_frequency`

Procedimientos de mantenimiento

Mantenimiento rutinario

Realizar mensualmente:

1. **Revisar métricas**:
    - Analizar tendencias de volumen de mensajes
    - Comprobar tasas de éxito de entrega
    - Identificar oportunidades de optimización
2. **Actualizar documentación**:
    - Documentar cualquier cambio de configuración
    - Actualizar información de contacto
    - Anotar ventanas de mantenimiento del operador
3. **Auditoría de credenciales**:
    - Revisar todas las contraseñas de SMPP
    - Planificar la rotación de credenciales
    - Verificar que las listas blancas de IP estén actualizadas
4. **Planificación de capacidad**:
    - Revisar tasas de mensajes pico

- Comprobar contra límites de TPS
- Planificar para el crecimiento

Reinicio del servicio

Cuando sea necesario:

- Después de cambios en el archivo de configuración
- Después de actualizaciones del sistema
- Durante la solución de problemas

Pasos:

```
# Verificar estado actual
sudo systemctl status omnimessage-smpp

# Reiniciar servicio
sudo systemctl restart omnimessage-smpp

# Verificar reinicio
sudo systemctl status omnimessage-smpp

# Verificar registros
sudo journalctl -u omnimessage-smpp -n 50
```

Verificar a través de la Interfaz Web:

1. Acceder al panel de control (puede tardar de 30 a 60 segundos en estar en línea)
2. Navegar a: SMPP → Estado en vivo
3. Esperar a que todas las conexiones se establezcan (1-2 minutos)
4. Verificar registros en busca de errores

Copia de seguridad de configuración

Hacer copia de seguridad de archivos críticos antes de realizar cambios:

```
# Copia de seguridad de configuración
sudo cp /opt/omnimessage-smpp/config/runtime.exs \
  /opt/omnimessage-smpp/config/runtime.exs.backup.$(date +%Y%m%d)

# Copia de seguridad de certificados
sudo tar -czf /tmp/smpp-certs-$(date +%Y%m%d).tar.gz \
  /opt/omnimessage-smpp/priv/cert/
```

Restaurar si es necesario:

```
# Restaurar configuración
sudo cp /opt/omnimessage-smpp/config/runtime.exs.backup.YYYYMMDD \
  /opt/omnimessage-smpp/config/runtime.exs

# Reiniciar servicio
sudo systemctl restart omnimessage-smpp
```

Procedimientos de emergencia

Corte total del servicio

Pasos:

1. **Verificar estado del servicio**:

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

2. **Si el servicio está detenido, iniciarlo**:

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

3. **Verificar registros en busca de razones de caída**:

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

4. **Si no se inicia**:
   - Verificar errores de sintaxis en la configuración
   - Verificar que existan certificados SSL
   - Verificar espacio en disco: `df -h`
   - Verificar memoria: `free -h`
5. **Contactar soporte** si no se resuelve

Solicitudes de desconexión de emergencia del operador

Pasos:

1. **Caer la conexión inmediatamente**:
   - Navegar a: SMPP → Estado en vivo
   - Encontrar la conexión afectada
   - Hacer clic en "Caer conexión"
2. **Documentar razón**:
   - Anotar nombre del operador
   - Registrar hora y razón
   - Guardar correspondencia
3. **Investigar el problema**:
   - Verificar patrones recientes de mensajes
   - Revisar registros en busca de errores
   - Identificar la causa raíz
4. **Coordinar resolución**:
   - Trabajar con el operador
   - Implementar correcciones

- Probar antes de reconectar

Pico de volumen alto

Síntomas: Tráfico de mensajes inesperadamente alto

Pasos:

1. **Verificar límites de TPS**:
   - Navegar a: SMPP → Estado en vivo
   - Verificar que las conexiones no estén limitadas
   - Puede ser necesario aumentar temporalmente los límites de TPS
2. **Monitorear estabilidad del operador**:
   - Observar desconexiones
   - Verificar tasas de éxito de entrega
3. **Coordinar con el backend**:
   - Verificar que la fuente de mensajes sea legítima
   - Puede ser necesario implementar limitación de tasa aguas arriba
4. **Escalar si es necesario**:
   - Puede ser necesario agregar instancias adicionales de la puerta de enlace
   - Contactar soporte para asesoramiento sobre escalado

Mejores prácticas

Lista de verificación diaria

- [ ] Verificar que todas las conexiones SMPP estén conectadas
- [ ] Revisar registros de errores en busca de problemas
- [ ] Monitorear la cola de mensajes para acumulaciones

- [ ] Verificar paneles de Prometheus/Grafana
- [ ] Verificar tasas de éxito de entrega > 98%

Tareas semanales

- [ ] Revisar tendencias de métricas
- [ ] Verificar anomalías en patrones
- [ ] Probar procedimientos de recuperación ante desastres
- [ ] Actualizar documentación según sea necesario
- [ ] Revisar y reconocer alertas

Tareas mensuales

- [ ] Auditoría de credenciales
- [ ] Revisión de planificación de capacidad
- [ ] Actualizar contactos del operador
- [ ] Revisar y optimizar configuraciones de TPS
- [ ] Copia de seguridad de archivos de configuración

Documentación relacionada

- **CONFIGURATION.md** - Configurar conexiones y ajustes
- **SOURCE_ADDRESS_WHITELIST.md** - Restringir direcciones de origen por par de servidor
- **MONITORING.md** - Configurar alertas de Prometheus
- **TROUBLESHOOTING.md** - Resolver problemas comunes
- **README.md** - Visión general del sistema
