



SMSc - Aplicativo do Centro de Serviço de SMS

Aplicativo do Centro de SMS construído com Phoenix/Elixir, fornecendo uma fila de mensagens centralizada e uma API REST para roteamento e entrega de SMS.

Documentação

Documentação Principal

- [Índice Completo da Documentação](#) - Comece aqui para toda a documentação
- [Referência de Configuração](#) - Opções de configuração completas
- [Referência da API](#) - Documentação da API REST
- [Guia de Operações](#) - Operações e monitoramento do dia a dia
- [Guia de Roteamento de SMS](#) - Gerenciamento e configuração de rotas
- [Guia de Tradução de Números](#) - Normalização e reescrita de números
- [Ajuste de Desempenho](#) - Otimização para diferentes cargas de trabalho
- [Guia de Métricas](#) - Métricas e monitoramento do Prometheus
- [Guia de Solução de Problemas](#) - Problemas comuns e soluções
- [Esquema CDR](#) - Formato de registro de detalhes de chamadas

Documentação de Conformidade

- [Conformidade de Interceptação ANSSI R226](#) - Especificações técnicas de interceptação legal na França

Desempenho

- [Benchmarks](#) - Testes de desempenho

Visão Geral da Arquitetura

O núcleo do SMS_C fornece uma fila de mensagens independente de protocolo e uma API REST. Frontends externos de SMSC (SMPP, IMS, SS7/MAP) se conectam como gateways independentes que se comunicam com o núcleo via API REST.

Fluxo de Mensagens

Fluxo de Mensagens de Saída (MT - Mobile Terminated)

Fluxo de Mensagens de Entrada (MO - Mobile Originated)

Principais Recursos

1. Design Independente de Protocolo

- O núcleo SMS_C lida com persistência de mensagens, roteamento e API
- Frontends externos (SMPP, IMS, SS7/MAP) lidam com comunicação específica do protocolo
- Todos os frontends se comunicam via API REST unificada
- Adicione novos protocolos sem alterar o núcleo
- Cada um pode ser escalado independentemente

2. Roteamento de Mensagens

- **Motor de roteamento dinâmico** com configuração em tempo de execução
- Roteamento baseado em prefixo (números chamadores/chamados)
- Filtragem de SMSC e tipo de origem (IMS/Circuit Switched/SMPP)
- Roteamento baseado em prioridade com balanceamento de carga baseado em peso
- Capacidade de roteamento de resposta automática e descarte de mensagens
- Controle de cobrança por rota
- UI Web para gerenciamento de rotas
- Atualizações de rotas em tempo real sem interrupção do serviço

◇ Veja [Guia de Roteamento de SMS](#) para documentação abrangente

3. Lógica de Retentativa com Backoff Exponencial

- Retentativa automática em caso de falha na entrega
- Backoff exponencial: 1min, 2min, 4min, 8min, etc.
- Tentativas máximas de retentativa configuráveis
- Tratamento de expiração de mensagens
- Rastreamento de retentativa por mensagem

Guia de Operações

Pontos de Acesso:

- API REST: <https://localhost:8443> (ou <http://localhost:8080> sem

- TLS)
- Painel de Controle: <https://localhost:8086>
- Documentação da API (Swagger UI): <https://localhost:8443/api/docs>

Iniciar Frontends Externos: Cada frontend de protocolo é um aplicativo independente. Veja a documentação individual do frontend para instruções de inicialização.

Configuração

Toda a configuração é gerenciada diretamente em `config/runtime.exs`. Nenhuma variável de ambiente é utilizada.

Configuração do Núcleo

Nenhuma variável de ambiente de configuração do aplicativo núcleo está atualmente em uso. As portas do servidor e os nomes dos hosts são configurados em `config/runtime.exs`:

- Servidor API: Porta 8443 (HTTPS), ouvindo em 0.0.0.0
- Painel de Controle: Porta 80 (HTTP), ouvindo em 0.0.0.0

Configuração do Banco de Dados

As configurações do banco de dados são configuradas em `config/runtime.exs`:

- Nome de usuário: `omnitouch`
- Senha: `omnitouch2024`
- Nome do host: `localhost`
- Porta: `3306`
- Nome do banco de dados: `smcsc_new`
- Tamanho do pool: `1`

Configuração do Cluster

As configurações do cluster são configuradas em `config/runtime.exs`:

- Nós do cluster: `""` (vazio - sem clustering por padrão)
- Consulta de cluster DNS: `nil`

Configuração da Fila de Mensagens

As configurações da fila de mensagens são configuradas em `config/runtime.exs`:

- Tempo de carta morta: `1440` minutos (24 horas antes da expiração da mensagem)

Integração de Cobrança

As configurações de cobrança são configuradas em `config/runtime.exs`:

- URL: `http://localhost:2080/jsonrpc`
- Inquilino: `mnc057.mcc505.3gppnetwork.org`
- Destino: `55512341234`
- Origem: `00101900000257`
- Assunto: `00101900000257`
- Conta: `00101900000257`

Configuração de Roteamento de SMS

O sistema de roteamento de SMS utiliza rotas dinâmicas, apoiadas por banco de dados, que podem ser gerenciadas via UI Web ou arquivo de configuração. As rotas são carregadas de `config/runtime.exs` na primeira inicialização.

Exemplo de Configuração:

```
config :sms_c, :sms_routes, [  
  %{  
    called_prefix: "+44",  
    dest_smsc: "InternationalGW",  
    weight: 100,  
    priority: 100,  
    description: "SMS Internacional do Reino Unido",  
    enabled: true  
  },  
  %{  
    called_prefix: "1900",  
    dest_smsc: "PremiumGW",  
    charged: :yes,  
    priority: 50,  
    description: "Números Premium dos EUA",  
    enabled: true  
  }  
]
```

Recursos:

- Correspondência baseada em prefixo (números chamadores/chamados)
- Filtragem de SMSC de origem e tipo
- Roteamento baseado em prioridade e peso
- Capacidade de resposta automática e descarte
- Controle de cobrança por rota
- Gerenciamento em tempo de execução via UI Web em `/routing`

❖ Veja [Guia de Roteamento de SMS](#) para documentação completa,

exemplos e referência da API.

Endpoints da API REST

Operações na Fila de Mensagens

Enviar SMS (Criar Mensagem)

POST /api/messages

Content-Type: application/json

```
{
  "source_msisdn": "+1234567890",
  "destination_msisdn": "+0987654321",
  "message_body": "Olá, Mundo!",
  "source_smsc": "web-app",
  "dest_smsc": "smpp-provider",          # Opcional - motor de
roteamento atribui se nulo
  "tp_dcs_character_set": "gsm7",        # Opcional: gsm7, 8bit,
latin1, ucs2
  "tp_dcs_coding_group": "general_data_coding",
  "expires": "2025-10-17T10:30:00Z"     # Opcional - padrão para 24h
a partir de agora
}
```

Campos Obrigatórios:

- destination_msisdn - Número de telefone de destino
- message_body - Conteúdo do texto da mensagem
- source_msisdn - Número de telefone de origem
- source_smsc - Identificador do sistema de origem

Campos Opcionais:

- dest_smsc - Gateway de destino (motor de roteamento atribui se não fornecido)
- source_imsi, dest_imsi - Identificadores IMSI
- tp_dcs_character_set - Codificação de caracteres (gsm7, 8bit, latin1, ucs2)
- tp_dcs_coding_group - Grupo de codificação DCS
- tp_dcs_compressed - Sinalizador de compressão (booleano)
- tp_dcs_has_message_class - Sinalizador de classe de mensagem (booleano)
- tp_dcs_message_class - Valor da classe de mensagem
- tp_user_data_header - Cabeçalho de dados do usuário (mapa)
- message_part_number, message_parts - Campos de mensagem multipart
- expires - Timestamp de expiração (padrão para 24 horas)

- deliver_after - Timestamp de entrega atrasada
- deadletter, raw_data_flag, raw_sip_flag - Sinalizadores booleanos

Resposta:

```
{
  "status": "success",
  "data": {
    "id": 123,
    "source_msisdn": "+1234567890",
    "destination_msisdn": "+0987654321",
    "dest_smsc": "smpp-provider",
    "message_body": "Olá, Mundo!",
    "deliver_time": null,
    "delivery_attempts": 0,
    "expires": "2025-10-17T10:30:00Z",
    "inserted_at": "2025-10-16T10:30:00Z"
  }
}
```

Obter Mensagens para SMSC

GET /api/messages/get_by_smsc?smsc=meu-nome-smsc

Retorna todas as mensagens não entregues onde:

- destination_smsc é nulo OU corresponde ao nome SMSC fornecido
- A mensagem não está expirada
- Pronta para enviar (deliver_after é nulo ou no passado)

Resposta:

```
{
  "status": "success",
  "data": [
    {
      "id": 123,
      "source_msisdn": "+1234567890",
      "destination_msisdn": "+0987654321",
      "message_body": "Olá",
      "destination_smsc": "meu-nome-smsc",
      "delivery_attempts": 0
    }
  ]
}
```

Listar Mensagens com Filtragem Opcional de SMSC

```
# Listar todas as mensagens na fila
GET /api/messages
```

```
# Listar mensagens para SMSC específico (com filtragem de cabeçalho)
GET /api/messages
SMSC: meu-nome-smsc
```

Sem Cabeçalho SMSC: Retorna todas as mensagens na fila, independentemente do status de entrega ou expiração.

Com Cabeçalho SMSC: Retorna mensagens não entregues onde:

- `dest_smsc` corresponde ao valor do cabeçalho OU `dest_smsc` é nulo
- `deliver_time` é nulo (não entregue ainda)
- `deliver_after` é nulo ou antes/igual ao tempo atual (pronto para entrega)
- `expires` é após o tempo atual (não expirada)
- Ordenado pelo tempo de inserção (mais antigo primeiro)

Nota: A abordagem do cabeçalho SMSC permite que frontends externos consultem suas mensagens usando o mesmo padrão de endpoint, com o cabeçalho controlando o comportamento de filtragem.

Resposta:

```
[
  {
    "id": 123,
    "source_msisdn": "+1234567890",
    "destination_msisdn": "+0987654321",
    "message_body": "Olá, Mundo!",
    "dest_smsc": "meu-nome-smsc",
    "deliver_time": null,
    "delivery_attempts": 0,
    "expires": "2025-10-17T10:30:00Z",
    "inserted_at": "2025-10-16T10:30:00Z"
  }
]
```

Obter Mensagem Única

```
GET /api/messages/{id}
```

Atualizar Mensagem

```
PATCH /api/messages/{id}
Content-Type: application/json
```

```
{
  "status": "entregue",
  "delivered_at": "2025-10-16T10:30:00Z"
}
```

Deletar SMS

```
DELETE /api/messages/{id}
```

Lidar com Falha na Entrega (Incrementar Contador de Retentativas)

Quando a entrega de uma mensagem falha temporariamente, incremente o contador de tentativas de entrega e agende uma retentativa com backoff exponencial.

Método 1: Usando PUT (Recomendado)

```
# Simples e semântico - PUT indica atualização do estado de entrega
PUT /api/messages/{id}
```

Método 2: Usando Endpoint Explícito

```
# Endpoint alternativo explícito
POST /api/messages/{id}/increment_delivery_attempt
```

Ambos os métodos incrementam `delivery_attempts` e definem o atraso de backoff exponencial via `deliver_after`:

Tentativa Fórmula de Backoff Atraso Tempo Total

1ª	2^1 minutos	2 min	2 min
2ª	2^2 minutos	4 min	6 min
3ª	2^3 minutos	8 min	14 min
4ª	2^4 minutos	16 min	30 min
5ª	2^5 minutos	32 min	1h 2min
6ª	2^6 minutos	64 min	2h 6min

Resposta:

```
{
  "id": 123,
  "delivery_attempts": 1,
  "deliver_after": "2025-10-20T19:05:00Z",
  "deliver_time": null,
  "expires": "2025-10-21T19:03:00Z",
  ...
}
```


Nota: Mensagens com `deliver_after` futuro são automaticamente filtradas de solicitações GET até que o período de backoff expire.

Atualizar Mensagem (Atualização Parcial)

Para atualizar campos específicos da mensagem (comportamento inalterado):

```
PATCH /api/messages/{id}
Content-Type: application/json

{
  "dest_smsc": "gateway-atualizado",
  "status": "entregue"
}
```

Importante: PUT e PATCH se comportam de maneira diferente:

- **PUT** → Incrementa tentativas de entrega com backoff (sem corpo necessário)
- **PATCH** → Realiza atualizações parciais de campo (corpo necessário)

Monitoramento de Saúde do Frontend

O núcleo SMS_C rastreia a saúde e a disponibilidade dos frontends externos por meio de um sistema de registro. Isso permite monitorar o tempo de atividade do frontend, detectar falhas e manter dados históricos de disponibilidade.

Nota: O registro do frontend NÃO é usado para entrega ou roteamento de mensagens. As mensagens são roteadas com base no campo `dest_smsc`. O sistema de registro existe apenas para monitoramento de saúde e visibilidade operacional.

Como Funciona o Registro do Frontend

Cada frontend externo (gateway SMPP, IMS, SS7/MAP) envia periodicamente um registro de heartbeat para o núcleo SMS_C:

1. **Intervalo de Heartbeat:** Frontends devem registrar a cada 30-60 segundos
2. **Tempo de Expiração:** Registros expiram após 90 segundos sem uma atualização
3. **Gerenciamento Automático de Estado:**
 - Novos frontends criam um novo registro de registro
 - Frontends ativos existentes atualizam seu registro (estendem a expiração)
 - Frontends expirados que voltam online criam um novo período de registro

Endpoints de Registro do Frontend

Registrar/Atualizar Frontend (Heartbeat)

POST /api/frontends

Content-Type: application/json

```
{
  "frontend_name": "gateway-smpp-1",
  "frontend_type": "SMPP",
  "ip_address": "10.0.1.5",
  "hostname": "smpp-gw-01",
  "uptime_seconds": 3600,
  "configuration": "{\"port\": 2775, \"system_id\": \"usuario_SMPP\"}"
}
```

Campos Obrigatórios:

- frontend_name - Identificador único para a instância do frontend
- frontend_type - Tipo de frontend (SMPP, IMS, MAP, etc.)

Campos Opcionais:

- ip_address - Endereço IP do frontend (detectado automaticamente a partir da origem da solicitação se não fornecido)
- hostname - Nome do host do servidor frontend
- uptime_seconds - Segundos desde que o frontend foi iniciado
- configuration - String JSON com configuração específica do frontend

Nota: Se ip_address não for fornecido, o núcleo SMS_C usará automaticamente o IP de origem da solicitação HTTP. Isso funciona com conexões diretas e solicitações proxy (via cabeçalho X-Forwarded-For).

Resposta:

```
{
  "id": 42,
  "frontend_name": "gateway-smpp-1",
  "frontend_type": "SMPP",
  "ip_address": "10.0.1.5",
  "hostname": "smpp-gw-01",
  "uptime_seconds": 3600,
  "status": "ativo",
  "last_seen_at": "2025-10-20T10:30:00Z",
  "expires_at": "2025-10-20T10:31:30Z",
  "inserted_at": "2025-10-20T10:00:00Z"
}
```

Listar Todos os Registros de Frontend

```
GET /api/frontends
```

Retorna todos os registros de frontend (ativos e expirados), ordenados pela atividade mais recente.

Listar Apenas Frontends Ativos

```
GET /api/frontends/active
```

Retorna apenas frontends atualmente ativos (não expirados).

Obter Estatísticas do Frontend

```
GET /api/frontends/stats
```

Retorna estatísticas resumidas:

```
{
  "active": 5,
  "expired": 12,
  "unique_frontends": 8
}
```

Obter Histórico do Frontend

```
GET /api/frontends/history/{frontend_name}
```

Retorna todos os registros históricos para um frontend específico, útil para analisar padrões de tempo de atividade/inatividade.

Exemplo:

```
GET /api/frontends/history/gateway-smpp-1
```

Obter Registro Específico

```
GET /api/frontends/{id}
```

Implementação em Frontends Externos

Frontends externos devem implementar uma tarefa em segundo plano que envia heartbeats:

Exemplo (pseudocódigo):

```
import time
```

```

import requests

def send_heartbeat():
    """Enviar heartbeat a cada 30 segundos"""
    while True:
        try:
            data = {
                "frontend_name": "meu-gateway-sMPP",
                "frontend_type": "SMPP",
                "ip_address": get_local_ip(),
                "hostname": get_hostname(),
                "uptime_seconds": get_uptime()
            }

            response = requests.post(
                "https://smc-core:8443/api/frontends",
                json=data,
                timeout=5
            )

            if response.status_code in [200, 201]:
                logger.debug("Heartbeat enviado com sucesso")
            else:
                logger.error(f"Falha no heartbeat: {response.status_code}")

        except Exception as e:
            logger.error(f"Erro no heartbeat: {e}")

        time.sleep(30)  # Enviar a cada 30 segundos

# Iniciar heartbeat em thread de fundo
threading.Thread(target=send_heartbeat, daemon=True).start()

```

Monitoramento da Saúde do Frontend

Painel de Controle - A UI web em <https://localhost:8086> mostra:

- Frontends atualmente ativos
- Timestamp da última visualização para cada frontend
- Rastreamento de tempo de atividade
- Disponibilidade histórica

Consultas da API:

```

# Obter todos os frontends ativos
curl https://localhost:8443/api/frontends/active

```

```
# Verificar se um frontend específico está ativo
curl https://localhost:8443/api/frontends/history/gateway-sMPP-1 | jq
'.[0].status'

# Obter estatísticas de saúde
curl https://localhost:8443/api/frontends/stats
```

Outros Endpoints

Status

```
GET /api/status
```

Localizações

```
GET /api/locations
POST /api/locations
GET /api/locations/{id}
PATCH /api/locations/{id}
DELETE /api/locations/{id}
```

Eventos SS7

```
GET /api/ss7_events
POST /api/ss7_events
GET /api/ss7_events/{id}
PATCH /api/ss7_events/{id}
DELETE /api/ss7_events/{id}
```

Fila de Mensagens MMS

```
GET /api/mms_message_queues
POST /api/mms_message_queues
GET /api/mms_message_queues/{id}
PATCH /api/mms_message_queues/{id}
DELETE /api/mms_message_queues/{id}
```

Desempenho

O núcleo SMS_C oferece um desempenho excepcional usando Mnesia para armazenamento de mensagens em memória com arquivamento automático para SQL para retenção de CDR a longo prazo.

Resultados de Benchmark

Medido em Intel i7-8650U @ 1.90GHz (8 núcleos):

Desempenho de Inserção de Mensagens:

- `insert_message` (com roteamento): **1.750 msg/sec** (0,58ms de latência média)
- `insert_message` (simples): **1.750 msg/sec** (0,57ms de latência média)
- **Capacidade de ~150 milhões de mensagens por dia**

Desempenho de Consulta:

- `get_messages_for_smsc`: 800 msg/sec (1,25ms em média)
- `list_message_queues`: Acesso rápido em memória
- Uso de memória: 62 KB por operação de inserção

Arquitetura

Estratégia de Armazenamento:

- **Mensagens Ativas**: Armazenadas em Mnesia (em memória + disco) para acesso ultra-rápido
- **Arquivo de Mensagens**: Arquivadas automaticamente para SQL para armazenamento de CDR a longo prazo
- **Retenção**: Período de retenção configurável (padrão: 24 horas)
- **Sem gargalo SQL**: Todas as operações de mensagens ativas ignoram SQL

Configuração

Armazenamento e retenção de mensagens configurados em `config/runtime.exs`:

```
config :sms_c,
  message_retention_hours: 24,           # Arquivar mensagens mais
  antigas que 24 horas
  batch_insert_batch_size: 100,         # Tamanho do lote de CDR para
  arquivamento SQL
  batch_insert_flush_interval_ms: 100   # Intervalo de flush de CDR
```

Para orientações detalhadas de ajuste, veja: [docs/PERFORMANCE_TUNING.md](#)

Monitoramento

Painel de Controle - UI Web em `https://localhost:8086`

- Visualizar fila de mensagens
- Enviar mensagens de teste
- Gerenciar roteamento de SMS (veja [Guia de Roteamento](#))
- Simular decisões de roteamento
- Visualizar recursos do sistema
- Rastrear estatísticas de trabalhadores em lote

Estatísticas do Trabalhador em Lote:

```
# Obter estatísticas atuais do trabalhador em lote
SmsC.Messaging.BatchInsertWorker.stats()
```

Retorna:

```
%{
  total_enqueued: 10000,
  total_flushed: 9900,
  current_queue_size: 100,
  last_flush_duration_ms: 45
}
```

Logs - Logs da aplicação escritos no stdout

```
# Visualizar logs em tempo real
tail -f log/dev.log
```

Solução de Problemas

Porta Já em Uso

```
# Encontrar processo usando a porta
lsof -i :4000

# Matar o processo
kill -9 <PID>
```

Frontend Externo Não Conectando

Sintomas: Mensagens presas na fila, logs do frontend mostram erros de conexão

Verifique:

- Verifique se `API_BASE_URL` está configurado corretamente no frontend
- Verifique se o núcleo `SMS_C` está em execução e acessível
- Revise as regras de rede/firewall
- Verifique a configuração do frontend

Solução:

```
# Testar conectividade da API a partir do frontend
curl http://localhost:4000/api/status

# Reiniciar frontend
export API_BASE_URL="http://localhost:4000"
# Iniciar aplicação frontend
```

Mensagens Não Sendo Entregues

Sintomas: Mensagens permanecem não entregues, tentativas de reententativa aumentando

Verifique:

1. Logs do frontend para erros de envio
2. Conectividade da rede externa
3. Configuração do frontend (credenciais, endereços)
4. Compatibilidade do formato da mensagem

Visualizar mensagens com falha:

```
# Obter mensagens com tentativas de reententativa
curl https://localhost:8443/api/messages | jq '.data[] |
select(.delivery_attempts > 0)'
```

Alta Latência de Mensagem

Sintomas: Mensagens levando mais tempo que o esperado, backlog na fila

Verifique:

1. Intervalo de polling do frontend (pode ser necessário diminuir para polling mais frequente)
2. Desempenho do banco de dados
3. Latência de rede para sistemas externos

Monitorar profundidade da fila:

```
watch -n 5 'curl -s https://localhost:8443/api/messages | jq ".data |
length"'
```




Benchmarks

Este diretório contém benchmarks de desempenho para o sistema SMS-C usando Benchee.

Benchmarks Disponíveis

1. Benchmark de SMS Bruto (`raw_sms_bench.exs`)

Benchmark do endpoint da API `submit_message_raw` usando PDUs de SMS reais.

Características:

- Usa PDUs de SMS reais (adicione seus PDUs à lista `@sample_pdus` no arquivo)
- Desabilita a detecção de duplicatas limpando as impressões digitais antes de cada iteração
- Gera relatórios tanto em console quanto em HTML

Uso:

```
mix run benchmarks/raw_sms_bench.exs
```

Saída: `benchmarks/output/raw_sms_benchmark.html`

2. Benchmark da API de Mensagens (`message_api_bench.exs`)

Benchmark de várias operações da API de mensagens, incluindo inserção, recuperação e roteamento.

Características:

- Testa `insert_message` (simples e com roteamento)
- Testa `get_messages_for_smsc`
- Testa `list_message_queues`
- Pré-popula o banco de dados com dados de teste para cenários realistas

Uso:

```
mix run benchmarks/message_api_bench.exs
```

Saída: `benchmarks/output/message_api_benchmark.html`

Configuração

Todos os benchmarks usam Benchee com as seguintes configurações padrão:

- Aquecimento: 2 segundos
- Tempo: 10 segundos
- Tempo de memória: 2 segundos
- Estatísticas estendidas habilitadas
- Relatórios HTML gerados automaticamente

Saídas

Relatórios de benchmark em HTML são gerados em benchmarks/output/ e incluem:

- Métricas de desempenho detalhadas
- Gráficos de comparação
- Estatísticas de uso de memória
- Análise estatística



Documentação de Operações SMS-C

[← Voltar para o README Principal](#)

Bem-vindo à documentação de operações do SMS-C. Este guia abrangente cobre todos os aspectos da configuração, operação, monitoramento e solução de problemas do sistema SMS-C.

Visão Geral da Documentação

Começando

- [Referência de Configuração](#) - Opções de configuração completas e exemplos

Operações Diárias

- [Guia de Operações](#) - Tarefas diárias, monitoramento e manutenção
- [Guia de Roteamento SMS](#) - Gerenciamento e configuração de rotas
- [Referência da API](#) - Documentação completa da API com exemplos

Desempenho & Monitoramento

- [Ajuste de Desempenho](#) - Otimização para diferentes cargas de trabalho
- [Guia de Métricas](#) - Métricas Prometheus e monitoramento

Solução de Problemas

- [Guia de Solução de Problemas](#) - Problemas comuns e soluções

Conformidade & Regulamentação

- [Conformidade de Interceptação ANSSI R226](#) - Especificações técnicas de interceptação legal francesas
 - Integração de frontend multi-protocolo (IMS/SIP, SMPP, SS7/MAP)
 - Interfaces de interceptação legal ETSI X1/X2/X3
 - Arquitetura de armazenamento de dois níveis Mnesia + SQL
 - Esquema CDR para consultas de interceptação legal
 - Capacidades de criptografia e criptoanálise

Links Rápidos

Tarefas Comuns

- [Submetendo uma Mensagem](#)
- [Criando uma Rota](#)
- [Verificando o Status da Mensagem](#)
- [Monitorando a Saúde do Sistema](#)
- [Lidando com Falhas de Entrega](#)

Exemplos de Configuração

- [Armazenamento & Retenção de Mensagens](#)
- [Configuração de Exportação CDR](#)
- [Controles de Privacidade](#)
- [Configuração de Alto Volume](#)
- [Roteamento Geográfico](#)
- [Balanceamento de Carga](#)
- [Configuração ENUM/NAPTR](#)
- [Cobrança OCS](#)
- [Tradução de Números](#)

Monitoramento & Alertas

- [Métricas Chave](#)
- [Alertas Recomendados](#)
- [Modelos de Dashboard](#)

Visão Geral da Arquitetura do Sistema

O SMS-C é uma plataforma de roteamento de mensagens distribuída e de alto desempenho com os seguintes componentes principais:

Componentes Principais

- **Armazenamento de Mensagens** - Armazenamento rápido baseado em Mnesia com retenção configurável e exportação CDR
- **Motor de Roteamento** - Regras de roteamento baseadas em Mnesia com correspondência de prefixo e balanceamento de carga
- **Tradução de Números** - Normalização de números baseada em regex com ordenação de prioridade
- **Integração de Cobrança** - Cobrança online OCS com políticas baseadas em rotas
- **Consulta ENUM** - Roteamento de números baseado em DNS com cache
- **Registro de Eventos** - Rastreamento do ciclo de vida da mensagem
- **Exportação CDR** - Exportação automática para banco de dados SQL para

faturamento/análise a longo prazo

Interfaces Externas

- **API REST** - Submissão e gerenciamento de mensagens (HTTPS)
- **Interface Web** - Gerenciamento de rotas, navegador de mensagens, monitoramento
- **Prometheus** - Exposição de métricas para monitoramento
- **OCS** - Integração de cobrança/faturamento
- **DNS** - Consultas ENUM/NAPTR para roteamento

Distribuição & HA

- **Cluster de Múltiplos Nós** - Processamento de mensagens distribuído
- **Replicação Mnesia** - Sincronização de rotas entre nós
- **Failover Automático** - Tratamento de falhas de nós
- **Balanceamento de Carga** - Distribuição de rotas ponderadas

Documentação Relacionada

- [Benchmarks de Desempenho](#) - Testes de desempenho e resultados
- [Referência do Esquema CDR](#) - Esquema completo do banco de dados CDR com exemplos SQL

Requisitos do Sistema

Requisitos Mínimos

- **CPU**: 2 núcleos
- **RAM**: 4 GB
- **Disco**: 50 GB (cresce com a retenção de mensagens)
- **OS**: Linux (recomendado), macOS (desenvolvimento)
- **Erlang/OTP**: 26.x ou posterior
- **Elixir**: 1.15.x ou posterior
- **Banco de Dados SQL**: MySQL 8.0+, MariaDB 10.5+ ou PostgreSQL 13+ (para armazenamento CDR)

Produção Recomendada

- **CPU**: 8+ núcleos
- **RAM**: 16+ GB
- **Disco**: 500+ GB SSD
- **Rede**: 1 Gbps+
- **Banco de Dados SQL**: Servidor dedicado com replicação (para armazenamento CDR)

Portas de Rede

- **80/443** - Interface Web (HTTP/HTTPS)
- **8443** - API (HTTPS)
- **4369** - Mapeador de Portas Erlang (clusterização)
- **9100-9200** - Distribuição Erlang (clusterização)
- **9568** - Métricas Prometheus

Suporte & Recursos

Logs

- **Logs da Aplicação:** /var/log/sms_c/ (produção) ou console (desenvolvimento)
- **Logs da Interface Web:** Visualizador de logs em tempo real em /logs
- **Logs de Eventos:** Rastreamento de eventos por mensagem via API

Diagnósticos

- **Verificação de Saúde:** GET /api/status
- **Métricas:** GET http://localhost:9568/metrics (formato Prometheus)
- **Status do Frontend:** Interface Web em /frontend_status
- **Fila de Mensagens:** Interface Web em /message_queue

Obtendo Ajuda

1. Verifique o [Guia de Solução de Problemas](#)
2. Revise os logs da aplicação
3. Verifique as métricas do Prometheus para anomalias
4. Use o simulador de roteamento para testar a lógica de roteamento
5. Examine os logs de eventos por mensagem

Informações da Versão

Esta documentação está atualizada até:

- **Última Atualização:** 2025-10-30
- **Versão SMS-C:** Última versão de desenvolvimento
- **Elixir Suportado:** 1.15.x - 1.17.x
- **Erlang/OTP Suportado:** 26.x - 27.x

Convenções da Documentação

Ao longo desta documentação:

- **Exemplos de configuração** mostram valores típicos; ajuste para o seu

ambiente

- **Exemplos de API** usam o formato de linha de comando curl
- **Endereços IP e domínios** são apenas exemplos; substitua pelos seus valores reais
- **Nomes de métricas** seguem as convenções de nomenclatura do Prometheus
- **Todos os timestamps** estão em UTC, a menos que especificado de outra forma

Início Rápido

1. **Configuração:** Configure via `config/runtime.exs` - veja [Referência de Configuração](#)
2. **Rotas Iniciais:** Crie regras de roteamento via Interface Web ou arquivo de configuração - veja [Guia de Roteamento SMS](#)
3. **Submeter Mensagem de Teste:** Use API ou Interface Web - veja [Referência da API](#)
4. **Monitorar:** Configure a coleta do Prometheus - veja [Guia de Métricas](#)

Feedback da Documentação

Esta documentação é mantida juntamente com a base de código do SMS-C. Para correções ou melhorias, atualize os arquivos markdown no diretório docs/.



Referência da API SMS-C

[← Voltar para o Índice da Documentação](#) | [README Principal](#)

Referência completa para todos os endpoints da API REST SMS-C com exemplos de requisições/respostas.

Índice

- [Visão Geral da API](#)
- [Autenticação](#)
- [Formatos Comuns de Resposta](#)
- [Endpoint de Status](#)
- [API de Fila de Mensagens](#)
- [API de PDU SMS Bruto](#)
- [API de Gerenciamento de Localização](#)
- [API de Registro de Frontend](#)
- [API de Registro de Eventos](#)
- [API de Mensagens MMS](#)
- [API de Eventos SS7](#)
- [Códigos de Erro](#)
- [Limitação de Taxa](#)
- [Melhores Práticas](#)

Visão Geral da API

A API REST SMS-C fornece acesso programático à submissão, roteamento e gerenciamento de mensagens.

URL Base

```
https://api.example.com:8443/api
```

Porta Padrão: 8443 (configurável)

Protocolo: HTTPS (TLS necessário em produção)

Tipo de Conteúdo

Todas as requisições e respostas usam JSON:

```
Content-Type: application/json
```


Versionamento da API

A API atual é a versão 1 (implícita). Versões futuras usarão versionamento de URL:

```
https://api.example.com:8443/api/v2/...
```

Autenticação

Certificados de Cliente TLS (Recomendado)

Implantações em produção devem usar autenticação de certificado de cliente TLS:

```
curl --cert client.crt --key client.key \  
https://api.example.com:8443/api/status
```

Autenticação por Chave de API

Autenticação personalizada por chave de API via cabeçalho X-API-Key:

```
curl -H "X-API-Key: your_api_key_here" \  
https://api.example.com:8443/api/status
```

Lista de IPs Permitidos

Restringir o acesso à API a endereços IP confiáveis no nível do firewall.

Formatos Comuns de Resposta

Resposta de Sucesso

```
{  
  "data": {  
    ...  
  }  
}
```

Resposta de Erro

```
{  
  "errors": {  
    "detail": "Mensagem de erro descrevendo o que deu errado"  
  }  
}
```

Resposta de Lista

```
{
  "data": [
    {...},
    {...}
  ]
}
```

Endpoint de Status

Endpoint de verificação de saúde para monitoramento e balanceadores de carga.

Obter Status da API

Requisição:

```
GET /api/status
```

Resposta (200 OK):

```
{
  "status": "ok",
  "application": "OmniMessage",
  "timestamp": "2025-10-30T12:34:56Z"
}
```

Exemplo:

```
curl https://api.example.com:8443/api/status
```

Casos de Uso:

- Verificações de saúde do balanceador de carga
- Monitoramento da conectividade do sistema
- Verificação da disponibilidade do serviço

API de Fila de Mensagens

Endpoints principais de submissão e gerenciamento de mensagens.

Listar Mensagens

Recuperar mensagens da fila.

Requisição:

GET /api/messages

Cabeçalhos Opcionais:

- `smc: frontend_name` - Filtrar por SMSC de destino
- `include-unrouted: true|false|1|0` - Incluir mensagens sem registro de localização (padrão: false)
 - `false` (padrão): Retornar apenas mensagens com roteamento explícito ou registro de localização
 - `true`: Incluir mensagens sem registro de localização (modo compatível com versões anteriores)

Parâmetros de Consulta:

- `status` - Filtrar por status: pending, delivered, expired, dropped
- `source_smc` - Filtrar por SMSC de origem
- `dest_smc` - Filtrar por SMSC de destino
- `limit` - Limitar resultados (padrão: 100, máximo: 1000)
- `offset` - Deslocamento de paginação

Resposta (200 OK):

```
{
  "data": [
    {
      "id": 12345,
      "source_msisdn": "+15551234567",
      "destination_msisdn": "+447700900000",
      "message_body": "Hello World",
      "source_smc": "api_client",
      "dest_smc": "uk_gateway",
      "status": "pending",
      "send_time": "2025-10-30T12:00:00Z",
      "deliver_time": null,
      "delivery_attempts": 0,
      "inserted_at": "2025-10-30T12:00:00Z"
    }
  ]
}
```

Exemplos:

Obter mensagens pendentes para SMSC específica (apenas com roteamento explícito ou localização):

```
curl -H "smc: uk_gateway" \
https://api.example.com:8443/api/messages
```

Obter mensagens pendentes incluindo mensagens não roteadas (compatível com

versões anteriores):

```
curl -H "smc: uk_gateway" \  
  -H "include-unrouted: true" \  
  https://api.example.com:8443/api/messages
```

Obter todas as mensagens entregues:

```
curl "https://api.example.com:8443/api/  
messages?status=delivered&limit=50"
```

Obter Mensagem Única

Recuperar detalhes de uma mensagem específica.

Requisição:

```
GET /api/messages/:id
```

Resposta (200 OK):

```
{  
  "data": {  
    "id": 12345,  
    "source_msisdn": "+15551234567",  
    "destination_msisdn": "+447700900000",  
    "message_body": "Hello World",  
    "source_smc": "api_client",  
    "dest_smc": "uk_gateway",  
    "source_imsi": null,  
    "dest_imsi": null,  
    "message_parts": 1,  
    "message_part_number": 1,  
    "tp_data_coding_scheme": "00",  
    "tp_user_data_header": null,  
    "status": "pending",  
    "send_time": "2025-10-30T12:00:00Z",  
    "deliver_time": null,  
    "expires": "2025-10-31T12:00:00Z",  
    "deadletter": false,  
    "delivery_attempts": 0,  
    "charge_failed": false,  
    "deliver_after": "2025-10-30T12:00:00Z",  
    "raw_data_flag": false,  
    "raw_sip_flag": false,  
    "raw_pdu": null,  
    "inserted_at": "2025-10-30T12:00:00Z",  
    "updated_at": "2025-10-30T12:00:00Z"  
  }  
}
```

```
}
```

Exemplo:

```
curl https://api.example.com:8443/api/messages/12345
```

Submeter Mensagem (Síncrona)

Submeter uma mensagem e receber o ID da mensagem imediatamente.

Requisição:

```
POST /api/messages
Content-Type: application/json
```

Corpo:

```
{
  "source_msisdn": "+15551234567",
  "destination_msisdn": "+447700900000",
  "message_body": "Hello World",
  "source_smsc": "api_client"
}
```

Campos Opcionais:

- dest_smsc - Sobrescrever decisão de roteamento
- send_time - Agendar para entrega futura (ISO 8601)
- message_parts - Total de partes para mensagem multipart
- message_part_number - Número da parte (indexado a partir de 1)
- tp_data_coding_scheme - DCS SMS (padrão: "00")
- source_imsi - IMSI do assinante de origem
- dest_imsi - IMSI do assinante de destino

Resposta (201 Criado):

```
{
  "data": {
    "id": 12345,
    "source_msisdn": "+15551234567",
    "destination_msisdn": "+447700900000",
    "message_body": "Hello World",
    "source_smsc": "api_client",
    "dest_smsc": "uk_gateway",
    "status": "pending",
    "send_time": "2025-10-30T12:00:00Z",
    "inserted_at": "2025-10-30T12:00:00Z"
  }
}
```

Exemplo:

```
curl -X POST https://api.example.com:8443/api/messages \
-H "Content-Type: application/json" \
-d '{
  "source_msisdn": "+15551234567",
  "destination_msisdn": "+447700900000",
  "message_body": "Hello World",
  "source_smsc": "api_client"
}'
```

Desempenho: ~70 mensagens/segundo, 14ms de tempo médio de resposta

Usar Quando:

- Necessita do ID da mensagem imediatamente
- Processando mensagens/segundo
- Requer confirmação imediata

Submeter Mensagem (Assíncrona)

Submeter uma mensagem com alta taxa de transferência (processamento em lote).

Requisição:

```
POST /api/messages/create_async
Content-Type: application/json
```

Corpo: Mesmo que o endpoint síncrono

Resposta (202 Aceito):

```
{
  "data": {
    "status": "accepted",
    "message": "Mensagem na fila para processamento"
  }
}
```

Exemplo:

```
curl -X POST https://api.example.com:8443/api/messages/create_async \
-H "Content-Type: application/json" \
-d '{
  "source_msisdn": "+15551234567",
  "destination_msisdn": "+447700900000",
  "message_body": "Mensagem de notificação em massa",
  "source_smsc": "bulk_api"
}'
```

```
}'
```

Desempenho: ~4.650 mensagens/segundo, 0.22ms de tempo médio de resposta

Latência: Mensagem aparece no banco de dados dentro de 100ms (configurável)

Usar Quando:

- Mensagens em massa de alto volume (> 100 msg/sec)
- Não precisa do ID da mensagem na resposta da API
- Taxa de transferência mais importante que a confirmação instantânea

Atualizar Mensagem

Atualizar parcialmente os campos da mensagem.

Requisição:

```
PATCH /api/messages/:id
Content-Type: application/json
```

Corpo:

```
{
  "dest_smsc": "alternate_gateway",
  "deliver_after": "2025-10-30T14:00:00Z"
}
```

Campos Atualizáveis:

- dest_smsc - Alterar destino
- deliver_after - Atrasar entrega
- message_body - Atualizar texto da mensagem
- status - Alterar status

Resposta (200 OK):

```
{
  "data": {
    "id": 12345,
    "dest_smsc": "alternate_gateway",
    "deliver_after": "2025-10-30T14:00:00Z",
    ...
  }
}
```

Exemplo:

```
curl -X PATCH https://api.example.com:8443/api/messages/12345 \
```

```
-H "Content-Type: application/json" \  
-d '{  
  "dest_smsc": "backup_gateway"  
'
```

Marcar Mensagem como Entregue

Marcar uma mensagem como entregue com sucesso.

Requisição:

```
POST /api/messages/:id/mark_delivered  
Content-Type: application/json
```

Corpo:

```
{  
  "dest_smsc": "uk_gateway"  
}
```

Resposta (200 OK):

```
{  
  "data": {  
    "id": 12345,  
    "status": "delivered",  
    "deliver_time": "2025-10-30T12:05:30Z",  
    "dest_smsc": "uk_gateway",  
    ...  
  }  
}
```

Exemplo:

```
curl -X POST https://api.example.com:8443/api/messages/12345/  
mark_delivered \  
-H "Content-Type: application/json" \  
-d '{  
  "dest_smsc": "uk_gateway"  
'
```

Caso de Uso: Chamado por sistemas frontend após entrega bem-sucedida

Incrementar Tentativa de Entrega

Incrementar contador de tentativas e aplicar backoff exponencial.

Requisição:


```
PUT /api/messages/:id
```

Resposta (200 OK):

```
{
  "data": {
    "id": 12345,
    "delivery_attempts": 2,
    "deliver_after": "2025-10-30T12:08:00Z",
    ...
  }
}
```

Cálculo de Backoff:

```
deliver_after = agora + 2^(delivery_attempts) minutos
```

Exemplo:

```
curl -X PUT https://api.example.com:8443/api/messages/12345
```

Caso de Uso: Chamado pelo frontend após falha de entrega para agendar nova tentativa

Deletar Mensagem

Remover mensagem da fila.

Requisição:

```
DELETE /api/messages/:id
```

Resposta (204 Sem Conteúdo)

Exemplo:

```
curl -X DELETE https://api.example.com:8443/api/messages/12345
```

Aviso: Deletar mensagens as remove permanentemente. Use com cautela.

API de PDU SMS Bruto

Submeter mensagens SMS como PDU bruto (Unidade de Dados de Protocolo) para máxima compatibilidade com sistemas legados.

Submeter SMS Bruto (Síncrono)

Requisição:

```
POST /api/messages_raw
Content-Type: application/json
```

Corpo:

```
{
  "pdu": "0001000B916407007009F0000004D4F29C0E",
  "source_smsc": "legacy_system"
}
```

Formato PDU: SMS TPDU (Unidade de Dados de Protocolo de Transporte) codificado em hexadecimal

Resposta (201 Criado):

```
{
  "data": {
    "id": 12346,
    "source_msisdn": "+447700900000",
    "destination_msisdn": "+447700900000",
    "message_body": "Teste",
    "source_smsc": "legacy_system",
    "raw_pdu": "0001000B916407007009F0000004D4F29C0E",
    ...
  }
}
```

Exemplo:

```
curl -X POST https://api.example.com:8443/api/messages_raw \
-H "Content-Type: application/json" \
-d '{
  "pdu": "0001000B916407007009F0000004D4F29C0E",
  "source_smsc": "legacy_system"
}'
```

Submeter SMS Bruto (Assíncrono)

Requisição:

```
POST /api/messages_raw/async
Content-Type: application/json
```

Corpo: Mesmo que o síncrono

Resposta (202 Aceito):

```
{
  "data": {
```

```
{
  "status": "accepted",
  "message": "PDU na fila para processamento"
}
```

Exemplo:

```
curl -X POST https://api.example.com:8443/api/messages_raw/async \
-H "Content-Type: application/json" \
-d '{
  "pdu": "0001000B916407007009F0000004D4F29C0E",
  "source_smsc": "legacy_gateway"
}'
```

Manipulação de PDU

O sistema automaticamente:

1. Decodifica PDU usando padrões SMS (3GPP TS 23.040)
2. Extrai números de telefone, texto da mensagem, DCS
3. Detecta relatórios de entrega (CP-ACK, RP-ACK, etc.)
4. Realiza pesquisa de IMSI para MSISDN se necessário
5. Aplica regras de roteamento
6. Armazena PDU original para referência

Deteção de Relatório de Entrega:

- CP-ACK, CP-ERROR - Acknowledgments do Protocolo de Conexão
- RP-ACK, RP-ERROR, RP-SMMA - Respostas do Protocolo de Relé
- Relatórios de entrega são registrados, mas não armazenados como mensagens

API de Gerenciamento de Localização

Gerenciar informações de localização do assinante para entrega de mensagens terminadas em dispositivos móveis.

Listar Localizações

Requisição:

```
GET /api/locations
```

Resposta (200 OK):

```
{
  "data": [
    {
```

```
{
  "id": 1,
  "msisdn": "+15551234567",
  "imsi": "001001000000001",
  "location": "msc1.region1.example.com",
  "ran_location": "cell_tower_12345",
  "imei": "123456789012345",
  "ims_capable": true,
  "csfb": false,
  "registered": true,
  "expires": "2025-10-30T13:00:00Z",
  "user_agent": "Samsung Galaxy",
  "inserted_at": "2025-10-30T12:00:00Z",
  "updated_at": "2025-10-30T12:00:00Z"
}
```

Exemplo:

```
curl https://api.example.com:8443/api/locations
```

Obter Localização

Requisição:

```
GET /api/locations/:id
```

Resposta (200 OK):

```
{
  "data": {
    "id": 1,
    "msisdn": "+15551234567",
    "imsi": "001001000000001",
    ...
  }
}
```

Exemplo:

```
curl https://api.example.com:8443/api/locations/1
```

Criar/Atualizar Localização

Cria nova localização ou atualiza existente com base no IMSI (identificador único).

Requisição:

POST /api/locations
Content-Type: application/json

Corpo:

```
{
  "msisdn": "+15551234567",
  "imsi": "001001000000001",
  "location": "msc1.region1.example.com",
  "ran_location": "cell_tower_12345",
  "imei": "123456789012345",
  "ims_capable": true,
  "csfb": false,
  "registered": true,
  "expires": "2025-10-30T13:00:00Z",
  "user_agent": "Samsung Galaxy"
}
```

Campos Obrigatórios:

- imsi - Identificador único do assinante
- msisdn - Número de telefone

Campos Opcionais:

- location - Endereço MSC/VLR
- ran_location - ID da torre/setor celular
- imei - Identificador do dispositivo
- ims_capable - Capacidade IMS VoLTE
- csfb - Sinalizador de fallback de circuito
- registered - Atualmente registrado
- expires - Expiração do registro
- user_agent - Modelo/informações do dispositivo

Resposta (201 Criado ou 200 OK):

```
{
  "data": {
    "id": 1,
    "msisdn": "+15551234567",
    ...
  }
}
```

Exemplo:

```
curl -X POST https://api.example.com:8443/api/locations \
  -H "Content-Type: application/json" \
  -d '{
```

```
"msisdn": "+15551234567",  
"imsi": "001001000000001",  
"location": "msc1.region1.example.com",  
"ims_capable": true,  
"registered": true  
}'
```

Caso de Uso: Chamado por sistemas de gerenciamento de mobilidade (HSS, MME, etc.) quando o assinante se registra

Atualizar Localização

Requisição:

```
PATCH /api/locations/:id  
Content-Type: application/json
```

Corpo: Atualização parcial com quaisquer campos de localização

Resposta (200 OK):

```
{  
  "data": {  
    "id": 1,  
    ...  
  }  
}
```

Exemplo:

```
curl -X PATCH https://api.example.com:8443/api/locations/1 \  
-H "Content-Type: application/json" \  
-d '{  
  "location": "msc2.region2.example.com",  
  "ran_location": "cell_tower_67890"  
}'
```

Deletar Localização

Requisição:

```
DELETE /api/locations/:id
```

Resposta (204 Sem Conteúdo)

Exemplo:

```
curl -X DELETE https://api.example.com:8443/api/locations/1
```

Caso de Uso: Chamado quando o assinante se desregistra ou expira

API de Registro de Frontend

Rastrear e gerenciar conexões de SMSC frontend.

Listar Todos os Frontends

Requisição:

```
GET /api/frontends
```

Resposta (200 OK):

```
{
  "data": [
    {
      "id": 1,
      "frontend_name": "uk_gateway_1",
      "frontend_type": "smpp",
      "ip_address": "10.0.1.50",
      "hostname": "gateway1.uk.example.com",
      "uptime_seconds": 86400,
      "configuration": {
        "max_throughput": 1000,
        "bind_type": "transceiver"
      },
      "status": "active",
      "expires_at": "2025-10-30T12:02:00Z",
      "last_seen_at": "2025-10-30T12:00:30Z",
      "inserted_at": "2025-10-29T12:00:00Z",
      "updated_at": "2025-10-30T12:00:30Z"
    }
  ]
}
```

Exemplo:

```
curl https://api.example.com:8443/api/frontends
```

Listar Apenas Frontends Ativos

Requisição:

```
GET /api/frontends/active
```

Resposta (200 OK): Mesmo formato, apenas frontends ativos

Exemplo:

```
curl https://api.example.com:8443/api/frontends/active
```

Caso de Uso: Obter lista de destinos disponíveis para roteamento

Obter Estatísticas do Frontend**Requisição:**

```
GET /api/frontends/stats
```

Resposta (200 OK):

```
{
  "data": {
    "active_count": 5,
    "expired_count": 2,
    "unique_frontends": 7,
    "total_registrations": 1523
  }
}
```

Exemplo:

```
curl https://api.example.com:8443/api/frontends/stats
```

Obter Histórico do Frontend**Requisição:**

```
GET /api/frontends/history/:name
```

Resposta (200 OK):

```
{
  "data": [
    {
      "id": 1,
      "frontend_name": "uk_gateway_1",
      "status": "active",
      "inserted_at": "2025-10-30T12:00:00Z",
      ...
    },
    {
      "id": 2,
      "frontend_name": "uk_gateway_1",
      "status": "expired",
      ...
    }
  ]
}
```



```
    "inserted_at": "2025-10-29T12:00:00Z",  
    ...  
  }  
]  
}
```

Exemplo:

```
curl https://api.example.com:8443/api/frontends/history/uk_gateway_1
```

Registrar Frontend

Registrar ou atualizar a conexão do frontend.

Requisição:

```
POST /api/frontends/register  
Content-Type: application/json
```

Corpo:

```
{  
  "frontend_name": "uk_gateway_1",  
  "frontend_type": "smpp",  
  "ip_address": "10.0.1.50",  
  "hostname": "gateway1.uk.example.com",  
  "uptime_seconds": 86400,  
  "configuration": {  
    "max_throughput": 1000,  
    "bind_type": "transceiver",  
    "system_id": "gateway1"  
  }  
}
```

Campos Obrigatórios:

- frontend_name - Identificador único para o frontend
- frontend_type - Tipo: smpp, sip, http, etc.

Campos Opcionais:

- ip_address - IP do frontend
- hostname - Nome do host do frontend
- uptime_seconds - Uptime desde o início
- configuration - Objeto de configuração personalizada

Resposta (201 Criado):

```
{
```

```
{
  "data": {
    "id": 1,
    "frontend_name": "uk_gateway_1",
    "status": "active",
    "expires_at": "2025-10-30T12:01:30Z",
    ...
  }
}
```

Exemplo:

```
curl -X POST https://api.example.com:8443/api/frontends/register \
-H "Content-Type: application/json" \
-d '{
  "frontend_name": "uk_gateway_1",
  "frontend_type": "smpp",
  "ip_address": "10.0.1.50",
  "hostname": "gateway1.uk.example.com"
}'
```

Tempo Limite de Registro: 90 segundos (frontends devem se re-registrar a cada 60-90 segundos)

Caso de Uso: Chamado periodicamente por sistemas frontend para manter status ativo

API de Registro de Eventos

Rastrear eventos do ciclo de vida das mensagens.

Obter Eventos de Mensagens

Requisição:

```
GET /api/events/:message_id
```

Resposta (200 OK):

```
{
  "data": [
    {
      "event_epoch": 1698672000,
      "name": "message_inserted",
      "description": "Mensagem inserida na fila",
      "event_source": "node1@server.example.com"
    },
    {
      "event_epoch": 1698672001,
```

```

    "name": "message_routed",
    "description": "Roteada para uk_gateway via route_id=42",
    "event_source": "node1@server.example.com"
  },
  {
    "event_epoch": 1698672005,
    "name": "message_delivered",
    "description": "Entregue com sucesso",
    "event_source": "node2@server.example.com"
  }
]
}

```

Exemplo:

```
curl https://api.example.com:8443/api/events/12345
```

Tipos de Evento:

- message_inserted - Mensagem criada
- message_routed - Decisão de roteamento feita
- message_delivered - Entrega bem-sucedida
- message_failed - Falha na entrega
- message_dropped - Removida pelo roteamento
- auto_reply_sent - Resposta automática acionada
- number_translated - Transformação de número aplicada
- routing_failed - Nenhum roteamento encontrado
- charging_failed - Erro de cobrança

Registrar Evento

Requisição:

```
POST /api/events
Content-Type: application/json
```

Corpo:

```

{
  "message_id": 12345,
  "name": "custom_event",
  "description": "Descrição do evento personalizado",
  "event_source": "external_system"
}

```

Resposta (201 Criado):

```

{
  "data": {

```

```
{
  "message_id": 12345,
  "name": "custom_event",
  "description": "Descrição do evento personalizado",
  "event_source": "external_system",
  "event_epoch": 1698672010
}
```

Exemplo:

```
curl -X POST https://api.example.com:8443/api/events \
-H "Content-Type: application/json" \
-d '{
  "message_id": 12345,
  "name": "external_delivery_confirmed",
  "description": "Confirmado pelo sistema downstream"
}'
```

Retenção de Eventos: 7 dias (configurável)

API de Mensagens MMS

Gerenciar mensagens do Serviço de Mensagens Multimídia (MMS).

Listar Mensagens MMS

Requisição:

```
GET /api/mms_messages
```

Resposta (200 OK): Semelhante às mensagens SMS com campos MMS adicionais

Criar Mensagem MMS

Requisição:

```
POST /api/mms_messages
Content-Type: application/json
```

Corpo:

```
{
  "source_msisdn": "+15551234567",
  "destination_msisdn": "+447700900000",
  "subject": "Foto",
  "content_type": "image/jpeg",
  "content_location": "https://cdn.example.com/media/12345.jpg",
  "message_size": 524288
}
```

```
}
```

Resposta (201 Criado): Objeto completo da mensagem MMS

API de Eventos SS7

Rastrear eventos de sinalização SS7.

Listar Eventos SS7

Requisição:

```
GET /api/ss7_events
```

Resposta (200 OK):

```
{
  "data": [
    {
      "id": 1,
      "event_type": "MAP_UPDATE_LOCATION",
      "imsi": "001001000000001",
      "msisdn": "+15551234567",
      "timestamp": "2025-10-30T12:00:00Z",
      ...
    }
  ]
}
```

Criar Evento SS7

Requisição:

```
POST /api/ss7_events
Content-Type: application/json
```

Corpo:

```
{
  "event_type": "MAP_UPDATE_LOCATION",
  "imsi": "001001000000001",
  "msisdn": "+15551234567"
}
```

Resposta (201 Criado): Objeto completo do evento

Códigos de Erro

Códigos de Status HTTP

Código	Significado	Descrição
200	OK	Requisição bem-sucedida
201	Criado	Recurso criado com sucesso
202	Aceito	Requisição aceita para processamento
204	Sem Conteúdo	Exclusão bem-sucedida
400	Requisição Inválida	Formato de requisição inválido
401	Não Autorizado	Autenticação necessária
403	Proibido	Permissões insuficientes
404	Não Encontrado	Recurso não existe
422	Entidade Não Processável	Erros de validação
429	Muitas Requisições	Limite de taxa excedido
500	Erro Interno do Servidor	Erro no servidor
503	Serviço Indisponível	Temporariamente indisponível

Formato de Resposta de Erro

```
{
  "errors": {
    "detail": "Validação falhou: destination_msisdn é obrigatório"
  }
}
```

Mensagens de Erro Comuns

Erro	Causa	Solução
"destination_msisdn é obrigatório"	Campo obrigatório ausente	Incluir destination_msisdn na requisição
"Formato de número de telefone inválido"	Número malformado	Usar formato E.164: +15551234567
"Mensagem muito longa"	Excede o limite de tamanho	Dividir em várias partes
"Nenhum roteamento encontrado"	Falha no roteamento	Verificar configuração de roteamento
"Falha na cobrança"	Erro no OCS	Verificar conectividade do sistema de cobrança
"Mensagem não encontrada"	ID de mensagem inválido	Verificar se o ID existe
"Frontend não registrado"	SMSC desconhecido	Registrar frontend primeiro

Limitação de Taxa

Limites Padrão

Endpoint	Limite	Janela
POST /api/messages	100 req/sec	Por IP
POST /api/messages/create_async	1000 req/sec	Por IP
POST /api/messages_raw	100 req/sec	Por IP
GET /api/*	1000 req/sec	Por IP

Cabeçalhos de Limitação de Taxa

```
X-RateLimit-Limit: 100
X-RateLimit-Remaining: 95
X-RateLimit-Reset: 1698672060
```

Limite de Taxa Excedido

Resposta (429 Muitas Requisições):

```
{
  "errors": {
    "detail": "Limite de taxa excedido. Tente novamente após 5 segundos."
  }
}
```

Melhores Práticas

Submissão de Mensagens

1. **Use Async para Bulk:** Use /create_async para > 100 msg/sec
2. **Incluir source_smsc:** Sempre identifique seu sistema
3. **Validar Números:** Use formato E.164 (+código do país)
4. **Tratar Erros:** Implementar lógica de repetição para erros 5xx
5. **Verificar Roteamento:** Testar rotas antes da submissão em massa

Integração com Frontend

1. **Registrar Regularmente:** Re-registrar a cada 60 segundos
2. **Consultar Mensagens:** Consultar com cabeçalho smsc para suas mensagens
3. **Usar include-unrouted com Sabedoria:** Por padrão, apenas mensagens com roteamento explícito ou registro de localização são retornadas. Defina include-unrouted: true apenas se precisar de comportamento compatível com versões anteriores para receber todas as mensagens não

- roteadas
4. **Marcar como Entregue:** Sempre chamar mark_delivered após sucesso
 5. **Incrementar em Caso de Falha:** Usar endpoint PUT para lógica de repetição
 6. **Monitorar Eventos:** Verificar log de eventos para problemas de entrega

Desempenho

1. **Pooling de Conexões:** Reutilizar conexões HTTP
2. **Requisições em Lote:** Agrupar várias mensagens por requisição
3. **Processamento Paralelo:** Fazer chamadas API concorrentes
4. **Monitorar Métricas:** Observar o Prometheus para gargalos
5. **Definir Timeouts:** Usar timeout de 30 segundos para chamadas API

Segurança

1. **Usar TLS:** Sempre usar HTTPS em produção
2. **Validar Certificados:** Não pular a validação de certificados
3. **Rotacionar Chaves de API:** Alterar chaves regularmente
4. **Lista de IPs Permitidos:** Restringir a fontes conhecidas
5. **Registrar Atividade da API:** Monitorar padrões suspeitos

Tratamento de Erros

1. **Repetir Erros 5xx:** Erros de servidor são geralmente temporários
2. **Não Repetir 4xx:** Erros de cliente precisam de correções de código
3. **Backoff Exponencial:** Esperar mais entre tentativas
4. **Disjuntor:** Parar após falhas repetidas
5. **Alertar sobre Padrões:** Monitorar taxas de erro

Integração de Exemplo (Python)

```
import requests
import time

class SMSCClient:
    def __init__(self, base_url, api_key=None):
        self.base_url = base_url
        self.session = requests.Session()
        if api_key:
            self.session.headers.update({"X-API-Key": api_key})

    def submit_message(self, from_num, to_num, text,
async_mode=False):
        endpoint = "/messages/create_async" if async_mode else
"/messages"
        url = f"{self.base_url}{endpoint}"
```



```

        payload = {
            "source_msisdn": from_num,
            "destination_msisdn": to_num,
            "message_body": text,
            "source_smsc": "python_client"
        }

        try:
            response = self.session.post(url, json=payload,
timeout=30)
            response.raise_for_status()
            return response.json()["data"]
        except requests.exceptions.RequestException as e:
            print(f"Erro na API: {e}")
            return None

    def get_pending_messages(self, smsc_name,
include_unrouted=False):
        url = f"{self.base_url}/messages"
        headers = {"smsc": smsc_name}

        # Incluir mensagens não roteadas se solicitado (modo
        # compatível com versões anteriores)
        if include_unrouted:
            headers["include-unrouted"] = "true"

        try:
            response = self.session.get(url, headers=headers,
timeout=30)
            response.raise_for_status()
            return response.json()["data"]
        except requests.exceptions.RequestException as e:
            print(f"Erro na API: {e}")
            return []

    def mark_delivered(self, message_id, smsc_name):
        url = f"{self.base_url}/messages/{message_id}/mark_delivered"
        payload = {"dest_smsc": smsc_name}

        try:
            response = self.session.post(url, json=payload,
timeout=30)
            response.raise_for_status()
            return True
        except requests.exceptions.RequestException as e:
            print(f"Erro na API: {e}")
            return False

```

```

# Uso
client = SMSCClient("https://api.example.com:8443/api",
api_key="your_key")

# Submeter mensagem única
result = client.submit_message("+15551234567", "+447700900000",
"Olá")
print(f"ID da Mensagem: {result['id']}")

# Submeter mensagens em massa (assíncronas)
for i in range(1000):
    client.submit_message("+15551234567", f"+44770090{i:04d}", f"Em
massa {i}", async_mode=True)

# Loop de polling do frontend
while True:
    # Obter mensagens com roteamento explícito ou registro de
localização
    messages = client.get_pending_messages("my_gateway")

    # Ou usar include_unrouted=True para comportamento compatível com
versões anteriores
    # messages = client.get_pending_messages("my_gateway",
include_unrouted=True)

    for msg in messages:
        # Entregar mensagem via seu protocolo
        success = deliver_via_smpp(msg)

        if success:
            client.mark_delivered(msg["id"], "my_gateway")
        else:
            # Incrementar para nova tentativa
            requests.put(f"{client.base_url}/messages/{msg['id']}")

    time.sleep(5) # Poll a cada 5 segundos

```

Registro de Alterações da API

Versão 1 (Atual)

- Lançamento inicial
- CRUD da fila de mensagens
- Submissão de PDU bruto
- Gerenciamento de localização
- Registro de frontend
- Registro de eventos

Recursos Planejados

- Submissão de mensagens em lote (uma única requisição, várias mensagens)
- Modelos de mensagens
- API de entrega agendada
- Webhooks em tempo real para eventos
- Endpoint da API GraphQL
- Autenticação OAuth2

Para perguntas ou problemas com a API, consulte o [Guia de Solução de Problemas](#) ou entre em contato com o suporte.



Referência do Esquema CDR (Registro de Detalhes de Chamadas)

[← Voltar ao Índice da Documentação](#) | [README Principal](#)

Referência completa para a tabela do banco de dados CDR usada para armazenamento de mensagens a longo prazo, faturamento e análises.

Índice

- [Visão Geral](#)
- [Esquema da Tabela](#)
- [Descrições dos Campos](#)
- [Exemplos de SQL](#)
- [Índices](#)
- [Tipos de Dados por Banco de Dados](#)
- [Considerações de Privacidade](#)
- [Retenção e Arquivamento](#)
- [Integração de Faturamento](#)

Visão Geral

A tabela cdrs armazena Registros de Detalhes de Chamadas para todas as mensagens SMS processadas pelo sistema. Os CDRs são gravados quando:

- Mensagens são entregues com sucesso
- Mensagens expiram sem entrega
- Mensagens falham permanentemente
- Mensagens são rejeitadas

Os CDRs fornecem armazenamento a longo prazo separado do banco de dados operacional Mnesia, permitindo:

- Faturamento e cobrança
- Análises e relatórios
- Conformidade e auditoria
- Histórico de mensagens além do período de retenção do Mnesia

Esquema da Tabela

MySQL / MariaDB

```
CREATE TABLE cdrs (  
  id BIGINT AUTO_INCREMENT PRIMARY KEY,  
  
  -- Identificação da mensagem  
  message_id BIGINT NOT NULL,  
  
  -- Números de telefone  
  calling_number VARCHAR(255) NOT NULL,  
  called_number VARCHAR(255) NOT NULL,  
  
  -- Roteamento SMSC  
  source_smsc VARCHAR(255),  
  dest_smsc VARCHAR(255),  
  
  -- Informações do nó (para implantações em cluster)  
  origin_node VARCHAR(255),  
  destination_node VARCHAR(255),  
  
  -- Carimbos de data/hora  
  submission_time DATETIME NOT NULL,  
  delivery_time DATETIME,  
  expiry_time DATETIME,  
  
  -- Status e metadados  
  status VARCHAR(50) NOT NULL,  
  delivery_attempts INT DEFAULT 0,  
  message_parts INT,  
  deadletter BOOLEAN DEFAULT FALSE,  
  
  -- Corpo da mensagem opcional (controles de privacidade)  
  message_body TEXT,  
  
  -- Carimbos de auditoria  
  inserted_at DATETIME NOT NULL,  
  updated_at DATETIME NOT NULL,  
  
  -- Índices  
  INDEX idx_cdrs_message_id (message_id),  
  INDEX idx_cdrs_calling_number (calling_number),  
  INDEX idx_cdrs_called_number (called_number),  
  INDEX idx_cdrs_status (status),  
  INDEX idx_cdrs_submission_time (submission_time),  
  INDEX idx_cdrs_dest_smsc (dest_smsc)  
);
```

PostgreSQL

```
CREATE TABLE cdrs (  
    id BIGSERIAL PRIMARY KEY,  
  
    -- Identificação da mensagem  
    message_id BIGINT NOT NULL,  
  
    -- Números de telefone  
    calling_number VARCHAR(255) NOT NULL,  
    called_number VARCHAR(255) NOT NULL,  
  
    -- Roteamento SMSC  
    source_smsc VARCHAR(255),  
    dest_smsc VARCHAR(255),  
  
    -- Informações do nó (para implantações em cluster)  
    origin_node VARCHAR(255),  
    destination_node VARCHAR(255),  
  
    -- Carimbos de data/hora  
    submission_time TIMESTAMP NOT NULL,  
    delivery_time TIMESTAMP,  
    expiry_time TIMESTAMP,  
  
    -- Status e metadados  
    status VARCHAR(50) NOT NULL,  
    delivery_attempts INTEGER DEFAULT 0,  
    message_parts INTEGER,  
    deadletter BOOLEAN DEFAULT FALSE,  
  
    -- Corpo da mensagem opcional (controles de privacidade)  
    message_body TEXT,  
  
    -- Carimbos de auditoria  
    inserted_at TIMESTAMP NOT NULL,  
    updated_at TIMESTAMP NOT NULL  
);  
  
-- Índices  
CREATE INDEX idx_cdrs_message_id ON cdrs(message_id);  
CREATE INDEX idx_cdrs_calling_number ON cdrs(calling_number);  
CREATE INDEX idx_cdrs_called_number ON cdrs(called_number);  
CREATE INDEX idx_cdrs_status ON cdrs(status);  
CREATE INDEX idx_cdrs_submission_time ON cdrs(submission_time);  
CREATE INDEX idx_cdrs_dest_smsc ON cdrs(dest_smsc);
```

Descrições dos Campos

Chave Primária

Campo	Tipo	Nulo	Descrição
id	BIGINT	NÃO	Chave primária auto-incrementada para o registro CDR

Identificação da Mensagem

Campo	Tipo	Nulo	Descrição
message_id	BIGINT	NÃO	Identificador único da mensagem da fila de mensagens SMS-C. Referencia o ID da mensagem original no Mnesia.

Números de Telefone

Campo	Tipo	Nulo	Descrição
calling_number	VARCHAR(255)	NÃO	MSISDN de origem (número de celular) do remetente da mensagem. Normalmente no formato E.164 (ex: +15551234567).
called_number	VARCHAR(255)	NÃO	MSISDN de destino (número de celular) do destinatário da mensagem. Normalmente no formato E.164 (ex: +15551234567).

Roteamento SMSC

Campo	Tipo	Nulo	Descrição
source_smsc	VARCHAR(255)	SIM	Nome ou identificador do SMSC de origem que enviou a mensagem. NULL se enviado via API ou outra interface que não seja SMSC.
dest_smsc	VARCHAR(255)	SIM	Nome ou identificador do SMSC de destino que entregou (ou tentou entregar) a mensagem. NULL se a mensagem nunca foi roteada.

Informações do Nó

Para implantações em cluster, rastreia quais nós manipularam a mensagem:

Campo	Tipo	Nulo	Descrição
origin_node	VARCHAR(255)	SIM	Nome do nó Erlang onde a mensagem foi originalmente recebida (ex: sms@node1.example.com). Útil para solução de problemas e análise de distribuição de carga.

Campo	Tipo	Nulo	Descrição
destination_node	VARCHAR(255)	SIM	Nome do nó Erlang de onde a mensagem foi entregue (se diferente da origem). NULL para implantações de nó único ou se a mensagem nunca foi entregue.

Carimbos de Data/Hora

Todos os carimbos de data/hora são armazenados em UTC:

Campo	Tipo	Nulo	Descrição
submission_time	DATETIME	NÃO	Quando a mensagem foi enviada pela primeira vez ao SMS-C. Usado como o horário de início para cálculos de faturamento.
delivery_time	DATETIME	SIM	Quando a mensagem foi entregue com sucesso. NULL se a mensagem expirou, falhou ou foi rejeitada.
expiry_time	DATETIME	SIM	Quando a mensagem expirou (se tornou indeliverável). NULL se a mensagem foi entregue ou ainda está pendente.

Cálculo da Duração da Entrega:

```
TIMESTAMPDIFF(SECOND, submission_time, delivery_time) AS
delivery_duration_seconds
```

Status e Metadados

Campo	Tipo	Nulo	Descrição
status	VARCHAR(50)	NÃO	Status final da mensagem. Valores válidos: delivered, expired, failed, rejected
delivery_attempts	INT	NÃO	Número de tentativas de entrega feitas antes do status final. Padrão: 0. Faixa: 0-255 normalmente.
message_parts	INT	SIM	Número de segmentos SMS para mensagens concatenadas. 1 para mensagens de uma parte, 2+ para multi-partes. NULL se desconhecido.
deadletter	BOOLEAN	NÃO	Se a mensagem foi movida para a fila de cartas mortas. TRUE indica que a mensagem não pôde ser entregue e esgotou todas as tentativas. Padrão: FALSE

Valores de Status:

Status	Descrição	Faturável	Tempo de Entrega
delivered	Entregue com sucesso ao destinatário	Sim	Definido
expired	Excedeu o período de validade sem entrega	Depende da política de faturamento	NULL
failed	Falha permanente na entrega (número inválido, etc.)	Depende da política de faturamento	NULL
rejected	Rejeitado por regras de roteamento ou validação	Não	NULL

Corpo da Mensagem

Campo	Tipo	Nulo	Descrição
message_body	TEXT	SIM	O conteúdo real da mensagem SMS. Pode ser NULL se delete_message_body_after_delivery estiver habilitado para privacidade. O comprimento máximo varia por banco de dados (normalmente 65.535 caracteres para o tipo TEXT).

Modos de Privacidade:

- **Retenção total:** Corpo da mensagem armazenado no CDR para conformidade/arquivamento
- **Modo de privacidade:** Corpo da mensagem definido como NULL quando delete_message_body_after_delivery: true
- **Modo de conformidade:** Corpo armazenado criptografado ou hash (requer implementação personalizada)

Carimbos de Auditoria

Campo	Tipo	Nulo	Descrição
inserted_at	DATETIME	NÃO	Quando o registro CDR foi inserido pela primeira vez no banco de dados. Normalmente o mesmo que ou logo após delivery_time/expiry_time.
updated_at	DATETIME	NÃO	Quando o registro CDR foi atualizado pela última vez. O mesmo que inserted_at se nunca atualizado.

Exemplos de SQL

Consultas Básicas

Encontrar todos os CDRs para um número de telefone específico:

```
SELECT * FROM cdrs
WHERE calling_number = '+15551234567'
```

```
    OR called_number = '+15551234567'
ORDER BY submission_time DESC
LIMIT 100;
```

Contar mensagens por status:

```
SELECT status, COUNT(*) as count
FROM cdrs
GROUP BY status;
```

Tempo médio de entrega para mensagens entregues:

```
SELECT AVG(TIMESTAMPDIFF(SECOND, submission_time, delivery_time)) AS
avg_delivery_seconds
FROM cdrs
WHERE status = 'delivered'
    AND delivery_time IS NOT NULL;
```

Consultas de Faturamento

Volume diário de mensagens por SMSC de destino:

```
SELECT
    DATE(submission_time) AS date,
    dest_smsc,
    COUNT(*) AS message_count,
    SUM(CASE WHEN status = 'delivered' THEN 1 ELSE 0 END) AS
delivered_count,
    SUM(message_parts) AS total_segments
FROM cdrs
WHERE submission_time >= DATE_SUB(NOW(), INTERVAL 30 DAY)
GROUP BY DATE(submission_time), dest_smsc
ORDER BY date DESC, message_count DESC;
```

Mensagens faturáveis para um cliente (por prefixo de número de chamada):

```
SELECT
    DATE(submission_time) AS date,
    COUNT(*) AS message_count,
    SUM(message_parts) AS total_segments,
    SUM(message_parts) * 0.01 AS total_cost
FROM cdrs
WHERE calling_number LIKE '+1555%'
    AND status = 'delivered'
    AND submission_time >= '2025-10-01'
    AND submission_time < '2025-11-01'
GROUP BY DATE(submission_time);
```

Análise de desempenho de rotas:

```
SELECT
    dest_smsc,
    COUNT(*) AS total_messages,
    SUM(CASE WHEN status = 'delivered' THEN 1 ELSE 0 END) AS delivered,
    ROUND(100.0 * SUM(CASE WHEN status = 'delivered' THEN 1 ELSE 0
END) / COUNT(*), 2) AS delivery_rate_pct,
    AVG(delivery_attempts) AS avg_attempts,
    AVG(TIMESTAMPDIFF(SECOND, submission_time, delivery_time)) AS
avg_delivery_seconds
FROM cdrs
WHERE submission_time >= DATE_SUB(NOW(), INTERVAL 7 DAY)
    AND dest_smsc IS NOT NULL
GROUP BY dest_smsc
ORDER BY delivery_rate_pct DESC;
```

Consultas de Análise

Mensagens por hora do dia (padrão de tráfego):

```
SELECT
    HOUR(submission_time) AS hour,
    COUNT(*) AS message_count
FROM cdrs
WHERE submission_time >= DATE_SUB(NOW(), INTERVAL 7 DAY)
GROUP BY HOUR(submission_time)
ORDER BY hour;
```

Análise de mensagens multi-partes:

```
SELECT
    message_parts,
    COUNT(*) AS message_count,
    AVG(TIMESTAMPDIFF(SECOND, submission_time, delivery_time)) AS
avg_delivery_seconds
FROM cdrs
WHERE message_parts IS NOT NULL
    AND status = 'delivered'
GROUP BY message_parts
ORDER BY message_parts;
```

Análise de mensagens falhadas:

```
SELECT
    called_number,
    COUNT(*) AS failure_count,
    AVG(delivery_attempts) AS avg_attempts,
```

```

    MAX(submission_time) AS last_failure
FROM cdrs
WHERE status IN ('failed', 'expired')
    AND submission_time >= DATE_SUB(NOW(), INTERVAL 7 DAY)
GROUP BY called_number
HAVING failure_count >= 5
ORDER BY failure_count DESC;

```

Consultas de Conformidade e Auditoria

Encontrar todas as mensagens entre duas partes em um intervalo de tempo:

```

SELECT
    submission_time,
    calling_number,
    called_number,
    status,
    message_body,
    delivery_time
FROM cdrs
WHERE (
    (calling_number = '+15551234567' AND called_number =
'+15559876543')
    OR
    (calling_number = '+15559876543' AND called_number =
'+15551234567')
)
AND submission_time >= '2025-10-01'
AND submission_time < '2025-11-01'
ORDER BY submission_time;

```

Aplicação da política de retenção (deletar CDRs antigos):

```

-- Encontrar registros mais antigos que o período de retenção
(exemplo: 2 anos)
SELECT COUNT(*) FROM cdrs
WHERE submission_time < DATE_SUB(NOW(), INTERVAL 2 YEAR);

-- Deletar registros antigos (usar com cautela!)
DELETE FROM cdrs
WHERE submission_time < DATE_SUB(NOW(), INTERVAL 2 YEAR)
LIMIT 10000; -- Deletar em lote para evitar bloqueio

```

Análise de Cluster

Distribuição de mensagens entre nós:

```

SELECT
    origin_node,
    COUNT(*) AS message_count,
    SUM(CASE WHEN status = 'delivered' THEN 1 ELSE 0 END) AS
delivered_count
FROM cdrs
WHERE submission_time >= DATE_SUB(NOW(), INTERVAL 1 DAY)
GROUP BY origin_node;

```

Índices

Os seguintes índices são criados para otimizar consultas comuns:

Nome do Índice	Colunas	Propósito
PRIMARY	id	Chave primária, garante registro único
idx_cdrs_message_id	message_id	Localizar CDR por ID da mensagem original
idx_cdrs_calling_number	calling_number	Encontrar mensagens de um remetente específico
idx_cdrs_called_number	called_number	Encontrar mensagens para um destinatário específico
idx_cdrs_status	status	Filtrar por status de entrega
idx_cdrs_submission_time	submission_time	Consultas baseadas em tempo, períodos de faturamento
idx_cdrs_dest_smsc	dest_smsc	Análise de desempenho de rotas

Recomendações de Índice Adicionais

Para implantações de alto volume, considere esses índices adicionais:

Índice composto para consultas de faturamento:

```

CREATE INDEX idx_cdrs_billing ON cdrs(calling_number,
submission_time, status);

```

Índice composto para análise de rotas:

```

CREATE INDEX idx_cdrs_route_perf ON cdrs(dest_smsc, submission_time,
status);

```

Índice composto para pesquisas de conformidade:

```

CREATE INDEX idx_cdrs_party_time ON cdrs(calling_number,
called_number, submission_time);

```

Índice de texto completo para pesquisas de corpo de mensagem (MySQL):

```
ALTER TABLE cdrs ADD FULLTEXT INDEX idx_cdrs_message_body_ft
(message_body);

-- Uso:
SELECT * FROM cdrs
WHERE MATCH(message_body) AGAINST('keyword' IN NATURAL LANGUAGE
MODE);
```

Tipos de Dados por Banco de Dados

Mapeamentos de tipo de campo entre bancos de dados suportados:

Campo	MySQL/MariaDB	PostgreSQL	Notas
id	BIGINT AUTO_INCREMENT	BIGSERIAL	Inteiro de 64 bits, auto-incrementado
message_id	BIGINT	BIGINT	Inteiro de 64 bits
Campos de string	VARCHAR(255)	VARCHAR(255)	String de comprimento variável, max 255 caracteres
message_body	TEXT	TEXT	Texto grande, até 65.535 bytes (MySQL), ilimitado (PostgreSQL)
Carimbos de data/hora	DATETIME	TIMESTAMP	Carimbos de data/hora UTC recomendados
Inteiros	INT	INTEGER	Inteiro assinado de 32 bits
Booleanos	BOOLEAN (TINYINT(1))	BOOLEAN	MySQL armazena como 0/1

Considerações de Privacidade

A tabela CDR pode conter informações pessoais sensíveis (números de telefone, conteúdo da mensagem). Considere essas medidas de privacidade:

1. Privacidade do Corpo da Mensagem

Opções de configuração em config/runtime.exs:

```
config :sms_c,
  # Deletar corpo da mensagem após entrega bem-sucedida
  delete_message_body_after_delivery: true,

  # Ocultar corpo da mensagem na UI
  hide_message_body_in_ui: true,

  # Ocultar corpo da mensagem em exportações
  hide_message_body_in_export: true
```

2. Mascaramento de Números de Telefone

Para análises que não requerem números completos:

```
-- Mascarar os últimos 4 dígitos dos números de telefone
SELECT
  CONCAT(SUBSTRING(calling_number, 1, LENGTH(calling_number) - 4),
  'XXXX') AS masked_calling,
  CONCAT(SUBSTRING(called_number, 1, LENGTH(called_number) - 4),
  'XXXX') AS masked_called,
  COUNT(*) AS message_count
FROM cdrs
GROUP BY masked_calling, masked_called;
```

3. Criptografia de Banco de Dados

Ativar criptografia em repouso para o servidor de banco de dados:

MySQL:

```
-- Ativar criptografia de tabela
ALTER TABLE cdrs ENCRYPTION='Y';
```

PostgreSQL: Use criptografia de dados transparente do PostgreSQL (TDE) ou criptografia em nível de sistema de arquivos.

4. Controles de Acesso

Restringir o acesso à tabela CDR:

```
-- Criar usuário de faturamento somente leitura
CREATE USER 'billing_ro'@'%' IDENTIFIED BY 'secure_password';
GRANT SELECT ON sms_c.cdrs TO 'billing_ro'@'%';

-- Criar usuário de análise limitado (sem acesso ao corpo da mensagem)
CREATE USER 'analytics'@'%' IDENTIFIED BY 'secure_password';
GRANT SELECT (id, message_id, calling_number, called_number,
source_smsc,
                dest_smsc, submission_time, delivery_time, status,
                delivery_attempts, message_parts)
ON sms_c.cdrs TO 'analytics'@'%';
```

Retenção e Arquivamento

Políticas de Retenção

Defina períodos de retenção com base em requisitos regulatórios e de negócios:

Indústria	Retenção Típica	Base Regulatória
Telecom (EUA)	18-24 meses	FCC, leis estaduais
Telecom (UE)	6 meses - 2 anos	GDPR, ePrivacy
Financeira	5-7 anos	SOX, SEC
Saúde	6 anos	HIPAA

Estratégia de Arquivamento

1. Particionamento por Data (MySQL 8.0+, PostgreSQL 11+)

```
-- Particionamento MySQL por mês
ALTER TABLE cdrs PARTITION BY RANGE (TO_DAYS(submission_time)) (
  PARTITION p202510 VALUES LESS THAN (TO_DAYS('2025-11-01')),
  PARTITION p202511 VALUES LESS THAN (TO_DAYS('2025-12-01')),
  PARTITION p202512 VALUES LESS THAN (TO_DAYS('2026-01-01')),
  PARTITION p_future VALUES LESS THAN MAXVALUE
);

-- Deletar partição antiga (arquivamento rápido)
ALTER TABLE cdrs DROP PARTITION p202510;
```

2. Arquivar para Armazenamento Frio

```
-- Exportar CDRs antigos para tabela de arquivamento
CREATE TABLE cdrs_archive LIKE cdrs;

INSERT INTO cdrs_archive
SELECT * FROM cdrs
WHERE submission_time < DATE_SUB(NOW(), INTERVAL 2 YEAR);

-- Verificar e deletar da tabela principal
DELETE FROM cdrs
WHERE submission_time < DATE_SUB(NOW(), INTERVAL 2 YEAR);
```

3. Script de Limpeza Automatizado

```
#!/bin/bash
# cleanup_old_cdrs.sh - Executar via cron

MYSQL_USER="cleanup_user"
MYSQL_PASS="secure_password"
```



```

MYSQL_DB="sms_c"
RETENTION_DAYS=730 # 2 anos

# Arquivar registros antigos
mysql -u"$MYSQL_USER" -p"$MYSQL_PASS" "$MYSQL_DB" <<EOF
INSERT INTO cdrs_archive
SELECT * FROM cdrs
WHERE submission_time < DATE_SUB(NOW(), INTERVAL $RETENTION_DAYS DAY)
LIMIT 100000;

DELETE FROM cdrs
WHERE submission_time < DATE_SUB(NOW(), INTERVAL $RETENTION_DAYS DAY)
LIMIT 100000;
EOF

```

Entrada do Cron:

```

# Executar diariamente às 2 AM
0 2 * * * /usr/local/bin/cleanup_old_cdrs.sh >> /var/log/sms_c/
cleanup.log 2>&1

```

Integração de Faturamento

Esquema de Tabela de Tarifas

Crie uma tabela de tarifas separada para faturamento:

```

CREATE TABLE billing_rates (
  id INT AUTO_INCREMENT PRIMARY KEY,
  destination_prefix VARCHAR(20) NOT NULL,
  description VARCHAR(255),
  rate_per_message DECIMAL(10, 6) NOT NULL,
  rate_per_segment DECIMAL(10, 6) NOT NULL,
  currency VARCHAR(3) DEFAULT 'USD',
  effective_date DATE NOT NULL,
  expiry_date DATE,
  INDEX idx_prefix (destination_prefix),
  INDEX idx_dates (effective_date, expiry_date)
);

-- Tarifas de exemplo
INSERT INTO billing_rates (destination_prefix, description,
rate_per_message, rate_per_segment, effective_date) VALUES
('+1', 'Estados Unidos/Canadá', 0.0050, 0.0050, '2025-01-01'),
('+44', 'Reino Unido', 0.0080, 0.0080, '2025-01-01'),
('+61', 'Austrália', 0.0100, 0.0100, '2025-01-01'),
('+', 'Padrão internacional', 0.0150, 0.0150, '2025-01-01');

```

Consulta de Faturamento

Junte CDRs com tarifas para faturamento:

```
SELECT
    DATE(c.submission_time) AS date,
    c.dest_smsc AS route,
    LEFT(c.called_number,
        CASE
            WHEN c.called_number LIKE '+1%' THEN 2
            WHEN c.called_number LIKE '+%' THEN
LENGTH(SUBSTRING_INDEX(c.called_number, '', 4))
            ELSE 0
        END
    ) AS destination_prefix,
    COUNT(*) AS message_count,
    SUM(c.message_parts) AS segment_count,
    COALESCE(r.rate_per_segment, 0.015) AS rate,
    SUM(c.message_parts) * COALESCE(r.rate_per_segment, 0.015) AS
total_cost
FROM cdrs c
LEFT JOIN billing_rates r ON c.called_number LIKE
CONCAT(r.destination_prefix, '%')
    AND c.submission_time >= r.effective_date
    AND (r.expiry_date IS NULL OR c.submission_time < r.expiry_date)
WHERE c.status = 'delivered'
    AND c.submission_time >= '2025-10-01'
    AND c.submission_time < '2025-11-01'
GROUP BY date, route, destination_prefix
ORDER BY date DESC, total_cost DESC;
```

Exportação para Sistemas de Faturamento

Exportação CSV:

```
mysql -u billing_ro -p -D sms_c -e "
SELECT
    id,
    message_id,
    calling_number,
    called_number,
    dest_smsc,
    submission_time,
    delivery_time,
    status,
    message_parts
FROM cdrs
WHERE submission_time >= '2025-10-01'
```

```
AND submission_time < '2025-11-01'  
AND status = 'delivered'  
" --batch --silent | sed 's/\t/,/g' > billing_export_202510.csv
```

Veja Também

- [Guia de Configuração](#) - Configurar as definições de exportação do CDR
- [Guia de Operações](#) - Procedimentos de manutenção do CDR
- [Referência da API](#) - Consultar CDRs via API REST



Referência de Configuração do SMS-C

[← Voltar para o Índice da Documentação](#) | [README Principal](#)

Referência completa para todas as opções de configuração do SMS-C com exemplos para cenários de implantação comuns.

Índice

- [Arquivos de Configuração](#)
- [Configuração do Banco de Dados](#)
- [Configuração da API](#)
- [Configuração da Interface Web](#)
- [Configuração do Cluster](#)
- [Configuração da Fila de Mensagens](#)
- [Configuração de Cobrança](#)
- [Configuração ENUM](#)
- [Configuração de Tradução de Números](#)
- [Configuração de Roteamento](#)
- [Configuração de Ajuste de Desempenho](#)
- [Configuração de Registro](#)
- [Cenários Comuns de Configuração](#)

Arquivos de Configuração

O SMS-C usa três arquivos de configuração principais:

config/config.exs

Configuração estática carregada em tempo de compilação. Contém:

- Padrões globais da aplicação
- Configuração do logger
- Configurações de desenvolvimento/teste
- Parâmetros de ajuste de desempenho

config/runtime.exs

Configuração em tempo de execução carregada na inicialização. Contém:

- Configurações de conexão do banco de dados

- Configuração do cluster
- Integração com serviços externos (OCS, ENUM)
- Rotas iniciais e regras de tradução
- Configurações específicas do ambiente

config/prod.exs (opcional)

Substituições específicas de produção.

Melhor Prática: Use variáveis de ambiente em `runtime.exs` para valores sensíveis como senhas e chaves de API.

Configuração de Armazenamento de CDR SQL

O SMS-C usa **Mnesia** para dados operacionais (fila de mensagens, regras de roteamento, traduções de números) e suporta **bancos de dados SQL externos** para armazenamento de CDR (Registro de Detalhes de Chamadas) a longo prazo, cobrança e análises.

Bancos de Dados SQL Suportados

O sistema suporta os seguintes bancos de dados SQL para exportação de CDR:

Banco de Dados	Versão	Adaptador	Porta Padrão	Melhor Para
MySQL	8.0+	<code>Ecto.Adapters.MyXQL</code>	3306	Uso geral, confiabilidade comprovada
MariaDB	10.5+	<code>Ecto.Adapters.MyXQL</code>	3306	Compatível com MySQL, código aberto
PostgreSQL	13+	<code>Ecto.Adapters.Postgres</code>	5432	Recursos avançados, suporte a JSON

Nota: O Mnesia é usado automaticamente para dados operacionais (fila de mensagens, roteamento, traduções) e não requer configuração. O banco de dados SQL é **apenas** usado para exportação de CDR e armazenamento a longo prazo.

Configuração MySQL / MariaDB

```
# config/runtime.exs
config :sms_c, SmsC.Repo,
  adapter: Ecto.Adapters.MyXQL,
  username: System.get_env("DB_USERNAME") || "sms_user",
  password: System.get_env("DB_PASSWORD") || "secure_password",
  hostname: System.get_env("DB_HOSTNAME") || "localhost",
  port: String.to_integer(System.get_env("DB_PORT") || "3306"),
  database: System.get_env("DB_NAME") || "sms_c_prod",
  pool_size: String.to_integer(System.get_env("DB_POOL_SIZE") ||
```

```
"20")
```

Configuração PostgreSQL

```
# config/runtime.exs
config :sms_c, SmsC.Repo,
  adapter: Ecto.Adapters.Postgres,
  username: System.get_env("DB_USERNAME") || "sms_user",
  password: System.get_env("DB_PASSWORD") || "secure_password",
  hostname: System.get_env("DB_HOSTNAME") || "localhost",
  port: String.to_integer(System.get_env("DB_PORT") || "5432"),
  database: System.get_env("DB_NAME") || "sms_c_prod",
  pool_size: String.to_integer(System.get_env("DB_POOL_SIZE") ||
"20")
```

Escolhendo um Banco de Dados SQL

MySQL/MariaDB - Recomendado para a maioria das implantações:

- Excelente desempenho para gravações de CDR
- Confiabilidade comprovada em ambientes de telecomunicações
- Amplo suporte de ferramentas para sistemas de cobrança
- Configuração de replicação fácil

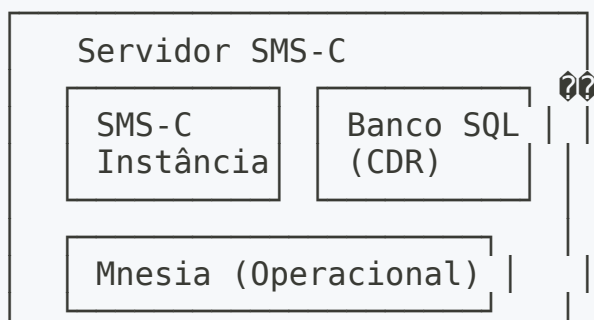
PostgreSQL - Considere se você precisar:

- Recursos avançados de JSON/JSONB para análises
- Consultas complexas em dados de CDR
- Infraestrutura existente do PostgreSQL
- PostGIS para análise geográfica

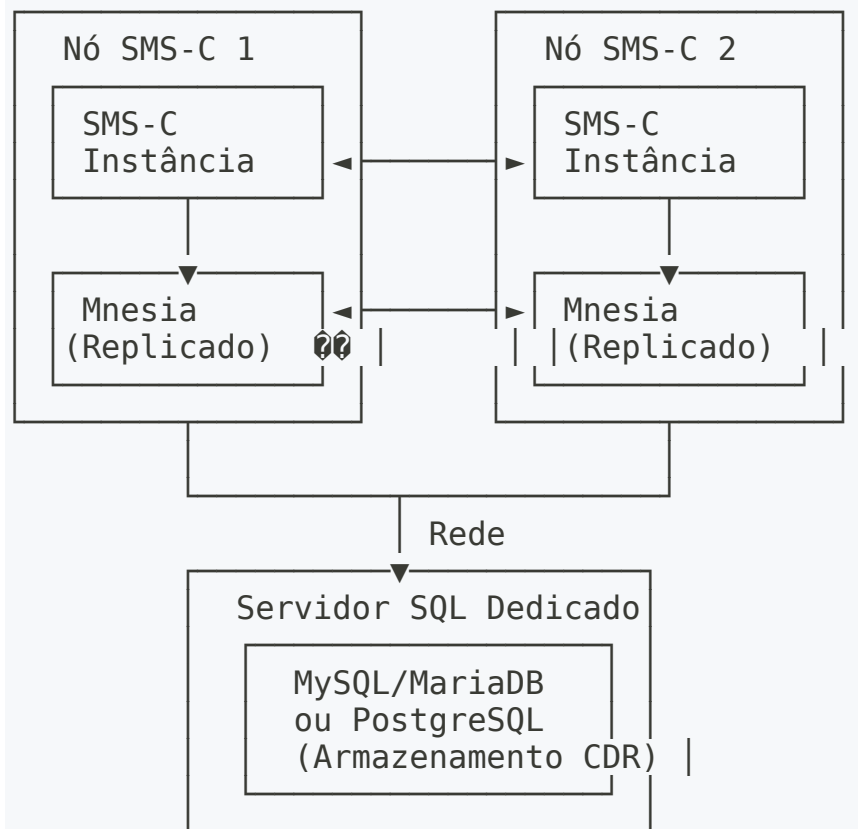
Topologias de Implantação

Importante: O banco de dados SQL CDR pode ser executado em um **servidor separado** de suas instâncias SMS-C. Esta é a abordagem recomendada para implantações em produção.

Implantação em Servidor Único (Desenvolvimento/Teste):



Implantação Distribuída (Produção - Recomendado):



Benefícios de um Servidor SQL Separado:

- **Isolamento de Desempenho:** Gravações de CDR não impactam o processamento de mensagens
- **Escalabilidade:** Escala independentemente o banco de dados e o processamento de mensagens
- **Confiabilidade:** Manutenção do banco de dados não afeta o tempo de atividade do SMS-C
- **Gerenciamento de Dados:** Armazenamento centralizado de CDR para várias instâncias SMS-C
- **Flexibilidade de Backup:** Agendas de backup independentes e políticas de retenção

Diretrizes de Tamanho do Pool

Carga de Trabalho	Tamanho do Pool	Descrição
Desenvolvimento	5-10	Concorrência mínima
Baixo Volume (< 100 msg/sec)	10-15	Pequenas implantações
Volume Médio (100-1000 msg/	20-30	Produção típica

Carga de Trabalho	Tamanho do Pool	Descrição
sec)		
Alto Volume (> 1000 msg/sec)	40-100	Cenários de alto rendimento

Cálculo: `pool_size = (operações DB concorrentes esperadas) * 1.5`

Exemplos de Conexão ao Banco de Dados

Usando Variáveis de Ambiente (Recomendado para Produção):

```
# Definir variáveis de ambiente
export DB_USERNAME=sms_prod_user
export DB_PASSWORD=strong_password_here
export DB_HOSTNAME=db-primary.internal.example.com
export DB_PORT=3306
export DB_NAME=sms_c_production
export DB_POOL_SIZE=30
```

Configuração Direta (Apenas para Desenvolvimento):

```
config :sms_c, SmsC.Repo,
  username: "dev_user",
  password: "dev_password",
  hostname: "localhost",
  database: "sms_c_dev",
  pool_size: 5
```

Monitoramento do Pool de Conexões

Monitore o uso do pool através de métricas do Prometheus:

- `ecto_pools_queue_time` - Tempo de espera pela conexão
- `ecto_pools_query_time` - Tempo de execução da consulta
- `ecto_pools_connected_count` - Conexões ativas

Alerta se o tempo de espera exceder 100ms consistentemente - indica necessidade de um pool maior.

Configuração da API

A API REST fornece capacidades de envio e gerenciamento de mensagens.

Configuração Básica da API

```
# config/runtime.exs
```



```
config :api_ex,  
  port: String.to_integer(System.get_env("API_PORT") || "8443"),  
  listen_ip: System.get_env("API_LISTEN_IP") || "0.0.0.0",  
  enable_tls: System.get_env("API_ENABLE_TLS") != "false"
```

Configuração TLS/SSL

Configuração de Produção com TLS (Recomendado):

```
config :api_ex,  
  port: 8443,  
  listen_ip: "0.0.0.0",  
  enable_tls: true,  
  tls_cert_path: "/etc/sms_c/certs/server.crt",  
  tls_key_path: "/etc/sms_c/certs/server.key"
```

Configuração de Desenvolvimento sem TLS:

```
config :api_ex,  
  port: 8080,  
  listen_ip: "127.0.0.1",  
  enable_tls: false
```

Configuração do Certificado da API

Gere um certificado autoassinado para testes:

```
# Criar diretório de certificados  
mkdir -p priv/cert  
  
# Gerar chave privada  
openssl genrsa -out priv/cert/server.key 2048  
  
# Gerar solicitação de assinatura de certificado  
openssl req -new -key priv/cert/server.key -out priv/cert/server.csr \  
  -subj "/C=US/ST=State/L=City/O=Organization/CN=sms-api.example.com"  
  
# Gerar certificado autoassinado (válido por 365 dias)  
openssl x509 -req -days 365 -in priv/cert/server.csr \  
  -signkey priv/cert/server.key -out priv/cert/server.crt  
  
# Definir permissões  
chmod 600 priv/cert/server.key  
chmod 644 priv/cert/server.crt
```

Para produção, use certificados de uma CA confiável (Let's Encrypt, CA comercial, etc.).

Controle de Acesso da API

Whitelist de IP (Firewall de Aplicação):

```
# Usando iptables (Linux)
iptables -A INPUT -p tcp --dport 8443 -s 10.0.0.0/8 -j ACCEPT
iptables -A INPUT -p tcp --dport 8443 -j DROP

# Usando firewalld (Red Hat/CentOS)
firewall-cmd --permanent --add-rich-rule='rule family="ipv4" source
address="10.0.0.0/8" port protocol="tcp" port="8443" accept'
firewall-cmd --reload
```

Autenticação por Chave da API (Nível de Aplicação):

Configure via plug personalizado no roteador - veja o Guia de Operações para detalhes de implementação.

Configuração da Interface Web

A interface web fornece gerenciamento de rotas, navegação de mensagens e monitoramento.

Configuração Básica da Interface Web

```
# config/runtime.exs
config :control_panel,
  port: String.to_integer(System.get_env("WEB_PORT") || "80"),
  hostname: System.get_env("WEB_HOSTNAME") || "localhost",
  enable_tls: System.get_env("WEB_ENABLE_TLS") == "true"
```

Configuração da Interface Web de Produção

```
config :control_panel,
  port: 443,
  hostname: "sms-admin.example.com",
  enable_tls: true,
  tls_cert_path: "/etc/sms_c/certs/web.crt",
  tls_key_path: "/etc/sms_c/certs/web.key"
```

Configuração de Proxy Reverso (Recomendado)

Use Nginx ou Apache como proxy reverso para segurança e recursos adicionais:

Exemplo de Configuração do Nginx:

```
upstream sms_web {
```

```

server 127.0.0.1:4000;
keepalive 32;
}

server {
    listen 80;
    server_name sms-admin.example.com;
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl http2;
    server_name sms-admin.example.com;

    ssl_certificate /etc/letsencrypt/live/sms-admin.example.com/
fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/sms-admin.example.com/
privkey.pem;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;

    # Autenticação básica para segurança adicional
    auth_basic "SMS-C Admin";
    auth_basic_user_file /etc/nginx/.htpasswd;

    location / {
        proxy_pass http://sms_web;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    # Suporte a WebSocket para LiveView
    location /live {
        proxy_pass http://sms_web;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_read_timeout 86400;
    }
}

```

Configuração do Cluster

O SMS-C suporta clustering de múltiplos nós para alta disponibilidade e distribuição de carga.

Configuração de Um Único Nó

```
# config/runtime.exs
config :sms_c,
  cluster_nodes: [], # Lista vazia = modo de nó único
  smsc_node_name: "node1"
```

Cluster Estático de Múltiplos Nós

```
# Nó 1: config/runtime.exs
config :sms_c,
  cluster_nodes: [
    "sms@node1.internal.example.com",
    "sms@node2.internal.example.com",
    "sms@node3.internal.example.com"
  ],
  smsc_node_name: "node1"

# Nó 2: config/runtime.exs
config :sms_c,
  cluster_nodes: [
    "sms@node1.internal.example.com",
    "sms@node2.internal.example.com",
    "sms@node3.internal.example.com"
  ],
  smsc_node_name: "node2"
```

Descoberta Automática Baseada em DNS

```
config :sms_c,
  dns_cluster_query: "sms-cluster.internal.example.com",
  smsc_node_name: System.get_env("NODE_NAME") || "node1"
```

Configuração de DNS para Descoberta Automática:

```
# Configurar registros SRV ou A para nós do cluster
# Registro SRV (preferido):
_sms._tcp.sms-cluster.internal.example.com. IN SRV 0 0 0
node1.internal.example.com.
_sms._tcp.sms-cluster.internal.example.com. IN SRV 0 0 0
node2.internal.example.com.
_sms._tcp.sms-cluster.internal.example.com. IN SRV 0 0 0
```

```
node3.internal.example.com.
```

```
# Registros A (alternativa):  
sms-cluster.internal.example.com. IN A 10.0.1.10  
sms-cluster.internal.example.com. IN A 10.0.1.11  
sms-cluster.internal.example.com. IN A 10.0.1.12
```

Configuração de Distribuição Erlang

Iniciar Nós com Nomes Apropriados:

```
# Nó 1  
export NODE_NAME=sms@node1.internal.example.com  
export ERLANG_COOKIE=shared_secret_cookie_here  
elixir --name $NODE_NAME --cookie $ERLANG_COOKIE -S mix phx.server  
  
# Nó 2  
export NODE_NAME=sms@node2.internal.example.com  
export ERLANG_COOKIE=shared_secret_cookie_here  
elixir --name $NODE_NAME --cookie $ERLANG_COOKIE -S mix phx.server
```

Importante: Todos os nós em um cluster DEVEM usar o mesmo cookie Erlang por motivos de segurança.

Requisitos de Rede do Cluster

Abra estas portas entre os nós do cluster:

Faixa de Portas		Protocolo	Propósito
4369		TCP	Erlang Port Mapper Daemon (EPMD)
9100-9200		TCP	Distribuição Erlang

Exemplo de Configuração de Firewall:

```
# Permitir tráfego do cluster da rede interna  
iptables -A INPUT -p tcp -s 10.0.0.0/8 --dport 4369 -j ACCEPT  
iptables -A INPUT -p tcp -s 10.0.0.0/8 --dport 9100:9200 -j ACCEPT
```

Configuração da Fila de Mensagens

Controla o comportamento de retenção e expiração de mensagens.

Expiração de Mensagens

```
# config/runtime.exs  
config :sms_c,  
  dead_letter_time_minutes: 1440 # 24 horas
```

Valores Comuns:

- **60** - 1 hora (teste/desenvolvimento)
- **1440** - 24 horas (produção típica)
- **4320** - 3 dias (retenção estendida)
- **10080** - 7 dias (retenção máxima)

Mensagens mais antigas que este valor tornam-se indesejáveis e são marcadas para limpeza.

Configuração de Tentativas de Entrega

O comportamento de tentativas utiliza retrocesso exponencial:

Delay de Tentativa = $2^{(\text{contagem_tentativas})}$ minutos

Tentativa	Atraso
1	2 minutos
2	4 minutos
3	8 minutos
4	16 minutos
5	32 minutos
6	64 minutos
7	128 minutos
8	256 minutos

Tentativas máximas antes da carta morta: Limitadas por `dead_letter_time_minutes`.

Configuração de Limpeza

```
# config/config.exs
config :sms_c,
  cleanup_interval_minutes: 10,
  fingerprint_ttl_minutes: 5,
  event_ttl_days: 7
```

Intervalos de Limpeza:

- **cleanup_interval_minutes**: Com que frequência o trabalhador de limpeza é executado (padrão: 10)
- **fingerprint_ttl_minutes**: Janela de detecção de duplicatas (padrão: 5)
- **event_ttl_days**: Retenção de logs de eventos (padrão: 7)

Configuração de Cobrança

Integração com OCS para cobrança e faturamento online.

Habilitar Cobrança

```
# config/runtime.exs
config :sms_c,
  default_charging_enabled: true,
  ocs_url: "http://ocs.internal.example.com:2080/jsonrpc",
  ocs_tenant: "sms.example.com",
  ocs_destination: "default",
  ocs_source: "sms_platform",
  ocs_subject: "sms_user",
  ocs_account: "default_account"
```

Desabilitar Cobrança

```
# config/runtime.exs
config :sms_c,
  default_charging_enabled: false
```

Quando desabilitado, todas as mensagens são processadas sem verificações de cobrança.

Configuração de Cobrança por Inquilino

```
config :sms_c,
  ocs_url: System.get_env("OCS_URL") || "http://localhost:2080/
jsonrpc",
  ocs_tenant: System.get_env("OCS_TENANT") || "tenant1.example.com",
  ocs_account: System.get_env("OCS_ACCOUNT") || "default"
```

Variáveis de Ambiente por Inquilino:

```
# Inquilino 1
export OCS_TENANT=tenant1.example.com
export OCS_ACCOUNT=tenant1_account

# Inquilino 2
export OCS_TENANT=tenant2.example.com
export OCS_ACCOUNT=tenant2_account
```

Comportamento em Caso de Falha de Cobrança

Configure o que acontece quando a cobrança falha:

```
config :sms_c,
  charging_failure_action: :allow # ou :deny
```

- **:allow** - Processar mensagem mesmo se a cobrança falhar (registrar erro)
- **:deny** - Rejeitar mensagem se a cobrança falhar

Exemplo de Conexão OCS

Testar Conectividade OCS:

```
# Testar API OCS
curl -X POST http://ocs.internal.example.com:2080/jsonrpc \
  -H "Content-Type: application/json" \
  -d '{
    "method": "SessionSv1.AuthorizeEvent",
    "params": [{
      "Tenant": "sms.example.com",
      "Account": "test_account",
      "Destination": "1234567890",
      "Usage": 100
    }],
    "id": 1
  }'
```

Resposta esperada:

```
{
  "id": 1,
  "result": {
    "Attributes": {},
    "MaxUsage": 100,
    ...
  }
}
```

Configuração ENUM

Pesquisas de números E.164 baseadas em DNS para roteamento inteligente.

Desabilitar ENUM (Padrão)

```
# config/runtime.exs
config :sms_c,
  enum_enabled: false
```

Habilitar ENUM com DNS Padrão

```
config :sms_c,
  enum_enabled: true,
  enum_domains: ["e164.arpa", "e164.org"],
  enum_dns_servers: [], # Usar DNS padrão do sistema
  enum_timeout: 5000 # 5 segundos
```


Habilitar ENUM com Servidores DNS Personalizados

```
config :sms_c,  
  enum_enabled: true,  
  enum_domains: ["e164.internal.example.com", "e164.arpa"],  
  enum_dns_servers: [  
    {"10.0.1.53", 53}, # Servidor DNS interno  
    {"8.8.8.8", 53},   # DNS Público do Google (fallback)  
    {"1.1.1.1", 53}    # DNS do Cloudflare (fallback)  
  ],  
  enum_timeout: 3000 # 3 segundos (failover mais rápido)
```

Prioridade do Domínio ENUM

Os domínios são consultados em ordem até uma pesquisa bem-sucedida:

```
config :sms_c,  
  enum_domains: [  
    "e164.internal.example.com", # Tentar interno primeiro  
    "e164.carrier.net",         # Depois carrier  
    "e164.arpa"                 # Depois registro público  
  ]
```

Ajuste de Desempenho ENUM

Para Redes de Baixa Latência:

```
enum_timeout: 2000 # 2 segundos
```

Para Alta Latência/Links de Satélite:

```
enum_timeout: 10000 # 10 segundos
```

Exemplo de Configuração DNS ENUM

Configurar Zona ENUM Privada (formato BIND9):

```
; Arquivo de zona para e164.internal.example.com  
$ORIGIN e164.internal.example.com.  
$TTL 300  
  
; Número: +1-555-0100 torna-se  
0.0.1.0.5.5.5.1.e164.internal.example.com  
0.0.1.0.5.5.5.1.e164.internal.example.com. IN NAPTR 100 10 "u"  
"E2U+sip" "!.*$.!sip:15550100@voip-gateway.example.com!" .  
0.0.1.0.5.5.5.1.e164.internal.example.com. IN NAPTR 100 20 "u"  
"E2U+pstn" "!.*$.!pstn:gateway-a.example.com!" .
```

```
; Número: +1-555-0200
0.0.2.0.5.5.5.1.e164.internal.example.com. IN NAPTR 100 10 "u"
"E2U+sip" "!.*${sip:15550200@voip-gateway.example.com!" .
```

Testar Resolução ENUM:

```
# Consultar domínio ENUM
dig @10.0.1.53 NAPTR 0.0.1.0.5.5.5.1.e164.internal.example.com

# A saída esperada inclui registros NAPTR:
# 0.0.1.0.5.5.5.1.e164.internal.example.com. 300 IN NAPTR 100 10 "u"
"E2U+sip" "!.*${sip:15550100@voip-gateway.example.com!" .
```

Configuração de Tradução de Números

Normalização de números baseada em regex aplicada antes do roteamento.

Desabilitar Tradução de Números

```
# config/runtime.exs
config :sms_c,
  translation_rules: []
```

Exemplos Básicos de Tradução de Números

Adicionar Código de País a Números Locais:

```
config :sms_c,
  translation_rules: [
    %{
      calling_prefix: nil,
      called_prefix: "",
      source_smsc: nil,
      calling_match: "^(\\d{10})$",           # Correspondência de
números de 10 dígitos
      calling_replace: "+1\\1",               # Prepend +1
      called_match: "^(\\d{10})$",
      called_replace: "+1\\1",
      priority: 100,
      description: "Adicionar +1 a números norte-americanos de 10
dígitos",
      enabled: true
    }
  ]
```

Normalizar Formato Internacional:

```
%{
```

```

    calling_prefix: nil,
    called_prefix: nil,
    source_smsc: nil,
    calling_match: "^00(\d+)$",          # Correspondência de
prefixo 00
    calling_replace: "+\1",              # Substituir por +
    called_match: "^00(\d+)$",
    called_replace: "+\1",
    priority: 10,
    description: "Converter prefixo internacional 00 em +",
    enabled: true
}

```

Remover Caracteres de Formatação:

```

%{
    calling_prefix: nil,
    called_prefix: nil,
    source_smsc: nil,
    calling_match: "^+?1?[\s\-\.\(\)]*(\d{3})[\s\-\.\(\)]*(\d{3})[\s\-\.\(\)]*(\d{4})$",
    calling_replace: "+1\1\2\3",
    called_match: "^+?1?[\s\-\.\(\)]*(\d{3})[\s\-\.\(\)]*(\d{3})[\s\-\.\(\)]*(\d{4})$",
    called_replace: "+1\1\2\3",
    priority: 50,
    description: "Normalizar formatação de números de telefone dos
EUA",
    enabled: true
}

```

Tradução Específica de Operadora

Remoção de Código de Roteamento:

```

%{
    calling_prefix: nil,
    called_prefix: "101",                # Apenas para prefixo
101
    source_smsc: "carrier_a",            # Apenas desta
operadora
    calling_match: nil,                   # Não alterar chamada
    calling_replace: nil,
    called_match: "^101(\d+)$",           # Remover código de
roteamento 101
    called_replace: "\1",
    priority: 5,
    description: "Remover código de roteamento da operadora do número

```

```
chamado",
  enabled: true
}
```

Tradução de Múltiplas Regras

As regras são avaliadas em ordem de prioridade (número menor = maior prioridade):

```
config :sms_c,
  translation_rules: [
    # Prioridade 1: Regras mais específicas primeiro
    %{
      calling_prefix: "1555",
      called_prefix: nil,
      source_smsc: nil,
      calling_match: "^(1555\d{7})$",
      calling_replace: "+\1",
      called_match: nil,
      called_replace: nil,
      priority: 1,
      description: "Normalização de número premium",
      enabled: true
    },

    # Prioridade 50: Regras gerais
    %{
      calling_prefix: nil,
      called_prefix: nil,
      source_smsc: nil,
      calling_match: "^(\\d{10})$",
      calling_replace: "+1\1",
      called_match: "^(\\d{10})$",
      called_replace: "+1\1",
      priority: 50,
      description: "Normalização geral de 10 dígitos",
      enabled: true
    }
  ]
]
```

Configuração de Roteamento

Regras de roteamento iniciais carregadas na primeira inicialização. Veja o [Guia de Roteamento SMS](#) para a documentação completa de roteamento.

Carregar Rotas da Configuração

```
# config/runtime.exs
config :sms_c,
  sms_routes: [
    # Exemplo de roteamento geográfico
    %{
      calling_prefix: nil,
      called_prefix: "+1",
      source_smsc: nil,
      dest_smsc: "north_america_gateway",
      source_type: nil,
      enum_domain: nil,
      auto_reply: false,
      auto_reply_message: nil,
      drop: false,
      charged: :default,
      weight: 100,
      priority: 50,
      description: "Roteamento para a América do Norte",
      enabled: true
    },

    # Exemplo de roteamento balanceado
    %{
      calling_prefix: nil,
      called_prefix: "+44",
      source_smsc: nil,
      dest_smsc: "uk_gateway_1",
      source_type: nil,
      enum_domain: nil,
      auto_reply: false,
      auto_reply_message: nil,
      drop: false,
      charged: :default,
      weight: 70,
      priority: 50,
      description: "Gateway primário do Reino Unido (70%)",
      enabled: true
    },
    %{
      calling_prefix: nil,
      called_prefix: "+44",
      source_smsc: nil,
      dest_smsc: "uk_gateway_2",
      source_type: nil,
      enum_domain: nil,
      auto_reply: false,
```

```

    auto_reply_message: nil,
    drop: false,
    charged: :default,
    weight: 30,
    priority: 50,
    description: "Gateway de backup do Reino Unido (30%)",
    enabled: true
  }
]

```

Pular Carregamento Inicial de Rotas

```

# Não carregar rotas da configuração (gerenciar apenas via Interface
Web)
config :sms_c,
  sms_routes: []

```

As rotas definidas na configuração SÃO carregadas apenas se a tabela de roteamento estiver vazia (primeira inicialização).

Configuração de Ajuste de Desempenho

Veja o [Guia de Ajuste de Desempenho](#) para estratégias detalhadas de otimização.

Trabalhador de Inserção em Lote

```

# config/config.exs
config :sms_c,
  batch_insert_batch_size: 100,           # Mensagens por lote
  batch_insert_flush_interval_ms: 100    # Tempo máximo de espera em
ms

```

Perfis de Desempenho:

Perfil	Tamanho do Lote	Intervalo	Vazão	Latência
Alto Volume	200	200ms	~5,000 msg/sec	Até 200ms
Balanceado	100	100ms	~4,500 msg/sec	Até 100ms
Baixa Latência	50	20ms	~3,000 msg/sec	Até 20ms
Tempo Real	10	10ms	~1,500 msg/sec	Até 10ms

Configuração de Registro

Níveis de Registro

```

# config/config.exs
config :logger, :console,

```

```
level: :info, # :debug, :info, :warning, :error
format: "$time $metadata[$level] $message\n",
metadata: [:request_id, :message_id, :route_id]
```

Recomendado para Produção: :info ou :warning **Recomendado para Desenvolvimento:** :debug

Destinos de Saída de Registro

Apenas Console (Desenvolvimento):

```
config :logger,
  backends: [:console]
```

Logger de Arquivo (Produção):

```
config :logger,
  backends: [:console, {LoggerFileBackend, :file_log}]

config :logger, :file_log,
  path: "/var/log/sms_c/application.log",
  level: :info,
  format: "$time $metadata[$level] $message\n",
  metadata: [:request_id, :message_id]
```

Rotação de Registro

Usando logrotate (Linux):

```
# /etc/logrotate.d/sms_c
/var/log/sms_c/*.log {
    daily
    rotate 30
    compress
    delaycompress
    notifempty
    create 0644 sms_user sms_group
    sharedscripts
    postrotate
        # Sinalizar aplicação para reabrir arquivo de log
        systemctl reload sms_c
    endscript
}
```

Cenários Comuns de Configuração

Agregador de Alto Volume

Otimize para máxima vazão (5,000+ mensagens/segundo):

```
# Banco de Dados
config :sms_c, SmsC.Repo,
  pool_size: 50

# Trabalhador em lote
config :sms_c,
  batch_insert_batch_size: 200,
  batch_insert_flush_interval_ms: 200

# Retenção de mensagens
config :sms_c,
  dead_letter_time_minutes: 1440 # 24 horas

# Cobrança (desabilitada para desempenho)
config :sms_c,
  default_charging_enabled: false

# Limpeza (intervalos estendidos)
config :sms_c,
  cleanup_interval_minutes: 30
```

Mensagens em Tempo Real para Empresas

Otimize para baixa latência (< 20ms):

```
# Banco de Dados
config :sms_c, SmsC.Repo,
  pool_size: 20

# Trabalhador em lote (baixa latência)
config :sms_c,
  batch_insert_batch_size: 20,
  batch_insert_flush_interval_ms: 10

# Retenção de mensagens
config :sms_c,
  dead_letter_time_minutes: 4320 # 3 dias

# Cobrança (habilitada)
config :sms_c,
  default_charging_enabled: true,
  ocs_url: "http://ocs.local:2080/jsonrpc"
```


Desenvolvimento/Teste

Otimize para depuração e visibilidade:

```
# Banco de Dados
config :sms_c, SmsC.Repo,
  pool_size: 5

# Trabalhador em lote (imediato)
config :sms_c,
  batch_insert_batch_size: 1,
  batch_insert_flush_interval_ms: 10

# Registro (verbose)
config :logger, :console,
  level: :debug

# Retenção de mensagens (curta)
config :sms_c,
  dead_letter_time_minutes: 60 # 1 hora

# Cobrança (desabilitada)
config :sms_c,
  default_charging_enabled: false
```

Provedor de Serviço Multi-Inquilino

Configuração separada por inquilino:

```
# Ambiente do Inquilino 1
export DB_NAME=sms_c_tenant1
export OCS_TENANT=tenant1.example.com
export OCS_ACCOUNT=tenant1_account
export NODE_NAME=sms_tenant1@node1.example.com

# Ambiente do Inquilino 2
export DB_NAME=sms_c_tenant2
export OCS_TENANT=tenant2.example.com
export OCS_ACCOUNT=tenant2_account
export NODE_NAME=sms_tenant2@node1.example.com
```

Redundância Geográfica

Cluster em regiões:

```
# Cluster da Costa Leste dos EUA
config :sms_c,
  cluster_nodes: [
```

```

    : "sms@us-east-1a.example.com",
    : "sms@us-east-1b.example.com",
    : "sms@us-west-1a.example.com" # Cross-region para DR
  ],
  smsc_node_name: "us-east-1a"

```

Validação de Configuração

Teste a configuração antes da implantação:

```

# Verificar sintaxe da configuração
mix compile

# Validar conexão com o banco de dados
mix ecto.create
mix ecto.migrate

# Testar conectividade OCS (se habilitada)
curl -X POST http://localhost:2080/jsonrpc -H "Content-Type:
application/json" \
  -d '{"method":"SessionSv1.Ping","params":[],"id":1}'

# Iniciar aplicação em modo interativo
iex -S mix phx.server

```

Referência de Variáveis de Ambiente

Variáveis de ambiente comuns usadas na configuração:

Variável	Propósito	Exemplo
DB_USERNAME	Nome de usuário do banco de dados	sms_prod_user
DB_PASSWORD	Senha do banco de dados	strong_password
DB_HOSTNAME	Host do banco de dados	db.internal.example.com
DB_PORT	Porta do banco de dados	3306
DB_NAME	Nome do banco de dados	sms_c_production
DB_POOL_SIZE	Tamanho do pool de conexão	30
API_PORT	Porta de escuta da API	8443
API_LISTEN_IP	IP de escuta da API	0.0.0.0
WEB_PORT	Porta da Interface Web	443
NODE_NAME	Nome do nó Erlang	sms@node1.example.com
ERLANG_COOKIE	Segredo do cluster	shared_cookie_value
OCS_URL	URL da API OCS	http://ocs.local:2080/ jsonrpc
OCS_TENANT	Inquilino OCS	sms.example.com

Melhores Práticas de Configuração

1. **Use Variáveis de Ambiente** para valores sensíveis (senhas, chaves de API)
2. **Teste Alterações de Configuração** em staging antes da produção
3. **Documente Configurações Personalizadas** em notas de implantação
4. **Controle de Versão dos Arquivos de Configuração** (excluindo segredos)
5. **Monitore Após Alterações** para regressões de desempenho
6. **Mantenha Backups** de configurações funcionais
7. **Valide Antes de Reiniciar** para evitar falhas na inicialização
8. **Use Nomenclatura Consistente** entre ambientes
9. **Defina Limites de Recursos** apropriados ao hardware
10. **Revise Periodicamente** para remover recursos não utilizados

Solução de Problemas de Configuração

Sintoma	Causa Provável	Solução
A aplicação não inicia	Erro de sintaxe na configuração	Verifique os logs, valide a sintaxe
Falha na conexão com o banco	Credenciais/host errados	Verifique as variáveis de ambiente DB_*
API não acessível	Porta/IP de vinculação errados	Verifique API_PORT e listen_ip
Nós do cluster não se conectam	Incompatibilidade de cookie, firewall	Verifique ERLANG_COOKIE, verifique as portas 4369, 9100-9200
Falhas de cobrança	OCS inacessível	Teste a conectividade com o ocs_url
Pesquisas ENUM falham	Servidor DNS inacessível	Teste a conectividade DNS, verifique o timeout
Desempenho ruim	Configurações de lote erradas	Revise o Guia de Ajuste de Desempenho
Mensagens não roteadas	Rotas não carregadas	Verifique a configuração sms_routes ou a Interface Web

Para ajuda adicional, veja o [Guia de Solução de Problemas](#).

Configuração de Armazenamento de Mensagens (Mnesia)

Retenção de Mensagens

As mensagens são armazenadas no Mnesia para acesso rápido com limpeza automática configurável.

```
config :sms_c,  
  # Quanto tempo manter mensagens no Mnesia (horas)
```

```
message_retention_hours: 24,  
  
# Com que frequência verificar mensagens antigas (minutos)  
retention_check_interval_minutes: 60
```

Recomendações:

- **Produção:** 24-72 horas (equilibrar necessidades operacionais vs memória)
- **Desenvolvimento:** 4-8 horas (limpeza mais rápida para testes)
- **Alto volume:** 12-24 horas (conservar memória)

Impacto na Memória:

- Mensagem média: ~1KB
- 10.000 mensagens: ~10MB
- 100.000 mensagens: ~100MB

Exportação de CDR (Registro de Detalhes de Chamadas)

Quando as mensagens são entregues ou expiram, os CDRs podem ser escritos automaticamente no seu banco de dados Ecto para armazenamento a longo prazo e análises de faturamento.

```
config :sms_c,  
  # Habilitar/desabilitar gravação de CDR  
  cdr_enabled: true
```

Registros CDR Incluem:

- ID da mensagem, números de chamada/chamada
- SMSC de origem/destino
- Nó de origem/destino (para clusters)
- Carimbos de data/hora de submissão, entrega, expiração
- Status, tentativas de entrega
- Corpo da mensagem opcional (veja controles de privacidade)

Quando Desabilitar:

- Ambientes de teste onde CDRs não são necessários
- Resolução temporária de problemas para reduzir carga no banco de dados

Controles de Privacidade

Configure a visibilidade e retenção do corpo da mensagem para conformidade com a privacidade.

```
config :sms_c,  
  # Excluir corpo da mensagem do Mnesia após entrega bem-sucedida
```

```
delete_message_body_after_delivery: false,

# Ocultar corpo da mensagem na interface web
hide_message_body_in_ui: false,

# Ocultar corpo da mensagem em exportações CSV
hide_message_body_in_export: false
```

Casos de Uso:

Configuração	Caso de Uso
delete_message_body_after_delivery: true	Economizar espaço no Mnesia, conformidade com a privacidade
hide_message_body_in_ui: true	Impedir que o operador veja o conteúdo da mensagem
hide_message_body_in_export: true	Conformidade com exportação de dados, relatórios sanitizados

Exemplos de Configurações:

Privacidade Máxima (Conformidade)

```
config :sms_c,
  delete_message_body_after_delivery: true,
  hide_message_body_in_ui: true,
  hide_message_body_in_export: true,
  cdr_enabled: true # Manter CDRs sem corpos
```

Desenvolvimento (Visibilidade Total)

```
config :sms_c,
  delete_message_body_after_delivery: false,
  hide_message_body_in_ui: false,
  hide_message_body_in_export: false,
  cdr_enabled: true
```

Registro de Inicialização

Na inicialização da aplicação, o status da configuração é registrado:

```
[info] Armazenamento de mensagens: Mnesia (retenção: 24h)
[info] Exportação de CDR: HABILITADA
[info] Exclusão após entrega: DESABILITADA
[info] Cobrança OCS: HABILITADA (url: http://..., inquilino: ...)
```

Isso fornece visibilidade imediata sobre os recursos ativos.



Documentação de Métricas Prometheus do SMS-C

[← Voltar ao Índice da Documentação](#) | [README Principal](#)

Visão Geral

Este documento descreve todas as métricas Prometheus expostas pelo sistema SMS-C. Essas métricas são projetadas para a equipe de operações monitorar a saúde do sistema, desempenho e solucionar problemas.

Acessando Métricas

O endpoint de métricas Prometheus está disponível em:

```
http://localhost:9568/metrics
```

Este endpoint expõe métricas no formato de texto Prometheus que podem ser coletadas por um servidor Prometheus. As métricas são atualizadas em tempo real à medida que o sistema processa mensagens.

Convenção de Nomenclatura de Métricas

Todas as métricas seguem o padrão: `sms_c.<categoria>.<nome_da_métrica>.<tipo>`

Categorias:

- `message` - Métricas de processamento de mensagens
- `routing` - Métricas de decisão de roteamento
- `enum` - Métricas de consulta ENUM/NAPTR
- `delivery` - Métricas de entrega de mensagens
- `queue` - Métricas de gerenciamento de filas
- `charging` - Métricas de cobrança/faturamento
- `mnesia` - Métricas de banco de dados
- `frontend` - Métricas de conexão do frontend
- `location` - Métricas de localização/registo
- `phoenix.endpoint` - Métricas de requisições da API HTTP
- `vm` - Métricas do sistema Erlang VM

Métricas de Processamento de Mensagens

`sms_c_message_received_count`

Tipo: Contador

Descrição: Número total de mensagens recebidas pelo SMS-C de todas as fontes.

Rótulos:

- `source_smsc`: Nome do SMSC de origem que enviou a mensagem
- `source_type`: Tipo de conexão de origem (ims, circuit_switched, smpp)
- `message_type`: Tipo de mensagem (sms, mms)

Caso de Uso: Monitorar o volume de mensagens recebidas por fonte e tipo. Usar para detectar padrões de tráfego, identificar períodos de pico e detectar anomalias no fluxo de mensagens.

Alerta: Definir alertas para quedas súbitas (potenciais problemas de conectividade da fonte) ou picos (potenciais ataques/spam).

sms_c_message_validated_count

Tipo: Contador

Descrição: Número total de validações de mensagens realizadas.

Rótulos:

- `valid`: Se a validação foi aprovada (true ou false)

Caso de Uso: Acompanhar taxas de sucesso/falha de validação. Altas taxas de falha podem indicar mensagens malformadas ou problemas de integração.

Alerta: Alertar quando a taxa de falha de validação exceder o limite (por exemplo, > 5% de falhas).

sms_c_message_processing_stop_duration

Tipo: Histograma

Descrição: Tempo levado para processar uma mensagem desde o recebimento até a conclusão (inclui validação, roteamento e enfileiramento).

Unidade: Milissegundos

Buckets: 10, 50, 100, 250, 500, 1000, 2500, 5000 ms

Rótulos:

- `success`: Se o processamento foi bem-sucedido (true ou false)

Caso de Uso: Monitorar o desempenho do processamento de mensagens de ponta a ponta. Identificar desacelerações no pipeline de processamento.

Alerta: Alertar quando a latência p95 ou p99 exceder os limites do SLA.

Métricas de Roteamento

sms_c_routing_route_matched_count

Tipo: Contador

Descrição: Número total de vezes que uma rota específica foi correspondida e selecionada para

roteamento de mensagens.

Rótulos:

- route_id: Identificador único da rota correspondida
- dest_smsc: SMSC de destino selecionada pela rota
- priority: Valor de prioridade da rota correspondida

Caso de Uso: Entender quais rotas estão sendo usadas com mais frequência. Identificar rotas subutilizadas ou sobrecarregadas. Útil para planejamento de capacidade e otimização de rotas.

Alerta: Alertar se rotas de alta prioridade raramente forem correspondidas (pode indicar configuração de roteamento incorreta).

sms_c_routing_failed_count

Tipo: Contador

Descrição: Número total de falhas de roteamento onde nenhuma rota adequada pôde ser encontrada.

Rótulos:

- reason: Motivo da falha (no_route_found, validation_failed, etc.)

Caso de Uso: Acompanhar falhas de roteamento para identificar lacunas de configuração ou padrões de tráfego inesperados.

Alerta: Alertar sobre quaisquer falhas de roteamento, pois indicam que as mensagens não podem ser entregues.

sms_c_routing_action_count

Tipo: Contador

Descrição: Número total de ações especiais de roteamento realizadas.

Rótulos:

- action: Tipo de ação (drop, auto_reply, forward)
- route_id: Rota que acionou a ação

Caso de Uso: Monitorar regras de drop (anti-spam), uso de auto-resposta e padrões de encaminhamento.

Alerta: Alertar sobre picos inesperados em ações de drop (pode indicar ataque de spam).

sms_c_routing_stop_duration

Tipo: Histograma

Descrição: Tempo levado para avaliar todas as rotas e selecionar a melhor correspondência.

Unidade: Milissegundos

Buckets: 1, 5, 10, 25, 50, 100, 250, 500 ms

Rótulos:

- dest_smsc: SMSC de destino selecionada

Caso de Uso: Monitorar o desempenho do mecanismo de roteamento. Roteamento lento indica muitas rotas ou lógica de correspondência complexa.

Alerta: Alertar quando o roteamento levar consistentemente mais tempo do que o esperado (por exemplo, p95 > 50ms).

Métricas de Consulta ENUM/NAPTR

sms_c_enum_cache_hit_count

Tipo: Contador

Descrição: Número total de consultas ENUM atendidas a partir do cache (não exigiu consulta DNS).

Rótulos:

- domain: Domínio ENUM consultado

Caso de Uso: Monitorar a eficácia do cache. Altas taxas de acerto de cache reduzem a carga do DNS e melhoram o desempenho.

Alerta: Alertar se a taxa de acerto do cache cair abaixo do limite (pode indicar problemas no cache ou tráfego incomum).

sms_c_enum_cache_miss_count

Tipo: Contador

Descrição: Número total de consultas ENUM que exigiram uma consulta DNS (não estão no cache).

Rótulos:

- domain: Domínio ENUM consultado

Caso de Uso: Acompanhar falhas de cache para entender a eficácia do cache. Usar com a contagem de acertos para calcular a taxa de acerto.

Cálculo: $\text{cache_hit_rate} = \text{hits} / (\text{hits} + \text{misses})$

sms_c_enum_cache_size_size

Tipo: Gauge

Descrição: Número atual de entradas no cache ENUM.

Caso de Uso: Monitorar o tamanho do cache para garantir que não esteja crescendo sem limites. Ajudar a ajustar as configurações de TTL do cache.

Alerta: Alertar se o tamanho do cache exceder os limites esperados (pode indicar vazamento de memória).

sms_c_enum_lookup_stop_duration

Tipo: Histograma

Descrição: Tempo levado para completar uma consulta ENUM (incluindo consulta DNS se não estiver em cache).

Unidade: Milissegundos

Buckets: 10, 50, 100, 250, 500, 1000, 2500, 5000 ms

Rótulos:

- domain: Domínio ENUM consultado
- success: Se a consulta foi bem-sucedida (true ou false)
- cache_hit: Se o resultado foi servido a partir do cache (true ou false)

Caso de Uso: Monitorar o desempenho da consulta ENUM. Identificar servidores DNS lentos ou problemas de rede.

Alerta: Alertar quando o tempo de consulta p95 exceder o limite de tempo limite.

sms_c_enum_naptr_records_record_count

Tipo: Histograma

Descrição: Número de registros NAPTR retornados por uma consulta ENUM bem-sucedida.

Buckets: 0, 1, 2, 3, 5, 10

Rótulos:

- domain: Domínio ENUM consultado

Caso de Uso: Entender a distribuição de registros ENUM. A maioria das consultas deve retornar de 1 a 3 registros.

Alerta: Alertar se frequentemente retornar 0 registros (problema de configuração DNS).

Métricas de Entrega

sms_c_delivery_queued_count

Tipo: Contador

Descrição: Número total de mensagens enfileiradas para entrega a um SMSC de destino.

Rótulos:

- `dest_smsc`: Nome do SMSC de destino

Caso de Uso: Monitorar o fluxo de mensagens para cada destino. Útil para planejamento de capacidade.

Alerta: Comparar com contagens de sucesso/falha de entrega para detectar acúmulo.

`sms_c_delivery_attempted_count`

Tipo: Contador

Descrição: Número total de tentativas de entrega realizadas (inclui tentativas de reenvio).

Rótulos:

- `dest_smsc`: Nome do SMSC de destino

Caso de Uso: Acompanhar o volume de tentativas de entrega. Alta contagem de tentativas em relação à contagem enfileirada indica comportamento de reenvio.

`sms_c_delivery_succeeded_count`

Tipo: Contador

Descrição: Número total de mensagens entregues com sucesso ao SMSC de destino.

Rótulos:

- `dest_smsc`: Nome do SMSC de destino

Caso de Uso: Acompanhar entregas bem-sucedidas por destino. Métrica de sucesso primária.

Alerta: Alertar se a taxa de sucesso cair abaixo do limite do SLA.

Cálculo: $\text{success_rate} = \text{succeeded} / \text{queued}$

`sms_c_delivery_failed_count`

Tipo: Contador

Descrição: Número total de mensagens que falharam na entrega após todas as tentativas de reenvio.

Rótulos:

- `dest_smsc`: Nome do SMSC de destino
- `reason`: Motivo da falha

Caso de Uso: Acompanhar falhas de entrega para identificar destinos problemáticos ou padrões de falha.

Alerta: Alertar sobre taxas de falha elevadas ou motivos de falha específicos.

sms_c_delivery_dead_letter_count

Tipo: Contador

Descrição: Número total de mensagens movidas para a fila de cartas mortas (indisponíveis para entrega).

Rótulos:

- reason: Motivo para a carta morta (por exemplo, max_retries_exceeded, expired)

Caso de Uso: Monitorar mensagens indesejadas que requerem intervenção manual.

Alerta: Alertar sobre quaisquer eventos de carta morta, pois representam falha completa de entrega.

sms_c_delivery_succeeded_duration

Tipo: Histograma

Descrição: Tempo de ponta a ponta desde que a mensagem foi enfileirada até a entrega bem-sucedida.

Unidade: Milissegundos

Buckets: 100, 500, 1000, 5000, 10000, 30000, 60000 ms

Rótulos:

- dest_smsc: Nome do SMSC de destino

Caso de Uso: Monitorar a latência de entrega. Identificar destinos lentos ou problemas de rede.

Alerta: Alertar quando o tempo de entrega p95 exceder os limites do SLA.

sms_c_delivery_succeeded_attempt_count

Tipo: Histograma

Descrição: Número de tentativas de entrega necessárias antes da entrega bem-sucedida.

Buckets: 1, 2, 3, 5, 10

Rótulos:

- dest_smsc: Nome do SMSC de destino

Caso de Uso: Entender o comportamento de reenvio. A maioria das entregas deve ser bem-sucedida na primeira tentativa.

Alerta: Alertar se a contagem média de tentativas exceder 2 (indica problemas de confiabilidade do destino).

sms_c_delivery_failed_attempt_count

Tipo: Histograma

Descrição: Número de tentativas de entrega feitas antes da falha final.

Buckets: 1, 2, 3, 5, 10

Rótulos:

- dest_smsc: Nome do SMSC de destino

Caso de Uso: Entender quantas tentativas de reenvio ocorrem antes de desistir.

Métricas de Fila

sms_c_queue_size_size

Tipo: Gauge

Descrição: Número total atual de mensagens na fila (todos os estados combinados).

Rótulos:

- queue_type: Tipo de fila (message_queue, dead_letter)

Caso de Uso: Monitorar a profundidade da fila para detectar acúmulos ou problemas de processamento.

Alerta: Alertar quando o tamanho da fila exceder os limites de capacidade.

sms_c_queue_size_pending

Tipo: Gauge

Descrição: Número atual de mensagens pendentes de entrega (ainda não tentadas).

Rótulos:

- queue_type: Tipo de fila

Caso de Uso: Monitorar a contagem de mensagens pendentes. Altas contagens pendentes indicam atrasos no processamento.

Alerta: Alertar quando a contagem pendente exceder o limite por um período prolongado.

sms_c_queue_size_failed

Tipo: Gauge

Descrição: Número atual de mensagens em estado de falha (aguardando reenvio).

Rótulos:

- queue_type: Tipo de fila

Caso de Uso: Monitorar a acumulação de mensagens falhadas. Indica problemas de entrega.

Alerta: Alertar sobre contagem elevada de falhas, pois isso impacta as taxas de entrega.

sms_c_queue_size_delivered

Tipo: Gauge

Descrição: Número atual de mensagens entregues aguardando limpeza/remoção da fila.

Rótulos:

- queue_type: Tipo de fila

Caso de Uso: Monitorar o atraso na limpeza. Altas contagens indicam que o processo de limpeza está atrasado.

Alerta: Alertar se mensagens entregues se acumularem significativamente.

sms_c_queue_oldest_message_age_seconds

Tipo: Gauge

Descrição: Idade (em segundos) da mensagem mais antiga atualmente em estado pendente.

Rótulos:

- queue_type: Tipo de fila

Caso de Uso: Detectar envelhecimento de mensagens e paralisações no processamento. Crítico para monitoramento de SLA.

Alerta: Alertar quando a idade da mensagem mais antiga exceder o limite do SLA (por exemplo, > 300 segundos).

Métricas de Cobrança

sms_c_charging_requested_count

Tipo: Contador

Descrição: Número total de solicitações de cobrança/faturamento feitas ao OCS ou sistema de faturamento.

Rótulos:

- account: Identificador da conta sendo cobrada

Caso de Uso: Acompanhar o volume de cobrança por conta. Útil para reconciliação de faturamento.

sms_c_charging_succeeded_count

Tipo: Contador

Descrição: Número total de operações de cobrança bem-sucedidas.

Rótulos:

- account: Identificador da conta cobrada

Caso de Uso: Monitorar a taxa de sucesso da cobrança por conta.

Cálculo: $\text{success_rate} = \text{succeeded} / \text{requested}$

sms_c_charging_failed_count

Tipo: Contador

Descrição: Número total de operações de cobrança falhadas.

Rótulos:

- account: Identificador da conta
- reason: Motivo da falha

Caso de Uso: Identificar falhas de cobrança que podem impactar a receita ou exigir intervenção na conta.

Alerta: Alertar sobre taxas elevadas de falha de cobrança.

sms_c_charging_succeeded_duration

Tipo: Histograma

Descrição: Tempo levado para completar uma solicitação de cobrança bem-sucedida.

Unidade: Milissegundos

Buckets: 10, 50, 100, 250, 500, 1000, 2500, 5000 ms

Rótulos:

- account: Identificador da conta

Caso de Uso: Monitorar o desempenho do sistema de faturamento. Cobranças lentas podem atrasar a entrega de mensagens.

Alerta: Alertar quando o tempo de cobrança p95 exceder o limite.

Métricas de Saúde do Sistema

sms_c_mnesia_table_size_record_count

Tipo: Gauge

Descrição: Número atual de registros em cada tabela do banco de dados Mnesia.

Rótulos:

- table: Nome da tabela (por exemplo, sms_route)

Caso de Uso: Monitorar o crescimento do banco de dados. Detectar acúmulo inesperado de dados.

Alerta: Alertar sobre taxas de crescimento inesperadas da tabela.

sms_c_frontend_status_count

Tipo: Gauge

Descrição: Número de frontends em cada status de conexão.

Rótulos:

- frontend_name: Identificador do frontend
- status: Status da conexão (connected, disconnected)

Caso de Uso: Monitorar a conectividade do frontend. Detectar falhas de conexão.

Alerta: Alertar quando frontends esperados se desconectarem.

sms_c_location_registered_count

Tipo: Contador

Descrição: Número total de registros de localização/assinante recebidos pelo sistema.

Rótulos:

- location: Nome do frontend/SMSC onde o assinante está registrado
- ims_capable: Se o assinante suporta IMS (true/false)

Caso de Uso: Monitorar a atividade de registro de assinantes. Acompanhar assinantes IMS vs não-IMS. Detectar tempestades ou falhas de registro.

Alerta: Definir alertas para:

- Quedas na taxa de registro (pode indicar problemas de rede)
- Picos incomuns em registros
- Alta proporção de registros não-IMS (influxo de dispositivos legados)

Exemplo de Consulta:


```
# Taxa de registro por minuto
rate(sms_c_location_registered_count[1m])

# Proporção de registro IMS vs não-IMS
sum(rate(sms_c_location_registered_count{ims_capable="true"}[5m])) /
sum(rate(sms_c_location_registered_count[5m]))
```

Métricas de Requisições da API HTTP

phoenix_endpoint_stop_duration

Tipo: Distribuição (Histograma)

Descrição: Duração do processamento de requisições HTTP em milissegundos, desde o início da requisição até a conclusão da resposta.

Rótulos:

- route: Rota do endpoint da API (por exemplo, /api/messages, /api/frontends)

Buckets: 10ms, 50ms, 100ms, 250ms, 500ms, 1s, 2.5s, 5s

Caso de Uso: Monitorar o desempenho da API. Identificar endpoints lentos. Acompanhar SLAs de tempo de resposta.

Alerta: Definir alertas para:

- Latência P95 > 500ms para endpoints críticos
- Latência P99 > 1s para qualquer endpoint
- Tendências de latência crescente

Exemplo de Consulta:

```
# Tempo de resposta P95 por endpoint
histogram_quantile(0.95,
  rate(phoenix_endpoint_stop_duration_bucket[5m]))

# Requisições mais lentas que 1 segundo
sum(rate(phoenix_endpoint_stop_duration_bucket{le="1000"}[5m]))
```

phoenix_endpoint_stop_count

Tipo: Contador

Descrição: Número total de requisições HTTP concluídas, categorizadas por rota e código de status HTTP.

Rótulos:

- route: Rota do endpoint da API
- status: Código de status HTTP (200, 201, 400, 404, 500, etc.)

Caso de Uso: Monitorar o volume de requisições da API e taxas de sucesso. Acompanhar taxas de erro por endpoint.

Alerta: Definir alertas para:

- Taxa de erro > 5% para qualquer endpoint
- Erros 5xx em endpoints críticos
- Quedas súbitas no volume de requisições

Exemplo de Consulta:

```
# Taxa de requisições por endpoint
sum by (route) (rate(phoenix_endpoint_stop_count[5m]))

# Taxa de erro por endpoint
sum by (route) (rate(phoenix_endpoint_stop_count{status=~"5.."}[5m])) /
sum by (route) (rate(phoenix_endpoint_stop_count[5m]))

# Taxa de sucesso
sum(rate(phoenix_endpoint_stop_count{status=~"2.."}[5m])) /
sum(rate(phoenix_endpoint_stop_count[5m]))
```

phoenix_router_dispatch_exception_count

Tipo: Contador

Descrição: Número total de exceções/erros levantados durante o processamento de requisições HTTP.

Rótulos:

- route: Rota do endpoint da API onde ocorreu a exceção
- kind: Tipo de exceção (error, exit, throw)

Caso de Uso: Acompanhar erros de aplicação. Identificar endpoints problemáticos. Monitorar a estabilidade do sistema.

Alerta: Definir alertas para qualquer valor diferente de zero em endpoints críticos.

Exemplo de Consulta:

```
# Taxa de exceções por endpoint
rate(phoenix_router_dispatch_exception_count[5m])

# Total de exceções na última hora
increase(phoenix_router_dispatch_exception_count[1h])
```

Métricas da VM Erlang

vm_memory_total

Tipo: Gauge

Descrição: Memória total alocada pela VM Erlang em bytes.

Caso de Uso: Monitorar o uso geral de memória. Detectar vazamentos de memória. Planejar capacidade.

Alerta: Alertar quando o uso de memória > 80% da memória disponível do sistema.

vm_memory_processes

Tipo: Gauge

Descrição: Memória usada pelos processos Erlang em bytes.

Caso de Uso: Acompanhar o consumo de memória dos processos. Fonte mais comum de crescimento de memória.

Alerta: Alertar sobre taxa de crescimento elevada sustentada.

vm_total_run_queue_lengths_total

Tipo: Gauge

Descrição: Número total de processos aguardando para serem agendados em todos os escalonadores de CPU.

Caso de Uso: Medir a carga do sistema. Valores altos indicam saturação da CPU.

Alerta: Alertar quando consistentemente > 10 * número de núcleos de CPU.

vm_system_counts_process_count

Tipo: Gauge

Descrição: Número atual de processos em execução na VM.

Caso de Uso: Monitorar padrões de criação de processos. Detectar vazamentos de processos.

Alerta: Alertar quando se aproximar do limite de processos (padrão 262.144).

Coleta e Polling de Métricas

O sistema coleta automaticamente as seguintes métricas a cada 10 segundos:

- Tamanhos e idades das filas
- Tamanhos das tabelas Mnesia
- Estatísticas do cache ENUM

Todas as outras métricas são acionadas por eventos e emitidas quando a ação correspondente ocorre.

Padrões Comuns de Monitoramento

Taxa de Sucesso de Entrega por Destino

Acompanhar a taxa de sucesso da entrega de mensagens para cada SMSC de destino:

Fórmula: $(\text{sms_c_delivery_succeeded_count}) / (\text{sms_c_delivery_queued_count})$

Interpretação: Deve ser > 95% para destinos saudáveis. Taxas mais baixas indicam problemas de entrega.

Latência de Mensagem de Ponta a Ponta

Monitorar o tempo total desde o recebimento da mensagem até a entrega:

Métricas:

- `sms_c_message_processing_stop_duration` (processamento)
- `sms_c_delivery_succeeded_duration` (entrega)

Interpretação: A soma representa a latência total enfrentada pelo usuário.

Eficácia do Cache ENUM

Medir o quão bem o cache ENUM está funcionando:

Fórmula: $(\text{sms_c_enum_cache_hit_count}) / (\text{sms_c_enum_cache_hit_count} + \text{sms_c_enum_cache_miss_count})$

Interpretação: Deve ser > 80% após o aquecimento. Taxas mais baixas podem indicar TTL curto ou alta variação de tráfego.

Utilização de Rotas

Identificar quais rotas lidam com o maior tráfego:

Métrica: `sms_c_routing_route_matched_count` agrupada por `route_id`

Interpretação: Usar para identificar rotas quentes para otimização e planejamento de capacidade.

Tendência de Acúmulo de Filas

Monitorar se a fila de mensagens está crescendo (acúmulo) ou diminuindo (recuperando):

Métricas:

- `sms_c_queue_size_pending` (pendente atual)
- `sms_c_queue_oldest_message_age_seconds` (tendência de idade)

Interpretação: Contagem pendente crescente + idade crescente = acúmulo se formando.

Taxa de Reenvio

Entender com que frequência os reenvios de entrega são necessários:

Métrica: `sms_c_delivery_succeeded_attempt_count` percentis do histograma

Interpretação: Se $p95 > 1$, a maioria das mensagens requer reenvios. Indica problemas de confiabilidade do destino.

Alertas Recomendados

Alerta	Condição	Severidade	Descrição
Alta Taxa de Falha de Roteamento	<code>routing_failed_count</code> aumento	Crítico	Mensagens não podem ser roteadas
Acúmulo de Fila	<code>queue_size_pending</code> > limite	Aviso	Mensagens acumulando
Mensagens Antigas na Fila	<code>queue_oldest_message_age_seconds</code> > 300	Crítico	Violação do SLA
Pico de Falhas de Entrega	<code>delivery_failed_count</code> pico	Alto	Problemas de destino
Eventos de Carta Morta	<code>delivery_dead_letter_count</code> > 0	Alto	Mensagens indesejáveis
Timeouts de Consulta ENUM	<code>enum_lookup_stop_duration</code> p95 > 5000ms	Aviso	Problemas de DNS
Baixa Taxa de Acerto de Cache	Taxa de acerto do cache ENUM < 0.7	Aviso	Cache ineficaz
Frontend Desconectado	<code>frontend_status_count{status="disconnected"}</code> > 0	Alto	Perda de conectividade
Falhas de Cobrança	<code>charging_failed_count</code> > limite	Alto	Problemas de faturamento
Processamento Lento de Mensagens	<code>message_processing_stop_duration</code> p95 > 1000ms	Aviso	Degradação de desempenho

Recomendações de Dashboard

Dashboard de Operações

Propósito: Monitoramento em tempo real da saúde do sistema

Painéis:

- Throughput de mensagens (recebidas/processadas/entregues por minuto)
 - Tamanhos de fila (pendentes, falhadas, entregues)
 - Taxa de sucesso de entrega por destino
 - Latência de processamento e entrega p95
 - Status dos frontends ativos
 - Alertas atuais
-

Dashboard de Desempenho

Propósito: Análise de desempenho do sistema

Painéis:

1. Histograma da duração do processamento de mensagens
 2. Histograma da duração do roteamento
 3. Histograma da duração da consulta ENUM
 4. Histograma da duração da cobrança
 5. Distribuição das tentativas de entrega
 6. Taxas de acerto do cache
-

Dashboard de Negócios

Propósito: Análise de tráfego e uso

Painéis:

1. Mensagens por SMSC de origem
 2. Mensagens por SMSC de destino
 3. Mapa de calor de utilização de rotas
 4. Contagens de ações de auto-resposta e drop
 5. Estatísticas de uso de ENUM
 6. Volume de cobrança por conta
-

Retenção de Métricas

Configurações de retenção recomendadas para Prometheus:

- **Métricas brutas:** 15 dias
- **Agregados de 5 minutos:** 90 dias
- **Agregados de 1 hora:** 2 anos

Isso fornece um histórico recente detalhado enquanto mantém tendências de longo prazo para planejamento de capacidade.

Solução de Problemas com Métricas

Cenário: Mensagens Não Estão Sendo Entregues

Passos de Investigação:

1. Verifique `sms_c_message_received_count` - As mensagens estão sendo recebidas?
 2. Verifique `sms_c_routing_failed_count` - Elas estão sendo roteadas?
 3. Verifique `sms_c_delivery_queued_count` - Elas estão sendo enfileiradas?
 4. Verifique `sms_c_delivery_failed_count` - As tentativas de entrega estão falhando?
 5. Verifique os rótulos `dest_smsc` para identificar o destino problemático.
-

Cenário: Processamento Lento de Mensagens

Passos de Investigação:

1. Verifique o histograma `sms_c_message_processing_stop_duration` - Tempo total de processamento
2. Verifique `sms_c_routing_stop_duration` - O roteamento está lento?
3. Verifique `sms_c_enum_lookup_stop_duration` - As consultas ENUM estão lentas?

4. Verifique sms_c_charging_succeeded_duration - A cobrança está lenta?
5. Identifique o gargalo e investigue o componente específico.

Cenário: Crescimento da Fila de Mensagens

Passos de Investigação:

1. Verifique a tendência de sms_c_queue_size_pending - Está crescendo?
2. Verifique sms_c_delivery_attempted_count - Estão ocorrendo tentativas de entrega?
3. Verifique sms_c_delivery_failed_count - Estão falhando?
4. Verifique sms_c_delivery_succeeded_duration - A entrega está demorando demais?
5. Verifique os rótulos dest_smsc para identificar destinos lentos.

Exemplos de Consultas Prometheus

Throughput de Mensagens

Mensagens Recebidas Por Segundo (média de 5 minutos):

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

Mensagens Recebidas Por Minuto (média de 1 hora):

```
rate(sms_c_message_received_count[1h]) * 60
```

Total de Mensagens Hoje:

```
increase(sms_c_message_received_count[24h])
```

Mensagens por Tipo de Origem:

```
sum by (source_type) (rate(sms_c_message_received_count[5m]))
```

Mensagens por SMSC de Origem:

```
sum by (source_smsc) (rate(sms_c_message_received_count[5m]))
```

Desempenho de Entrega

Taxa de Sucesso de Entrega (Porcentagem):

```
(rate(sms_c_delivery_succeeded_count[5m]) / rate(sms_c_delivery_queued_count[5m])) * 100
```

Taxa de Falha de Entrega (Porcentagem):

```
(rate(sms_c_delivery_failed_count[5m]) / rate(sms_c_delivery_queued_count[5m])) * 100
```

Tentativas Médias de Entrega (p95):

```
histogram_quantile(0.95, sms_c_delivery_succeeded_attempt_count_bucket)
```

Sucesso de Entrega por Destino:

```
sum by (dest_smsc) (rate(sms_c_delivery_succeeded_count[5m]))
```

Razões de Falha de Entrega:

```
sum by (reason) (rate(sms_c_delivery_failed_count[5m]))
```

Tempo para Entrega (p95):

```
histogram_quantile(0.95, sms_c_delivery_succeeded_duration_bucket)
```

Tempo para Entrega (p99):

```
histogram_quantile(0.99, sms_c_delivery_succeeded_duration_bucket)
```

Métricas de Fila

Mensagens Pendentes Atuais:

```
sms_c_queue_size_pending
```

Mensagens Falhadas Aguardando Reenvio:

```
sms_c_queue_size_failed
```

Idade da Mensagem Mais Antiga (Minutos):

```
sms_c_queue_oldest_message_age_seconds / 60
```

Taxa de Crescimento da Fila (Mensagens/Hora):

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

Mensagens Entrando na Fila:

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

Mensagens Saindo da Fila:

```
rate(sms_c_delivery_succeeded_count[5m]) + rate(sms_c_delivery_failed_count[5m])
```

Acúmulo de Fila (Entrando - Saindo):

```
rate(sms_c_delivery_queued_count[5m]) - (rate(sms_c_delivery_succeeded_count[5m]) + rate(sms_c_delivery_failed_count[5m]))
```

Desempenho de Roteamento

Taxa de Sucesso de Roteamento:

```
(1 - (rate(sms_c_routing_failed_count[5m]) / (rate(sms_c_routing_route_matched_count[5m]) + rate(sms_c_routing_failed_count[5m])))) * 100
```

Rotas Mais Usadas:

```
topk(10, sum by (route_id, dest_smsc) (rate(sms_c_routing_route_matched_count[1h])))
```


Latência de Roteamento (p50, p95, p99):

```
histogram_quantile(0.50, sms_c_routing_stop_duration_bucket)
histogram_quantile(0.95, sms_c_routing_stop_duration_bucket)
histogram_quantile(0.99, sms_c_routing_stop_duration_bucket)
```

Falhas de Roteamento Por Minuto:

```
rate(sms_c_routing_failed_count[5m]) * 60
```

Ações de Drop Por Hora:

```
increase(sms_c_routing_action_count{action="drop"}[1h])
```

Ações de Auto-Resposta Por Hora:

```
increase(sms_c_routing_action_count{action="auto_reply"}[1h])
```

Desempenho ENUM

Taxa de Acerto do Cache ENUM:

```
rate(sms_c_enum_cache_hit_count[5m]) / (rate(sms_c_enum_cache_hit_count[5m]) +
rate(sms_c_enum_cache_miss_count[5m]))
```

Porcentagem de Acerto do Cache ENUM:

```
(rate(sms_c_enum_cache_hit_count[5m]) / (rate(sms_c_enum_cache_hit_count[5m]) +
rate(sms_c_enum_cache_miss_count[5m]))) * 100
```

Latência de Consulta ENUM (p95):

```
histogram_quantile(0.95, sms_c_enum_lookup_stop_duration_bucket)
```

Consultas ENUM Por Segundo (Cacheadas vs Não Cacheadas):

```
# Cacheadas (rápido)
rate(sms_c_enum_cache_hit_count[5m])

# Não Cacheadas (exige consulta DNS)
rate(sms_c_enum_cache_miss_count[5m])
```

Média de Registros NAPTR Retornados:

```
rate(sms_c_enum_naptr_records_record_count_sum[5m]) /
rate(sms_c_enum_naptr_records_record_count_count[5m])
```

Tamanho do Cache ENUM:

```
sms_c_enum_cache_size_size
```

Desempenho de Processamento

Latência de Processamento de Mensagens (p95):

```
histogram_quantile(0.95, sms_c_message_processing_stop_duration_bucket)
```

Latência de Processamento de Mensagens (p99):

```
histogram_quantile(0.99, sms_c_message_processing_stop_duration_bucket)
```

Falhas de Processamento:

```
rate(sms_c_message_processing_stop_duration_count{success="false"}[5m])
```

Taxa de Falha de Validação:

```
rate(sms_c_message_validated_count{valid="false"}[5m]) /  
rate(sms_c_message_validated_count[5m])
```

Métricas de Cobrança

Taxa de Sucesso de Cobrança:

```
rate(sms_c_charging_succeeded_count[5m]) /  
rate(sms_c_charging_requested_count[5m])
```

Falhas de Cobrança Por Minuto:

```
rate(sms_c_charging_failed_count[5m]) * 60
```

Latência de Cobrança (p95):

```
histogram_quantile(0.95, sms_c_charging_succeeded_duration_bucket)
```

Volume de Cobrança por Conta:

```
sum by (account) (rate(sms_c_charging_requested_count[1h]))
```

Saúde do Frontend

Frontends Ativos:

```
sum(sms_c_frontend_status_count{status="connected"})
```

Frontends Desconectados:

```
sum(sms_c_frontend_status_count{status="disconnected"})
```

Frontends por Nome:

```
sum by (frontend_name) (sms_c_frontend_status_count{status="connected"})
```

Saúde do Sistema

Tamanhos das Tabelas Mnesia:

```
sms_c_mnesia_table_size_record_count
```

Contagem de Rotas:

```
sms_c_mnesia_table_size_record_count{table="sms_route"}
```

Contagem de Regras de Tradução:

```
sms_c_mnesia_table_size_record_count{table="translation_rule"}
```

Exemplos de Dashboard do Grafana

Dashboard 1: Operações em Tempo Real

Propósito: Monitorar a atividade e saúde atuais do sistema.

Painéis:

1. Throughput de Mensagens (Gráfico)

- Consulta: `rate(sms_c_message_received_count[5m])`
- Consulta: `rate(sms_c_delivery_succeeded_count[5m])`
- Unidade: mensagens/segundo
- Legenda: `{{source_type}}`

2. Taxa de Sucesso de Entrega (Gauge)

- Consulta: $(\text{rate}(\text{sms_c_delivery_succeeded_count}[5m]) / \text{rate}(\text{sms_c_delivery_queued_count}[5m])) * 100$
- Unidade: porcentagem (0-100)
- Limites:
 - Vermelho: < 90
 - Amarelo: 90-95
 - Verde: > 95

3. Profundidade da Fila (Gráfico)

- Consulta: `sms_c_queue_size_pending`
- Consulta: `sms_c_queue_size_failed`
- Unidade: mensagens
- Legenda: `{{queue_type}}`

4. Idade da Mensagem Mais Antiga (Estatística)

- Consulta: `sms_c_queue_oldest_message_age_seconds / 60`
- Unidade: minutos
- Limites:
 - Verde: < 5
 - Amarelo: 5-10
 - Vermelho: > 10

5. Frontends Ativos (Estatística)

- Consulta: `sum(sms_c_frontend_status_count{status="connected"})`
- Unidade: contagem
- Cor: Azul

6. Falhas de Roteamento (Gráfico)

- Consulta: `rate(sms_c_routing_failed_count[5m]) * 60`
- Unidade: falhas/minuto
- Limite de alerta: > 0

Dashboard 2: Análise de Desempenho

Propósito: Analisar o desempenho do sistema e identificar gargalos.

Painéis:

1. Latência de Ponta a Ponta (Gráfico)

- Consulta: `histogram_quantile(0.50, sms_c_message_processing_stop_duration_bucket)` (p50)
- Consulta: `histogram_quantile(0.95, sms_c_message_processing_stop_duration_bucket)` (p95)
- Consulta: `histogram_quantile(0.99, sms_c_message_processing_stop_duration_bucket)` (p99)
- Unidade: milissegundos
- Legenda: Percentil

2. Latências de Componentes (Barra Gauge)

- Roteamento: `histogram_quantile(0.95, sms_c_routing_stop_duration_bucket)`
- ENUM: `histogram_quantile(0.95, sms_c_enum_lookup_stop_duration_bucket)`
- Cobrança: `histogram_quantile(0.95, sms_c_charging_succeeded_duration_bucket)`
- Entrega: `histogram_quantile(0.95, sms_c_delivery_succeeded_duration_bucket)`
- Unidade: milissegundos
- Barras horizontais

3. Distribuição de Tentativas de Entrega (Mapa de Calor)

- Consulta: `sms_c_delivery_succeeded_attempt_count_bucket`
- Mostra quantas tentativas são normalmente necessárias
- Escala de cores: Azul (1 tentativa) a Vermelho (muitas tentativas)

4. Desempenho do Cache ENUM (Gráfico)

- Taxa de Acerto: $\text{rate}(\text{sms_c_enum_cache_hit_count}[5m]) / (\text{rate}(\text{sms_c_enum_cache_hit_count}[5m]) + \text{rate}(\text{sms_c_enum_cache_miss_count}[5m]))$
- Tamanho do Cache: `sms_c_enum_cache_size_size`
- Eixo Y duplo (taxa vs tamanho)

5. Taxa de Sucesso de Processamento (Gauge)

- Consulta: $(\text{rate}(\text{sms_c_message_processing_stop_duration_count}\{\text{success}=\text{"true"}\}[5m]) / \text{rate}(\text{sms_c_message_processing_stop_duration_count}[5m])) * 100$
- Unidade: porcentagem
- Limites:
 - Vermelho: < 95
 - Amarelo: 95-99
 - Verde: > 99

Dashboard 3: Análise de Tráfego

Propósito: Analisar padrões de tráfego de mensagens e distribuição de roteamento.

Painéis:

1. Mensagens por Tipo de Origem (Gráfico de Pizza)

- Consulta: `sum by (source_type) (increase(sms_c_message_received_count[1h]))`
- Mostra a distribuição: IMS vs CS vs SMPP

2. Mensagens por SMSC de Origem (Gráfico de Barras)

- Consulta: `sum by (source_smsc) (rate(sms_c_message_received_count[1h]))`
- Top 10 fontes
- Barras horizontais

3. Utilização de Rotas (Tabela)

- Colunas:
 - Route ID
 - SMSC de Destino
 - Mensagens (1h): `sum by (route_id, dest_smsc) (increase(sms_c_routing_route_matched_count[1h]))`
 - Prioridade
 - Taxa de Sucesso
- Ordenado pela contagem de mensagens

4. Entrega por Destino (Gráfico)

- Consulta: `sum by (dest_smsc) (rate(sms_c_delivery_succeeded_count[5m]))`
- Unidade: mensagens/segundo
- Gráfico de área empilhada
- Legenda: `{{dest_smsc}}`

5. Ações de Drop/Auto-Resposta (Estatística)

- Drop: `increase(sms_c_routing_action_count{action="drop"}[1h])`
- Auto-Resposta:
`increase(sms_c_routing_action_count{action="auto_reply"}[1h])`
- Estatísticas lado a lado

6. Padrão de Tráfego Horário (Gráfico)

- Consulta: `rate(sms_c_message_received_count[1h]) * 3600`
- Intervalo de tempo: Últimos 7 dias
- Mostra padrões diários

Dashboard 4: Capacidade e Recursos

Propósito: Monitorar o uso de recursos e limites de capacidade.

Painéis:

1. Capacidade da Fila (Gráfico)

- Atual: `sms_c_queue_size_size`
- Linha de capacidade: Valor fixo com base nos limites do sistema
- Mostra a tendência de utilização

2. Crescimento das Tabelas do Banco de Dados (Gráfico)

- Mensagens: sms_c_mnesia_table_size_record_count{table="sms_route"}
- Traduções: sms_c_mnesia_table_size_record_count{table="translation_rule"}
- Tendência nos últimos 30 dias

3. Tendência de Acúmulo de Mensagens (Gráfico)

- Consulta: $\text{rate}(\text{sms_c_delivery_queued_count}[5m]) - (\text{rate}(\text{sms_c_delivery_succeeded_count}[5m]) + \text{rate}(\text{sms_c_delivery_failed_count}[5m]))$
- Positivo = acúmulo crescente
- Negativo = recuperando

4. Pico de Tráfego (Estatística)

- Consulta: max_over_time(rate(sms_c_message_received_count[5m])[24h:])
- Mostra a maior taxa de 5m na última 24h
- Unidade: mensagens/segundo

5. Utilização de Capacidade (Gauge)

- Consulta: $(\text{rate}(\text{sms_c_message_received_count}[5m]) / \text{MAX_CAPACITY}) * 100$
- Substitua MAX_CAPACITY pelo seu limite de sistema
- Unidade: porcentagem
- Limites:
 - Verde: < 70
 - Amarelo: 70-85
 - Vermelho: > 85

Dashboard 5: Conformidade com SLA

Propósito: Acompanhar métricas de SLA e conformidade.

Painéis:

1. Conformidade com SLA (Gauge)

- Sucesso de Entrega: $(\text{rate}(\text{sms_c_delivery_succeeded_count}[1h]) / \text{rate}(\text{sms_c_delivery_queued_count}[1h])) * 100$
- Linha alvo em 99%
- Limites:
 - Vermelho: < 95
 - Amarelo: 95-99
 - Verde: >= 99

2. Mensagens Entregues Dentro do SLA (Estatística)

- Consulta: $\text{count}(\text{sms_c_delivery_succeeded_duration_bucket}\{\text{le}="5000"\}) / \text{count}(\text{sms_c_delivery_succeeded_duration_bucket})$
- Mostra a porcentagem entregue dentro de 5 segundos
- Unidade: porcentagem

3. Violações de SLA (Contador)

- Mensagens excedendo 5 minutos:

- `increase(sms_c_queue_oldest_message_age_seconds{} > 300)[24h:]`
 - Deve ser 0

4. Uptime (Estatística)

- Consulta: `up{job="sms-c"}`
- Binário: 1 = ativo, 0 = inativo
- Mostra o status atual

5. Tendência da Taxa de Sucesso Diária (Gráfico)

- Consulta: `avg_over_time((rate(sms_c_delivery_succeeded_count[1h]) / rate(sms_c_delivery_queued_count[1h]))[24h:1h])`
- Intervalo de tempo: Últimos 30 dias
- Linha do SLA em 99%

Exemplos de Regras de Alerta

Alertas Críticos

Falhas de Roteamento:

```
alert: RoutingFailuresDetected
expr: increase(sms_c_routing_failed_count[5m]) > 0
for: 2m
labels:
  severity: critical
annotations:
  summary: "{{ $value }}" falhas de roteamento nos últimos 5 minutos"
  description: "Mensagens não podem ser roteadas. Verifique a configuração de roteamento."
```

Acúmulo na Fila de Mensagens:

```
alert: MessageQueueBacklog
expr: sms_c_queue_size_pending > 10000
for: 5m
labels:
  severity: critical
annotations:
  summary: "A fila de mensagens tem {{ $value }} mensagens pendentes"
  description: "A fila está se acumulando. Verifique o desempenho de entrega."
```

Mensagens Antigas na Fila:

```
alert: OldMessagesInQueue
expr: sms_c_queue_oldest_message_age_seconds > 300
for: 2m
labels:
  severity: critical
annotations:
  summary: "A mensagem mais antiga tem {{ $value }} segundos"
  description: "Mensagens não estão sendo entregues. Verifique os frontends."
```

Todos os Frontends Desconectados:

```
alert: NoActiveFrontends
```

```
expr: sum(sms_c_frontend_status_count{status="connected"}) == 0
for: 1m
labels:
  severity: critical
annotations:
  summary: "Nenhum frontend conectado"
  description: "Nenhum caminho de entrega disponível. Verifique a conectividade do frontend."
```

Crescimento da Fila de Cartas Mortas:

```
alert: DeadLetterMessagesIncreasing
expr: rate(sms_c_delivery_dead_letter_count[10m]) > 0
for: 5m
labels:
  severity: critical
annotations:
  summary: "{{ $value }}" mensagens movidas para a fila de cartas mortas"
  description: "Mensagens estão se tornando indesejáveis. Investigue as falhas."
```

Alertas de Aviso

Baixa Taxa de Sucesso de Entrega:

```
alert: LowDeliverySuccessRate
expr: (rate(sms_c_delivery_succeeded_count[10m]) /
rate(sms_c_delivery_queued_count[10m])) < 0.95
for: 10m
labels:
  severity: warning
annotations:
  summary: "A taxa de sucesso de entrega é {{ $value | humanizePercentage }}"
  description: "Taxa de sucesso abaixo de 95%. Investigue as falhas de entrega."
```

Alta Taxa de Reenvio:

```
alert: HighDeliveryRetryRate
expr: histogram_quantile(0.95, sms_c_delivery_succeeded_attempt_count_bucket) > 2
for: 15m
labels:
  severity: warning
annotations:
  summary: "Tentativas de entrega no percentil 95: {{ $value }}"
  description: "Mensagens exigindo múltiplas tentativas. Verifique a confiabilidade do destino."
```

Processamento Lento de Mensagens:

```
alert: SlowMessageProcessing
expr: histogram_quantile(0.95, sms_c_message_processing_stop_duration_bucket) > 1000
for: 10m
labels:
  severity: warning
annotations:
  summary: "Tempo de processamento no percentil 95: {{ $value }}ms"
  description: "O processamento de mensagens está lento. Verifique os recursos do"
```


sistema."

Consultas ENUM Falhando:

```
alert: HighEnumFailureRate
expr: rate(sms_c_enum_lookup_stop_duration_count{success="false"}[10m]) > 0.1
for: 10m
labels:
  severity: warning
annotations:
  summary: "Taxa de falha de consulta ENUM: {{ $value }}"
  description: "Consultas DNS falhando. Verifique os servidores DNS."
```

Baixa Taxa de Acerto do Cache ENUM:

```
alert: LowEnumCacheHitRate
expr: rate(sms_c_enum_cache_hit_count[10m]) /
(rate(sms_c_enum_cache_hit_count[10m]) + rate(sms_c_enum_cache_miss_count[10m])) <
0.70
for: 30m
labels:
  severity: warning
annotations:
  summary: "Taxa de acerto do cache ENUM: {{ $value | humanizePercentage }}"
  description: "Baixa eficiência do cache. Pode indicar tráfego de números únicos."
```

Falhas de Cobrança:

```
alert: ChargingFailuresDetected
expr: rate(sms_c_charging_failed_count[10m]) > 0.05
for: 10m
labels:
  severity: warning
annotations:
  summary: "Taxa de falha de cobrança: {{ $value }}"
  description: "Erros no sistema de cobrança. Verifique a conectividade do OCS."
```

Notas Adicionais

- Todas as métricas de duração usam precisão de nanosegundos internamente, mas são convertidas para milissegundos para relatórios.
- Métricas de contador são cumulativas e devem ser usadas com as funções `rate()` ou `increase()` nas consultas Prometheus.
- Métricas de gauge representam valores instantâneos no momento da coleta.
- Métricas de histograma fornecem cálculos percentuais (p50, p95, p99) e podem ser usadas para criar mapas de calor.
- Todas as métricas incluem rótulos padrão adicionados pelo Prometheus (instância, trabalho, etc.).
- Ao criar dashboards, use intervalos de tempo apropriados: 5m para tempo real, 1h para tendências, 24h+ para planejamento de capacidade.
- Configure regras de gravação no Prometheus para consultas complexas frequentemente usadas para melhorar o desempenho do dashboard.
- Use modelagem de variáveis no Grafana para dashboards dinâmicos (selecionar `dest_smsc`, `source_smsc`, etc.).



Guia de Operações SMS-C

[← Voltar ao Índice da Documentação](#) | [README Principal](#)

Procedimentos operacionais diários, monitoramento e tarefas de manutenção para equipes de operações SMS-C.

Índice

- [Operações Diárias](#)
- [Monitoramento](#)
- [Rastreamento de Mensagens](#)
- [Gerenciamento de Rotas](#)
- [Gerenciamento de Frontend](#)
- [Gerenciamento de Tradução de Números](#)
- [Manutenção do Sistema](#)
- [Backup e Recuperação](#)
- [Planejamento de Capacidade](#)
- [Resposta a Incidentes](#)

Operações Diárias

Verificação de Saúde Matinal

Realize estas verificações no início de cada dia:

1. Verificar Status do Sistema

```
# Verificação de saúde da API
curl https://api.example.com:8443/api/status

# Resposta esperada:
#
{"status":"ok","application":"OmniMessage","timestamp":"2025-10-30T08:00:00Z"}
```

2. Revisar Métricas do Prometheus

Acesse o painel do Prometheus e verifique:

- Taxa de mensagens (últimas 24 horas)
- Taxa de falhas de roteamento (deve ser < 1%)
- Pendências na fila (deve ser < 1000 pendentes)
- Taxa de sucesso de entrega (deve ser > 95%)
- Status de conexão do frontend (todos os frontends esperados ativos)

3. Verificar Fila de Mensagens

Acesse a interface da Web: https://sms-admin.example.com/message_queue

Revise:

- Total de mensagens pendentes (deve ser baixo)
- Idade da mensagem mais antiga (deve ser < 5 minutos)
- Mensagens com altas tentativas de entrega (investigar se > 3)
- Mensagens de carta morta (investigar quaisquer presentes)

4. Revisar Status do Frontend

Acesse a interface da Web: https://sms-admin.example.com/frontend_status

Verifique:

- Todos os frontends esperados estão ativos
- Sem desconexões não expiradas
- Sem erros de frontend nas últimas 24 horas

5. Verificar Logs da Aplicação

Acesse a interface da Web: <https://sms-admin.example.com/logs> ou verifique os arquivos de log

Procure por:

- Mensagens de nível de erro
- Falhas de roteamento
- Falhas de cobrança
- Problemas de conexão com o banco de dados
- Problemas de nós do cluster

Monitoramento do Volume de Mensagens

Verificar Contagem de Mensagens Horárias:

Use a consulta do Prometheus:

```
# Mensagens recebidas por hora
increase(sms_c_message_received_count[1h])

# Mensagens entregues por hora
increase(sms_c_delivery_succeeded_count[1h])

# Calcular taxa de entrega
rate(sms_c_delivery_succeeded_count[1h]) /
rate(sms_c_message_received_count[1h])
```

Padrões Esperados:

- Horário comercial: Volume mais alto
- Noites/fins de semana: Volume mais baixo
- Taxa de entrega: Deve ser > 95%

Condições de Alerta:

- Queda súbita nas mensagens (> 50% de diminuição)

- Aumento súbito nas mensagens (> 200% de aumento)
- Queda da taxa de entrega abaixo de 90%

Monitoramento

Métricas Chave a Observar

Métricas de Processamento de Mensagens

Contagem de Mensagens Recebidas (sms_c_message_received_count):

- **O que:** Total de mensagens entrando no sistema
- **Alerta:** Queda ou pico súbito
- **Consulta:** rate(sms_c_message_received_count[5m])

Duração do Processamento de Mensagens

(sms_c_message_processing_stop_duration):

- **O que:** Tempo de processamento de ponta a ponta
- **Alerta:** p95 > 1000ms
- **Consulta:** histogram_quantile(0.95, sms_c_message_processing_stop_duration)

Métricas de Roteamento

Falhas de Roteamento (sms_c_routing_failed_count):

- **O que:** Mensagens que não puderam ser roteadas
- **Alerta:** Quaisquer falhas (> 0)
- **Consulta:** increase(sms_c_routing_failed_count[5m])

Rota Correspondida (sms_c_routing_route_matched_count):

- **O que:** Quais rotas estão sendo usadas
- **Alerta:** Rotas de alta prioridade não correspondendo
- **Consulta:** sms_c_routing_route_matched_count

Métricas de Entrega

Taxa de Sucesso de Entrega:

- **O que:** Percentual de entregas bem-sucedidas
- **Alerta:** Taxa < 95%
- **Consulta:** rate(sms_c_delivery_succeeded_count[5m]) / rate(sms_c_delivery_queued_count[5m])

Tentativas de Entrega (sms_c_delivery_succeeded_attempt_count):

- **O que:** Retentativas necessárias para entrega
- **Alerta:** p95 > 2 (muitas retentativas)
- **Consulta:** histogram_quantile(0.95, sms_c_delivery_succeeded_attempt_count)

Métricas de Fila

Tamanho da Fila (sms_c_queue_size_size):

- **O que:** Total de mensagens na fila
- **Alerta:** Tamanho > 10.000
- **Consulta:** sms_c_queue_size_size

Idade da Mensagem Mais Antiga (sms_c_queue_oldest_message_age_seconds):

- **O que:** Idade da mensagem pendente mais antiga
- **Alerta:** Idade > 300 segundos
- **Consulta:** sms_c_queue_oldest_message_age_seconds

Configuração do Painel

Painéis do Painel Operacional:

1. Taxa de Mensagens (Gráfico)

- Mensagens recebidas (taxa de 5 minutos)
- Mensagens entregues (taxa de 5 minutos)
- Intervalo de tempo: Últimas 24 horas

2. Status da Fila (Estatísticas Únicas)

- Mensagens pendentes atuais
- Idade da mensagem mais antiga
- Contagem de mensagens falhadas

3. Desempenho de Entrega (Gráfico)

- Taxa de sucesso ao longo do tempo
- Taxa de falha ao longo do tempo
- Intervalo de tempo: Últimas 24 horas

4. Status de Roteamento (Tabela)

- ID da Rota
- Contagem de correspondências (última hora)
- SMSC de destino
- Prioridade

5. Status do Frontend (Tabela)

- Nome do frontend
- Status (ativo/expirado)
- Última vez visto
- Contagem de mensagens (última hora)

6. Saúde do Sistema (Estatísticas Únicas)

- Tempo de resposta da API (p95)
- Tempo de consulta ao banco de dados (p95)

- Tempo de consulta ENUM (p95)

Configuração de Alertas

Alertas Críticos (Resposta Imediata Necessária):

```
# Nenhuma rota encontrada - mensagens não podem ser entregues
- alert: RoutingFailures
  expr: increase(sms_c_routing_failed_count[5m]) > 0
  severity: critical
  description: "{{ $value }}" mensagens falharam no roteamento nos últimos 5 minutos"

# Fila acumulando - processamento ficando para trás
- alert: QueueBacklog
  expr: sms_c_queue_size_pending > 10000
  severity: critical
  description: "A fila tem {{ $value }}" mensagens pendentes"

# Mensagens envelhecendo - entrega travada
- alert: OldMessagesInQueue
  expr: sms_c_queue_oldest_message_age_seconds > 300
  severity: critical
  description: "A mensagem mais antiga tem {{ $value }}" segundos"

# Frontend desconectado - sem caminho de entrega
- alert: FrontendDisconnected
  expr: sms_c_frontend_status_count{status="disconnected"} > 0
  severity: critical
  description: "{{ $value }}" frontends desconectados"
```

Alertas de Aviso (Investigação Necessária):

```
# Taxa de sucesso de entrega caindo
- alert: LowDeliveryRate
  expr: rate(sms_c_delivery_succeeded_count[10m]) /
rate(sms_c_delivery_queued_count[10m]) < 0.90
  severity: warning
  description: "A taxa de sucesso de entrega é {{ $value }}"

# Muitas tentativas de entrega
- alert: HighRetryRate
  expr: histogram_quantile(0.95, sms_c_delivery_succeeded_attempt_count) > 2
  severity: warning
  description: "Tentativas de entrega no 95º percentil: {{ $value }}"

# Consultas ENUM lentas ou falhando
- alert: SlowEnumLookups
  expr: histogram_quantile(0.95, sms_c_enum_lookup_stop_duration) > 5000
  severity: warning
  description: "Consultas ENUM levando > 5 segundos"

# Baixa taxa de acerto de cache ENUM
- alert: LowEnumCacheHitRate
```

```
expr: rate(sms_c_enum_cache_hit_count[10m]) /  
(rate(sms_c_enum_cache_hit_count[10m]) +  
rate(sms_c_enum_cache_miss_count[10m])) < 0.70  
severity: warning  
description: "Taxa de acerto de cache ENUM: {{ $value }}"
```

Rastreamento de Mensagens

Encontrar Mensagem Específica

Por ID da Mensagem:

1. **Interface da Web:** Navegue até /message_queue
2. Insira o ID da mensagem na caixa de pesquisa
3. Veja os detalhes completos e o histórico de eventos

Via API:

```
curl https://api.example.com:8443/api/messages/12345
```

Por Número de Telefone:

1. **Interface da Web:** Navegue até /message_queue
2. Insira o número de telefone na caixa de pesquisa
3. Veja todas as mensagens para esse número

Rastrear Ciclo de Vida da Mensagem

Ver Histórico de Eventos:

1. **Interface da Web:** Clique na mensagem na fila, veja a seção "Eventos"
2. **API:** GET /api/events/12345

Sequência Comum de Eventos:

1. message_inserted - Mensagem criada
↓
2. number_translated - Números normalizados (se configurado)
↓
3. message_routed - Decisão de roteamento tomada
↓
4. charging_attempted - Verificação de cobrança (se habilitada)
↓
5. message_delivered - Entregue com sucesso

Sequência de Entrega Falhada:

1. message_inserted
↓
2. message_routed
↓
3. delivery_attempt_1 - Primeira tentativa falhou
↓
4. delivery_attempt_2 - Segunda tentativa falhou (2min de atraso)

- ↓
5. `delivery_attempt_3` - Terceira tentativa falhou (4min de atraso)
↓
6. `message_dead_letter` - Limite de tentativas excedido

Verificar Status de Entrega

Mensagens Pendentes:

- Status: "pendente"
- `deliver_after`: Timestamp futuro
- `delivery_attempts`: 0 ou número baixo

Mensagens Entregues:

- Status: "entregue"
- `deliver_time`: Timestamp de entrega
- `dest_smsc`: Frontend que entregou

Mensagens Falhadas:

- Status: "pendente" com altas `delivery_attempts`
- `deadletter`: true (se expirado)
- Verifique o log de eventos para razões de falha

Roteamento de Mensagens Baseado em Localização

O SMS-C suporta a recuperação de mensagens baseada em localização, permitindo que frontends recebam automaticamente mensagens destinadas a assinantes registrados em sua localização.

Como Funciona:

Quando um frontend consulta por mensagens pendentes usando `get_messages_for_smsc(smsc_name)`, o sistema retorna mensagens de duas maneiras:

1. **Roteamento Explícito** - Mensagens onde `dest_smsc` corresponde explicitamente ao nome do frontend
2. **Roteamento Baseado em Localização** - Mensagens onde:
 - `dest_smsc` é null (não roteado explicitamente)
 - `destination_msisdn` tem um registro de localização ativo
 - O campo `location` da localização corresponde ao nome do frontend
 - A localização não expirou

Cenário de Exemplo:

Um assinante com MSISDN +447700900123 se registra no frontend `uk_gateway`:

```
# Assinante se registra (cria registro de localização)
POST /api/locations
{
  "msisdn": "+447700900123",
  "imsi": "234150123456789",
  "location": "uk_gateway",
```



```
"expires": "2025-11-01T12:00:00Z"
}
```

Quando uma mensagem chega para esse assinante sem roteamento explícito:

```
# Mensagem submetida sem dest_smsc
POST /api/messages
{
  "source_msisdn": "+15551234567",
  "destination_msisdn": "+447700900123",
  "message_body": "Hello",
  "source_smsc": "api"
  # Nota: dest_smsc é null
}
```

O frontend uk_gateway receberá automaticamente essa mensagem quando fizer a consulta:

```
# Frontend consulta por mensagens
GET /api/messages/queue?smsc=uk_gateway

# Retorna a mensagem mesmo que dest_smsc seja null
# porque o assinante de destino está registrado em uk_gateway
```

Requisitos de Localização:

Para que o roteamento baseado em localização não funcione:

- A tabela locations deve ter uma entrada para o destination_msisdn
- O campo location deve corresponder ao nome do SMSC consultado
- O timestamp expires deve estar no futuro

Monitoramento do Roteamento Baseado em Localização:

Verifique os registros de localização:

```
# Via API
GET /api/locations/{msisdn}

# Verifique se a localização expirou
# o campo expires deve ser > hora atual
```

Problemas Comuns:

- **Mensagem não entregue:** Verifique se a localização expirou
- **Frontend errado:** Verifique se o campo location corresponde ao nome do frontend esperado
- **Localização não encontrada:** O assinante pode precisar se registrar novamente

Intervenções Manuais

Tentar Novamente Mensagem Falhada:

```
# Redefinir delivery_attempts e deliver_after
curl -X PATCH https://api.example.com:8443/api/messages/12345 \
```

```
-H "Content-Type: application/json" \  
-d '{  
  "delivery_attempts": 0,  
  "deliver_after": "2025-10-30T12:00:00Z"  
'
```

Alterar Destino:

```
# Roteamento para SMSC diferente  
curl -X PATCH https://api.example.com:8443/api/messages/12345 \  
-H "Content-Type: application/json" \  
-d '{  
  "dest_smsc": "backup_gateway"  
'
```

Excluir Mensagem Travada:

```
curl -X DELETE https://api.example.com:8443/api/messages/12345
```

Gerenciamento de Rotas

Ver Rotas Atuais

Interface da Web: Navegue até /sms_routing

Via API:

```
# Listar todas as rotas  
curl https://api.example.com:8443/api/routes
```

Verificar Uso da Rota:

Consulta do Prometheus:

```
# Mensagens roteadas por cada rota (última hora)  
increase(sms_c_routing_route_matched_count[1h])
```

Adicionar Nova Rota

Interface da Web:

1. Navegue até /sms_routing
2. Clique em "Adicionar Nova Rota"
3. Preencha os campos:
 - **Prefixo de Chamada:** Prefixo do número de origem (opcional)
 - **Prefixo Chamado:** Prefixo do número de destino (obrigatório para roteamento geográfico)
 - **SMSC de Origem:** Filtro do sistema de origem (opcional)
 - **SMSC de Destino:** Gateway de destino (obrigatório, a menos que seja resposta automática/descarte)
 - **Prioridade:** Prioridade da rota (1-255, menor = maior prioridade)
 - **Peso:** Peso de balanceamento de carga (1-100)
 - **Descrição:** Descrição legível por humanos

- **Habilitado:** Marque para ativar imediatamente
4. Clique em "Salvar Rota"

Exemplo: Rota Geográfica:

- Prefixo Chamado: +44
- SMSC de Destino: uk_gateway
- Prioridade: 50
- Peso: 100
- Descrição: "Roteamento do Reino Unido"

Exemplo: Rota Balanceada:

Crie duas rotas com os mesmos critérios, mas pesos diferentes:

Rota 1:

- Prefixo Chamado: +44
- SMSC de Destino: uk_primary
- Prioridade: 50
- Peso: 70
- Descrição: "Reino Unido primário (70%)"

Rota 2:

- Prefixo Chamado: +44
- SMSC de Destino: uk_backup
- Prioridade: 50
- Peso: 30
- Descrição: "Reino Unido de backup (30%)"

Testar Rotas

Simulador de Roteamento:

1. Navegue até /simulator
2. Insira parâmetros de teste:
 - Número Chamador: +15551234567
 - Número Chamado: +447700900000
 - SMSC de Origem: (opcional)
 - Tipo de Origem: (opcional)
3. Clique em "Simular Roteamento"
4. Revise os resultados:
 - **Rota Selecionada:** Qual rota foi escolhida
 - **Todas as Correspondências:** Quais rotas corresponderam aos critérios
 - **Avaliação:** Por que cada rota correspondeu ou não correspondeu

Testar Antes da Produção:

- Teste todas as novas rotas no simulador
- Verifique se a rota correta é selecionada
- Verifique a ordem de prioridade
- Valide a distribuição de peso

Modificar Rota Existente

Interface da Web:

1. Navegue até /sms_routing
2. Encontre a rota na lista
3. Clique em "Editar"
4. Modifique os campos
5. Clique em "Salvar Rota"

Modificações Comuns:

- **Desabilitar Rota:** Desmarque "Habilitado" (remoção temporária)
- **Ajustar Peso:** Alterar distribuição de balanceamento de carga
- **Alterar Prioridade:** Reordenar avaliação da rota
- **Atualizar Destino:** Mudar para SMSC diferente

Excluir Rota

Interface da Web:

1. Navegue até /sms_routing
2. Encontre a rota na lista
3. Clique em "Excluir"
4. Confirme a exclusão

Aviso: Excluir rotas é permanente. Considere desabilitar em vez disso.

Exportar/Importar Rotas

Exportar Rotas (Backup):

1. Navegue até /sms_routing
2. Clique em "Exportar Rotas"
3. Salve o arquivo JSON

Importar Rotas:

1. Navegue até /sms_routing
2. Clique em "Importar Rotas"
3. Selecione o arquivo JSON
4. Escolha o modo de importação:
 - **Mesclar:** Adicionar às rotas existentes
 - **Substituir:** Excluir todas e importar

Casos de Uso:

- Backup antes de grandes alterações
- Copiar rotas entre ambientes
- Recuperação de desastres
- Versionamento de configuração

Gerenciamento de Frontend

Monitorar Conexões de Frontend

Interface da Web: Navegue até /frontend_status

Verifique:

- Todos os frontends esperados estão "ativos"
- Os horários da última visualização são recentes (< 90 segundos)
- Sem frontends expirados inesperados

Via API:

```
# Obter frontends ativos
curl https://api.example.com:8443/api/frontends/active

# Obter estatísticas
curl https://api.example.com:8443/api/frontends/stats
```

Investigar Desconexões

Frontend Expirado:

1. Verifique os logs do frontend para erros
2. Verifique a conectividade de rede com o SMS-C
3. Confirme se o frontend está em execução
4. Verifique a lógica de registro do frontend (deve se registrar a cada 60s)

Registro Não Aparecendo:

1. Verifique se o frontend está chamando POST /api/frontends/register
2. Verifique os logs da API para erros de registro
3. Verifique o formato do payload JSON
4. Teste o registro manualmente com curl

Exemplo de Registro Manual:

```
curl -X POST https://api.example.com:8443/api/frontends/register \
-H "Content-Type: application/json" \
-d '{
  "frontend_name": "test_gateway",
  "frontend_type": "smpp",
  "ip_address": "10.0.1.50",
  "hostname": "gateway.example.com"
}'
```

Ver Histórico do Frontend

Interface da Web:

1. Navegue até /frontend_status
2. Encontre o frontend na lista

3. Clique em "Histórico"
4. Revise registros de registros anteriores

Via API:

```
curl https://api.example.com:8443/api/frontends/history/uk_gateway
```

Casos de Uso:

- Investigar a confiabilidade da conexão
- Rastrear padrões de tempo de atividade do frontend
- Identificar alterações de configuração

Gerenciamento de Tradução de Números

As regras de tradução de números são gerenciadas via `config/runtime.exs`. Alterações requerem reinício da aplicação.

Ver Regras de Tradução Ativas

Verifique o arquivo de configuração:

```
cat config/runtime.exs | grep -A 20 "translation_rules:"
```

Tarefas Comuns de Tradução

Adicionar Código de País a Números Locais:

Edite `config/runtime.exs`:

```
%{
  calling_prefix: nil,
  called_prefix: nil,
  source_smsc: nil,
  calling_match: "^(\\d{10})$",
  calling_replace: "+1\\1",
  called_match: "^(\\d{10})$",
  called_replace: "+1\\1",
  priority: 100,
  description: "Adicionar +1 a números de 10 dígitos dos EUA",
  enabled: true
}
```

Normalizar Formato Internacional:

```
%{
  calling_prefix: nil,
  called_prefix: nil,
  source_smsc: nil,
  calling_match: "^00(\\d+)$",
  calling_replace: "+\\1",
  called_match: "^00(\\d+)$",
  called_replace: "+\\1",
  priority: 10,
}
```

```
description: "Converter prefixo 00 para +",
enabled: true
}
```

Remoção de Código Específico do Transportador:

```
%{
  calling_prefix: nil,
  called_prefix: "101",
  source_smsc: "carrier_a",
  calling_match: nil,
  calling_replace: nil,
  called_match: "^101(\\d+)$",
  called_replace: "\\1",
  priority: 5,
  description: "Remover código do transportador do transportador A",
  enabled: true
}
```

Testar Regras de Tradução

Após alterações na configuração:

1. Reinicie a aplicação para carregar novas regras
2. Envie uma mensagem de teste com origem/destino que deve corresponder
3. Verifique o log de eventos para o evento `number_translated`
4. Verifique se os números foram transformados corretamente

Desabilitar Regra de Tradução

Defina `enabled: false` na regra:

```
%{
  ...
  enabled: false
}
```

Reinicie a aplicação.

Manutenção do Sistema

Manutenção do Banco de Dados

Verificar Tamanho do Banco de Dados:

Use suas ferramentas de gerenciamento de banco de dados para monitorar o tamanho de armazenamento de CDR:

- **MySQL/MariaDB:** Consulte `information_schema.tables` para o tamanho do banco de dados
- **PostgreSQL:** Use a função `pg_database_size()` ou o comando `\l+` no `psql`

Limpar Registros CDR Antigos:

Os registros CDR devem ser arquivados e purgados periodicamente com base na sua política de retenção:

- Configure arquivamento automático com base nos requisitos de negócios (tipicamente 30-90 dias no banco de dados operacional)
- Arquive registros mais antigos para um data warehouse ou armazenamento frio
- Exclua registros arquivados do banco de dados operacional em lotes para evitar contenção de bloqueio

Otimizar Tabelas:

Periodicamente otimize as tabelas do banco de dados para manter o desempenho:

- **MySQL/MariaDB:** Execute o comando `OPTIMIZE TABLE` durante períodos de baixo tráfego
- **PostgreSQL:** Execute `VACUUM ANALYZE` regularmente (ou habilite autovacuum)

Executar Semanalmente durante período de baixo tráfego para manter desempenho ideal.

Manutenção do Banco de Dados Mnesia

Verificar Tamanho da Tabela Mnesia:

```
# No console IEx
:mnesia.table_info(:sms_route, :size)
:mnesia.table_info(:translation_rule, :size)
```

Backup das Tabelas Mnesia:

```
# Exportar rotas (Interface da Web)
# Navegue até /sms_routing
# Clique em "Exportar Rotas"

# Ou via backup do Mnesia
:mnesia.backup("/var/backups/sms_c/mnesia_backup.bup")
```

Restaurar Mnesia:

```
# Via importação da Interface da Web
# Ou restaurar backup:
:mnesia.restore("/var/backups/sms_c/mnesia_backup.bup", [])
```

Rotação de Logs

Configure logrotate para logs da aplicação:

```
# /etc/logrotate.d/sms_c
/var/log/sms_c/*.log {
    daily
    rotate 30
    compress
    delaycompress
    notifempty
```



```
create 0644 sms_user sms_group
sharedscripts
postrotate
    systemctl reload sms_c || true
endscript
}
```

Reiniciar a Aplicação

Reinício Gracioso (zero downtime em cluster):

```
# Reinicie um nó de cada vez
systemctl restart sms_c

# Aguarde o nó se juntar ao cluster
# Repita para cada nó
```

Reinício de Emergência (todos os nós):

```
systemctl restart sms_c
```

Após o Reinício:

- Verifique se todos os frontends se reconectam
- Verifique o Prometheus para continuidade de métricas
- Monitore logs para erros
- Verifique se o processamento de mensagens é retomado

Backup e Recuperação

O que Fazer Backup

1. Arquivos de Configuração:

- config/runtime.exs
- config/config.exs
- config/prod.exs (se existir)

2. Tabelas de Roteamento (Mnesia):

- Exportar via Interface da Web
- Ou comando de backup do Mnesia

3. Banco de Dados SQL CDR:

- Backup completo diário
- Backups de log de transações (contínuos)

4. Certificados TLS:

- priv/cert/*.crt
- priv/cert/*.key

Procedimentos de Backup

Backup Diário de Configuração:

```
#!/bin/bash
# /opt/sms_c/scripts/backup_config.sh

BACKUP_DIR="/var/backups/sms_c/$(date +%Y%m%d)"
mkdir -p $BACKUP_DIR

# Backup de configuração
cp -r /opt/sms_c/config $BACKUP_DIR/

# Backup de certificados
cp -r /opt/sms_c/priv/cert $BACKUP_DIR/

# Definir permissões
chmod 600 $BACKUP_DIR/cert/*

echo "Backup de configuração concluído: $BACKUP_DIR"
```

Backup do Banco de Dados:

```
#!/bin/bash
# /opt/sms_c/scripts/backup_database.sh

BACKUP_DIR="/var/backups/sms_c/database"
DATE=$(date +%Y%m%d_%H%M%S)

mkdir -p $BACKUP_DIR

# Backup do banco de dados SQL CDR
# MySQL/MariaDB: Use mysqldump com --single-transaction para consistência
# PostgreSQL: Use pg_dump -F c para formato personalizado

# Estrutura de exemplo (adapte ao seu banco de dados):
# - Use a ferramenta de backup apropriada (mysqldump, pg_dump)
# - Habilite backups seguros para consistência
# - Comprimir saída para economizar espaço
# - Configurar período de retenção (por exemplo, 30 dias)

# Remover backups antigos
find $BACKUP_DIR -name "sms_c_*.gz" -mtime +30 -delete

echo "Backup do banco de dados concluído: sms_c_${DATE}"
```

Backup da Tabela de Roteamento:

```
#!/bin/bash
# /opt/sms_c/scripts/backup_routes.sh

BACKUP_DIR="/var/backups/sms_c/routes"
DATE=$(date +%Y%m%d)
```

```
mkdir -p $BACKUP_DIR

# Exportar via API
curl https://api.example.com:8443/api/routes/export \
  > $BACKUP_DIR/routes_$(date +%Y%m%d).json

echo "Backup das rotas concluído: routes_$(date +%Y%m%d).json"
```

Agendar Backups (crontab):

```
# Diário às 2 AM
0 2 * * * /opt/sms_c/scripts/backup_config.sh
0 2 * * * /opt/sms_c/scripts/backup_database.sh
0 2 * * * /opt/sms_c/scripts/backup_routes.sh
```

Procedimentos de Recuperação

Restaurar Configuração:

```
# Parar aplicação
systemctl stop sms_c

# Restaurar arquivos de configuração
cp -r /var/backups/sms_c/20251030/config/* /opt/sms_c/config/

# Restaurar certificados
cp -r /var/backups/sms_c/20251030/cert/* /opt/sms_c/priv/cert/

# Iniciar aplicação
systemctl start sms_c
```

Restaurar Banco de Dados SQL CDR:

Use as ferramentas de restauração apropriadas para seu banco de dados:

- **MySQL/MariaDB:** Descompactar e canalizar para o cliente mysql
- **PostgreSQL:** Use pg_restore com dumps de formato personalizado

Importante: Pare a aplicação SMS-C antes de restaurar o banco de dados para evitar conflitos de dados.

Restaurar Tabelas de Roteamento:

1. Navegue até a Interface da Web /sms_routing
2. Clique em "Importar Rotas"
3. Selecione o arquivo JSON de backup
4. Escolha o modo "Substituir"
5. Confirme a importação

Planejamento de Capacidade

Monitorar Tendências de Crescimento

Tendência de Volume de Mensagens:

Consulta do Prometheus (média de 30 dias):

```
avg_over_time(sms_c_message_received_count[30d])
```

Taxa de Crescimento do Banco de Dados:

```
-- Crescimento de dados mensal
SELECT
  DATE_FORMAT(inserted_at, '%Y-%m') AS month,
  COUNT(*) AS message_count,
  ROUND(SUM(LENGTH(message_body)) / 1024 / 1024, 2) AS data_mb
FROM message_queues
GROUP BY month
ORDER BY month DESC
LIMIT 12;
```

Indicadores de Capacidade

Uso de CPU:

- **Normal:** < 50% em média
- **Alto:** > 70% sustentado
- **Crítico:** > 90%

Uso de Memória:

- **Normal:** < 70% do disponível
- **Alto:** > 80%
- **Crítico:** > 90%

Uso de Disco:

- **Normal:** < 60% cheio
- **Alto:** > 75%
- **Crítico:** > 85%

Profundidade da Fila:

- **Normal:** < 1000 pendentes
- **Alto:** > 5000 pendentes
- **Crítico:** > 10.000 pendentes

Recomendações de Escalonamento

Quando Escalonar Verticalmente (Atualizar Recursos):

- CPU consistentemente > 70%

- Memória consistentemente > 80%
- Gargalo em um único nó

Quando Escalonar Horizontalmente (Adicionar Nós):

- CPU > 50% em todos os nós
- Volume de mensagens > 5.000 msg/sec
- Distribuição geográfica necessária
- Alta disponibilidade requerida

Escalonamento do Banco de Dados:

- Réplicas de leitura para consultas de relatórios
- Otimização de pool de conexões
- Otimização de índices
- Particionar tabelas grandes por data

Resposta a Incidentes

Níveis de Severidade

Crítico (Resposta Imediata):

- Nenhuma mensagem sendo entregue
- Todos os frontends desconectados
- Banco de dados indisponível
- API completamente fora do ar

Alto (Resposta em até 1 hora):

- Taxa de sucesso de entrega < 80%
- Vários frontends desconectados
- Falhas de roteamento > 10%
- Fila de pendências crescendo

Médio (Resposta em até 4 horas):

- Um único frontend desconectado
- Taxa de sucesso de entrega 80-95%
- Processamento de mensagens lento
- Consultas ENUM falhando

Baixo (Resposta em até 24 horas):

- Degradação de desempenho menor
- Problema em uma única rota
- Alertas de aviso não críticos

Checklist de Incidentes

1. Avaliar Severidade:

- Verifique os alertas do Prometheus
- Revise as métricas do painel

- Verifique o status da fila de mensagens
- Verifique as conexões do frontend

2. Coletar Informações:

- Alterações recentes na configuração?
- Implantações recentes?
- Status de dependências externas (OCS, DNS)?
- Mensagens de erro nos logs?

3. Ações Imediatas:

- Pare as alterações em andamento
- Reverta implantações recentes se suspeitar da causa
- Habilite logs detalhados se necessário
- Notifique as partes interessadas

4. Investigação:

- Revise os logs da aplicação
- Verifique o uso de recursos do sistema
- Examine o desempenho do banco de dados
- Teste dependências externas

5. Resolução:

- Aplique a correção
- Teste no simulador
- Implante em produção
- Monitore para melhorias

6. Pós-Incidente:

- Documente a causa raiz
- Atualize monitoramento/alertas
- Implemente medidas preventivas
- Atualize os runbooks

Incidentes Comuns

Alta Pendência na Fila:

1. Verifique a taxa de sucesso de entrega
2. Verifique se os frontends estão conectados e consultando
3. Verifique o desempenho do banco de dados
4. Revise o Prometheus em busca de gargalos
5. Considere aumentar o tamanho do lote/intervalo

Falhas de Roteamento:

1. Revise a configuração de roteamento
2. Teste no simulador de roteamento
3. Verifique se há rotas ausentes
4. Verifique se a rota catch-all existe

5. Verifique os logs de eventos para razões de falha

Desconexões de Frontend:

1. Verifique o status do sistema do frontend
2. Verifique a conectividade de rede
3. Revise os logs do frontend
4. Teste o registro manual da API
5. Verifique as regras de firewall

Processamento de Mensagens Lento:

1. Verifique o desempenho das consultas do banco de dados
2. Revise a configuração do trabalhador de lotes
3. Verifique se há recursos adequados (CPU/Memória)
4. Verifique se há atrasos nas consultas ENUM
5. Revise o desempenho do sistema de cobrança

Para procedimentos detalhados de solução de problemas, consulte o [Guia de Solução de Problemas](#).



Guia de Otimização de Desempenho

[← Voltar ao Índice da Documentação](#) | [README Principal](#)

Este guia explica como otimizar o desempenho do SMS-C para diferentes cenários de carga de trabalho.

Visão Geral de Desempenho

O SMS-C oferece um throughput de **1.750 mensagens/segundo** usando Mnesia para armazenamento de mensagens em memória com arquivamento automático de banco de dados SQL para retenção de CDR.

Principais Métricas de Desempenho

Medido em Intel i7-8650U @ 1.90GHz (8 núcleos):

Operação	Throughput	Latência (média)	Melhoria
Inserção de Mensagem (com roteamento)	1.750 msg/sec	0.58ms	21x mais rápido que SQL
Inserção de Mensagem (simples)	1.750 msg/sec	0.57ms	21x mais rápido que SQL
Obter Mensagens para SMSC	800 msg/sec	1.25ms	Consulta em memória
Memória por Inserção	62 KB	-	Redução de 50%

Capacidade: ~150 milhões de mensagens por dia em um único nó

Índice

- [Arquitetura de Armazenamento de Mensagens](#)
- [Otimização do Mnesia](#)
- [Configuração de Arquivamento de CDR](#)
- [Otimização de Consultas](#)
- [Benchmarking](#)

Arquitetura de Armazenamento de Mensagens

O SMS-C utiliza uma arquitetura de armazenamento dual para desempenho ideal:

Armazenamento Ativo de Mensagens (Mnesia)

- **Propósito:** Inserção, roteamento e entrega de mensagens ultra-rápidos
- **Armazenamento:** Em memória com persistência em disco (disc_copies)
- **Desempenho:** 1.750 msg/sec de throughput de inserção, 0.58ms de latência
- **Retenção:** Configurável (padrão: 24 horas)
- **Clusterização:** Suporta Mnesia distribuído para escalabilidade horizontal

Arquivo de CDR (Banco de Dados SQL)

- **Propósito:** Histórico de mensagens a longo prazo e relatórios
- **Armazenamento:** Banco de dados SQL (MySQL/MariaDB ou PostgreSQL) para arquivamento durável
- **Desempenho:** Escritas em lotes para minimizar a carga do banco de dados
- **Retenção:** Permanente (ou conforme política de retenção de dados)
- **Consultas:** Análises, relatórios, conformidade

Fluxo de Dados

Otimização do Mnesia

Configuração de Retenção de Mensagens

```
# config/runtime.exs
config :sms_c,
  message_retention_hours: 24 # Padrão: 24 horas
```

Diretrizes de Ajuste:

- **Alto volume (>1M msg/dia):** 12-24 horas de retenção
 - Minimiza o tamanho da tabela Mnesia
 - Consultas mais rápidas
 - Arquivamento mais frequente para MySQL
- **Volume médio (100K-1M msg/dia):** 24-48 horas de retenção
 - Bom equilíbrio para a maioria das implantações
 - Buffer adequado para lógica de reintentos
- **Baixo volume (<100K msg/dia):** 48-168 horas de retenção
 - Histórico de mensagens mais longo em armazenamento rápido
 - Arquivamento menos frequente

Índices da Tabela Mnesia

O MessageStore cria automaticamente índices em:

- status - Para filtrar mensagens pendentes/entregues
- dest_smsc - Para consultas específicas do SMSC
- expires - Para gerenciamento de expiração
- destination_msisdn - Para consultas de assinantes
- source_msisdn - Para consultas de assinantes

Persistência em Disco do Mnesia

As mensagens são armazenadas como disc_copies, proporcionando:

- ⚡ Desempenho em memória
- ⚡ Persistência automática em disco
- ⚡ Recuperação de falhas
- ⚡ Sem perda de dados na reinicialização

Configuração de Arquivamento de CDR

O BatchInsertWorker gerencia o arquivamento de CDR para MySQL usando escritas em lotes:

```
# config/runtime.exs
config :sms_c,
  batch_insert_batch_size: 100,      # Tamanho do lote de CDR
  batch_insert_flush_interval_ms: 100 # Intervalo de auto-flush
```

Diretrizes de Ajuste de CDR

Arquivamento de Alto Volume

```
batch_insert_batch_size: 200
batch_insert_flush_interval_ms: 200
```

- Lotes maiores reduzem a carga do MySQL
- Latência mais alta para escritas de CDR (aceitável para arquivamento)

Equilibrado (Recomendado)

```
batch_insert_batch_size: 100
batch_insert_flush_interval_ms: 100
```

- Bom equilíbrio para a maioria das implantações
- CDRs escritos dentro de 100ms

Requisitos de CDR em Tempo Real

```
batch_insert_batch_size: 20  
batch_insert_flush_interval_ms: 20
```

- Escritas de CDR mais rápidas para conformidade
- Mais operações de escrita no MySQL

Otimização de Consultas

Usando Índices do Mnesia de Forma Eficiente

Consultas que utilizam campos indexados são as mais rápidas:

```
# Consultas rápidas (use índices)  
MessageStore.list(status: :pending)  
MessageStore.list(dest_smsc: "gateway-1")  
Messaging.get_messages_for_smsc("gateway-1")  
  
# Consultas mais lentas (varredura completa da tabela)  
MessageStore.list(limit: :infinity) # Retorna todas as mensagens
```

Pool de Conexões MySQL

Para consultas e arquivamento de CDR, configure o pool de conexões MySQL:

```
# config/runtime.exs  
config :sms_c, SmsC.Repo,  
  pool_size: 10 # Aumentar para relatórios pesados de CDR
```

Diretrizes:

- Implantação padrão: pool_size: 10
- Relatórios pesados de CDR: pool_size: 20-30
- Apenas arquivamento: pool_size: 5

Benchmarking

Executando Benchmarks

O projeto inclui benchmarks baseados em Benchec para testes de desempenho:

```
# Benchmark da API SMS bruta (compara sync vs async)  
mix run benchmarks/raw_sms_bench.exs  
  
# Benchmark da API de mensagens geral  
mix run benchmarks/message_api_bench.exs
```

Interpretando Resultados

Exemplo de saída:

Nome	ips	média	desvio
mediana 99º %			
submit_message_raw_async (batch)	4.65 K	0.22 ms	
±41.72% 0.184 ms 0.55 ms			
submit_message_raw (sync)	0.0696 K	14.36 ms	
±33.42% 12.57 ms 33.71 ms			

Métricas principais:

- **ips**: Iterações por segundo (quanto maior, melhor)
- **média**: Tempo médio de execução (quanto menor, melhor)
- **mediana**: Valor médio, mais representativo do que a média para distribuições assimétricas
- **99º %**: Latência do percentil 99 (importante para conformidade com SLA)

Linha de Base de Desempenho

Desempenho esperado em hardware moderno (Intel i7-8650U, 8 núcleos):

Métrica	insert_message (Mnesia)	Anterior (MySQL)
Throughput (com roteamento)	1.750 msg/sec	83 msg/sec
Throughput (simples)	1.750 msg/sec	89 msg/sec
Tempo de Resposta (média)	0.58ms	16ms
Tempo de Resposta (p99)	<5ms	30ms
Memória por operação	62 KB	121 KB
Ganho de Desempenho	21x mais rápido	-

Principais Melhorias:

- ✧ Remoção de chamadas duplicadas de tradução de números
- ✧ Pós-processamento assíncrono (roteamento, cobrança, eventos)
- ✧ Armazenamento em memória do Mnesia vs I/O de disco do MySQL
- ✧ Redução de 50% na memória

Monitoramento

Estatísticas de Tempo de Execução

Verifique as estatísticas do trabalhador de lote:

```
SmsC.Messaging.BatchInsertWorker.stats()
```

Retorna:

```
%{
  total_enqueued: 10000,
  total_flushed: 9900,
  total_batches: 99,
  current_queue_size: 100,
  flush_errors: 0,
  last_flush_at: ~U[2025-10-22 12:34:56Z],
  last_flush_count: 100,
  last_flush_duration_ms: 45
}
```

Métricas Chave a Monitorar

1. **Tamanho da Fila:** `current_queue_size` - Deve estar abaixo de `batch_size` na maior parte do tempo
2. **Duração do Flush:** `last_flush_duration_ms` - Deve ser $< 100\text{ms}$ para `batch_size=100`
3. **Erros de Flush:** `flush_errors` - Deve ser 0 ou muito baixo
4. **Throughput:** `total_flushed / uptime` - Deve corresponder à carga esperada

Alertas

Configure alertas de monitoramento para:

- Tamanho da fila consistentemente no máximo (indica pressão de retorno)
- Duração do flush aumentando (degradação do desempenho do banco de dados)
- Erros de flush > 0 (problemas de conectividade com o banco de dados)
- Throughput abaixo do esperado (degradação do desempenho)

Solução de Problemas

Sintoma: Baixo Throughput

Possíveis causas:

1. Pool de conexões do banco de dados esgotado: Aumente `pool_size`
2. Banco de dados lento: Verifique o desempenho da consulta, adicione índices
3. Latência de rede: Otimize o caminho de rede para o banco de dados
4. Tamanho do lote muito pequeno: Aumente `batch_insert_batch_size`

Sintoma: Alta Latência

Possíveis causas:

1. Intervalo de flush muito alto: Reduza `batch_insert_flush_interval_ms`
2. Tamanho do lote muito alto: Reduza `batch_insert_batch_size`
3. Escritas lentas no banco de dados: Verifique I/O de disco, otimize tabelas
4. Usando API assíncrona quando você precisa de síncrona: Mude para o endpoint síncrono

Sintoma: Problemas de Memória

Possíveis causas:

1. Fila acumulando: Mensagens se acumulando mais rápido do que o flush
2. Tamanho do lote muito grande: Reduza `batch_insert_batch_size`
3. Falhas de flush: Verifique `flush_errors` nas estatísticas
4. Necessidade de reiniciar o trabalhador: `Supervisor.terminate_child/2` e reiniciar

Melhores Práticas

1. **Comece com os padrões** (100/100ms) e ajuste com base no comportamento observado
2. **Monitore em produção** por pelo menos 1 semana antes de otimizar
3. **Teste alterações de configuração** em staging com carga semelhante à produção
4. **Use benchmarks** para validar alterações de configuração
5. **Documente suas decisões de ajuste** para referência futura
6. **Configure alertas** antes de otimizar para capturar regressões
7. **Considere fusos horários** - a carga máxima varia por região

Configurações de Exemplo

Configuração: Agregador de Alto Volume

```
# config/prod.exs
config :sms_c,
  batch_insert_batch_size: 200,
  batch_insert_flush_interval_ms: 200

config :sms_c, SmsC.Repo,
  pool_size: 50
```

Configuração: Mensagens em Tempo Real para Empresas

```
# config/prod.exs
config :sms_c,
  batch_insert_batch_size: 20,
  batch_insert_flush_interval_ms: 10
```

```
config :sms_c, SmsC.Repo,  
  pool_size: 20
```

Configuração: Desenvolvimento/Teste

```
# config/dev.exs  
config :sms_c,  
  batch_insert_batch_size: 10,  
  batch_insert_flush_interval_ms: 50  
  
config :sms_c, SmsC.Repo,  
  pool_size: 5
```

Leitura Adicional

- [Guia de Desempenho do Ecto](#)
- [Documentação do Benchee](#)
- [Phoenix Sob Pressão](#)



Guia de Solução de Problemas do SMS-C

[← Voltar ao Índice da Documentação](#) | [README Principal](#)

Guia abrangente para diagnosticar e resolver problemas comuns do SMS-C.

Índice

- [Ferramentas de Diagnóstico](#)
- [Problemas de Entrega de Mensagens](#)
- [Problemas de Roteamento](#)
- [Problemas de Desempenho](#)
- [Problemas de Banco de Dados](#)
- [Problemas de Conexão no Frontend](#)
- [Problemas de Cobrança/Faturamento](#)
- [Problemas de Pesquisa ENUM](#)
- [Problemas de Cluster](#)
- [Problemas de API](#)
- [Problemas na Interface Web](#)
- [Problemas de Recursos do Sistema](#)

Ferramentas de Diagnóstico

Verificação Rápida de Saúde

```
# 1. Verificar status da API
curl https://api.example.com:8443/api/status

# 2. Verificar endpoint de métricas do Prometheus
curl https://api.example.com:9568/metrics | grep sms_c

# 3. Verificar logs da aplicação
tail -f /var/log/sms_c/application.log

# 4. Verificar status do processo
systemctl status sms_c

# 5. Verificar conectividade com o banco de dados SQL CDR (MySQL/MariaDB)
mysql -u sms_user -p -h db.example.com -e "SELECT 1"
```



```
# Para PostgreSQL:  
# psql -U sms_user -h db.example.com -d sms_c_prod -c "SELECT 1"
```

Análise de Logs

Ver Erros Recentes:

```
# Últimas 100 entradas de log de nível de erro  
tail -1000 /var/log/sms_c/application.log | grep "\[error\]"  
  
# Procurar por padrões de erro específicos  
grep "routing_failed" /var/log/sms_c/application.log  
  
# Encontrar erros de banco de dados SQL  
grep -i "database\|sql\|ecto" /var/log/sms_c/application.log | grep  
error
```

Monitorar Logs em Tempo Real:

```
# Acompanhar logs com filtro  
tail -f /var/log/sms_c/application.log | grep -E  
"(error|warning|critical)"
```

Consultas de Métricas

Verificar Taxa de Processamento de Mensagens:

```
# Mensagens por segundo  
rate(sms_c_message_received_count[5m])  
  
# Taxa de sucesso de entrega  
rate(sms_c_delivery_succeeded_count[5m]) /  
rate(sms_c_delivery_queued_count[5m])
```

Verificar Status da Fila:

```
# Profundidade atual da fila  
sms_c_queue_size_pending  
  
# Idade da mensagem mais antiga (segundos)  
sms_c_queue_oldest_message_age_seconds
```

Verificar Desempenho do Sistema:

```
# Latência de processamento de mensagens (p95)  
histogram_quantile(0.95,  
sms_c_message_processing_stop_duration_bucket)
```

```
# Latência de roteamento (p95)
histogram_quantile(0.95, sms_c_routing_stop_duration_bucket)
```

Problemas de Entrega de Mensagens

Mensagens Não Sendo Entregues

Sintomas:

- Mensagens presas no status "pendente"
- Alta contagem de mensagens pendentes
- Sem notificações de entrega

Passos de Diagnóstico:

1. Verificar Conexões do Frontend:

```
curl https://api.example.com:8443/api/frontends/active
```

Esperado: Lista de frontends ativos

Problema: Lista vazia ou frontends esperados ausentes

2. Verificar Fila de Mensagens:

Acesse a Interface Web: /message_queue

- Filtrar por status: "pendente"
- Verificar valor dest_smsc
- Verificar se deliver_after não está no futuro

3. Verificar Roteamento:

Acesse a Interface Web: /simulator

- Testar com parâmetros de mensagem reais
- Verificar se a rota corresponde e o destino está correto

4. Verificar Polling do Frontend:

Revisar logs do sistema do frontend:

- O frontend está consultando /api/messages?
- O frontend está enviando o cabeçalho smsc corretamente?

Soluções:

Sem Frontends Conectados:

```
# Verificar status do sistema do frontend
systemctl status frontend_service

# Verificar se o frontend pode alcançar a API
curl -k https://api.example.com:8443/api/status

# Registrar frontend manualmente
curl -X POST https://api.example.com:8443/api/frontends/register \
  -H "Content-Type: application/json" \
  -d '{
    "frontend_name": "test_gateway",
    "frontend_type": "smpp",
    "ip_address": "10.0.1.50"
  }'
```

Mensagens Roteadas para o SMSC Errado:

- Revisar configuração de roteamento
- Verificar prioridades de rota
- Testar no simulador de roteamento
- Verificar se o nome do frontend corresponde ao dest_smsc nas mensagens

Mensagens Agendadas para o Futuro:

- Verificar timestamp deliver_after
- Reiniciar se necessário:

```
curl -X PATCH https://api.example.com:8443/api/messages/12345 \
  -H "Content-Type: application/json" \
  -d '{"deliver_after": "2025-10-30T12:00:00Z"}'
```

Mensagens Falhando com Tentativas de Reenvio

Sintomas:

- Contador delivery_attempts aumentando
- Mensagens com alta contagem de tentativas (> 3)
- Atrasos de backoff exponencial

Passos de Diagnóstico:

1. Verificar Log de Eventos:

```
curl https://api.example.com:8443/api/events/12345
```

Procurar por:

- Eventos de falha de entrega
- Descrições de erro

- Timestamps de reenvio

2. **Verificar Logs do Frontend:**

- Por que o frontend está falhando ao entregar?
- Erros de rede?
- Erros de protocolo?
- Sistema downstream indisponível?

Soluções:

Problemas Temporários de Rede:

- Aguardar reenvio (automático)
- Monitorar para entrega bem-sucedida

Falhas Persistentes:

```
# Roteamento para gateway alternativo
curl -X PATCH https://api.example.com:8443/api/messages/12345 \
  -H "Content-Type: application/json" \
  -d '{"dest_smsc": "backup_gateway"}'

# Reiniciar contador de tentativas
curl -X PATCH https://api.example.com:8443/api/messages/12345 \
  -H "Content-Type: application/json" \
  -d '{"delivery_attempts": 0, "deliver_after":
"2025-10-30T12:00:00Z"}'
```

Número de Destino Inválido:

- Verificar formato do número
- Verificar regras de tradução de números
- Excluir mensagem se realmente inválida

Mensagens de Carta Morta

Sintomas:

- deadletter: true na mensagem
- Mensagens além do tempo de expiração
- Status ainda "pendente"

Passos de Diagnóstico:

1. Encontrar Mensagens de Carta Morta:

Acesse a Interface Web: /message_queue

- Filtrar por status expirado
- Verificar timestamps de expiração

2. **Verificar Por Que Expirou:**

- Revisar log de eventos
- Verificar histórico de tentativas de entrega
- Verificar se o roteamento foi bem-sucedido

Soluções:

Estender Expiração:

```
# Adicionar 24 horas à expiração
curl -X PATCH https://api.example.com:8443/api/messages/12345 \
-H "Content-Type: application/json" \
-d '{"expires": "2025-10-31T12:00:00Z", "deadletter": false}'
```

Problemas de Roteamento

Nenhuma Rota Encontrada

Sintomas:

- Erro: no_route_found
- Métrica sms_c_routing_failed_count aumentando
- Log de eventos mostra "routing_failed"

Passos de Diagnóstico:

1. Verificar se as Rotas Existem:

Acesse a Interface Web: /sms_routing

- Verificar se as rotas estão configuradas
- Verificar se pelo menos uma rota está habilitada

2. Testar Roteamento:

Acesse a Interface Web: /simulator

- Inserir parâmetros de mensagem (número chamador, número chamado, SMSC de origem)
- Revisar resultados de avaliação
- Verificar por que as rotas não corresponderam

3. Verificar Critérios de Rota:

- Prefixos correspondem?
- Filtro de SMSC de origem muito restritivo?
- Todas as rotas desabilitadas?

Soluções:

Nenhuma Rota Configurada:

Adicionar rota catch-all:

```
Prefixo Chamador: (vazio)
Prefixo Chamado: (vazio)
SMSC de Origem: (vazio)
SMSC de Destino: default_gateway
Prioridade: 255
Peso: 100
Habilitado: ✓
Descrição: Rota padrão catch-all
```

Rotas Muito Específicas:

Adicionar rota mais ampla:

```
Prefixo Chamado: +
SMSC de Destino: international_gateway
Prioridade: 200
Peso: 100
Habilitado: ✓
Descrição: Catch-all internacional
```

Todas as Rotas Desabilitadas:

- Habilitar rotas apropriadas via Interface Web
- Verificar se a configuração não desabilitou acidentalmente as rotas

Rota Errada Selecionada

Sintomas:

- Mensagens roteadas para destino inesperado
- Gateway errado recebendo tráfego
- Balanceamento de carga não distribuindo como esperado

Passos de Diagnóstico:

1. Usar Simulador de Roteamento:

Acesse a Interface Web: /simulator

- Testar com parâmetros de mensagem reais
- Revisar seção "Todas as Correspondências"
- Verificar prioridades e pontuações de especificidade

2. **Verificar Prioridades de Rota:**

- Número menor = maior prioridade
- Rotas avaliadas em ordem de prioridade
- Dentro da mesma prioridade, pesos se aplicam

3. **Verificar Especificidade da Rota:**

Pontuação de especificidade:

- Prefixo chamado mais longo: +100 pontos por caractere
- Prefixo chamador mais longo: +50 pontos por caractere
- SMSC de origem especificado: +25 pontos
- Tipo de origem especificado: +10 pontos
- Domínio ENUM especificado: +15 pontos

Soluções:

Ajustar Prioridades:

Tornar rota específica de maior prioridade:

Rota Premium:

Prefixo Chamado: +1555

Prioridade: 10 (alta prioridade)

Rota Geral:

Prefixo Chamado: +1

Prioridade: 50 (prioridade mais baixa)

Ajustar Pesos:

Alterar distribuição de balanceamento de carga:

Primário (70%):

Peso: 70

Backup (30%):

Peso: 30

Adicionar Rota Mais Específica:

Substituir rota geral para caso específico:

Rota Específica:

```
Prefixo Chamado: +15551234
SMSC de Destino: dedicated_gateway
Prioridade: 1
```

```
Rota Geral:
Prefixo Chamado: +1
SMSC de Destino: general_gateway
Prioridade: 50
```

Resposta Automática Não Funcionando

Sintomas:

- Rota de resposta automática configurada, mas não acionando
- Nenhuma mensagem de resposta sendo enviada
- Log de eventos sem evento de resposta automática

Passos de Diagnóstico:

1. Verificar Configuração da Rota:

- `auto_reply: true`
- `auto_reply_message` contém texto
- Rota está habilitada
- Rota corresponde aos critérios da mensagem

2. Testar no Simulador:

- Verificar se a rota é selecionada
- Verificar indicação de "auto_reply"

3. Verificar Log de Eventos:

```
curl https://api.example.com:8443/api/events/12345 | grep auto_reply
```

Soluções:

Rota Não Correspondendo:

- Ampliar critérios (remover filtros)
- Verificar prioridade (deve ser maior que rotas normais)
- Verificar status habilitado

Mensagem Não Definida:

Editar rota, adicionar mensagem:

```
Resposta Automática: ✓
Mensagem de Resposta Automática: "Obrigado pela sua mensagem."
```


Responderemos em breve."

Prioridade Errada:

Rotas de resposta automática devem ter alta prioridade (número baixo):

Rota de Resposta Automática:

Prioridade: 10

Rota Normal:

Prioridade: 50

Problemas de Desempenho

Alta Latência de Processamento de Mensagens

Sintomas:

- sms_c_message_processing_stop_duration p95 > 1000ms
- Respostas lentas da API
- Fila acumulando

Passos de Diagnóstico:

1. Verificar Latências dos Componentes:

Latência de roteamento

```
histogram_quantile(0.95, sms_c_routing_stop_duration_bucket)
```

Latência de pesquisa ENUM

```
histogram_quantile(0.95, sms_c_enum_lookup_stop_duration_bucket)
```

Latência de cobrança

```
histogram_quantile(0.95, sms_c_charging_succeeded_duration_bucket)
```

Latência de entrega

```
histogram_quantile(0.95, sms_c_delivery_succeeded_duration_bucket)
```

2. Verificar Recursos do Sistema:

Uso de CPU

```
top -b -n 1 | grep sms_c
```

Uso de memória

```
ps aux | grep beam.smp
```

Soluções:

Roteamento Lento (Muitas rotas):

- Reduzir número de rotas habilitadas
- Combinar rotas semelhantes
- Otimizar critérios de rota

Pesquisas ENUM Lentas:

- Verificar latência do servidor DNS
- Aumentar timeout
- Usar servidores DNS mais rápidos/próximos
- Desabilitar ENUM se não necessário

Cobrança Lenta:

- Verificar desempenho do OCS
- Aumentar timeout do OCS
- Desabilitar cobrança se não necessário
- Usar cobrança assíncrona

Banco de Dados Lento:

- Aumentar tamanho do pool de conexões
- Adicionar índices
- Otimizar consultas
- Atualizar recursos do banco de dados

Alterações de Configuração:

```
# config/config.exs
# Aumentar tamanho do lote para throughput
config :sms_c,
  batch_insert_batch_size: 200,
  batch_insert_flush_interval_ms: 200

# Aumentar pool de banco de dados
config :sms_c, SmsC.Repo,
  pool_size: 50
```

Baixo Throughput de Mensagens

Sintomas:

- Processamento < 100 msg/sec
- Usando API assíncrona, mas ainda lenta
- Altos tempos de resposta da API

Passos de Diagnóstico:

1. Verificar Trabalhador de Lote:

```
# No console de produção (iex)  
SmsC.Messaging.BatchInsertWorker.stats()
```

Procurar por:

- `current_queue_size` próximo do máximo
- `flush_errors > 0`
- `last_flush_duration_ms` muito alto

2. Verificar Gargalos:

```
# Tempo de consulta do banco de dados  
ecto_pools_query_time
```

```
# Tempo de fila do pool de conexões  
ecto_pools_queue_time
```

Soluções:

Gargalo no Banco de Dados:

Aumentar tamanho do pool:

```
config :sms_c, SmsC.Repo,  
  pool_size: 50 # Aumentar de 20
```

Configuração de Lote:

Ajustar para throughput:

```
config :sms_c,  
  batch_insert_batch_size: 200, # Lotes maiores  
  batch_insert_flush_interval_ms: 200 # Intervalo mais longo
```

Usar Endpoint Assíncrono:

```
# Alto throughput: usar /create_async  
curl -X POST https://api.example.com:8443/api/messages/create_async
```

```
# NÃO: /api/messages (síncrono)
```

Acúmulo de Fila Crescendo

Sintomas:

- `sms_c_queue_size_pending` aumentando
- Idade da mensagem mais antiga aumentando

- Processamento não consegue acompanhar a taxa de entrada

Passos de Diagnóstico:

1. Verificar Taxa de Entrada vs Entrega:

```
# Taxa de entrada  
rate(sms_c_message_received_count[5m])  
  
# Taxa de entrega  
rate(sms_c_delivery_succeeded_count[5m])
```

2. Verificar Capacidade do Frontend:

- Os frontends estão consultando com frequência suficiente?
- Os frontends estão processando mensagens rapidamente o suficiente?
- Algum erro no frontend?

3. Verificar Taxa de Sucesso de Entrega:

```
rate(sms_c_delivery_succeeded_count[5m]) /  
rate(sms_c_delivery_attempted_count[5m])
```

Soluções:

Frontends Não Consultando:

- Verificar conectividade do frontend
- Verificar intervalo de polling (deve ser de 5 a 10 segundos)
- Reiniciar serviços do frontend

Frontends Muito Lentos:

- Adicionar mais instâncias de frontend
- Otimizar processamento do frontend
- Aumentar concorrência do frontend

Alta Taxa de Reenvio:

- Investigar falhas de entrega
- Corrigir problemas downstream
- Roteamento para gateways alternativos

Pico Temporário:

- Aguardar a fila esvaziar
- Monitorar até normal
- Considerar upgrades de capacidade se recorrente

Problemas de Banco de Dados

Falhas de Conexão

Sintomas:

- Erro: "unable to connect to database"
- API retornando erros 500
- Aplicação não inicia

Passos de Diagnóstico:

1. Verificar Status do Banco de Dados SQL CDR:

```
# MySQL/MariaDB
systemctl status mysql

# PostgreSQL
systemctl status postgresql

# Testar conectividade (MySQL/MariaDB)
mysql -u sms_user -p -h db.example.com -e "SELECT 1"

# Testar conectividade (PostgreSQL)
psql -U sms_user -h db.example.com -d sms_c_prod -c "SELECT 1"
```

2. Verificar Rede:

```
# Pingar host do banco de dados
ping db.example.com

# Verificar conectividade de porta (MySQL/MariaDB: 3306, PostgreSQL: 5432)
telnet db.example.com 3306
# ou
telnet db.example.com 5432
```

3. Verificar Credenciais:

```
# Verificar variáveis de ambiente
echo $DB_USERNAME
echo $DB_HOSTNAME
echo $DB_PORT

# Tentar conexão manual com as mesmas credenciais (MySQL/MariaDB)
mysql -u $DB_USERNAME -p$DB_PASSWORD -h $DB_HOSTNAME

# Para PostgreSQL:
```

```
# psql -U $DB_USERNAME -h $DB_HOSTNAME -d sms_c_prod
```

Soluções:

Banco de Dados Indisponível:

```
# Iniciar banco de dados (MySQL/MariaDB)  
systemctl start mysql
```

```
# Iniciar banco de dados (PostgreSQL)  
systemctl start postgresql
```

Credenciais Erradas:

Atualizar configuração:

```
export DB_USERNAME=correct_user  
export DB_PASSWORD=correct_password
```

```
# Reiniciar aplicação  
systemctl restart sms_c
```

Problema de Rede:

- Verificar regras de firewall
- Verificar grupos de segurança (nuvem)
- Verificar conectividade VPN/rede

Pool de Conexões Esgotado:

Aumentar tamanho do pool:

```
config :sms_c, SmsC.Repo,  
  pool_size: 50 # Aumentar do valor atual
```

Consultas Lentas

Sintomas:

- Tempo de consulta do banco de dados alto
- Respostas lentas da API
- Fila de conexão acumulando

Passos de Diagnóstico:

1. Verificar Log de Consultas Lentas:

```
-- MySQL/MariaDB: Habilitar log de consultas lentas  
SET GLOBAL slow_query_log = 'ON';
```

```

SET GLOBAL long_query_time = 1; -- Registrar consultas > 1 segundo

-- Visualizar consultas lentas (MySQL/MariaDB)
SELECT * FROM mysql.slow_log ORDER BY query_time DESC LIMIT 10;

-- PostgreSQL: Habilitar log de consultas lentas no postgresql.conf
-- log_min_duration_statement = 1000 # milissegundos
-- Em seguida, verificar logs do PostgreSQL

```

2. Verificar Índices Ausentes:

```

-- Verificar índices da tabela
SHOW INDEX FROM message_queues;

-- Índices esperados:
-- - source_smsc
-- - dest_smsc
-- - send_time
-- - inserted_at

```

3. Verificar Estatísticas da Tabela:

```

-- Tamanhos das tabelas (MySQL/MariaDB)
SELECT
    table_name,
    table_rows,
    ROUND(data_length / 1024 / 1024, 2) AS data_mb,
    ROUND(index_length / 1024 / 1024, 2) AS index_mb
FROM information_schema.tables
WHERE table_schema = 'sms_c_prod';

-- Tamanhos das tabelas (PostgreSQL)
-- SELECT schemaname, tablename,
--
pg_size_pretty(pg_total_relation_size(schemaname||'.'||tablename)) AS
size
-- FROM pg_tables WHERE schemaname = 'public';

```

Soluções:

Índices Ausentes:

```

CREATE INDEX idx_message_queues_source_smsc ON
message_queues(source_smsc);
CREATE INDEX idx_message_queues_dest_smsc ON
message_queues(dest_smsc);
CREATE INDEX idx_message_queues_send_time ON
message_queues(send_time);
CREATE INDEX idx_message_queues_status ON message_queues(status);

```

Fragmentação da Tabela:

```
-- MySQL/MariaDB
OPTIMIZE TABLE message_queues;
OPTIMIZE TABLE frontend_registrations;

-- PostgreSQL
-- VACUUM ANALYZE message_queues;
-- VACUUM ANALYZE frontend_registrations;
```

Demais Dados:

Limpar registros antigos:

```
-- Excluir mensagens entregues com mais de 30 dias
DELETE FROM message_queues
WHERE status = 'delivered'
AND deliver_time < DATE_SUB(NOW(), INTERVAL 30 DAY)
LIMIT 10000;
```

Disco Cheio

Sintomas:

- Erro: "Disk full"
- Não é possível gravar no banco de dados
- Aplicação trava

Passos de Diagnóstico:

1. Verificar Uso do Disco:

```
df -h

# Verificar diretório do banco de dados SQL (MySQL/MariaDB)
du -sh /var/lib/mysql

# Verificar diretório do banco de dados SQL (PostgreSQL)
du -sh /var/lib/postgresql
```

2. Encontrar Arquivos Grandes:

```
# Encontrar maiores arquivos (MySQL/MariaDB)
find /var/lib/mysql -type f -exec du -h {} + | sort -rh |
head -20

# Encontrar maiores arquivos (PostgreSQL)
find /var/lib/postgresql -type f -exec du -h {} + | sort
-rh | head -20
```



```
# Verificar arquivos de log
du -sh /var/log/sms_c/*
```

Soluções:

Limpar Dados Antigos:

```
-- Excluir mensagens antigas
DELETE FROM message_queues
WHERE inserted_at < DATE_SUB(NOW(), INTERVAL 90 DAY)
LIMIT 100000;
```

Rotacionar Logs:

```
# Forçar logrotate
logrotate -f /etc/logrotate.d/sms_c

# Limpar arquivos de log antigos
find /var/log/sms_c -name "*.log.*" -mtime +30 -delete
```

Expandir Disco:

- Redimensionar volume (nuvem)
- Adicionar novo disco e estender volume
- Mover dados para disco maior

Problemas de Conexão no Frontend

Frontend Não Aparecendo como Ativo

Sintomas:

- Status do frontend mostra "expirado"
- Frontend não na lista ativa
- Mensagens não sendo entregues ao frontend

Passos de Diagnóstico:

1. Verificar Registro:

```
curl https://api.example.com:8443/api/frontends/active | grep
frontend_name
```

2. Verificar Logs do Frontend:

- O frontend está chamando /api/frontends/register?
- Algum erro na API?

- Frequência de registro (deve ser a cada 60s)

3. Verificar Logs da API:

```
grep "frontend.*register" /var/log/sms_c/application.log | tail -20
```

Soluções:

Frontend Não Registrando:

Testar registro manual:

```
curl -X POST https://api.example.com:8443/api/frontends/register \
-H "Content-Type: application/json" \
-d '{
  "frontend_name": "uk_gateway",
  "frontend_type": "smpp",
  "ip_address": "10.0.1.50"
}';
```

Se bem-sucedido, o problema está no código/configuração do frontend.

Registro Expirando:

Frontends expiram após 90 segundos. Garantir registro a cada 60 segundos:

```
# 0 frontend deve chamar o registro a cada 60 segundos
while True:
    register_with_smsc()
    time.sleep(60)
```

Problemas de Rede:

- Verificar firewall entre frontend e API
- Verificar resolução de DNS
- Testar com curl do servidor frontend

Frontend Conectando/Desconectando Repetidamente

Sintomas:

- Status do frontend alternando entre ativo/expirado
- Alta contagem de registros na história
- Conexão instável

Passos de Diagnóstico:

1. Verificar Saúde do Frontend:

- O processo do frontend está estável?
- Algum travamento ou reinício?
- Problemas de recursos (CPU/memória)?

2. Verificar Estabilidade da Rede:

```
# Verificar perda de pacotes
ping -c 100 api.example.com

# Verificar resets de conexão
netstat -s | grep -i reset
```

3. Verificar Tempo de Registro:

- Muito frequente? (a cada poucos segundos)
- Muito infrequente? (> 90 segundos)

Soluções:

Frontend Instável:

- Corrigir problemas da aplicação do frontend
- Aumentar recursos do frontend
- Verificar logs do frontend para erros

Problemas de Rede:

- Verificar conectividade intermitente
- Revisar logs do firewall
- Verificar verificações de saúde do balanceador de carga

Intervalo de Registro Errado:

Corrigir intervalo:

```
REGISTRATION_INTERVAL = 60 # segundos
```

Problemas de Cobrança/Faturamento

Falhas de Cobrança

Sintomas:

- sms_c_charging_failed_count aumentando
- Log de eventos mostra "charging_failed"
- Mensagens marcadas como charge_failed: true

Passos de Diagnóstico:

1. Verificar Conectividade com o OCS:

```
# Testar API do OCS
curl -X POST http://ocs.example.com:2080/jsonrpc \
  -H "Content-Type: application/json" \
  -d '{"method": "SessionSv1.Ping",
      "params": [],
      "id": 1
    }';
```

Esperado: {"result": "Pong"}

2. Verificar Logs do OCS:

```
tail -f /var/log/ocs/ocs.log
```

3. Verificar Configuração:

```
# Verificar URL do OCS
grep ocs_url config/runtime.exs
```

Soluções:

OCS Indisponível:

```
# Verificar status do OCS
systemctl status ocs

# Iniciar se necessário
systemctl start ocs
```

Erro de Configuração:

Atualizar configuração:

```
config :sms_c,
  ocs_url: "http://correct-host:2080/jsonrpc",
  ocs_tenant: "correct_tenant"
```

Desabilitar Cobrança Temporariamente:

```
config :sms_c,
  default_charging_enabled: false
```

Reiniciar aplicação.

Problemas de Conta:

- Verificar se a conta existe no OCS
- Verificar se a conta tem saldo
- Verificar se os planos de tarifação estão configurados

Cobrança Muito Lenta

Sintomas:

- sms_c_charging_succeeded_duration p95 > 500ms
- Processamento de mensagens lento quando a cobrança está habilitada
- Rápido quando a cobrança está desabilitada

Passos de Diagnóstico:

1. Verificar Latência de Cobrança:

```
histogram_quantile(0.95, sms_c_charging_succeeded_duration_bucket)
```

2. Verificar Desempenho do OCS:

```
# Tempo de resposta do OCS
curl -w "%#123;time_total}\n" -X POST
http://ocs.example.com:2080/jsonrpc \
-H "Content-Type: application/json" \
-d '{"method": "SessionSv1.Ping", "params": [], "id": 1}'
```

3. Verificar Latência da Rede:

```
# Pingar host do OCS
ping -c 10 ocs.example.com
```

Soluções:

OCS Lento:

- Otimizar configuração do OCS
- Adicionar recursos ao OCS
- Usar motor de tarifação mais rápido

Latência de Rede:

- Implantar OCS mais próximo do SMS-C
- Usar caminho de rede direto
- Evitar VPN/túneis se possível

Timeout Muito Baixo:

Aumentar timeout:

```
config :sms_c,  
  ocs_timeout: 5000 # 5 segundos
```

Problemas de Pesquisa ENUM

Pesquisas ENUM Falhando

Sintomas:

- sms_c_enum_lookup_stop_duration mostrando falhas
- Log de eventos mostra erros ENUM
- Rotas com enum_result_domain não correspondendo

Passos de Diagnóstico:

1. Verificar Configuração ENUM:

```
grep -A 10 "enum_" config/runtime.exs
```

2. Testar Conectividade DNS:

```
# Testar servidor DNS  
dig @8.8.8.8 e164.arpa
```

```
# Testar consulta ENUM  
# Para +15551234567:  
dig @8.8.8.8 NAPTR 7.6.5.4.3.2.1.5.5.5.1.e164.arpa
```

3. Verificar Servidor DNS:

```
# 0 servidor DNS personalizado é acessível?  
ping 10.0.1.53
```

```
# Testar porta  
nc -zv 10.0.1.53 53
```

Soluções:

Servidor DNS Inacessível:

Usar DNS alternativo:

```
config :sms_c,  
  enum_dns_servers: [  
    &#123;"8.8.8.8", 53&#125;, # Google Public DNS  
    &#123;"1.1.1.1", 53&#125;, # Cloudflare DNS  
  ]
```

Domínio ENUM Errado:

Atualizar domínio:

```
config :sms_c,  
  enum_domains: ["e164.arpa"] # Usar domínio padrão
```

Timeout Muito Curto:

Aumentar timeout:

```
config :sms_c,  
  enum_timeout: 10000 # 10 segundos
```

Desabilitar ENUM (se não necessário):

```
config :sms_c,  
  enum_enabled: false
```

Problemas de Cache ENUM

Sintomas:

- Baixa taxa de acerto de cache (< 70%)
- Tamanho do cache crescendo sem limites
- Uso de memória alto

Passos de Diagnóstico:

1. Verificar Estatísticas de Cache:

```
# Taxa de acerto de cache  
rate(sms_c_enum_cache_hit_count[5m]) /  
(rate(sms_c_enum_cache_hit_count[5m]) +  
rate(sms_c_enum_cache_miss_count[5m]))  
  
# Tamanho do cache  
sms_c_enum_cache_size_size
```

2. Verificar Padrão de Tráfego:

- Os números estão se repetindo?
- TTL do cache apropriado?

Soluções:

Baixa Taxa de Acerto (Esperado):

- Tráfego para números únicos (normal)

- Monitorar, mas não alarmar se $< 70\%$

Cache Crescendo:

Limpar cache via página de Teste NAPTR ou reiniciar aplicação.

Uso Alto de Memória:

- Esperado com cache grande
- Monitorar memória total do sistema
- Considerar ajuste de TTL

Problemas de Cluster

Nó Não Consegue Entrar no Cluster

Sintomas:

- Um único nó em execução
- Consultas de cluster retornando apenas resultados locais
- Erros de distribuição Erlang

Passos de Diagnóstico:

1. Verificar Nomes dos Nós:

```
# No console IEx
Node.self()
# Esperado: :sms@node1.example.com

Node.list()
# Esperado: Lista de outros nós
```

2. Verificar Cookie Erlang:

```
# Verificar arquivo de cookie
cat ~/.erlang.cookie

# Verificar se é o mesmo em todos os nós
```

3. Verificar Rede:

```
# Os nós conseguem se alcançar?
ping node2.example.com

# Verificar portas
nc -zv node2.example.com 4369
nc -zv node2.example.com 9100-9200
```


Soluções:

Cookie Desigual:

Definir o mesmo cookie em todos os nós:

```
export ERLANG_COOKIE=same_secret_value_here

# Ou atualizar ~/.erlang.cookie
echo "same_secret_value_here" > ~/.erlang.cookie
chmod 400 ~/.erlang.cookie
```

Firewall Bloqueando:

Abrir portas necessárias:

```
# EPMD
iptables -A INPUT -p tcp --dport 4369 -j ACCEPT

# Distribuição Erlang
iptables -A INPUT -p tcp --dport 9100:9200 -j ACCEPT
```

Problemas de DNS:

Usar endereços IP em vez de nomes de host:

```
config :sms_c,
  cluster_nodes: [
    : "sms@10.0.1.10",
    : "sms@10.0.1.11"
  ]
```

Split Brain do Cluster

Sintomas:

- Nós em execução, mas desconectados
- Dados diferentes em diferentes nós
- Inconsistências no Mnesia

Passos de Diagnóstico:

1. Verificar Conectividade do Nó:

```
# Em cada nó (IEx)
Node.list()
```

2. Verificar Mnesia:

```
:mnesia.system_info(:running_db_nodes)
```

Soluções:

Reconectar Nós:

```
# Parar todos os nós
systemctl stop sms_c

# Iniciar um nó primeiro
systemctl start sms_c # No node1

# Aguardar até que inicie completamente, então iniciar outros
systemctl start sms_c # No node2
systemctl start sms_c # No node3
```

Inconsistência do Mnesia:

- Exportar rotas do nó correto
- Parar todos os nós
- Deletar diretório do Mnesia
- Iniciar nós
- Importar rotas

Problemas de API

API Não Respondendo

Sintomas:

- Timeout de conexão
- Conexão recusada
- Sem resposta

Passos de Diagnóstico:

1. Verificar Processo da API:

```
# A aplicação está em execução?
systemctl status sms_c

# Verificar portas em escuta
netstat -tlnp | grep 8443
```

2. Verificar Firewall:

```
# Verificar iptables
iptables -L -n | grep 8443
```

```
# Testar conectividade local
curl -k https://localhost:8443/api/status
```

3. Verificar Configuração TLS:

```
# Verificar se o certificado existe
ls -l priv/cert/server.crt priv/cert/server.key

# Verificar validade do certificado
openssl x509 -in priv/cert/server.crt -noout -dates
```

Soluções:

Aplicação Não Executando:

```
systemctl start sms_c
```

Firewall Bloqueando:

```
# Permitir porta da API
iptables -A INPUT -p tcp --dport 8443 -j ACCEPT
```

Problemas de Certificado:

Gerar novo certificado (ver Guia de Configuração).

Porta Errada:

Verificar configuração:

```
grep "port:" config/runtime.exs
```

API Retornando Erros 500

Sintomas:

- Erro Interno do Servidor
- Código de status 500
- Erro nos logs

Passos de Diagnóstico:

1. Verificar Logs da Aplicação:

```
tail -100 /var/log/sms_c/application.log | grep "\[error\]"
```

2. Verificar Banco de Dados:

```
mysql -u sms_user -p -e "SELECT 1"
```

3. Verificar Recursos:

```
# Memória  
free -h  
  
# CPU  
top -b -n 1  
  
# Disco  
df -h
```

Soluções:

Banco de Dados Indisponível:

- Iniciar banco de dados
- Corrigir configuração de conexão

Fora de Memória:

- Reiniciar aplicação
- Aumentar memória do sistema
- Verificar vazamentos de memória

Erro na Aplicação:

- Verificar erro específico nos logs
- Corrigir problema de configuração
- Reiniciar aplicação

Problemas na Interface Web

Não É Possível Acessar a Interface Web

Sintomas:

- Timeout de conexão
- 404 Não Encontrado
- Página não carrega

Passos de Diagnóstico:

1. Verificar Status da Aplicação:

```
systemctl status sms_c
```

2. Verificar Porta:

```
netstat -tlnp | grep 80
```

3. Verificar URL:

- Nome do host correto?
- Porta correta?
- HTTP vs HTTPS?

Soluções:

Porta Errada:

Verificar configuração:

```
grep "control_panel" config/runtime.exs
```

Acessar na porta correta (padrão: 80 ou 4000).

Aplicação Não Executando:

```
systemctl start sms_c
```

Firewall:

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

LiveView Não Atualizando

Sintomas:

- Página carrega, mas não atualiza
- Dados estão desatualizados
- Erros de WebSocket no console do navegador

Passos de Diagnóstico:

1. Verificar Console do Navegador:

- Abrir Ferramentas de Desenvolvedor (F12)
- Procurar por erros de WebSocket
- Verificar aba de rede para requisições falhadas

2. Verificar Configuração do Proxy:

Se estiver usando proxy reverso, garantir suporte a WebSocket:

```
location /live {
```

```
proxy_http_version 1.1;  
proxy_set_header Upgrade $http_upgrade;  
proxy_set_header Connection "upgrade";  
&#125;
```

Soluções:

WebSocket Bloqueado:

- Configurar proxy para WebSocket
- Verificar firewall
- Verificar extensões do navegador

Atualizar Página:

- Atualização forçada (Ctrl+F5)
- Limpar cache do navegador

Problemas de Recursos do Sistema

Alta Utilização de CPU

Sintomas:

- CPU consistentemente > 80%
- Sistema lento
- Aplicação não responsiva

Passos de Diagnóstico:

1. Verificar Processo:

```
top -b -n 1 | grep beam.smp
```

2. Verificar Métricas:

```
# Taxa de processamento de mensagens  
rate(sms_c_message_received_count[5m])  
  
# Operações de roteamento  
rate(sms_c_routing_route_matched_count[5m])
```

Soluções:

Tráfego Alto:

- Escalar horizontalmente (adicionar nós)
- Escalar verticalmente (adicionar CPU)

Roteamento Ineficiente:

- Reduzir número de rotas
- Otimizar critérios de rota

Muitas Pesquisas ENUM:

- Verificar taxa de acerto de cache
- Considerar desabilitar se não necessário

Alto Uso de Memória

Sintomas:

- Uso de memória > 90%
- Aplicação trava
- Erros de falta de memória

Passos de Diagnóstico:

1. Verificar Memória:

```
free -h
```

```
ps aux | grep beam.smp
```

2. Verificar Tamanhos de Cache:

```
sms_c_enum_cache_size_size
```

Soluções:

Cache ENUM Muito Grande:

- Limpar cache
- Reduzir TTL
- Desabilitar ENUM se não necessário

Fila de Lote Crescendo:

```
# Verificar estatísticas do trabalhador (IEx)  
SmsC.Messaging.BatchInsertWorker.stats()
```

Se a fila estiver grande, esvaziar manualmente ou reiniciar.

Adicionar Memória:

- Escalar verticalmente
- Adicionar swap (temporário)

Vazamento de Memória:

- Reiniciar aplicação
 - Reportar problema para investigação
-

Para assistência adicional, consulte:

- [Guia de Operações](#) - Procedimentos diários
- [Guia de Configuração](#) - Opções de configuração
- [Guia de Métricas](#) - Configuração de monitoramento
- Logs da aplicação - `/var/log/sms_c/application.log`



Documentação de Conformidade de Interceptação ANSSI R226

Objetivo do Documento: Este documento fornece especificações técnicas necessárias para a autorização ANSSI R226 sob os Artigos R226-3 e R226-7 do Código Penal Francês para o Centro de Serviço de SMS OmniMessage (SMSc).

Classificação: Documentação de Conformidade Regulatória

Autoridade Alvo: Agence nationale de la sécurité des systèmes d'information (ANSSI)

Regulamentação: R226 - Proteção da Privacidade da Correspondência e Interceptação Legal

1. ESPECIFICAÇÕES TÉCNICAS DETALHADAS

1.1 Ficha Técnica Comercial

Nome do Produto: OmniMessage SMSc (Centro de Serviço de SMS)

Tipo de Produto: Centro de Mensagens de Telecomunicações

Função Principal: Roteamento, armazenamento e entrega de mensagens SMS

Protocolos de Rede: REST API (HTTPS), protocolos SMS (SMPP, IMS, SS7/MAP via frontends externos)

Modelo de Implantação: Aplicação de servidor local

Stack Tecnológico: Elixir/Erlang, Phoenix Framework, Mnesia, MySQL/PostgreSQL

Capacidades Principais

Processamento de Mensagens:

- Fila de mensagens SMS centralizada com REST API
- Design agnóstico a protocolos suportando frontends SMPP, IMS, SS7/MAP
- Motor de roteamento dinâmico com roteamento baseado em prefixo
- Lógica de repetição com retrocesso exponencial
- Expiração de mensagens e tratamento de filas de mensagens mortas
- Geração e arquivamento de Registros de Detalhes de Chamadas (CDR)
- Desempenho: taxa de inserção de ~1.750 mensagens/segundo, capacidade de 150 milhões de mensagens/dia

Armazenamento de Mensagens:

- **Fila de Mensagens Ativas:** Banco de dados em memória Mnesia com persistência em disco opcional
 - Armazenamento primário: RAM para acesso ultra-rápido (latência sub-milissegundo)
 - Backup em disco: modo `disc_copies` grava em disco para recuperação de falhas
 - Recuperação automática: Mensagens sobrevivem a reinicializações do sistema
 - Retenção: Configurável (padrão 24 horas), após isso limpeza automática
- **Arquivo CDR de Longo Prazo:** Banco de dados MySQL/PostgreSQL (separado da fila de mensagens)
 - CDRs gravados quando mensagens são entregues, expiradas, falhadas ou rejeitadas
 - Banco de dados SQL usado SOMENTE para exportação/arquivamento de CDR, NÃO para operações de mensagens ativas
 - Sem impacto de desempenho no roteamento de mensagens (gravação assíncrona)
- **Benefícios da Arquitetura de Dois Níveis:**
 - Fila ativa: Extremamente rápida (1.750 msg/sec) sem gargalo SQL
 - Arquivo CDR: Retenção de longo prazo (meses/anos) para faturamento e interceptação legal
 - Separação limpa: Operações de mensagens nunca tocam SQL
- Suporte a cluster para alta disponibilidade (replicação Mnesia entre nós)

Interfaces de Rede:

- **REST API:** HTTPS (porta 8443) para comunicação de frontend externo
- **Painel de Controle:** HTTPS (porta 8086) para gerenciamento baseado na web
- **Protocolos de Frontend:** SMPP, IMS, SS7/MAP (via aplicações de gateway externas)
- **Banco de Dados:** MySQL/PostgreSQL para armazenamento de CDR

Roteamento e Processamento:

- Roteamento SMS dinâmico com atualizações de configuração em tempo de execução
- Correspondência baseada em prefixo (números chamadores/chamados)
- Filtragem de SMSC de origem e tipo
- Balanceamento de carga baseado em prioridade e peso
- Tradução e normalização de números
- Suporte a pesquisa DNS ENUM (E.164 Number Mapping)
- Capacidade de resposta automática e descarte de mensagens
- Controle de cobrança por rota (integração CGRates)

◈ **Arquitetura e recursos completos documentados em [README.md](#)**

1.2 Capacidades de Intercepção

1.2.1 Aquisição de Mensagens

Captura de Mensagens SMS:

- O OmniMessage SMSc processa todas as mensagens SMS entre assinantes e redes externas
- Acesso completo aos metadados e conteúdo da mensagem, incluindo:
 - MSISDN de origem (número móvel)
 - MSISDN de destino (número móvel)
 - IMSI de origem (Identidade Internacional do Assinante Móvel)
 - IMSI de destino
 - Corpo da mensagem (conteúdo de texto)
 - Dados brutos de PDU (Unidade de Dados de Protocolo)
 - Informações TP-DCS (Esquema de Codificação de Dados)
 - Codificação da mensagem (GSM7, UCS-2, 8-bit, Latin-1)
 - Indicadores de mensagem multipart e dados de remontagem
 - Informações do Cabeçalho de Dados do Usuário (UDH)

Aquisição de Metadados de Mensagens:

- Registros de Detalhes de Chamadas (CDR) completos armazenados no banco de dados com:
 - ID da mensagem (identificador único)
 - Número chamador (MSISDN de origem)
 - Número chamado (MSISDN de destino)
 - Timestamp de submissão (quando a mensagem entrou no sistema)
 - Timestamp de entrega (quando a mensagem foi entregue)
 - Timestamp de expiração (quando a mensagem expirou se não entregável)
 - Status (entregue, expirado, falhado, rejeitado)
 - Contagem de tentativas de entrega
 - Partes da mensagem (para SMS concatenados/multiparte)
 - Identificador SMSC de origem
 - Identificador SMSC de destino
 - Nó de origem (nome do nó do cluster Erlang)
 - Nó de destino (para implantações distribuídas)
 - Flag de carta morta (indicador de exaustão de tentativas)

◆ **Esquema completo de CDR documentado em [CDR_SCHEMA.md](#)**

Acesso à Fila de Mensagens:

- Monitoramento em tempo real da fila de mensagens
- Endpoints da REST API para recuperação de mensagens
- Consultas de banco de dados para pesquisa histórica de mensagens
- Capacidades de filtragem por:

- Número de telefone (origem/destino)
- Gateway SMSC
- Intervalo de tempo
- Status da mensagem
- Tentativas de entrega

◆ **Documentação completa da API em [API_REFERENCE.md](#)**

1.2.2 Capacidades de Processamento de Dados

Arquitetura de Armazenamento de Mensagens (Sistema de Dois Níveis):

O SMSc utiliza uma sofisticada arquitetura de armazenamento de dois níveis que separa o processamento operacional de mensagens do arquivamento de longo prazo:

Nível 1: Fila de Mensagens Ativas (Mnesia)

- **Propósito:** Operações de roteamento e entrega de mensagens em tempo real
- **Tecnologia:** Banco de dados distribuído Erlang Mnesia
- **Modo de Armazenamento:** Em memória com backup `disc_copies`
 - Armazenamento primário em RAM para máxima velocidade
 - Sincronização automática em disco para recuperação de falhas
 - Mensagens persistem entre reinicializações do sistema
- **Desempenho:** Operações de leitura/gravação sub-milissegundo
- **Retenção:** Curto prazo (padrão 24 horas), configurável
- **Limpeza:** Arquivamento automático para o banco de dados CDR, depois exclusão do Mnesia
- **Operações:** Todas as operações da fila de mensagens (inserir, atualizar, status de entrega, roteamento)
- **Recurso Crítico:** Banco de dados SQL NUNCA é consultado durante roteamento/entrega de mensagens

Nível 2: Arquivo CDR (MySQL/PostgreSQL)

- **Propósito:** Armazenamento de longo prazo para faturamento, análises e interceptação legal
- **Tecnologia:** Banco de dados SQL tradicional (MySQL ou PostgreSQL)
- **Gatilho de Gravação:** CDRs gravados SOMENTE quando mensagens atingem estado final:
 - Mensagem entregue com sucesso
 - Mensagem expirou (excedeu o período de validade)
 - Mensagem falhou permanentemente
 - Mensagem rejeitada por regras de roteamento
- **Modo de Gravação:** Gravação em lote assíncrona (sem impacto no desempenho de roteamento de mensagens)
- **Retenção:** Longo prazo (meses a anos), configurável conforme requisitos

- regulatórios
- **Operações:** Consultas históricas, relatórios, conformidade, interceptação legal
- **Acesso:** Consultas SQL, REST API (futuro), exportação CSV/JSON

Principais Benefícios Arquitetônicos:

1. **Desempenho:** Operações de roteamento ativas nunca tocam SQL (sem gargalo de banco de dados)
2. **Escalabilidade:** Mnesia lida com 1.750+ mensagens/segundo sem sobrecarga SQL
3. **Confiabilidade:** Modo disc_copies garante que não haja perda de mensagens em caso de falha
4. **Conformidade:** Banco de dados CDR fornece trilha de auditoria permanente
5. **Separação de Preocupações:** Dados operacionais vs. dados arquivados claramente separados

Ciclo de Vida da Mensagem:

1. Mensagem submetida → Armazenada no Mnesia (RAM + backup em disco)
2. Mensagem roteada → Consulta Mnesia (ultra-rápida)
3. Mensagem entregue/expirada → CDR gravado no SQL (assíncrono)
4. Após 24h → Mensagem excluída do Mnesia (trabalhador de limpeza)
5. CDR permanece no SQL → Disponível para consultas de interceptação legal (anos)

Retenção e Recuperação de Dados:

- Retenção ou exclusão configurável do corpo da mensagem para privacidade
- Preservação de dados binários (armazenamento de PDU bruta em Mnesia e CDR)
- Capacidade de busca de texto completo (se habilitada no banco de dados CDR)
- Campos de CDR indexados para consultas rápidas de interceptação legal

Rastreamento de Frontend:

- Rastreamento em tempo real de frontends SMSC externos (SMPP, IMS, gateways MAP)
- Registro de frontend com monitoramento de heartbeat
- Rastreamento de status de saúde (ativo/expirado)
- Histórico de tempo de atividade/inatividade
- Rastreamento de endereço IP e nome do host
- Registro de configuração específico do frontend

1.2.3 Capacidades de Análise

Monitoramento em Tempo Real:

- Painel da interface web mostrando:
 - Fila de mensagens ativas
 - Submissão e entrega de mensagens
 - Decisões de roteamento e seleção de gateway
 - Status do gateway de frontend
 - Utilização de recursos do sistema
- Integração de métricas Prometheus para monitoramento operacional
- Métricas de desempenho (throughput, latência, taxas de sucesso)

❖ **Guia completo de monitoramento em [OPERATIONS_GUIDE.md](#)**

❖ **Documentação de métricas em [METRICS.md](#)**

Análise Histórica:

- Banco de dados CDR consultável por:
 - Intervalo de tempo
 - Número da parte chamadora/chamada
 - Status da mensagem
 - Gateway SMSC
 - Tentativas de entrega
 - Conteúdo da mensagem (busca de texto completo se habilitada)
- Capacidades de análise estatística:
 - Volume de mensagens por hora/dia/mês
 - Taxas de sucesso/falha por rota
 - Tempos médios de entrega
 - Análise de mensagens multiparte
 - Padrões de entrega falhada

Rastreamento de Assinantes:

- Histórico de mensagens por número de telefone (MSISDN)
- Rastreamento baseado em IMSI (quando disponível de frontends IMS/MAP)
- Análise de padrões de chamadas
- Correlação de partes de comunicação
- Análise temporal (frequência de mensagens, padrões de tempo)

Análise de Rede:

- Métricas de desempenho de rota
- Disponibilidade e saúde do gateway
- Visualização do fluxo de mensagens
- Distribuição de nós de cluster (implantações de múltiplos nós)
- Análise de tentativas de entrega
- Análise de padrões de repetição

Inteligência de Números:

- Normalização de números E.164
- Identificação de país/região a partir do prefixo do número
- Regras de tradução e reescrita de números
- Pesquisa DNS ENUM para inteligência de roteamento
- Decisões de roteamento baseadas em prefixo

◇ **Guia de tradução de números em [number_translation_guide.md](#)**

◇ **Guia de roteamento em [sms_routing_guide.md](#)**

1.3 Capacidades de Contramedidas

1.3.1 Mecanismos de Proteção da Privacidade

Confidencialidade da Comunicação:

- HTTPS/TLS para comunicações da REST API
- Autenticação baseada em certificado
- Criptografia de conexão de banco de dados (suporte a TLS)
- Exclusão configurável do corpo da mensagem após a entrega

Controle de Acesso:

- Controle de acesso à interface web
- Mecanismos de autenticação da API
- Controles de acesso ao banco de dados
- Autenticação de registro de frontend

Registro de Auditoria:

- Registro completo de eventos do sistema
- Registro de submissão/entrega de mensagens
- Rastreamento de alterações de configuração
- Registro de ações administrativas
- Registro estruturado com níveis configuráveis

1.3.2 Recursos de Proteção de Dados

Privacidade da Mensagem:

- Exclusão configurável do corpo da mensagem após a entrega
- Corpo da mensagem excluído da exibição da interface (opcional)
- Corpo da mensagem excluído das exportações (opcional)
- Campo do corpo da mensagem CDR pode ser definido como NULL para privacidade

Segurança do Banco de Dados:

- Suporte à criptografia de tabela MySQL (ENCRYPTION='Y')
- Suporte à criptografia de dados transparente do PostgreSQL
- Separação de funções de acesso ao banco de dados
- Contas de usuário somente leitura para análises
- Acesso restrito ao conteúdo da mensagem

Fortalecimento do Sistema:

- Portas de rede expostas mínimas
- Gerenciamento de certificados TLS
- Armazenamento seguro de configurações
- Separação de configurações baseadas em ambiente
- Segurança de cluster com protocolo de distribuição Erlang

1.4 Arquitetura de Armazenamento: Design de Dois Níveis Mnesia + SQL

Visão Geral

O OmniMessage SMSc emprega uma arquitetura de armazenamento de dois níveis única, especificamente projetada para separar o processamento operacional de mensagens de armazenamento de conformidade e arquivamento de longo prazo.

Nível 1: Fila de Mensagens em Memória Mnesia

O que é Mnesia?

- Banco de dados distribuído embutido no tempo de execução Erlang/OTP
- Armazenamento híbrido: Primário em memória com backup automático em disco
- Transações compatíveis com ACID
- Replicação de cluster entre vários nós

Modo de Armazenamento: disc_copies

- **Primário em Memória:** Todas as mensagens ativas armazenadas em RAM
 - Operações de leitura/gravação ultrarrápidas (sub-milissegundo)
 - Sem I/O em disco durante operações normais de roteamento de mensagens
 - Permite throughput de 1.750+ mensagens/segundo
- **Backup em Disco (Automático):** Mnesia sincroniza RAM com disco
 - Gravações ocorrem assíncronas em segundo plano
 - Cópia em disco atualizada em cada commit de transação
 - Recuperação de falhas: O sistema reinicia com todas as mensagens intactas
 - Localização: diretório Mnesia.* / nos dados da aplicação

Ciclo de Vida da Mensagem em Mnesia:

1. Mensagem chega via REST API → Inserida na RAM do Mnesia + backup em disco
2. Motor de roteamento consulta Mnesia → Resposta instantânea (acesso à memória)
3. Gateway externo consulta mensagens → Consulta Mnesia (acesso à memória)
4. Gateway atualiza status de entrega → Atualização Mnesia (memória + disco)
5. Após entrega/expiração → Mensagem marcada para limpeza
6. Trabalhador de limpeza (padrão 24h) → Mensagem excluída do Mnesia

Recurso de Desempenho Crítico:

- **ZERO consultas ao banco de dados SQL** durante roteamento/entrega ativa de mensagens
- SQL é completamente ignorado para processamento operacional de mensagens
- Isso elimina o tradicional gargalo do SMS-C (I/O de banco de dados)

Nível 2: Banco de Dados SQL para Exportação/Arquivamento de CDR

O que é CDR (Registro de Detalhes de Chamadas)?

- Registro de auditoria permanente de metadados e conteúdo da mensagem
- Gravado no banco de dados MySQL ou PostgreSQL
- Usado para faturamento, análises, conformidade e interceptação legal

Quando os CDRs são Gravados: Registros de CDR são criados SOMENTE quando mensagens atingem um estado final:

- ◇ Mensagem entregue com sucesso
- ◇ Mensagem expirou (excedeu o período de validade sem entrega)
- ◇ Mensagem falhou permanentemente (número inválido, erro de roteamento)
- ◇ Mensagem rejeitada (regras de roteamento, falha de validação)

Como os CDRs são Gravados:

- **Gravação em lote assíncrona:** CDRs gravados em processo de trabalhador em segundo plano
- **Sem bloqueio:** O roteamento de mensagens nunca espera pela gravação SQL
- **Inserções em lote:** Múltiplos CDRs agrupados (padrão 100) e gravados juntos
- **Intervalo de flush:** 100ms padrão (configurável)
- **Tratamento de erros:** Gravações de CDR com falha registradas,

processamento de mensagens continua

```
# Configuração em config/runtime.exs
config :sms_c,
  batch_insert_batch_size: 100,          # Tamanho do lote para
gravações de CDR
  batch_insert_flush_interval_ms: 100    # Intervalo de flush
```

Propósito do Banco de Dados SQL:

- ✧ NÃO usado para: Operações ativas da fila de mensagens
- ✧ NÃO usado para: Decisões de roteamento de mensagens
- ✧ NÃO usado para: Entrega de mensagens em tempo real
- ✧ SOMENTE usado para: Arquivamento de longo prazo de CDR e consultas históricas
- ✧ SOMENTE usado para: Consultas de interceptação legal (meses/anos de histórico)
- ✧ SOMENTE usado para: Relatórios de faturamento e análises

Diagrama de Arquitetura

Legenda:

- Linhas sólidas: Operações síncronas (em tempo real)
- Linhas tracejadas: Operações assíncronas (em segundo plano)
- Verde: Nível de alto desempenho (em memória)
- Azul: Nível de arquivamento (SQL persistente)

Implicações da Interceptação Legal

Mensagens Recentes (< 24 horas):

- Acessíveis via Mnesia (consultas da REST API)
- Recuperação ultra-rápida
- Conteúdo completo da mensagem disponível
- Monitoramento em tempo real possível

Mensagens Históricas (> 24 horas):

- Acessíveis via banco de dados SQL (tabela CDR)
- Desempenho padrão de consulta SQL
- Metadados completos da mensagem sempre disponíveis
- Corpo da mensagem disponível (a menos que o modo de privacidade esteja habilitado)

Benefícios de Conformidade:

1. **Sem perda de dados:** O modo `disc_copies` garante que mensagens

- sobrevivam a falhas
2. **Trilha de auditoria permanente:** CDRs retidos por anos no banco de dados SQL
 3. **Desempenho:** Consultas de interceptação legal não impactam o roteamento de mensagens
 4. **Flexibilidade:** Mensagens recentes (Mnesia) + mensagens históricas (SQL) ambas acessíveis

1.5 Arquitetura de Integração de Frontend Multi-Protocolo

O OmniMessage SMSc emprega um design central agnóstico a protocolos que se conecta a gateways externos específicos de protocolo (frontends) via uma REST API unificada. Esta arquitetura permite que a interceptação legal capture mensagens independentemente de qual protocolo de telecomunicações foi usado para enviá-las ou recebê-las.

Visão Geral da Arquitetura

Detalhes da Integração de Protocolo de Frontend

1. Integração de Frontend IMS/SIP

Redes IMS usam o protocolo SIP para mensagens SMS sobre IP. O gateway IMS traduz entre SIP e a REST API do SMSc.

Dados de Interceptação Específicos do IMS:

- IMSI de origem/destino (do registro IMS)
- Cabeçalhos SIP P-Asserted-Identity
- SIP Call-ID para correlação
- Localização da rede IMS (P-Access-Network-Info)
- Perfis de assinante do HSS IMS

2. Integração de Frontend SMPP

SMPP é o protocolo padrão da indústria para agregadores e provedores de SMS. O gateway SMPP traduz mensagens baseadas em PDU para chamadas da REST API.

Dados de Interceptação Específicos do SMPP:

- PDU SMPP completo (formato binário preservado)
- Detalhes do Esquema de Codificação de Dados (DCS)
- Cabeçalho de Dados do Usuário (UDH) para mensagens concatenadas
- Identificação do sistema ESME (identificação do cliente)
- Informações do plano de numeração TON/NPI
- Flags de entrega registrada

3. Integração de Frontend SS7/MAP

Redes legadas de comutação de circuito usam o protocolo SS7 MAP para SMS. O gateway MAP traduz entre sinalização SS7 e REST API.

Dados de Interceptação Específicos do SS7/MAP:

- IMSI de mensagens MAP
- Endereços de Título Global (GT)
- Endereço MSC/VLR (identificação do elemento de rede)
- Endereços de partes chamadoras/chamadas SCCP
- Códigos de operação MAP
- Formato binário TP-User-Data

Interceptação Unificada Através de Todos os Protocolos

Benefício Chave para Interceptação Legal: Independentemente de qual protocolo foi usado (IMS/SIP, SMPP ou SS7/MAP), todas as mensagens convergem no núcleo do SMSsc com estrutura de dados normalizada, permitindo:

1. **Monitoramento Agnóstico a Protocolos:** Ponto de interceptação único captura todos os tipos de mensagens
2. **Formato CDR Unificado:** Todos os protocolos gravam no mesmo esquema de CDR
3. **Correlação Entre Protocolos:** Rastrear mensagens através de fronteiras de protocolo
4. **Preservação Completa de Metadados:** Campos específicos de protocolo preservados no CDR

Resumo do Fluxo de Dados:

Identificação de Protocolo no CDR:

- O campo `source_smsc` indica o protocolo do frontend (por exemplo, "ims.gateway-01", "smpp.customer123", "map.msc-01")
- Permite filtragem e análise por tipo de protocolo
- Consultas de interceptação legal podem direcionar protocolos específicos ou todos os protocolos

1.6 Arquitetura Técnica para Interceptação Legal

Pontos de Integração de Interceptação Legal

A arquitetura de armazenamento de dois níveis fornece múltiplos pontos de acesso para interceptação legal, otimizados tanto para monitoramento em tempo real (Mnesia) quanto para análise histórica (SQL).

1. Acesso à REST API para Mensagens Recentes (Mnesia):

Acesso a mensagens ativas na fila Mnesia (tipicamente últimas 24 horas):

Endpoints da API para Intercepção em Tempo Real:

- GET /api/messages - Listar mensagens ativas com filtragem
- GET /api/messages/{id} - Obter detalhes de mensagem específica (do Mnesia)
- GET /api/messages/get_by_smsc?smsc=X - Obter mensagens por gateway
- Todas as consultas atingem Mnesia (em memória) para resposta instantânea

Nota: Esses endpoints consultam a fila de mensagens ativa do Mnesia, fornecendo acesso a mensagens que estão sendo processadas ou foram recentemente entregues (dentro do período de retenção).

Parâmetros de Consulta:

- Filtrar por MSISDN de origem/destino
- Filtrar por intervalo de tempo
- Filtrar por gateway SMSC
- Filtrar por status da mensagem
- Suporte a ordenação e paginação

2. Acesso Direto ao Banco de Dados CDR para Mensagens Históricas (SQL):

Acesso a mensagens arquivadas no banco de dados SQL (todas as mensagens entregues/expiradas/falhadas):

Acesso SQL Direto:

- Credenciais de banco de dados somente leitura para sistemas autorizados
- Acesso a consultas SQL na tabela cdrs (trilha de auditoria permanente)
- **Método de Acesso:** Cliente SQL padrão (mysql, psql, DBeaver, etc.)
- **Fonte de Dados:** Somente mensagens arquivadas (não fila ativa)
- Campos indexados para pesquisa eficiente:
 - calling_number (indexado) - Número de telefone de origem
 - called_number (indexado) - Número de telefone de destino
 - message_id (indexado) - Identificador único da mensagem
 - submission_time (indexado) - Quando a mensagem entrou no sistema
 - status (indexado) - Status final de entrega
 - dest_smsc (indexado) - Gateway usado para entrega

Nota: O banco de dados CDR contém registros permanentes de todas as mensagens processadas. Esta é a principal fonte de dados para consultas históricas de intercepção legal (meses/anos de dados).

3. Feed de Mensagens em Tempo Real (PubSub):

- Integração Phoenix PubSub para eventos em tempo real
- Notificações de submissão de mensagens
- Notificações de entrega de mensagens
- Eventos de mudança de status de mensagem
- Filtragem de eventos configurável por critérios
- Suporte a WebSocket para monitoramento ao vivo

4. Interface de Exportação em Lote:

- Exportação CSV de registros CDR
- Exportação JSON para acesso programático
- Campos de exportação configuráveis
- Exportações baseadas em intervalo de tempo
- Exportações cientes da privacidade (opcional exclusão do corpo da mensagem)

Interfaces Padrão de Interceptação Legal da ETSI

O OmniMessage SMSc fornece a base para implementar interfaces de interceptação legal compatíveis com a ETSI. Embora o núcleo do SMSc não implemente nativamente as interfaces X1/X2/X3, ele fornece todos os pontos de acesso de dados necessários que podem ser integrados com sistemas externos de Função de Mediação de Interceptação Legal (LIMF).

Interfaces Padrão de LI da ETSI:

Descrições das Interfaces:

Interface X1 - Função Administrativa:

- **Propósito:** Provisionamento de mandados e alvos de interceptação da aplicação de justiça para o sistema de interceptação
- **Direção:** LEMF → LIMF (bidirecional)
- **Funções:**
 - Ativar/desativar interceptação para alvos específicos (MSISDNs, IMSIs)
 - Definir duração e período de validade da interceptação
 - Configurar critérios de filtragem (números de telefone, janelas de tempo)
 - Recuperar status de interceptação
- **Integração com SMSc:**
 - LIMF mantém lista de alvos (banco de dados de mandados)
 - LIMF consulta CDR/API do SMSc para mensagens correspondentes
 - LIMF filtra com base nos critérios provisionados pelo X1

Interface X2 - Entrega de IRI (Informação Relacionada à Interceptação):

- **Propósito:** Entregar metadados de mensagens à aplicação de justiça

- **Direção:** LIMF → LEMF (unidirecional)
- **Formato de Dados:** XML/ASN.1 compatível com ETSI TS 102 232-x
- **Conteúdo do CDR do SMSc:**
 - ID da mensagem
 - Número chamador (MSISDN de origem)
 - Número chamado (MSISDN de destino)
 - IMSI (origem e destino, se disponível)
 - Timestamp de submissão
 - Timestamp de entrega
 - Status da mensagem (entregue/falhado/expirado)
 - Tentativas de entrega
 - Informações do gateway SMSC (origem/destino)
 - Localização da rede (se disponível)
- **Integração com SMSc:**
 - LIMF consulta o banco de dados CDR para números de telefone alvo
 - LIMF transforma registros CDR no formato IRI da ETSI
 - LIMF entrega IRI à LEMF via X2

Interface X3 - Entrega de CC (Conteúdo da Comunicação):

- **Propósito:** Entregar o conteúdo real da mensagem à aplicação de justiça
- **Direção:** LIMF → LEMF (unidirecional)
- **Formato de Dados:** Compatível com ETSI TS 102 232-x
- **Conteúdo do SMSc:**
 - Corpo da mensagem (conteúdo de texto)
 - Dados brutos de PDU (dados binários de SMS)
 - Informações de codificação de caracteres
 - Segmentos de mensagem multiparte
 - Informações TP-DCS
 - Cabeçalho de Dados do Usuário (UDH)
- **Integração com SMSc:**
 - LIMF recupera o conteúdo da mensagem do campo message_body do CDR
 - LIMF recupera dados brutos de PDU se disponíveis
 - LIMF empacota o conteúdo no formato ETSI CC
 - LIMF entrega CC à LEMF via X3

Arquitetura de Implementação:

Mapeamento de Dados do SMSc para Interfaces de LI:

Campo de Dados do SMSc	X2 (IRI)	X3 (CC)	Coluna da Tabela CDR
ID da Mensagem	◇ ID de correlação	◇ Referência	message_id
Número Chamador	◇ Parte A	-	calling_number
Número Chamado	◇ Parte B	-	called_number
Timestamp de Submissão	◇ Timestamp	-	submission_time

Campo de Dados do SMSc	X2 (IRI)	X3 (CC)	Coluna da Tabela CDR
Timestamp de Entrega	❖ Conclusão	-	delivery_time
Status	❖ Resultado	-	status
Corpo da Mensagem	-	❖ Conteúdo	message_body
PDU Bruto	-	❖ Binário	(Mnesia/CDR)
SMSC de Origem	❖ Elemento de rede	-	source_smsc
SMSC de Destino	❖ Elemento de rede	-	dest_smsc
IMSI	❖ ID do Assinante	-	(Via frontends)

Opções de Integração LIMF:

Opção 1: Arquitetura de Polling

- LIMF consulta periodicamente o banco de dados CDR (a cada 1-60 segundos)
- Consultas SQL filtram por números de telefone alvo da lista de mandados do X1
- Baixa complexidade, fácil de implementar
- Pequeno atraso entre entrega de mensagem e entrega de LI

Opção 2: Arquitetura de Feed em Tempo Real

- O SMSc PubSub publica eventos de mensagens
- LIMF se inscreve no fluxo de mensagens em tempo real
- LIMF filtra com base na lista de alvos
- Latência quase zero para interceptação legal
- Requer desenvolvimento de integração personalizada

Opção 3: Arquitetura Híbrida

- Mensagens recentes: Feed em tempo real PubSub (< 24 horas)
- Mensagens históricas: Polling do banco de dados CDR
- Equilíbrio ideal de latência e confiabilidade

Mecanismos de Gatilho de Interceptação

Interceptação Baseada em Alvo:

- Correspondência de número de telefone (MSISDN)
- Alvo baseado em IMSI (quando disponível)
- Listas de vigilância configuráveis
- Vistas de banco de dados para isolamento de alvos
- Filtragem da API por identificadores de alvos

Interceptação Baseada em Evento:

- Todas as mensagens para/de números específicos
- Mensagens via gateways SMSC específicos
- Mensagens com características específicas (multiparte, entrega falhada, etc.)
- Roteamento geográfico (via ENUM ou correspondência de prefixo)

Interceptação Baseada em Tempo:

- Filtragem de intervalo de data/hora em consultas CDR
- Aplicação de políticas de retenção
- Arquivamento automático de mensagens antigas
- Políticas de retenção de dados configuráveis

Exemplos de Consultas SQL para Interceptação Legal:

```
-- Obter todas as mensagens para número alvo
SELECT * FROM cdrs
WHERE calling_number = '+33612345678'
      OR called_number = '+33612345678'
ORDER BY submission_time DESC;

-- Obter mensagens em janela de tempo específica
SELECT * FROM cdrs
WHERE (calling_number = '+33612345678' OR called_number =
'+33612345678')
      AND submission_time BETWEEN '2025-11-01 00:00:00' AND '2025-11-30
23:59:59'
ORDER BY submission_time;

-- Obter conversa entre duas partes
SELECT * FROM cdrs
WHERE (calling_number = '+33612345678' AND called_number =
'+33687654321')
      OR (calling_number = '+33687654321' AND called_number =
'+33612345678')
ORDER BY submission_time;
```

2. CAPACIDADES DE CRIPTOGRAFIA E CRIPTOANÁLISE

2.1 Visão Geral das Capacidades Criptográficas

O OmniMessage SMSc implementa mecanismos criptográficos para proteger comunicações e proteger dados sensíveis. Esta seção documenta todas as

capacidades criptográficas de acordo com os requisitos da ANSSI.

2.2 Criptografia da Camada de Transporte

2.2.1 Implementação TLS/SSL

Protocolos Suportados:

- TLS 1.2 (RFC 5246)
- TLS 1.3 (RFC 8446) - Recomendado
- SSL 2.0/3.0: NÃO SUPORTADO (vulnerabilidades conhecidas)
- TLS 1.0/1.1: DESATIVADO (não recomendado)

Implementação:

- Biblioteca SSL/TLS Erlang/OTP (validada criptograficamente)
- Servidor web Cowboy com suporte a TLS
- Endpoints HTTPS do Phoenix Framework

Conjuntos de Cifras:

O sistema utiliza a seleção padrão de conjuntos de cifras seguras do Erlang/OTP, que inclui:

Preferido - TLS 1.3:

- TLS_AES_256_GCM_SHA384
- TLS_AES_128_GCM_SHA256
- TLS_CHACHA20_POLY1305_SHA256

Suportado - TLS 1.2:

- ECDHE-RSA-AES256-GCM-SHA384
- ECDHE-RSA-AES128-GCM-SHA256
- DHE-RSA-AES256-GCM-SHA384
- DHE-RSA-AES128-GCM-SHA256

Recursos de Segurança:

- Perfeita Secreção de Avanço (PFS) via troca de chaves ECDHE/DHE
- Grupos de Diffie-Hellman fortes (mínimo de 2048 bits)
- Suporte a Criptografia de Curva Elíptica
- Suporte a Indicação de Nome do Servidor (SNI)

Gerenciamento de Certificados:

- Suporte a certificados X.509
- Tamanhos de chave RSA: mínimo de 2048 bits, recomendado 4096 bits

- Suporte a ECDSA
- Validação de cadeia de certificados
- Certificados autoassinados (somente para desenvolvimento)
- Integração com CA externa

Localização da Configuração TLS:

```
# config/runtime.exs
config :api_ex,
  api: %{
    enable_tls: true,
    tls_cert_path: "priv/cert/omnitouch.crt",
    tls_key_path: "priv/cert/omnitouch.pem"
  }
```

◆ Referência de configuração completa em [CONFIGURATION.md](#)

Aplicações:

- HTTPS para REST API (porta 8443)
- HTTPS para painel de controle da web (porta 8086)
- Conexões de banco de dados (MySQL/PostgreSQL via TLS)

2.3 Criptografia de Dados em Repouso

2.3.1 Criptografia de Banco de Dados

Criptografia MySQL/MariaDB:

- Suporte à criptografia em nível de tabela
- Algoritmo de criptografia AES-256
- Criptografia de dados transparente (TDE)

```
-- Habilitar criptografia para tabela CDR
ALTER TABLE cdrs ENCRYPTION='Y';
```

Criptografia PostgreSQL:

- Suporte à criptografia de dados transparente
- Criptografia em nível de sistema de arquivos
- Criptografia em nível de coluna (extensão pgcrypto)

2.3.2 Armazenamento em Disco do Mnesia

Banco de Dados Mnesia:

- Cópias em disco para persistência de mensagens
- Criptografia em nível de sistema de arquivos recomendada (LUKS, dm-

- crypt)
- Proteção de memória via isolamento da VM Erlang

2.3.3 Criptografia de Sistema de Arquivos

Armazenamento de Dados Sensíveis:

- Arquivos de configuração: Criptografia de sistema de arquivos recomendada
- Chaves privadas: Permissões de arquivo (0600) + criptografia de sistema de arquivos
- Arquivos de log: Criptografia configurável para logs arquivados
- Exportações de CDR: Armazenamento criptografado para exportações sensíveis

Armazenamento de Chaves:

- Certificados e chaves TLS armazenados em priv/cert/
- Keystores baseados em arquivo com permissões restritas
- Procedimentos de rotação de chaves seguras

2.4 Autenticação e Controle de Acesso

2.4.1 Autenticação da API

Segurança da REST API:

- Criptografia de transporte HTTPS/TLS obrigatória
- Autenticação baseada em cabeçalho (cabeçalho SMS-Sc para identificação de frontend)
- Controle de acesso baseado em IP (nível de firewall)
- Autenticação de cliente baseada em certificado (opcional)

Registro de Frontend:

- Identificação única de frontend (nome, tipo, IP, hostname)
- Autenticação baseada em heartbeat
- Gerenciamento de sessão baseado em expiração (timeout de 90 segundos)
- Rastreamento e monitoramento de frontend

2.4.2 Autenticação de Banco de Dados

Controle de Acesso ao Banco de Dados:

- Autenticação de nome de usuário/senha
- Suporte a conexão TLS/SSL
- Restrições de conexão baseadas em IP
- Controle de acesso baseado em funções (RBAC)

Configuração:

```
# config/runtime.exs
config :sms_c, SmsC.Repo,
  username: "omnitouch",
  password: "omnitouch2024", # Deve usar senhas fortes em produção
  hostname: "localhost",
  ssl: true # Habilitar TLS para conexões de banco de dados
```

Recomendações de Controle de Acesso:

```
-- Criar usuário somente leitura para acesso à aplicação de justiça
CREATE USER 'li_readonly'@'%' IDENTIFIED BY 'secure_password';
GRANT SELECT ON sms_c.cdrrs TO 'li_readonly'@'%';

-- Criar usuário limitado sem acesso ao conteúdo da mensagem
CREATE USER 'analytics'@'%' IDENTIFIED BY 'secure_password';
GRANT SELECT (id, message_id, calling_number, called_number,
              source_smsc, dest_smsc, submission_time, delivery_time,
              status, delivery_attempts)
ON sms_c.cdrrs TO 'analytics'@'%';
```

2.5 Detalhes de Algoritmos Criptográficos

2.5.1 Algoritmos de Hashing

Disponíveis no Erlang/OTP:

- SHA-256, SHA-384, SHA-512 (recomendado)
- SHA-1 (descontinuado, compatibilidade legada apenas)
- MD5 (descontinuado, não usado para segurança)
- BLAKE2 (disponível em versões modernas do OTP)

Uso:

- Impressão digital de mensagens (detecção de duplicatas)
- Verificação de integridade de dados
- Integridade de logs de auditoria

2.5.2 Criptografia Simétrica

Algoritmos Disponíveis:

- AES (Padrão de Criptografia Avançada)
 - AES-128-GCM
 - AES-256-GCM
 - AES-128-CBC
 - AES-256-CBC

- ChaCha20-Poly1305

Tamanhos de Chave:

- 128 bits (mínimo)
- 256 bits (recomendado)

Uso:

- Criptografia de sessão TLS
- Criptografia de banco de dados em repouso
- Criptografia opcional do corpo da mensagem

2.5.3 Criptografia Assimétrica

Algoritmos Suportados:

- RSA (mínimo de 2048 bits, recomendado 4096 bits)
- ECDSA (Algoritmo de Assinatura Digital de Curva Elíptica)
 - Curvas P-256, P-384, P-521
- Ed25519 (EdDSA)

Uso:

- Autenticação de certificado TLS
- Assinaturas digitais
- Troca de chaves

2.6 Segurança do Protocolo SMS

2.6.1 Codificação de Mensagens SMS

Suporte a Codificação de Caracteres:

- GSM 7-bit (codificação padrão de SMS)
- UCS-2 (Unicode, 16-bit)
- Dados binários de 8 bits
- Latin-1

TP-DCS (Esquema de Codificação de Dados):

- Indicação da classe da mensagem
- Flags de compressão
- Especificação do grupo de codificação
- Identificação do conjunto de caracteres

Sem Criptografia Nativa de SMS:

- O protocolo SMS não fornece criptografia de ponta a ponta
- Conteúdo da mensagem acessível no nível do SMSc
- Permite interceptação legal conforme necessário

2.6.2 Considerações de Segurança do Protocolo

Protocolo SMPP (Frontend Externo):

- Autenticação de nome de usuário/senha no nível SMPP
- Suporte a TLS disponível (SMPP sobre TLS)
- Autenticação de bind

Protocolo IMS (Frontend Externo):

- Mensagens baseadas em SIP
- Mecanismos de autenticação SIP
- Integração com segurança da rede central IMS

Protocolo SS7/MAP (Frontend Externo):

- Segurança da rede SS7
- Autenticação do protocolo MAP
- Segurança da camada SCCP/TCAP

Nota: A segurança específica do protocolo é implementada nos gateways de frontend externos, não no núcleo do SMSc.

2.7 Capacidades de Criptoanálise e Avaliação de Segurança

2.7.1 Ferramentas de Análise de Protocolo

Capacidades de Depuração Integradas:

- Sistema de registro abrangente
- Rastreio do fluxo de mensagens
- Registro de solicitações/respostas da API
- Registro de consultas ao banco de dados
- Rastreamento de erros e exceções

Integração Externa:

- Saída de registro padrão (stdout/arquivos)
- Suporte a captura PCAP para análise de rede
- Registro de consultas ao banco de dados para forense
- Exportação de métricas Prometheus

2.7.2 Considerações de Avaliação de Vulnerabilidades

Limitações Conhecidas:

- Protocolo SMS inerentemente não criptografado (por design, permite interceptação legal)
- Credenciais de banco de dados em arquivos de configuração (deve usar gerenciamento de segredos)
- Suporte a certificados autoassinados (somente desenvolvimento/teste)

Recomendações de Fortalecimento de Segurança:

- Usar conjuntos de cifras TLS fortes
- Implementar criptografia de conexão de banco de dados
- Usar gerenciamento de segredos externo (Vault, AWS Secrets Manager)
- Atualizações regulares de segurança para Erlang/OTP e dependências
- Restrições de firewall nas portas da API
- Lista de permissões de IP para acesso ao frontend

Testes de Segurança:

- Escaneamento regular de vulnerabilidades de dependências
- Suporte a testes de penetração
- Validação da configuração TLS
- Auditorias de segurança de banco de dados
- Revisão de controle de acesso

2.8 Infraestrutura de Gerenciamento de Chaves

2.8.1 Geração de Chaves

Geração de Certificado TLS:

```
# Gerar chave privada (RSA 4096-bit)
openssl genrsa -out omnitouch.pem 4096

# Gerar solicitação de assinatura de certificado
openssl req -new -key omnitouch.pem -out omnitouch.csr

# Certificado autoassinado (desenvolvimento)
openssl x509 -req -days 365 -in omnitouch.csr -signkey omnitouch.pem
-out omnitouch.crt

# Produção: Obter certificado de CA confiável
```

Geração de Números Aleatórios:

- CSPRNG Erlang/OTP (Gerador de Números Aleatórios Pseudo-Aleatórios)

- Criptograficamente Seguro)
- Pool de entropia do sistema (/dev/urandom)
- Aleatoriedade forte para chaves de sessão, IDs, tokens

2.8.2 Armazenamento e Proteção de Chaves

Armazenamento de Chaves Privadas:

- Sistema de arquivos com permissões restritas (0600)
- Armazenado no diretório priv/cert/
- Formato PEM (opcionalmente criptografado)
- Procedimentos de backup seguro

Rotação de Chaves:

- Procedimentos de renovação de certificado TLS (anualmente recomendado)
- Rotação de credenciais de banco de dados
- Rotação de tokens da API (se implementado)

2.8.3 Distribuição de Chaves

Distribuição de Certificados:

- Instalação manual em priv/cert/
- Referências em arquivos de configuração
- Suporte ao protocolo ACME possível (Let's Encrypt)

Distribuição de Chaves Simétricas:

- Troca de chaves fora de banda para credenciais de banco de dados
- Acordo de chaves Diffie-Hellman em TLS
- Nenhuma transmissão de chave em texto claro

2.9 Conformidade e Padrões

Conformidade com Padrões Criptográficos:

- NIST SP 800-52: Diretrizes TLS
- NIST SP 800-131A: Transições de algoritmos criptográficos
- RFC 7525: Recomendações TLS
- ETSI TS 133 310: Segurança de rede (para integração IMS)

Regulamentações de Criptografia Francesas:

- Nenhuma criptografia restrita à exportação (todos os algoritmos padrão)
- Declaração de meios criptográficos (se aplicável)
- Certificação de produto criptográfico ANSSI (se necessário)

2.10 Resistência à Criptoanálise

2.10.1 Princípios de Design

Defesa Contra Criptoanálise:

- Nenhum algoritmo criptográfico proprietário/customizado
- Apenas algoritmos padrão da indústria, revisados por pares
- Atualizações regulares de segurança para bibliotecas criptográficas
- Descontinuação de algoritmos fracos
- Uso de criptografia autenticada (GCM, Poly1305)

2.10.2 Segurança Operacional

Rotação de Chaves:

- Procedimentos de renovação de certificado TLS
- Rotação de chaves de sessão (por sessão para TLS)
- Políticas de rotação de credenciais de banco de dados

Monitoramento e Detecção:

- Registro de tentativas de autenticação falhadas
 - Monitoramento de expiração de certificados
 - Registro de falhas de handshake TLS
 - Detecção de anomalias para falhas de criptografia
 - Alertas de eventos de segurança
-

3. CONTROLE E AUTORIZAÇÃO DE INTERCEPTAÇÃO

3.1 Controle de Acesso para Interceptação Legal

Autorização Administrativa:

- Acesso de administrador do sistema necessário para configuração
- Controles de acesso em nível de banco de dados para consultas CDR
- Acesso à API restrito por IP/autenticação
- Registro de auditoria de todos os acessos

Integração com Quadro Legal:

- Rastreamento de mandados de interceptação (integração de sistema externo)
- Listas de autorização de identificadores de alvos (vistas de banco de dados)

- Consultas com limite de tempo (cláusulas WHERE SQL)
- Aplicação automática via políticas de acesso

3.2 Retenção de Dados e Privacidade

Políticas de Retenção:

- Retenção de mensagens ativas: Configurável (padrão 24 horas no Mnesia)
- Retenção de CDR: Configurável (típico de 6 meses a 2 anos)
- Arquivamento automático do Mnesia para SQL
- Purgação automática de CDRs antigos (baseada em cron)

Proteções de Privacidade:

- Opção de exclusão do corpo da mensagem após a entrega
- Exclusão do corpo da mensagem da interface/exportações
- Criptografia de banco de dados em repouso
- Registro e monitoramento de acesso
- Princípio de coleta mínima de dados

Configuração:

```
# config/runtime.exs
config :sms_c,
  # Retenção de mensagens do Mnesia antes do arquivamento
  message_retention_hours: 24,

  # Excluir corpo da mensagem após entrega para privacidade
  delete_message_body_after_delivery: false, # Definir true para
modo de privacidade

  # Controle de gravação de CDR
  cdr_enabled: true,

  # Configurações de arquivamento em lote
  batch_insert_batch_size: 100,
  batch_insert_flush_interval_ms: 100
```

❖ Veja [CONFIGURATION.md](#) para todas as configurações de retenção

3.3 Interfaces de Transferência para Aplicações de Justiça

Interfaces Padrão:

1. Acesso à REST API:

- Endpoints HTTPS para recuperação de mensagens
- Troca de dados em formato JSON

- Autenticação e autorização
- Filtragem de consultas por critérios de alvo

2. Acesso Direto ao Banco de Dados:

- Credenciais SQL somente leitura
- Consultas SQL padrão
- Acesso à tabela CDR
- Capacidades de pesquisa indexadas

3. Exportação em Lote:

- Formato de exportação CSV
- Formato de exportação JSON
- Exportações baseadas em intervalo de tempo
- Seleção de campos configuráveis

Formatos de Entrega:

IRI (Informação Relacionada à Interceptação):

- Campos de metadados CDR:
 - ID da mensagem
 - Números chamadores/chamados
 - Timestamps (submissão, entrega, expiração)
 - Status
 - Tentativas de entrega
 - Informações de roteamento SMSC
 - Informações de nó (rastreamento de cluster)

CC (Conteúdo da Comunicação):

- Corpo da mensagem (conteúdo de texto)
- Dados brutos de PDU
- Informações de codificação
- Montagem de mensagem multiparte

Exemplo de Exportação:

```
# Exportação CSV para a aplicação de justiça
mysql -u li_readonly -p -D sms_c -e "
SELECT
  message_id,
  calling_number,
  called_number,
  message_body,
  submission_time,
  delivery_time,
  status
```

```
FROM cdrs
WHERE (calling_number = '+33612345678' OR called_number =
'+33612345678')
AND submission_time BETWEEN '2025-11-01' AND '2025-11-30'
ORDER BY submission_time
" --batch --silent | sed 's/\t/,/g' > interception_report.csv
```

4. SEGURANÇA E INTEGRIDADE DO SISTEMA

4.1 Segurança da Aplicação

Segurança Erlang/Erlang:

- Isolamento e sandboxing da VM Erlang
- Isolamento de processos e supervisão
- Recuperação de falhas e tolerância a falhas
- Sem vulnerabilidades de estouro de buffer (tempo de execução gerenciado)

Gerenciamento de Dependências:

- Bloqueio de versão de dependências (mix.lock)
- Escaneamento de vulnerabilidades de segurança
- Atualizações regulares de dependências
- Pegada mínima de dependências

4.2 Segurança de Rede

Exposição de Rede:

- Portas expostas mínimas:
 - 8443 (API REST HTTPS)
 - 8086 (Painel de Controle HTTPS)
 - 3306/5432 (Banco de dados - deve ser protegido por firewall)
- Configuração de firewall recomendada
- Lista de permissões de IP para acesso ao frontend
- Implantação DMZ para serviços voltados para a internet

Segmentação de Rede:

- Rede de gerenciamento separada
- Rede de banco de dados isolada
- Separação da rede do gateway de frontend
- Rede de comunicação de cluster (distribuição Erlang)

4.3 Monitoramento e Detecção de Intrusões

Capacidades de Registro:

- Registro estruturado da aplicação
- Níveis de registro configuráveis
- Rotação e arquivamento de logs
- Suporte à integração Syslog
- Registro centralizado (compatível com ELK stack)

Monitoramento de Eventos de Segurança:

- Tentativas de autenticação falhadas
- Padrões de mensagens incomuns
- Falhas de conexão ao banco de dados
- Falhas de handshake TLS
- Anomalias de recursos do sistema

Métricas e Alertas:

- Exportação de métricas Prometheus
- Monitoramento de throughput de mensagens
- Rastreamento de taxas de erro
- Utilização de recursos do sistema
- Regras de alerta personalizadas

❖ **Documentação completa de monitoramento em**
[OPERATIONS_GUIDE.md](#) e [METRICS.md](#)

4.4 Alta Disponibilidade e Recuperação de Desastres

Suporte a Cluster:

- Capacidade de cluster distribuído Erlang
- Replicação Mnesia entre nós
- Failover automático
- Descoberta e junção de nós

Redundância de Dados:

- Cópias em disco Mnesia em todos os nós do cluster
- Replicação de banco de dados SQL (nativa do MySQL/PostgreSQL)
- Procedimentos de backup de CDR
- Backup de configuração

Procedimentos de Recuperação:

- Backup e restauração do banco de dados

- Recuperação de tabelas Mnesia
 - Restauração de configuração
 - Procedimentos de substituição de nós
-

5. REFERÊNCIAS DA DOCUMENTAÇÃO

5.1 Manuais Técnicos

Documentação disponível no repositório do projeto:

- [README.md](#) - Visão geral do sistema, arquitetura e recursos
- [CONFIGURATION.md](#) - Referência completa de configuração
- [API_REFERENCE.md](#) - Documentação da REST API
- [OPERATIONS_GUIDE.md](#) - Procedimentos operacionais e monitoramento
- [CDR_SCHEMA.md](#) - Esquema do banco de dados de Registro de Detalhes de Chamadas
- [sms_routing_guide.md](#) - Configuração de roteamento de SMS
- [number_translation_guide.md](#) - Normalização de números
- [METRICS.md](#) - Métricas e monitoramento Prometheus
- [PERFORMANCE_TUNING.md](#) - Otimização de desempenho
- [TROUBLESHOOTING.md](#) - Problemas comuns e soluções

5.2 Certificações de Segurança

- **Relatórios de Testes de Penetração:** [A serem fornecidos mediante solicitação]
- **Relatórios de Auditoria de Segurança:** [A serem fornecidos mediante solicitação]
- **Avaliações de Vulnerabilidades:** [A serem fornecidas mediante solicitação]
- **Validação Criptográfica Erlang/OTP:** Biblioteca criptográfica padrão da indústria

5.3 Documentação de Conformidade

- **Solicitação de Autorização ANSSI R226:** Este documento
 - **Conformidade de Interceptação Legal:** Conforme exigido pelas regulamentações de telecomunicações francesas
 - **Conformidade com Proteção de Dados:** Considerações GDPR para dados de mensagens
-

6. INFORMAÇÕES DE CONTATO

Informações do Fornecedor/Operador:

- Nome da Empresa: Omnitouch Network Services Pty Ltd
- Endereço: PO BOX 296, QUINNS ROCKS WA 6030, AUSTRÁLIA
- Pessoa de Contato: Equipe de Conformidade
- Email: compliance@omnitouch.com.au

Contato de Segurança Técnica:

- Nome: Equipe de Conformidade
- Email: compliance@omnitouch.com.au

Contato Legal/Compliance:

- Nome: Equipe de Conformidade
 - Email: compliance@omnitouch.com.au
-

APÊNDICES

Apêndice A: Fluxo de Mensagens SMS com Pontos de Interceptação

A.1 Fluxo de SMS de Saída (Terminado no Móvel)



Guia de Tradução de Números SMS-C

[← Voltar ao Índice da Documentação](#) | [README Principal](#)

Visão Geral

O sistema de Tradução de Números SMS-C fornece uma transformação flexível e baseada em regex de números de telefone antes do roteamento. As regras de tradução podem normalizar números, adicionar prefixos internacionais, formatar números para gateways específicos e encadear múltiplas transformações. As regras são armazenadas no Mnesia para persistência e podem ser modificadas em tempo de execução sem interrupção do serviço.

Principais Recursos

- **Correspondência baseada em prefixo:** Correspondência de números por prefixo antes de aplicar transformações
- **Transformação baseada em regex:** Poderosa correspondência de padrões e substituição com grupos de captura
- **Filtragem de SMSC de origem:** Aplicar diferentes traduções com base na origem da mensagem
- **Avaliação baseada em prioridade:** Controlar a ordem das regras com prioridades configuráveis (1-255)
- **Encadeamento de regras:** Continuar o processamento através de múltiplas regras com prevenção de loops
- **Transformações separadas de chamador/chamado:** Transformação independente para números de origem e destino
- **Carregamento de arquivo de configuração:** Carregar regras iniciais de `runtime.exs` na primeira inicialização
- **Configuração em tempo de execução:** Adicionar, modificar ou desabilitar regras sem reiniciar
- **Interface Web:** Interface CRUD completa para gerenciamento de regras
- **Ferramenta de simulação:** Testar lógica de tradução com avaliação passo a passo
- **Backup/Restauração:** Exportar e importar configurações de tradução
- **Integração pré-roteamento:** Traduções aplicadas antes do roteamento para formatos de número consistentes

Arquitetura

Modelo de Dados

Cada regra de tradução contém os seguintes campos:

Campo	Tipo	Descrição	Necessário
rule_id	inteiro	Identificador único auto-incremental	Sim (auto)
calling_prefix	string/ nil	Correspondência de prefixo para número de chamada (nil = curinga)	Não
called_prefix	string/ nil	Correspondência de prefixo para número chamado (nil = curinga)	Não
source_smsc	string/ nil	Nome do SMSC de origem (nil = curinga)	Não
calling_match	string/ nil	Padrão regex para corresponder ao número de chamada	Não
calling_replace	string/ nil	Padrão de substituição para número de chamada	Não
called_match	string/ nil	Padrão regex para corresponder ao número chamado	Não
called_replace	string/ nil	Padrão de substituição para número chamado	Não
priority	inteiro	Prioridade da regra (1-255, menor = maior prioridade)	Sim
description	string	Descrição legível por humanos	Não
enabled	booleano	Habilitar/desabilitar regra	Sim
continue	booleano	Continuar avaliando regras após correspondência (padrão: falso)	Não

Nota: As regras são avaliadas em ordem de prioridade (número mais baixo primeiro). Apenas regras habilitadas são avaliadas.

Algoritmo de Tradução

Ao traduzir números, o sistema:

1. **Recupera regras habilitadas** ordenadas por prioridade (menor primeiro)
2. **Avalia regras sequencialmente** contra os parâmetros da mensagem:
 - Correspondência de calling_prefix (se especificado)
 - Correspondência de called_prefix (se especificado)
 - Correspondência de source_smsc (se especificado)
3. **Aplica a primeira regra correspondente:**
 - Transformar o número de chamada usando calling_match e calling_replace
 - Transformar o número chamado usando called_match e called_replace

4. **Verifica a flag continue:**

- Se continue: false → Parar o processamento, retornar resultado
- Se continue: true → Remover a regra correspondente das regras disponíveis, continuar com o passo 2 usando **números transformados**

5. **Retorna números finais** e lista de todas as regras aplicadas

Encadeamento de Regras com Prevenção de Loop

A flag continue permite um poderoso encadeamento de regras enquanto previne loops infinitos:

Curiosidades

- nil ou valores vazios atuam como curingas que correspondem a qualquer valor
- Uma regra sem critérios de correspondência é uma regra catch-all
- Uma regra sem padrões de transformação (nil match/replace) passa números sem alteração

Exemplo: Cenário de Encadeamento de Regras

Configuração

Carregando Regras do Arquivo de Configuração

As regras de tradução podem ser definidas em config/runtime.exs e serão carregadas automaticamente na primeira inicialização.

Importante: As regras da configuração só são carregadas quando a tabela de tradução está **vazia** (primeira inicialização). Isso preserva as regras adicionadas via a interface Web durante o tempo de execução e previne duplicatas em reinicializações.

Fluxo de Carregamento de Configuração

Exemplo de Configuração

```
# config/runtime.exs
config :sms_c, :translation_rules, [
  # Adicionar +1 a números de 10 dígitos dos EUA
  %{
    calling_prefix: nil,
    called_prefix: nil,
    source_smsc: "us_domestic_smsc",
    calling_match: "^(\\d{10})$",
    calling_replace: "+1\\1",
```

```
    called_match: "^(\\d{10})$",
    called_replace: "+1\\1",
    priority: 10,
    description: "Adicionar +1 a números de 10 dígitos dos EUA de
SMSC doméstico",
    enabled: true,
    continue: false
},

# Remover zeros à frente do formato internacional
%{
    calling_prefix: "00",
    called_prefix: nil,
    source_smsc: nil,
    calling_match: "^00(.+)$",
    calling_replace: "+\\1",
    called_match: nil,
    called_replace: nil,
    priority: 5,
    description: "Converter prefixo internacional 00 para +",
    enabled: true,
    continue: true # Continuar aplicando mais formatação
},

# Formatar números do Reino Unido para um gateway específico
%{
    calling_prefix: "+44",
    called_prefix: "+44",
    source_smsc: nil,
    calling_match: "^\\+44(.*)$",
    calling_replace: "0044\\1",
    called_match: "^\\+44(.*)$",
    called_replace: "0044\\1",
    priority: 20,
    description: "Formatar números do Reino Unido para gateway
legado",
    enabled: true,
    continue: false
}
]
```

Começando

Fluxo de Inicialização

Fluxo de Tradução de Mensagens

Casos de Uso Comuns

Normalização de Números Internacionais

Normalize vários formatos internacionais para E.164:

Formatação Específica de Gateway

Encadeie regras para formatar números de acordo com os requisitos de um gateway específico:

Traduções Específicas de SMSC

Aplique diferentes traduções com base na origem da mensagem:

Preparação de Roteamento Baseada em Prefixo

Normalize números antes do roteamento para garantir correspondência consistente de prefixos:

Manipulação de Portabilidade de Números

Lide com números portados que requerem alterações de prefixo:

Interface Web

UI de Gerenciamento de Regras de Tradução

Acesse a interface de gerenciamento de regras em `/number_translation` (via menu de navegação):

Recursos:

- Visualizar todas as regras em uma tabela classificável por prioridade
- Adicionar novas regras com validação de formulário
- Editar regras existentes
- Habilitar/desabilitar regras sem excluir
- Excluir regras com confirmação
- Indicador visual para regras com `continue: true`

- Importar/Exportar regras como JSON

Adicionando uma Regra:

1. Preencha os critérios de correspondência (opcional):
 - Prefixo de chamada (ex: "+1", "44")
 - Prefixo chamado (ex: "+639", "1555")
 - SMSC de origem (deixe vazio para qualquer)
2. Defina as transformações (opcional):
 - Correspondência e substituição de regex para número de chamada
 - Correspondência e substituição de regex para número chamado
3. Defina a prioridade (1-255, menor = maior prioridade)
4. Defina o status:
 - **Habilitado**: Regra está ativa
 - **Continuar Processando**: Continuar avaliando mais regras após esta
5. Adicione descrição
6. Clique em "Adicionar Regra" ou "Atualizar Regra"

Alternância de Continuar Processando:

- **Parar** (padrão): Parar o processamento após esta regra corresponder
- **Continuar**: Aplicar esta regra e continuar avaliando as regras restantes
- Regras com continue habilitado mostram um distintivo azul "↓ Continuar" na tabela

Editando uma Regra:

1. Clique em "Editar" ao lado da regra
2. Modifique os campos conforme necessário
3. Clique em "Atualizar Regra"

Indicadores da Tabela de Regras:

- Distintivo **Habilitado/Desabilitado** mostra o status da regra
- Distintivo ↓ **Continuar** mostra regras que continuarão o processamento
- Distintivo **Prioridade** mostra a ordem de avaliação
- Padrões regex exibidos em fonte monoespaçada para clareza

Simulador de Tradução

Acesse o simulador em `/translation_simulator` (via menu de navegação):

Recursos:

- Testar lógica de tradução com números reais
- **Transformação Passo a Passo** mostrando cada regra aplicada
- Ver valores antes/depois para cada transformação
- Visualizar quais regras corresponderam e por quê

- Carregar cenários de exemplo para testes rápidos
- Visualizar histórico de testes (últimos 10 testes)

Usando o Simulador:

1. Insira os parâmetros de teste:
 - Número de chamada (de)
 - Número chamado (para)
 - SMSC de origem (opcional)
2. Clique em "Testar Tradução"
3. Veja resultados abrangentes:
 - **Resultado da Tradução:** Números finais após todas as transformações
 - **Regras Aplicadas:** Contagem e lista de todas as regras que corresponderam
 - **Transformações Passo a Passo:** Visão detalhada de cada regra:
 - Número do passo e informações da regra
 - Descrição da regra
 - Antes → Depois para números de chamada e chamado
 - Indicador "↓ Continuar" para regras que continuaram o processamento
 - Transformações destacadas em verde
 - Valores inalterados marcados como "passaram através"
4. Carregue exemplos pré-configurados usando os botões de exemplo
5. Revise o histórico de testes para comparar diferentes cenários

Exemplo de Saída:

Resultado da Tradução

```

Número Chamador: 5551234567 → +1-555-123-4567
Número Chamado: 9078720155 → +1-907-872-0155
✓ Traduzido por 3 regra(s)
  
```

Transformações Passo a Passo

Passo 1

```

Regra #1 (Prioridade 10)           ↓ Continuar |
Adicionar código de país a números de 10 dígitos |
Chamado: 9078720155 → +19078720155
  
```

Passo 2

```

Regra #2 (Prioridade 20)           ↓ Continuar |
Formatar código de área com traços |
Chamado: +19078720155 → +1-907-8720155
  
```



```
Passo 3
Regra #3 (Prioridade 30)
Formatação final para gateway

Chamado: +1-907-8720155 → +1-907-872-0155
```

Referência da API

Visão Geral das Operações Principais

Parâmetros de Tradução

translate_numbers aceita os seguintes parâmetros:

- `calling_number` (opcional): Número de telefone de origem
- `called_number` (opcional): Número de telefone de destino
- `source_smsc` (opcional): Identificador do SMSC de origem
- `message_id` (opcional): Para registro de eventos

Retorna:

- `{:ok, translated_calling, translated_called, [rules_applied]}` - Sempre bem-sucedido
- Retorna números originais se nenhuma regra corresponder
- Retorna lista de todas as regras que foram aplicadas (na ordem)

```
# Exemplo de uso
{:ok, new_calling, new_called, rules} =
  NumberTranslation.translate_numbers(
    calling_number: "5551234567",
    called_number: "9078720155",
    source_smsc: "domestic_gateway",
    message_id: "msg_123"
  )

# Verificar se alguma tradução ocorreu
if rules != [] do
  Logger.info("Aplicadas #{length(rules)} regras de tradução")
  Enum.each(rules, fn rule ->
    Logger.info("  - Regra ##{rule.rule_id}: #{rule.description}")
  end)
end
```


Operações de Gerenciamento de Regras

```
# Adicionar uma nova regra
{:ok, rule} = NumberTranslation.add_rule(%{
  calling_prefix: nil,
  called_prefix: nil,
  source_smsc: "gateway1",
  calling_match: "^(\\d{10})$",
  calling_replace: "+1\\1",
  called_match: "^(\\d{10})$",
  called_replace: "+1\\1",
  priority: 10,
  description: "Adicionar +1 a números de 10 dígitos",
  enabled: true,
  continue: false
})

# Atualizar uma regra
{:ok, updated_rule} = NumberTranslation.update_rule(rule_id, %{
  enabled: false,
  description: "Desabilitado para teste"
})

# Excluir uma regra
:ok = NumberTranslation.delete_rule(rule_id)

# Obter uma regra específica
rule = NumberTranslation.get_rule(rule_id)

# Listar todas as regras
all_rules = NumberTranslation.list_rules()

# Listar apenas regras habilitadas (ordenadas por prioridade)
enabled_rules = NumberTranslation.list_enabled_rules()
```

Operações de Importação/Exportação

```
# Exportar todas as regras
backup = NumberTranslation.export_rules()
# Retorna: %{
#   version: "1.0",
#   exported_at: ~U[2024-01-15 10:30:00Z],
#   count: 5,
#   rules: [...]
# }

# Salvar em arquivo JSON
json = Jason.encode!(backup, pretty: true)
```

```
File.write!("translation_rules_backup.json", json)

# Importar regras (mesclar com existente)
{:ok, %{imported: 3, failed: 0}} =
  NumberTranslation.import_rules(backup, mode: :merge)

# Importar regras (substituir todas as existentes)
{:ok, %{imported: 5, failed: 0}} =
  NumberTranslation.import_rules(backup, mode: :replace)
```

Melhores Práticas

Design de Regras

1. Mantenha prioridades organizadas:

- **1-10:** Regras de normalização críticas (adicionar códigos de país, corrigir formatos)
- **11-50:** Formatação específica de gateway
- **51-100:** Transformações opcionais
- **101+:** Regras catch-all ou de depuração

2. Use continue estrategicamente:

- Habilite continue: true para regras de normalização que preparam números para processamento adicional
- Desabilite continue: false para regras de formatação final
- Evite longas cadeias (máximo de 3-4 regras) para manter o desempenho

3. Documente suas regras:

- Sempre adicione descrições claras
- Inclua exemplos na descrição (ex: "5551234567 → +15551234567")
- Documente o propósito e a entrada/saída esperada

4. Teste padrões regex:

- Teste padrões com o simulador antes de implantar
- Use grupos de captura (\1, \2) para transformações flexíveis
- Escape caracteres especiais de regex (pontos, parênteses, etc.)

Desempenho

1. Minimize a contagem de regras:

- Combine regras semelhantes sempre que possível
- Use correspondência de prefixo para reduzir avaliações de regex

- Remova ou desabilite regras não utilizadas

2. **Otimize padrões regex:**

- Use correspondência de prefixo primeiro (mais rápido que regex)
- Mantenha padrões regex simples
- Evite padrões pesados em retrocesso

3. **Limite o encadeamento de regras:**

- Cadeias longas (5+ regras) podem impactar o desempenho
- Considere combinar múltiplos passos em uma regra, se possível
- Monitore a latência de tradução com métricas de Telemetria

Operações

1. **Teste antes de implantar:**

- Use o simulador com exemplos do mundo real
- Teste casos extremos (números vazios, caracteres especiais)
- Verifique o comportamento da flag continue

2. **Faça backup regularmente:**

- Exporte regras antes de fazer alterações significativas
- Controle de versão suas exportações
- Teste importações em não-produção primeiro

3. **Monitore traduções:**

- Habilite o registro de message_id para depuração
- Verifique logs de eventos para decisões de tradução
- Monitore quais regras estão sendo aplicadas

4. **Implantação gradual:**

- Adicione novas regras como desabilitadas primeiro
- Teste com o simulador
- Habilite e monitore
- Ajuste conforme necessário

Dicas de Regex

1. **Padrões comuns:**

- Número de 10 dígitos dos EUA: `^(\\d{10})$`
- Formato internacional: `^\\+(\\d+)$`
- Remover zeros à frente: `^0+(\\.+) $`

- Adicionar traços: `^(\d{3})(\d{3})(\d{4})$` → `\1-\2-\3`

2. Grupos de captura:

- Use parênteses para capturar: `^(\d{3})(\d{7})$`
- Referência na substituição: `+1\1\2`
- Múltiplas capturas: `^\+(\d{1,3})(\d+)$` → `00\1\2`

3. Escape de caracteres especiais:

- Ponto literal: `\.`
- Mais literal: `\+`
- Parêntese literal: `\(` ou `\)`

Solução de Problemas

Regra Não Correspondente

Sintoma: Regra esperada não corresponde, números passam sem alteração

Possíveis causas:

- Prefixo não corresponde (verifique a correspondência exata do prefixo)
- SMSC de origem não corresponde
- Padrão regex não corresponde ao formato de entrada
- Regra está desabilitada
- Regra de prioridade mais alta correspondida primeiro (com `continue: false`)

Soluções:

1. Use o simulador para ver quais regras são avaliadas
2. Verifique o status da regra (habilitada/desabilitada)
3. Verifique a correspondência de prefixo (sensível a maiúsculas)
4. Teste o padrão regex separadamente
5. Verifique a ordem de prioridade

Transformação Errada Aplicada

Sintoma: Número transformado, mas resultado está incorreto

Possíveis causas:

- Padrão regex corresponde, mas padrão de substituição está errado
- Múltiplas regras aplicando em ordem inesperada
- Referências de grupos de captura incorretas (`\1`, `\2`, etc.)

Soluções:

1. Use o simulador para ver transformações passo a passo
2. Verifique se o padrão regex captura os grupos corretos
3. Verifique a sintaxe do padrão de substituição
4. Teste regex em um testador de regex online
5. Revise a prioridade da regra e as flags continue

Loop Infinito / Degradação de Desempenho

Sintoma: Tradução leva muito tempo ou parece travar

Nota: Isso não deve acontecer devido à prevenção de loops, mas se acontecer:

Possíveis causas:

- Bug na lógica de prevenção de loops
- Avaliação de regex extremamente longa
- Cadeia de regras muito longa

Soluções:

1. Verifique os logs da aplicação para erros
2. Revise regras com continue: true
3. Simplifique padrões regex
4. Reduza o número de regras encadeadas
5. Relate um bug se a prevenção de loops falhar

Encadeamento de Regras Inesperado

Sintoma: Mais regras aplicadas do que o esperado

Possíveis causas:

- Regras têm continue: true quando não deveriam
- A ordenação de prioridade permite múltiplas correspondências
- Número transformado corresponde a regras adicionais

Soluções:

1. Use o simulador para ver a cadeia exata de regras
2. Revise as flags continue em todas as regras
3. Ajuste prioridades para controlar a ordem
4. Defina continue: false na regra final

Tradução Não Aplicada Antes do Roteamento

Sintoma: Roteador vê números não traduzidos

Possíveis causas:

- Tradução não integrada no fluxo da mensagem
- Tradução ocorrendo após o roteamento
- Código da aplicação ignorando tradução

Soluções:

1. Verifique a integração da aplicação: a tradução deve ser chamada antes do roteamento
2. Verifique o pipeline de processamento de mensagens
3. Revise logs de eventos para eventos de tradução
4. Certifique-se de que `translate_numbers` é chamado na ordem correta

Tópicos Avançados

Integração com Roteamento

A tradução acontece **antes** do roteamento para garantir formatos de número consistentes:

Registro de Eventos

Decisões de tradução são registradas via o EventLogger:

- `translation_started`: A tradução começa
- `translation_candidates`: Número de regras habilitadas
- `translation_matched`: Regra correspondente e aplicada
- `translation_calling`: Número de chamada transformado
- `translation_called`: Número chamado transformado
- `translation_continue`: Regra tem `continue=true`, continuando a avaliação
- `translation_none`: Nenhuma regra correspondeu

Habilite o registro passando `message_id` para `translate_numbers/1`.

Métricas de Telemetria

Monitore o desempenho da tradução com Telemetria:

```
:telemetry.attach(  
  "number-translation-handler",  
  [:sms_c, :number_translation, :translate, :stop],  
  fn _event_name, measurements, metadata, _config ->  
    # measurements: %{duration: microsegundos}  
    # metadata: %{rules_applied: contagem, ...}  
  end,  
  nil  
)
```

Principais métricas a serem monitoradas:

- Duração da tradução (p50, p95, p99)
- Regras aplicadas por mensagem
- Regras correspondidas vs não correspondidas
- Uso da flag continue

Clustering

As tabelas Mnesia são automaticamente distribuídas entre nós em cluster. As regras de tradução são replicadas para alta disponibilidade.

Estratégias de Migração

Ao implantar novas regras de tradução:

Exemplos

Exemplo 1: Normalização de Números dos EUA

Requisito: Converter vários formatos de números dos EUA para E.164 (+1XXXXXXXXXX)

```
# Regra 1: Números de 10 dígitos (prioridade mais alta)
%{
  calling_match: "^(\\d{10})$",
  calling_replace: "+1\\1",
  called_match: "^(\\d{10})$",
  called_replace: "+1\\1",
  priority: 5,
  description: "Adicionar +1 a números de 10 dígitos",
  enabled: true,
  continue: false
}

# Regra 2: 1 + 10 dígitos (prioridade média)
%{
  calling_match: "^1(\\d{10})$",
  calling_replace: "+1\\1",
  called_match: "^1(\\d{10})$",
  called_replace: "+1\\1",
  priority: 10,
  description: "Converter 1XXXXXXXXXX para +1XXXXXXXXXX",
  enabled: true,
  continue: false
}
```

```
# Casos de teste:
# "5551234567" → "+15551234567" (Regra 1)
# "15551234567" → "+15551234567" (Regra 2)
# "+15551234567" → "+15551234567" (Sem correspondência, passar através)
```

Exemplo 2: Conversão de Prefixo Internacional com Encadeamento

Requisito: Converter prefixo 00 para +, depois formatar para gateway

```
# Regra 1: Converter 00 para + (continua para a próxima regra)
%{
  calling_match: "^00(.+)$",
  calling_replace: "+\1",
  called_match: "^00(.+)$",
  called_replace: "+\1",
  priority: 5,
  description: "Converter prefixo internacional 00 para +",
  enabled: true,
  continue: true # Continuar para formatar
}

# Regra 2: Formatar para gateway (para o processamento)
%{
  calling_match: "^\\+(\\d+)$",
  calling_replace: "00\1",
  called_match: "^\\+(\\d+)$",
  called_replace: "00\1",
  priority: 10,
  description: "Formatar números + como 00 para gateway",
  enabled: true,
  continue: false # Parar após isso
}

# Caso de teste:
# Passo 1: "00441234567890" → "+441234567890" (Regra 1, continuar)
# Passo 2: "+441234567890" → "00441234567890" (Regra 2, parar)
# Resultado: "00441234567890"
# Regras aplicadas: [Regra 1, Regra 2]
```

Exemplo 3: Manipulação Específica de SMSC

Requisito: Aplicar regras diferentes com base na SMSC de origem

```
# Regra 1: SMSC confiável - passar através (prioridade 5)
%{
  source_smsc: "trusted_gateway",
```



```

    calling_match: nil, # Sem transformação
    calling_replace: nil,
    called_match: nil,
    called_replace: nil,
    priority: 5,
    description: "Passar números de gateway confiável",
    enabled: true,
    continue: false
}

# Regra 2: SMSC não confiável - normalizar (prioridade 10)
%{
    source_smsc: "untrusted_gateway",
    calling_match: "^(.*)$",
    calling_replace: "+VALIDATE\1",
    called_match: "^(.*)$",
    called_replace: "+VALIDATE\1",
    priority: 10,
    description: "Adicionar prefixo de validação para fonte não
confiável",
    enabled: true,
    continue: false
}

# Regra 3: Catch-all para outras SMSCs (prioridade 100)
%{
    source_smsc: nil, # Curiga
    calling_match: "^(\\d{10})$",
    calling_replace: "+1\1",
    called_match: "^(\\d{10})$",
    called_replace: "+1\1",
    priority: 100,
    description: "Padrão: Adicionar +1 a números de 10 dígitos",
    enabled: true,
    continue: false
}

```

Exemplo 4: Cadeia de Formatação em Múltiplos Passos

Requisito: Normalizar → Adicionar código de país → Formatar com traços

```

# Regra 1: Remover zeros à frente (continuar)
%{
    calling_match: "^0+(.+)$",
    calling_replace: "\1",
    called_match: "^0+(.+)$",
    called_replace: "\1",
    priority: 5,

```

```

    description: "Remover zeros à frente",
    enabled: true,
    continue: true
}

# Regra 2: Adicionar código de país se ausente (continuar)
%{
    calling_match: "^(\\d{10})$",
    calling_replace: "+1\\1",
    called_match: "^(\\d{10})$",
    called_replace: "+1\\1",
    priority: 10,
    description: "Adicionar +1 a números de 10 dígitos",
    enabled: true,
    continue: true
}

# Regra 3: Formatar com traços (parar)
%{
    calling_match: "^\\+1(\\d{3})(\\d{3})(\\d{4})$",
    calling_replace: "+1-\\1-\\2-\\3",
    called_match: "^\\+1(\\d{3})(\\d{3})(\\d{4})$",
    called_replace: "+1-\\1-\\2-\\3",
    priority: 15,
    description: "Formatar como +1-XXX-XXX-XXXX",
    enabled: true,
    continue: false
}

# Caso de teste:
# Entrada: "005551234567"
# Passo 1: "005551234567" → "5551234567" (Regra 1, continuar)
# Passo 2: "5551234567" → "+15551234567" (Regra 2, continuar)
# Passo 3: "+15551234567" → "+1-555-123-4567" (Regra 3, parar)
# Resultado: "+1-555-123-4567"
# Regras aplicadas: [Regra 1, Regra 2, Regra 3]

```

Suporte

Para problemas ou perguntas:

- Verifique a suíte de testes em `test/sms_c/messaging/number_translation_test.exe` para exemplos
- Use o simulador para depurar a lógica de tradução
- Revise logs de eventos para decisões de tradução
- Verifique o conteúdo da tabela Mnesia:
`:mnesia.table_info(:translation_rule, :size)`
- Monitore métricas de Telemetria para problemas de desempenho



Guia de Roteamento SMS-C

[← Voltar ao Índice da Documentação](#) | [README Principal](#)

Visão Geral

O sistema de Roteamento SMS-C fornece roteamento flexível e de alto desempenho de mensagens SMS com base em múltiplos critérios, incluindo prefixos de números, identificadores SMSC, tipos de conexão e mais. As rotas são armazenadas no Mnesia para persistência e podem ser modificadas em tempo de execução sem interrupção do serviço.

Principais Recursos

- **Roteamento baseado em prefixo:** Roteamento com base em prefixos de números chamadores/chamados com lógica de maior correspondência ganha
- **Roteamento baseado em SMSC:** Roteamento com base na SMSC de origem ou destino
- **Roteamento baseado em tipo:** Roteamento com base no tipo de conexão de origem (IMS, Circuit Switched, SMPP)
- **Roteamento baseado em prioridade:** Controle da ordem de seleção de rotas com prioridades configuráveis
- **Balanceamento de carga baseado em peso:** Distribuição de tráfego entre várias rotas usando pesos
- **Roteamento de resposta automática:** Envio automático de respostas de volta aos originadores das mensagens
- **Roteamento de descarte:** Descarte de mensagens que correspondem a critérios específicos (filtragem de spam, etc.)
- **Controle de cobrança:** Configuração do comportamento de cobrança por rota (Sim/Não/Padrão)
- **Carregamento de arquivo de configuração:** Carregamento de rotas iniciais a partir de `runtime.exs` na primeira inicialização
- **Configuração em tempo de execução:** Adicionar, modificar ou desabilitar rotas sem reiniciar
- **Interface Web:** Interface completa CRUD para gerenciamento de rotas com dropdown no frontend
- **Ferramenta de simulação:** Testar lógica de roteamento antes da implantação
- **Backup/Restauração:** Exportar e importar configurações de roteamento
- **Suporte a ENUM:** Pesquisa de número baseada em DNS (para implementação futura)

Arquitetura

Modelo de Dados

Cada rota contém os seguintes campos:

Campo	Tipo	Descrição	Obrigatório
route_id	inteiro	Identificador único auto-incrementado	Sim (auto)
calling_prefix	string/ nil	Correspondência de prefixo para número chamador (nil = curinga)	Não
called_prefix	string/ nil	Correspondência de prefixo para número chamado (nil = curinga)	Não
source_smsc	string/ nil	Nome da SMSC de origem (nil = curinga)	Não
dest_smsc	string/ nil	Nome da SMSC de destino (obrigatório, a menos que auto_reply ou drop seja verdadeiro)	Condicional
source_type	atom/ nil	Tipo de origem: :ims, :circuit_switched, :smpp, ou nil	Não
enum_domain	string/ nil	Domínio DNS ENUM para pesquisa	Não
auto_reply	boolean	Se verdadeiro, envia resposta de volta ao originador	Não (padrão: falso)
auto_reply_message	string/ nil	Texto da mensagem para resposta automática (obrigatório se auto_reply for verdadeiro)	Condicional
drop	boolean	Se verdadeiro, descarta a mensagem (filtragem de spam)	Não (padrão: falso)
charged	atom	Comportamento de cobrança: :yes, :no, ou :default	Não (padrão: :default)
weight	inteiro	Peso de balanceamento de carga (1-100, padrão 100)	Sim
priority	inteiro	Prioridade da rota (1-255, menor = maior prioridade)	Sim
description	string	Descrição legível por humanos	Não
enabled	boolean	Habilitar/desabilitar rota	Sim

Nota: Uma rota deve ser um dos três tipos:

1. **Roteamento normal:** auto_reply=false, drop=false, requer dest_smsc
2. **Resposta automática:** auto_reply=true, requer auto_reply_message
3. **Descarte:** drop=true, descarta a mensagem

Algoritmo de Roteamento

Ao roteamento de uma mensagem, o sistema segue esta ordem de prioridade:

PRIORIDADE 1: Roteamento Baseado em Localização (Mais Alto)

1. **Verificar registro do assinante:** Se o MSISDN de destino estiver registrado na tabela de locais
2. **Roteamento direto para frontend atendendo:** Ignorar todas as regras de roteamento e enviar diretamente para o frontend que atende esse assinante
3. **Isso acontece DEPOIS da tradução do número** para garantir consistência com os registros de localização

PRIORIDADE 2: Regras de Roteamento Padrão (se nenhum registro de localização encontrado)

1. **Filtra rotas habilitadas** que correspondem a TODOS os critérios especificados
2. **Classifica por especificidade** (rotas mais específicas primeiro):
 - Prefixo chamado mais longo = maior especificidade (×100 pontos)
 - Prefixo chamador mais longo = especificidade média (×50 pontos)
 - SMSC de origem especificada = +25 pontos
 - Domínio de resultado ENUM especificado = +15 pontos
 - Tipo de origem especificado = +10 pontos
 - Domínio ENUM especificado = +5 pontos
3. **Agrupar por prioridade** (número menor = maior prioridade)
4. **Seleciona do grupo de maior prioridade** usando seleção aleatória ponderada
5. **Executa ação de rota:**
 - **Rota normal:** Retorna SMSC de destino para entrega da mensagem
 - **Rota de resposta automática:** Envia resposta de volta ao originador de forma assíncrona
 - **Rota de descarte:** Descartar mensagem e registrar evento

Curingas

- nil ou valores vazios atuam como curingas que correspondem a qualquer valor
- Uma rota sem critérios especificados é uma rota catch-all

Configuração

Carregando Rotas do Arquivo de Configuração

As rotas podem ser definidas em `config/runtime.exs` e serão carregadas automaticamente na primeira inicialização. Isso é útil para definir regras de

roteamento básicas que devem estar presentes quando o sistema inicia pela primeira vez.

Importante: As rotas da configuração só são carregadas quando a tabela de roteamento está **vazia** (primeira inicialização). Isso preserva as rotas adicionadas via UI Web durante o tempo de execução e evita duplicatas em reinicializações.

Fluxo de Carregamento de Configuração

Estrutura de Configuração de Exemplo de Rota

Veja `config/runtime.exs` e `config/sms_routes.example.exs` para exemplos completos, incluindo:

- Roteamento geográfico
- Rotas de resposta automática
- Rotas de descarte (filtragem de spam)
- Rotas balanceadas por carga
- Roteamento de números premium com cobrança

Começando

Fluxo de Inicialização

Visão Geral dos Tipos de Rota

Fluxo de Roteamento de Mensagens

Casos de Uso Comuns

Roteamento Baseado em Localização (Maior Prioridade)

Roteie mensagens diretamente para o frontend que atende um assinante registrado, ignorando todas as regras de roteamento:

Como funciona:

1. A mensagem chega com o número de destino
2. Os números são traduzidos (se configurado)
3. O sistema verifica se o MSISDN de destino traduzido está na tabela de locais
4. Se registrado, a mensagem é roteada diretamente para o frontend que atende esse assinante
5. As regras de roteamento padrão são **completamente ignoradas**
6. Se não registrado, aplicam-se as regras de roteamento normais

Benefícios:

- **Entrega garantida** ao frontend correto para assinantes registrados
- **Roteamento mais rápido** - não é necessária avaliação da tabela de rotas
- **Roteamento preciso** - a localização do assinante é a fonte da verdade
- **Substitui todas as regras de roteamento** - garante a acessibilidade do assinante

Casos de uso:

- Assinantes IMS/VoLTE registrados em núcleos IMS específicos
- Assinantes móveis conectados a MSCs específicos
- Assinantes SIP registrados em servidores de aplicação específicos

Roteamento Geográfico

Roteie mensagens para SMSCs regionais com base no país de destino:

Balanceamento de Carga

Distribua o tráfego entre várias SMSCs com pesos:

Roteamento de Números Premium

Roteie números premium para tratamento especial com prioridade:

Roteamento Específico de Protocolo

Roteie com base no tipo de conexão de origem:

Migração de Rede

Durante a migração, roteie prefixos específicos para a nova infraestrutura:

Roteamento Multi-Critério Complexo

Combine múltiplos critérios para controle detalhado:

Interface Web

UI de Gerenciamento de Rotas

Acesse a interface de gerenciamento de rotas em `/sms_routing` (configure em seu roteador):

Recursos:

- Visualizar todas as rotas em uma tabela ordenável
- Adicionar novas rotas com validação de formulário
- Editar rotas existentes
- Habilitar/desabilitar rotas sem excluir
- Excluir rotas com confirmação
- Atualizações em tempo real (atualização a cada 5 segundos)

Adicionando uma Rota:

1. Clique em "Adicionar Nova Rota"
2. Preencha os campos do formulário (apenas SMSC de destino é obrigatório)
3. Defina o peso (1-100, padrão 100) e prioridade (1-255, padrão 100)
4. Marque "Habilitado" para ativar imediatamente
5. Clique em "Salvar Rota"

Editando uma Rota:

1. Clique em "Editar" ao lado da rota
2. Modifique os campos conforme necessário
3. Clique em "Salvar Rota"

Desabilitando uma Rota:

- Clique em "Desabilitar" para desativar temporariamente sem excluir
- Clique em "Habilitar" para reativar

Simulador de Roteamento

Acesse o simulador em `/simulator` (via menu de navegação):

Recursos:

- Testar lógica de roteamento com vários parâmetros
- **Avaliação detalhada campo a campo** mostrando por que cada rota correspondeu ou não
- Veja todas as rotas avaliadas em ordem de prioridade
- Indicadores visuais para rotas correspondentes/selecionadas
- Carregar cenários de exemplo para testes rápidos
- Visualizar histórico de testes (últimos 10 testes)

Usando o Simulador:

1. Insira os parâmetros de teste:
 - Número chamador (de)
 - Número chamado (para)
 - SMSC de origem (opcional)
 - Tipo de origem (Qualquer/IMS/Circuit Switched/SMPP)
2. Clique em "Simular Roteamento"
3. Veja os resultados abrangentes:

- **Resultado do Roteamento:** Rota selecionada e destino (ou "Nenhuma Rota Encontrada")
 - **Avaliação da Rota:** Todas as rotas com análise campo a campo:
 - ✓ Marca de verificação verde = Campo correspondente
 - ✗ X vermelho = Campo não correspondeu
 - Razão para a correspondência/não correspondência de cada campo
 - **Indicadores visuais:**
 - Borda verde + distintivo "SELECIONADO" = Rota realmente usada
 - Borda roxa + distintivo "CORRESPONDIDO" = Rotas que corresponderam mas não foram selecionadas
 - Borda cinza = Rotas que não corresponderam
4. Carregue exemplos pré-configurados usando os botões de exemplo
 5. Revise o histórico de testes para comparar diferentes cenários

Saída de Avaliação de Exemplo: Para cada rota, você verá por que ela correspondeu ou não:

- **Prefixo chamador:** "Corresponde ao prefixo '1234'" ou "Não começa com '44'"
- **Prefixo chamado:** "Curinga (corresponde a qualquer)" ou "Não começa com '639'"
- **SMSC de origem:** "Corresponde a 'smc1'" ou "Esperado 'untrusted_smc', obtido 'none'"
- **Tipo de origem:** "Curinga (corresponde a qualquer)" ou "Esperado 'smpp', obtido 'IMS'"

Referência da API

Visão Geral das Operações Principais

Operações de Gerenciamento de Rotas

Parâmetros de Roteamento de Mensagens

route_message aceita os seguintes parâmetros:

- **calling_number** (opcional): Número de telefone de origem
- **called_number** (opcional): Número de telefone de destino
- **source_smc** (opcional): Identificador da SMSC de origem
- **source_type** (opcional): Tipo de conexão (:ims, :circuit_switched, :smpp)
- **message_id** (opcional): Para registro de eventos

Retorna:

- `{:ok, dest_smsc, route}` - Rota encontrada e selecionada
- `{:error, :no_route_found}` - Nenhuma rota correspondente

Operações de Importação/Exportação

Melhores Práticas

Design de Rota

1. **Use prioridades sabiamente:** Reserve prioridades baixas (1-10) para rotas críticas
2. **Mantenha simples:** Comece com rotas amplas e adicione específicas conforme necessário
3. **Documente rotas:** Sempre adicione descrições às rotas
4. **Use catch-all:** Sempre tenha uma rota padrão com baixa prioridade

Desempenho

1. **Minimize a contagem de rotas:** Combine rotas semelhantes sempre que possível
2. **Use prefixos mais longos:** Prefixos mais específicos reduzem o tempo de avaliação
3. **Desabilite rotas não utilizadas:** Não exclua rotas que você pode precisar mais tarde; desative-as

Operações

1. **Teste antes de implantar:** Use o simulador para verificar a lógica de roteamento
2. **Faça backup regularmente:** Exporte rotas antes de fazer alterações significativas
3. **Monitore o roteamento:** Verifique os logs de eventos para decisões de roteamento
4. **Implantação gradual:** Use pesos para mudar gradualmente o tráfego para novas rotas

Testes

1. **Escreva testes de integração:** Teste seus cenários de roteamento específicos
2. **Teste de carga:** Verifique o desempenho do roteamento sob carga
3. **Teste de failover:** Garanta que rotas de backup funcionem quando as primárias falharem

Solução de Problemas

Nenhuma Rota Encontrada

Sintoma: `{:error, :no_route_found}` retornado

Possíveis causas:

- Nenhuma rota configurada
- Todas as rotas correspondentes estão desabilitadas
- Critérios da rota não correspondem aos parâmetros da mensagem
- Prefixo não corresponde (verifique se há erros de digitação)

Soluções:

1. Verifique se as rotas existem: `SmsRouting.list_enabled_routes()`
2. Use o simulador para testar o roteamento com parâmetros de mensagem reais
3. Adicione uma rota catch-all para depuração: `add_route(%{dest_smsc: "debug_smsc", priority: 255})`
4. Verifique os logs de eventos para detalhes da avaliação de roteamento

Rota Errada Selecionada

Sintoma: Mensagem roteada para destino inesperado

Possíveis causas:

- Configuração de prioridade incorreta
- Rota curinga tem prioridade mais alta
- Cálculo de especificidade favorece rota diferente
- Múltiplas rotas com os mesmos critérios usando pesos

Soluções:

1. Use o simulador para ver todas as rotas correspondentes
2. Verifique os valores de prioridade (menor = maior prioridade)
3. Verifique as pontuações de especificidade no simulador
4. Revise a distribuição de pesos para rotas balanceadas por carga

Problemas de Desempenho

Sintoma: Roteamento lento

Possíveis causas:

- Muitas rotas no banco de dados
- Padrões de rota complexos

- Tabela Mnesia não indexada corretamente

Soluções:

1. Consolide rotas semelhantes
2. Remova rotas desabilitadas que não são mais necessárias
3. Verifique se os índices do Mnesia estão criados (automático em `init_tables`)
4. Considere armazenar em cache decisões de roteamento frequentemente usadas

Tópicos Avançados

Integração ENUM/NAPTR

ENUM (E.164 Number Mapping) fornece pesquisa de número baseada em DNS usando registros NAPTR. O SMS-C inclui suporte completo a ENUM com cache, servidores DNS configuráveis e correspondência de rota com base nos resultados da pesquisa ENUM.

O que é ENUM?

ENUM mapeia números de telefone E.164 para nomes DNS usando uma transformação simples:

- **Número de Telefone:** +1-212-555-1234
- **Consulta ENUM:** 4.3.2.1.5.5.5.2.1.2.1.e164.arpa
- **Tipo de Registro DNS:** NAPTR (Ponteiro de Autoridade de Nome)
- **Resultado:** URI SIP, informações de roteamento ou outros dados de serviço

Configuração

A funcionalidade ENUM é configurada em `config/runtime.exs`:

Ativar Pesquisas ENUM:

Defina `enum_enabled`: `true` para ativar pesquisas ENUM antes do roteamento. Quando ativado, o sistema realizará pesquisas DNS ENUM para mensagens recebidas e usará os resultados nas decisões de roteamento.

Domínios ENUM:

Liste os domínios ENUM a serem consultados em ordem de prioridade. O sistema tentará cada domínio até que uma pesquisa bem-sucedida ocorra.

Domínios ENUM comuns:

- e164.arpa - Domínio ENUM oficial da IETF
- e164.org - Registro ENUM alternativo
- Domínios ENUM privados personalizados

Servidores DNS:

Configure servidores DNS específicos para consultas ENUM. Formato: {ip_address, port}

Deixe vazio ou defina como [] para usar servidores DNS padrão do sistema.

Exemplo de configuração DNS personalizada:

- DNS Público do Google: {"8.8.8.8", 53}, {"8.8.4.4", 53}
- DNS do Cloudflare: {"1.1.1.1", 53}, {"1.0.0.1", 53}
- DNS ENUM personalizado: {"10.0.0.53", 53}

Timeout:

Defina o tempo limite da consulta DNS em milissegundos (padrão: 5000ms). Aumente para redes lentas, diminua para failover mais rápido.

Como Funcionam as Pesquisas ENUM

Cache ENUM

O sistema armazena em cache os resultados da pesquisa ENUM por 15 minutos para melhorar o desempenho e reduzir a carga DNS.

Benefícios do Cache:

- Reduz a carga de consulta DNS
- Melhora a latência de roteamento
- Protege contra falhas do servidor DNS (resultados em cache permanecem disponíveis)

Estatísticas do Cache:

- Visualize o tamanho e o status do cache na página de Teste NAPTR
- Monitore taxas de acerto/miss do cache via métricas do Prometheus
- Limpe o cache manualmente, se necessário (alterações de configuração, testes, etc.)

Comportamento do Cache:

- Tanto pesquisas bem-sucedidas quanto falhadas são armazenadas em cache
- Pesquisas falhadas são armazenadas em cache para evitar consultas repetidas para números inválidos

- O cache expira automaticamente após 15 minutos
- O cache sobrevive a reinicializações da aplicação (armazenado em ETS)

Usando ENUM em Rotas

As rotas podem corresponder aos resultados da pesquisa ENUM usando o campo `enum_result_domain`:

Cenário de Exemplo:

A pesquisa ENUM para +1-555-0100 retorna o registro NAPTR:

- Serviço: E2U+sip
- Substituição: sip:customer@voip-carrier.com
- **Domínio de Resultado:** voip-carrier.com

Configuração da Rota:

Crie uma rota com `enum_result_domain: "voip-carrier.com"` para corresponder a mensagens onde a pesquisa ENUM retornou esse domínio.

Lógica de Correspondência:

- Se a rota tiver `enum_result_domain: nil` - corresponde a todas as mensagens (curinga)
- Se a rota tiver `enum_result_domain: "specific.com"` - corresponde apenas se ENUM retornar esse domínio
- Rotas com domínios ENUM correspondentes recebem pontuações de especificidade mais altas

Cálculo de Prioridade:

Rotas com domínios de resultado ENUM recebem +15 pontos de especificidade, priorizando-as sobre rotas genéricas.

Testando Pesquisas ENUM

Acesse a página de Teste NAPTR em `/naptr_test` (via menu de navegação).

Recursos:

- Realizar pesquisas ENUM ao vivo contra servidores DNS configurados
- Visualizar informações detalhadas do registro NAPTR
- Ver domínios de resultado extraídos dos registros NAPTR
- Monitorar estatísticas do cache
- Limpar o cache para testes

Fluxo de Teste:

1. Insira um número de telefone (com ou sem prefixo +)
2. Especifique o domínio ENUM (padrão: e164.arpa)
3. Clique em "Realizar Pesquisa"
4. Revise os resultados:
 - Registros NAPTR encontrados
 - Valores de ordem e preferência
 - Tipos de serviço (E2U+sip, E2U+tel, etc.)
 - Expressões regulares
 - Valores de substituição
 - **Domínios de resultado extraídos** (usados para correspondência de rota)

Exibição da Configuração Atual:

- Servidores DNS sendo usados (ou "Padrão do Sistema")
- Configuração de tempo limite
- Tamanho e status do cache
- Botão para limpar o cache

Entendendo os Resultados:

Cada registro NAPTR contém:

- **Ordem:** Prioridade para processamento (menor primeiro)
- **Preferência:** Dentro da mesma ordem (menor primeiro)
- **Flags:** Instruções de processamento (u=terminal, s=continuar)
- **Serviço:** Tipo de serviço (E2U+sip, E2U+tel, etc.)
- **Regexp:** Expressão de substituição
- **Substituição:** Domínio ou endereço alternativo
- **Domínio de Resultado:** Domínio extraído para correspondência de rota

Casos de Uso Comuns de ENUM

1. Peering VoIP

Use ENUM para identificar números hospedados em redes SIP/VoIP e roteá-los diretamente para gateways VoIP:

- ENUM retorna URI SIP: sip:number@voip-carrier.com
- Domínio de resultado: voip-carrier.com
- Rota com enum_result_domain: "voip-carrier.com" selecionada
- Tráfego enviado para o gateway de peering VoIP direto

2. Identificação de Operadora

Identifique a operadora que atende a um número e roteie de acordo:

- ENUM retorna informações da operadora

- Domínio de resultado: carrier-a.com
- Roteie para a interconexão da operadora A
- Otimize custos e qualidade de roteamento

3. Portabilidade de Números

Lide com números portados que se moveram entre operadoras:

- Pesquisa ENUM retorna a operadora atual
- Roteie automaticamente para o destino correto
- Nenhuma atualização manual da tabela de roteamento necessária

4. Roteamento de Menor Custo

Combine ENUM com várias rotas:

- ENUM identifica a rede de destino
- Múltiplas rotas para o mesmo domínio com custos diferentes
- Use prioridade e pesos para preferir rotas de menor custo

5. Serviços de Emergência

Roteie números de emergência (911, 112, etc.) para os serviços de emergência adequados:

- Pesquisa ENUM identifica o gateway de emergência local
- Rota de alta prioridade garante roteamento imediato
- Sem atraso da avaliação de rota normal

Estratégia de Roteamento ENUM

Configuração Recomendada:

1. Rotas ENUM de Alta Prioridade (Prioridade 1-10)

- Rotas que correspondem a domínios de resultado ENUM específicos
- Usadas para peering direto, roteamento VoIP
- Maior especificidade, selecionadas primeiro

2. Rotas de Prefixo de Prioridade Média (Prioridade 50-100)

- Roteamento padrão baseado em prefixo
- Usadas quando a pesquisa ENUM falha ou não retorna registros
- Fallback confiável

3. Catch-All de Baixa Prioridade (Prioridade 200+)

- Rota padrão para tudo o mais
- Garante que nenhuma mensagem fique sem roteamento

Hierarquia de Rotas de Exemplo:

- Prioridade 1: enum_result_domain: "sip.carrier.com" → Gateway VoIP direto
- Prioridade 10: enum_result_domain: "tel.carrier.com" → Gateway PSTN da operadora
- Prioridade 50: called_prefix: "+1" → Gateway padrão da América do Norte
- Prioridade 100: called_prefix: "+" → Gateway padrão internacional
- Prioridade 200: Sem critérios → Fallback final

Considerações de Desempenho

Latência de Consulta DNS:

Pesquisas ENUM adicionam tempo de consulta DNS ao roteamento:

- **Em cache:** < 1ms (rápido)
- **Sem cache:** 10-100ms (depende do servidor DNS)

Recomendações:

- Use servidores DNS geograficamente próximos
- Configure o tempo limite apropriado (5000ms padrão)
- Monitore taxas de acerto de cache (meta > 80%)
- Considere aquecer o cache para números conhecidos

Escalabilidade:

O sistema de cache lida com cenários de alto volume:

- O cache é compartilhado entre todos os processos
- Tabela ETS de leitura concorrente para desempenho
- Limpeza automática do cache via TTL
- Escala para milhões de entradas em cache

Tratamento de Falhas:

Falhas ENUM retornam graciosamente para o roteamento regular:

- Timeout DNS → Passar para a próxima rota
- Sem registros NAPTR → Usar rotas baseadas em prefixo
- Formato NAPTR inválido → Registrar erro, continuar roteamento
- Servidor DNS indisponível → Usar resultados em cache ou fallback

Monitoramento de Operações ENUM

Use métricas do Prometheus para monitorar o desempenho ENUM:

- sms_c_enum_lookup_stop_duration - Latência de pesquisa
- sms_c_enum_cache_hit_count - Acertos de cache
- sms_c_enum_cache_miss_count - Falhas de cache
- sms_c_enum_cache_size_size - Tamanho atual do cache
- sms_c_enum_naptr_records_record_count - Registros NAPTR por pesquisa

Métricas Chave a Observar:

- **Taxa de acerto do cache:** Deve ser > 70% após aquecimento
- **Duração da pesquisa p95:** Deve ser < 1000ms
- **Pesquisas falhadas:** Monitore problemas de DNS

Veja docs/METRICS.md para documentação completa de métricas.

Solução de Problemas ENUM

Problema: Nenhum Registro NAPTR Encontrado

- Verifique a configuração do domínio ENUM
- Teste a conectividade do servidor DNS
- Verifique se o número está realmente no registro ENUM
- Tente um domínio ENUM alternativo (por exemplo, e164.org)
- Use a página de Teste NAPTR para diagnosticar

Problema: Pesquisas ENUM Lentas

- Verifique a latência do servidor DNS
- Verifique a conectividade da rede
- Aumente o tempo limite se necessário
- Considere usar servidores DNS mais próximos
- Verifique a taxa de acerto do cache

Problema: Rota Errada Selecionada Após ENUM

- Verifique o campo enum_result_domain nas rotas
- Use o Simulador de Rotas para testar a lógica de roteamento
- Verifique se a extração do domínio de resultado está correta
- Revise o formato do registro NAPTR na página de Teste

Problema: Pesquisas ENUM Desabilitadas

- Verifique enum_enabled: true em config/runtime.exs
- Verifique se a lista de enum_domains não está vazia
- Reinicie a aplicação após alterações de configuração
- Verifique os logs da aplicação para inicialização ENUM

Considerações de Segurança

Envenenamento de Cache DNS:

- Use apenas servidores DNS confiáveis
- Considere DNSSEC se disponível
- Valide formatos de registros NAPTR
- Monitore domínios de resultado inesperados

Exaustão de Recursos:

- Limites de cache evitam exaustão de memória
- Timeout evita travar em DNS lentos
- Pesquisas falhadas armazenadas em cache para evitar tempestades de tentativas

Divulgação de Informações:

- Pesquisas ENUM revelam intenções de roteamento para servidores DNS
- Use servidores DNS privados para roteamento sensível
- Considere VPN/DNS criptografado para privacidade

Registro de Eventos

Decisões de roteamento são registradas via EventLogger:

- sms_routing_started: Avaliação de roteamento começa
- sms_routing_candidates: Número de rotas habilitadas encontradas
- sms_routing_matches: Número de rotas correspondentes
- sms_routing_selected: Detalhes da rota selecionada
- sms_routing_failed: Nenhuma rota encontrada

Ative o registro passando message_id para route_message/1.

Clusterização

As tabelas Mnesia são distribuídas automaticamente entre nós em cluster. As rotas são replicadas para alta disponibilidade.

Exemplos

Veja a suíte de testes em test/sms_c/messaging/sms_routing_test.exs para exemplos abrangentes de:

- Correspondência de prefixo
- Roteamento baseado em prioridade
- Balanceamento de carga baseado em peso
- Roteamento multi-critérios

- Casos extremos

Migração do Antigo Roteamento

Se você estiver migrando do antigo roteamento baseado em configuração, siga este processo:

Detalhes dos Passos de Migração

1. Inicializar Tabelas

- Cria tabelas de roteamento Mnesia
- Prepara o sistema para novo roteamento

2. Analisar Rotas Antigas

- **Padrões de Regex** → Rotas baseadas em prefixo
- **Respostas pré-definidas** → Rotas de resposta automática
- **Lógica personalizada** → Rotas multi-critérios

3. Testar Minuciosamente

- Use o simulador de roteamento
- Verifique todos os cenários
- Verifique casos extremos

4. Atualizar Código

- Substitua chamadas de roteamento antigas
- Use a API `route_message/1`
- Atualize o tratamento de erros

5. Implantar e Monitorar

- Implemente o novo sistema de roteamento
- Monitore por problemas
- Mantenha a configuração antiga como backup inicialmente

6. Limpar

- Remova a configuração de roteamento antiga
- Remova o código de migração
- Atualize a documentação

Suporte

Para problemas ou perguntas:

- Verifique a suíte de testes para exemplos
- Use o simulador para depurar a lógica de roteamento
- Revise os logs de eventos para decisões de roteamento
- Verifique o conteúdo da tabela Mnesia: `:mnesia.table_info(:sms_route, :size)`