Guia da API REST

← Voltar para a Documentação Principal

Este guia fornece documentação abrangente para a **API REST** e **Swagger UI** do OmniSS7.

Índice

1. Visão Geral
2. Configuração do Servidor HTTP
3. Swagger UI
4. Endpoints da API
5. Exemplos Práticos
6. Autenticação
7. Formatos de Resposta
8. Tratamento de Erros
9. Métricas (Prometheus)
10. Exemplos de Requisições

Visão Geral

OmniSS7 expõe uma API REST na **porta 9090** para conduzir operações MAP (Mobile Application Part) e CAP (CAMEL Application Part) sobre a rede SS7, além de injeção bruta de SCCP/M3UA para testes. A API define **47 endpoints de operação** `POST /api/*` juntamente com os endpoints de métricas e Swagger. Ela permite que você:

- Dispare requisi��ões MAP (mobilidade, SMS, autenticação, dados do assinante, serviços suplementares, serviços de localização, GPRS, equipamentos)
- Conduza diálogos de controle de chamada CAP/CAMEL

- Injete cargas úteis brutas de SCCP/M3UA
- Monitore métricas do sistema via Prometheus

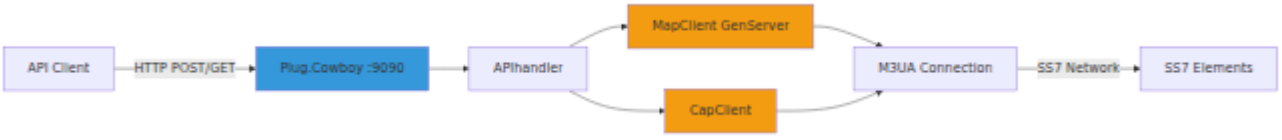
Para esquemas de requisição/resposta por operação e opcodes MAP, este guia faz referência cruzada ao **Guia do Cliente MAP**, que documenta cada operação em detalhes.

Portas e Transportes

OmniSS7 executa **três servidores HTTP distintos**. Este guia cobre o primeiro (:9090). Os outros estão documentados na Referência de Configuração.

Porta	Transporte	Servidor	Escopo deste guia
9090	HTTP (`Plug.Cowboy`)	`APIhandler`	**Sim.** MAP/CAP REST `/api/*`, `/metrics`, `/swagger`.
8087	HTTPS (`Bandit`, Phoenix)	`ControlPanelWeb.Endpoint`	Não: interface da web do painel de controle. Veja o Guia da Interface Web.
8445	HTTPS / TLS	`:api_ex` controladores REST	Não: status/routing/links REST. Veja a Referência de Configuração.

Arquitetura da API



Configuração do Servidor HTTP

Detalhes do Servidor

Os endpoints `/api/*`, `/metrics`, e `/swagger` são servidos por um ouvinte HTTP `Plug.Cowboy`.

Parâmetro	Valor	Configurável
Protocolo	HTTP	Não
Endereço IP	0.0.0.0 (todas as interfaces)	Apenas via código
Porta	9090	Sim: `config :omniss7, api_port: <port>` (padrão `9090`)
Transporte	`Plug.Cowboy`	Não

URL de Acesso: `http://[server-ip]:9090`

> O painel de controle (:8087, Bandit) e os controladores REST `:api_ex` (:8445) são servidores separados. Veja Portas e Transportes.

Habilitando/Desabilitando o Servidor HTTP

Controle se o servidor HTTP :9090 inicia:

```
config :omniss7,
  start_http_server: true  # Defina como false para desabilitar
```

Padrão: `true` (habilitado)

Quando Desabilitado: O servidor HTTP :9090 não será iniciado, e a API REST / Swagger UI / endpoint de métricas estarão indisponíveis. Os servidores do

painel de controle (:8087) e `:api_ex` (:8445) não são afetados (eles são controlados por `start_web_interfaces`).

Swagger UI

A API inclui um **Swagger UI** para documentação e testes interativos da API.

Acessando o Swagger UI

URL: `http://[server-ip]:9090/swagger`

Recursos:

- Documentação interativa da API
- Funcionalidade de teste para endpoints
- Esquemas de requisição/resposta
- Cargas úteis de exemplo

Swagger JSON

A especificação OpenAPI está disponível em:

URL: `http://[server-ip]:9090/swagger.json`

Casos de Uso:

- Importar para Postman ou outros clientes de API
- Gerar bibliotecas de cliente
- Automação da documentação da API

Endpoints da API

Cada endpoint de operação segue o padrão `POST /api/{operation}` e aceita um corpo de requisição JSON. As requisições são **síncronas**: o manipulador

bloqueia até que a resposta SS7 chegue ou a operação expire (padrão **10 segundos** para operações MAP aguardadas; veja Tratamento de Erros).

As tabelas abaixo listam todos os **47** endpoints de operação agrupados por categoria. Para cada campo de requisição, esquema de resposta e opcode MAP da operação, veja a seção correspondente do **Guia do Cliente MAP**, que foi atualizado juntamente com este guia e documenta cada operação em detalhes.

> **Códigos de resposta (todos os endpoints):** `200` = sucesso, `400` = requisição ruim/inválida (campos ausentes ou com tipo errado), `504` = timeout (sem resposta SS7 dentro do tempo limite da operação), `500` = erro de operação (não timeout), `503` = recurso não habilitado (USSD), `404` = caminho desconhecido. Veja Tratamento de Erros.

Gerenciamento de Mobilidade

Operação	Endpoint	Descrição
Atualizar Localização	`POST /api/updateLocation`	Registrar o VLR de atendimento de um assinante com o HLR. Em modo HLR, isso aciona a sequência InsertSubscriberData (ISD).
Cancelar Localização	`POST /api/cancelLocation`	Instruir um VLR a excluir um registro de assinante.
Fornecer Número de Roaming	`POST /api/prn`	Obter um MSRN do MSC de atendimento.
Purgar MS	`POST /api/purgeMS`	Marcar um assinante como purgado no HLR.
Enviar Identificação	`POST /api/sendIdentification`	Recuperar dados de IMSI/autenticação do VLR anterior (TMSI fornecido como hex).
Resetar	`POST /api/reset`	Reset iniciado pelo HLR em direção a um VLR (não confirmado: retorna `{"status":"sent"}`).
Restaurar Dados	`POST /api/restoreData`	O VLR solicita a restauração de dados do assinante do HLR.

SMS

Operação	Endpoint	Descrição
Enviar Informação de Roteamento para SM	`POST /api/sri-for-sm`	Consultar o HLR pelo nó de atendimento para entrega de SMS.
MT-Encaminhar SM	`POST /api/MT-forwardSM`	Entregar um SM terminado em móvel ao MSC/SGSN de atendimento (`smsPDU` como hex).
MO-Encaminhar SM	`POST /api/forwardSM`	Enviar um SM originado em móvel (`smsPDU` como hex).
Enviar SM	`POST /api/sendSM`	Construir e entregar um SMS-DELIVER a partir de texto GSM-7.
Construir PDU de Entrega	`POST /api/deliverPDU`	Construir um PDU SMS-DELIVER e retornar o hex (sem envio SS7).
Relatar Status de Entrega de SM	`POST /api/reportSM-DeliveryStatus`	Relatar o resultado da entrega de SM ao HLR.
Pronto para SM	`POST /api/readyForSM`	Notificar o HLR que um MS está acessível para SMS.
Alertar Centro de Serviço	`POST /api/alertServiceCentre`	Alertar um SMSC que um assinante está

Operação	Endpoint	Descrição
		disponível.

Autenticação

Operação	Endpoint	Descrição
Enviar Informação de Autenticação	`POST /api/send-auth-info`	Recuperar vetores de autenticação do HLR.
Enviar IMSI	`POST /api/sendIMSI`	Resolver um MSISDN para seu IMSI no HLR.

Dados do Assinante

Operação	Endpoint	Descrição
Enviar Informação de Roteamento	`POST /api/sri`	Consultar o HLR p informações de roteamento de chamadas de voz
Fornecer Informação do Assinante	`POST /api/provideSubscriberInfo`	Solicitar estado/localizaçã assinante ao VLR
Excluir Dados do Assinante	`POST /api/deleteSubscriberData`	Remover dados d assinante no VLR `withdraw` opcion
Interrogação a Qualquer Momento	`POST /api/anyTimeInterrogation`	gsmSCF consulta informações/loca do assinante no H
Interrogação de Assinatura a Qualquer Momento	`POST /api/anyTimeSubscriptionInterrogation`	Consultar dados assinatura no HLI
Modificação a Qualquer Momento	`POST /api/anyTimeModification`	Modificar dados d assinatura no HLI
Notar Dados do Assinante Modificados	`POST /api/noteSubscriberDataModified`	HLR notifica o gsı sobre dados alter

Operação	Endpoint	Descrição
Notar Evento MM	`POST /api/noteMM-Event`	Relatar um event gerenciamento d mobilidade ao gs

Serviços Suplementares

Operação	Endpoint	Descrição
Interrogar SS	`POST /api/interrogateSS`	Interrogar o status de um serviço suplementar.
Registrar SS	`POST /api/registerSS`	Registrar um serviço suplementar (ex: desvio de chamada / CFU).
Apagar SS	`POST /api/eraseSS`	Apagar um serviço suplementar.
Ativar SS	`POST /api/activateSS`	Ativar um serviço suplementar.
Desativar SS	`POST /api/deactivateSS`	Desativar um serviço suplementar.
Registrar Senha	`POST /api/registerPassword`	Registrar uma senha de SS (`ss_code` inteiro).
Obter Senha	`POST /api/getPassword`	Recuperar/verificar uma senha de SS.
Encaminhar Verificação de Indicação de SS	`POST /api/forwardCheckSS-Indication`	Verificação não confirmada em direção a um VLR (retorna `{"status":"sent"}`).

As cinco operações principais de SS (`interrogateSS`, `registerSS`, `eraseSS`, `activateSS`, `deactivateSS`) compartilham uma forma de requisição: `ss_code` (inteiro) e `hlr_gt`, com `forwarded_to`, `no_reply_time`, e `basic_service` opcionais.

Serviços de Localização (LCS)

Operação	Endpoint	Descrição
Enviar Informação de Roteamento para LCS	`POST /api/sendRoutingInfoForLCS`	GMLC consulta o HLR para roteamento de LCS.
Fornecer Localização do Assinante	`POST /api/provideSubscriberLocation`	Solicitar a localização de um assinante do nó de atendimento.
Relatório de Localização do Assinante	`POST /api/subscriberLocationReport`	Relatar a localização de um assinante ao GMLC.

GPRS

Operação	Endpoint	Descrição
Atualizar Localização GPRS	`POST /api/updateGprsLocation`	Registrar o SGSN de atendimento de um assinante no HLR.
Enviar Informação de Roteamento para GPRS	`POST /api/sendRoutingInfoForGprs`	Consultar o HLR para roteamento GPRS (seleção de GGSN).

Equipamento

Operação	Endpoint	Descrição
Verificar IMEI	`POST /api/checkIMEI`	Consultar o status do equipamento no EIR.

USSD

Operação	Endpoint	Descrição
USSD Enviar (originado pela rede)	`POST /api/ussd/send`	Enviar uma mensagem USSD para um assinante. Requer `ussd_gateway_enabled: true` (caso contrário, `503`). Veja o Guia do Gateway USSD.

CAP / CAMEL

Operação	Endpoint	Descrição
CAP InitialDP	`POST /api/cap/initialDP`	Acionar um diálogo CAMEL em direção a um gsmSCF. Retorna um `otid` para operações de acompanhamento.
CAP Connect	`POST /api/cap/connect`	Roteamento da chamada para um destino (`destination_number` + `otid`).
CAP Continue	`POST /api/cap/continue`	Continuar o processamento da chamada (`otid`).
CAP Release Call	`POST /api/cap/releaseCall`	Liberar a chamada com uma `cause` (`otid`).
CAP Apply Charging	`POST /api/cap/applyCharging`	Aplicar uma janela de cobrança (`duration` + `otid`).

As operações de acompanhamento CAP carregam o diálogo `otid` (hex, retornado por `initialDP`) e respondem com `{"status":"sent","otid":"<hex>"}`. Veja o Guia do Gateway CAMEL.

Injeção Bruta (teste / diagnósticos)

Operação	Endpoint	Descrição
Raw SCCP	`POST /api/raw/sccp`	Injetar uma carga útil SCCP construída à mão (hex). Sobrescritas SCCP opcionais (partido chamador/chamado, SSNs, OPC/DPC).
Raw M3UA	`POST /api/raw/m3ua`	Injetar uma carga útil M3UA construída à mão (hex). Sobrescritas OPC/DPC/contexto de roteamento opcionais.

Endpoints Utilitários

Endpoint	Método	Propósito
`/metrics`	GET	Métricas Prometheus (formato texto).
`/swagger`	GET	Swagger UI.
`/swagger.json`	GET	Especificação OpenAPI.

Exemplos Práticos

Os seguintes exemplos ilustram a forma de requisição/resposta para operações representativas. Para a referência completa de campos de cada operação, veja o Guia do Cliente MAP.

SendRoutingInfo (SRI)

Recuperar informações de roteamento para estabelecer uma chamada a um assinante móvel.

Endpoint: `POST /api/sri`

Corpo da Requisição:

```
{
  "msisdn": "1234567890",
  "gmsc": "5551234567"
}
```

Campo	Tipo	Obrigatório	Descrição
`msisdn`	String	Sim	MSISDN da parte chamada
`gmsc`	String	Sim	Título Global do MSC

Resposta (200 OK):

```
{
  "result": {
    "imsi": "001001234567890",
    "msrn": "5551234999",
    "vlr_number": "5551234800"
  }
}
```

Exemplo cURL:

```
curl -X POST http://localhost:9090/api/sri \
  -H "Content-Type: application/json" \
  -d '{"msisdn": "1234567890", "gmsc": "5551234567"}'
```

SendRoutingInfoForSM (SRI-for-SM)

Recuperar informações de roteamento para entregar um SMS a um assinante móvel.

Endpoint: `POST /api/sri-for-sm`

Corpo da Requisição:

```
{
  "msisdn": "1234567890",
  "service_center": "5551234567"
}
```

Campo	Tipo	Obrigatório	Descrição
`msisdn`	String	Sim	MSISDN de destino
`service_center`	String	Sim	Título Global do Centro de Serviço

Exemplo cURL:

```
curl -X POST http://localhost:9090/api/sri-for-sm \
  -H "Content-Type: application/json" \
  -d '{"msisdn": "1234567890", "service_center": "5551234567"}'
```

UpdateLocation

Notificar o HLR sobre uma mudança de localização de um assinante (registro VLR). Em modo HLR, isso aciona a sequência InsertSubscriberData (ISD).

Endpoint: `POST /api/updateLocation`

Corpo da Requisição:

```
{
  "imsi": "001001234567890",
  "vlr": "5551234800"
}
```

Campo	Tipo	Obrigatório	Descrição
`imsi`	String	Sim	IMSI do assinante
`vlr`	String	Sim	Endereço do Título Global do VLR

USSD Send (Originado pela Rede)

Enviar uma mensagem USSD para um assinante. Requer `ussd_gateway_enabled: true`. Para o protocolo completo de callback e ciclo de vida da sessão, veja o Guia do Gateway USSD.

Endpoint: `POST /api/ussd/send`

Corpo da Requisição:

```
{
  "msisdn": "+254712345678",
  "text": "Você tem uma fatura pendente. Responda 1 para pagar.",
  "callback_url": "http://billing-app:9000/ussd"
}
```

Campo	Tipo	Obrigatório	Descrição
`msisdn`	String	Sim	MSISDN do assinante de destino
`text`	String	Sim	Texto USSD a ser exibido (GSM 7-bit)
`callback_url`	String	Sim	URL para receber respostas do assinante via HTTP POST

Resposta (200 OK):

```
{
  "session_id": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
  "status": "sent"
}
```

Erro (503, gateway desativado):

```
{
  "error": "USSD gateway not enabled"
}
```

Autenticação

Status Atual: A API **não requer autenticação**.

Considerações de Segurança:

- A API é destinada ao uso em rede interna/confiável
- Considere usar regras de firewall para restringir o acesso
- Para implantações em produção, considere implementar middleware de autenticação

---

Formatos de Resposta

Todas as respostas usam o formato **JSON**.

Resposta de Sucesso

Status HTTP: 200 OK

Estrutura:

```
{
  "result": {
    // Dados de resposta específicos da operação
  }
}
```

Resposta de Erro

Status HTTP:

- 400 Bad Request - Campos de requisição inválidos ou ausentes
- 500 Internal Server Error - Operação falhou (erro não timeout retornado pela pilha SS7)
- 503 Service Unavailable - Recurso não habilitado (gateway USSD)
- 504 Gateway Timeout - Operação excedeu seu timeout (padrão 10 segundos)
- 404 Not Found - Endpoint inválido

Estrutura:

```
{
  "error": "timeout"
}
```

ou

```
{
  "error": "invalid request"
}
```

Tratamento de Erros

Erros Comuns

Erro	Código HTTP	Descrição	Solução
JSON Inválido	400	O corpo da requisição não é um JSON válido	Verifique a sintaxe do JSON
Campos ausentes/inválidos	400	Campos obrigatórios ausentes ou tipo errado (as respostas geralmente incluem uma lista `"required"`)	Inclua todos os parâmetros obrigatórios com os tipos corretos
Erro de operação	500	A pilha SS7 retornou um erro não timeout	Inspecione a mensagem `"error"`; verifique os logs de eventos SS7
Recurso desativado	503	Gateway USSD não habilitado	Defina `ussd_gateway_enabled: true`
Timeout	504	A operação excedeu seu timeout (padrão 10s)	Verifique a conectividade M3UA, disponibilidade do HLR/VLR

Erro	Código HTTP	Descrição	Solução
Não Encontrado	404	Endpoint inválido	Verifique a URL do endpoint

Comportamento de Timeout

Operações aguardadas usam um **timeout de 10 segundos** por padrão:

1. Requisi��ão enviada ao MapClient GenServer
2. Aguarda resposta por até 10 segundos
3. Se não houver resposta → retorna 504 Gateway Timeout
4. Se a resposta for recebida → retorna 200 OK com resultado

Solução de Problemas de Timeouts:

- Verifique o status da conexão M3UA (Interface Web → página M3UA)
- Verifique se o elemento da rede (HLR/VLR/MSC) está acessível
- Verifique a configuração de roteamento
- Revise os logs de eventos SS7 em busca de erros

Métricas (Prometheus)

A API expõe métricas Prometheus para monitoramento.

Endpoint de Métricas

URL: `http://[server-ip]:9090/metrics`

Formato: Formato de texto Prometheus

Exemplo de Saída:

```
# HELP map_requests_total Total de requisições MAP
# TYPE map_requests_total counter
map_requests_total{operation="sri"} 42
map_requests_total{operation="sri_for_sm"} 158
map_requests_total{operation="updateLocation"} 23

# HELP cap_requests_total Total de requisições CAP
# TYPE cap_requests_total counter
cap_requests_total{operation="initialDP"} 87
cap_requests_total{operation="requestReportBCSMEvent"} 91

# HELP map_request_duration_milliseconds Duração das
requisições/respostas MAP em ms
# TYPE map_request_duration_milliseconds histogram
map_request_duration_milliseconds_bucket{operation="sri",le="10"}
5
map_request_duration_milliseconds_bucket{operation="sri",le="50"}
12
map_request_duration_milliseconds_bucket{operation="sri",le="100"}
35
...

# HELP map_pending_requests Número de requisições MAP TID
pendentes
# TYPE map_pending_requests gauge
map_pending_requests 3

# HELP omniss7_license_status Status atual da licença (1 = válido,
0 = inválido)
# TYPE omniss7_license_status gauge
omniss7_license_status 1
```

Métricas Disponíveis

As métricas são agrupadas por camada de protocolo / subsistema. Todas são declaradas na inicialização e emitidas com ajudantes protegidos, de modo que uma métrica não registrada nunca interrompa um caminho quente de protocolo.

Aplicação (MAP / CAP / USSD / licença)

Métrica	Tipo	Rótulos	Descr
`map_requests_total`	Contador	`operation`	Total de requisiçõ MAP por de opera
`map_request_errors_total`	Contador	`operation`	Total de de requis MAP por de opera
`cap_requests_total`	Contador	`operation`	Total de requisiçõ CAP por de opera
`cap_request_errors_total`	Contador	`operation`	Total de de requis CAP por de opera
`map_request_duration_milliseconds`	Histograma	`operation`	Duraç requisiçã milisseg
`map_pending_requests`	Gauge	-	Número transaçõ MAP pendent
`ussd_requests_total`	Contador	`direction`	Total de requisiçõ USSD (entrada

Métrica	Tipo	Rótulos	Descr
`ussd_active_sessions`	Gauge	-	Número sessões ativas
`omniss7_license_status`	Gauge	-	Status at licença (válido, 0 inválido)

SCCP

Métrica	Tipo	Rótulos	
`sccp_messages_total`	Contador	`message_type`, `direction`	Mensage (`udt`/`xud` e direção
`sccp_decode_errors_total`	Contador	-	Mensage decodific
`sccp_segments_received_total`	Contador	-	Segment remontag
`sccp_reassembly_completed_total`	Contador	-	Mensage remontad
`sccp_reassembly_timeouts_total`	Contador	-	Buffers d temporiz

TCAP

Métrica	Tipo	Rótulos	
`tcap_dialogues_total`	Contador	`type`	Diálogos TCAP por tipo (
`tcap_aborts_total`	Contador	`reason`	Mensagens de abortos T `unrecognized_message_` (ou `p_abort_other`); U-
`tcap_decode_errors_total`	Contador	-	Mensagens TCAP de um
`tcap_active_dialogues`	Gauge	-	Transações MAP atualm

SCTP

Métrica	Tipo	Rótulos	
`sctp_association_state`	Gauge	`remote_ip`, `remote_port`	Estad inativ
`sctp_association_state_changes_total`	Contador	`event`	Trans (`comm`
`sctp_send_errors_total`	Contador	`reason`	Envio M2PA
`sctp_heartbeat_sent_total`	Contador	`type`	Batim (`beat`
`sctp_heartbeat_received_total`	Contador	`type`	Batim (`beat`

SMS / SMSc

Métrica	Tipo	Rótulos	Descrição
`sms_messages_total`	Contador	`direction`, `operation`	SMS por direção (`mo`/`mt`) e operação (`submit`/`deliver`
`sms_delivery_failures_total`	Contador	`reason`	Falhas de entrega de SMS por razão
`sms_tracked_subscribers`	Gauge	-	Assinantes atualmente rastreados pelo SubscriberTracke

M2PA (Adaptação MTP2 Peer-to-Peer)

Métrica	Tipo	Rótulos	Descrição
`m2pa_messages_sent_total`	Contador	`link_name`, `point_code`, `adjacent_pc`	Mensagens MTP3 enviadas sobre M2PA
`m2pa_messages_received_total`	Contador	`link_name`, `point_code`, `adjacent_pc`	Mensagens MTP3 recebidas sobre M2PA
`m2pa_bytes_sent_total`	Contador	`link_name`, `point_code`, `adjacent_pc`	Bytes enviados sobre M2PA
`m2pa_bytes_received_total`	Contador	`link_name`, `point_code`, `adjacent_pc`	Bytes recebidos sobre M2PA
`m2pa_link_state_changes_total`	Contador	`link_name`, `from_state`, `to_state`	Transições de estado do link M2PA
`m2pa_errors_total`	Contador	`link_name`, `error_type`	Erros M2PA por tipo

M3UA STP

Métrica	Tipo	Rótulos	Des
`m3ua_stp_messages_received_total`	Contador	`peer_name`, `point_code`	Mensagens recebidas p
`m3ua_stp_messages_sent_total`	Contador	`peer_name`, `point_code`	Mensagens enviadas p
`m3ua_stp_routing_failures_total`	Contador	`reason`	Falhas de r STP (`no_route`/

Configuração do Prometheus

Adicione ao seu `prometheus.yml`:

```yaml
scrape_configs:
  - job_name: 'omniss7'
    static_configs:
      - targets: ['server-ip:9090']
    metrics_path: '/metrics'
    scrape_interval: 15s
```

Exemplos de Requisições

Exemplo em Python

```python
import requests
import json

# Requisição SRI-for-SM
url = "http://localhost:9090/api/sri-for-sm"
payload = {
    "msisdn": "1234567890",
    "service_center": "5551234567"
}

response = requests.post(url, json=payload, timeout=15)

if response.status_code == 200:
    result = response.json()
    print(f"Sucesso: {result}")
elif response.status_code == 504:
    print("Timeout - sem resposta da rede")
else:
    print(f"Erro: {response.status_code} - {response.text}")
```

Exemplo em JavaScript

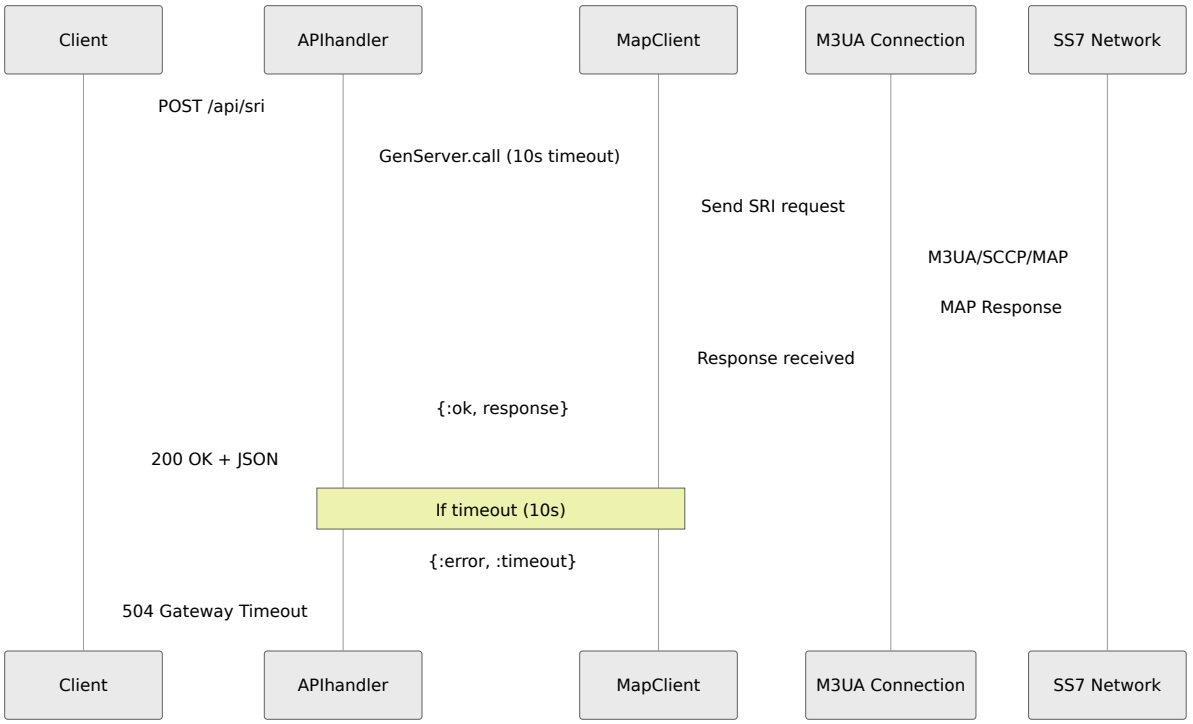
```
const axios = require('axios');

async function sendSRI() {
  try {
    const response = await
axios.post('http://localhost:9090/api/sri', {
      msisdn: '1234567890',
      gmsc: '5551234567'
    }, {
      timeout: 15000
    });

    console.log('Sucesso:', response.data);
  } catch (error) {
    if (error.code === 'ECONNABORTED') {
      console.error('Timeout - sem resposta da rede');
    } else {
      console.error('Erro:', error.response?.data ||
error.message);
    }
  }
}

sendSRI();
```

Exemplo em Bash/cURL

```
#!/bin/bash

# Requisição UpdateLocation
response=$(curl -s -w "\n%{http_code}" -X POST
http://localhost:9090/api/updateLocation \
  -H "Content-Type: application/json" \
  -d '{
    "imsi": "001001234567890",
    "vlr": "5551234800"
  }')

http_code=$(echo "$response" | tail -n 1)
body=$(echo "$response" | sed '$d')

if [ "$http_code" -eq 200 ]; then
  echo "Sucesso: $body"
elif [ "$http_code" -eq 504 ]; then
  echo "Timeout - sem resposta da rede"
else
  echo "Erro $http_code: $body"
fi
```

Diagramas de Fluxo

Fluxo de Requisição da API



Resumo

A API REST do OmniSS7 fornece:

- **47 endpoints de operação** em Mobilidade, SMS, Autenticação, Dados do Assinante, Serviços Suplementares, Serviços de Localização, GPRS, Equipamento, USSD, CAP/CAMEL e Injeção bruta.
- **Swagger UI** em `/swagger` para documentação interativa e testes.
- **Métricas Prometheus** em `/metrics` para monitoramento e observabilidade.
- **Timeout padrão de 10 segundos** para operações aguardadas (`504` em caso de timeout).
- Um servidor HTTP **Plug.Cowboy na porta 9090**, ativado por `start_http_server`. Este servidor é separado do painel de controle (:8087) e dos controladores REST `:api_ex` (:8445).

Para esquemas de requisição/resposta por operação e opcodes MAP, veja o Guia do Cliente MAP.

Para acesso à Interface Web, veja o Guia da Interface Web.

Para detalhes de configuração, veja a Referência de Configuração.

Referência Técnica (Apêndice)

← Voltar à Documentação Principal

Referência técnica para protocolos SS7 e implementação do OmniSS7.

Pilha de Protocolos SS7



Códigos de Operação MAP

Os opcodes abaixo são os códigos de operação `local` do MAP conforme 3GPP TS 29.002. Esta tabela cobre as operações que o OmniSS7 gera e/ou manipula; para o conjunto completo de operações acionáveis pelo cliente e seus parâmetros de solicitação, consulte o **Guia do Cliente MAP**.

Operação	Opcode	Propósito
updateLocation	2	Registrar localização do assinante (CS)
cancelLocation	3	Desregistrar assinante do VLR/SGSN
provideRoamingNumber	4	Solicitar MSRN
noteSubscriberDataModified	5	Notificar gsmSCF sobre alteração de dados do assinante
insertSubscriberData	7	Enviar perfil do assinante para VLR/SGSN
deleteSubscriberData	8	Retirar dados do assinante do VLR/SGSN
registerSS	10	Registrar um serviço suplementar (por exemplo, CFU)
eraseSS	11	Apagar um serviço suplementar
activateSS	12	Ativar um serviço suplementar
deactivateSS	13	Desativar um serviço suplementar
interrogateSS	14	Interrogar status do serviço suplementar
registerPassword	17	Registrar uma senha de SS

Operação	Opcode	Propósito
getPassword	18	Solicitar uma senha de SS do assinante
sendRoutingInfo	22	Consultar roteamento de chamada (SRI)
updateGprsLocation	23	Registrar localização de comutação de pacotes (SGSN)
sendRoutingInfoForGprs	24	Consultar roteamento GPRS em direção ao GGSN
reset	37	Reinicialização do HLR em direção ao VLR/SGSN
forwardCheckSS-Indication	38	Indicar que os dados de SS devem ser verificados novamente
checkIMEI	43	Consultar EIR sobre status do equipamento
mt-forwardSM	44	Entregar SMS ao assinante (MT)
sendRoutingInfoForSM	45	Consultar roteamento de SMS (SRI-for-SM)
mo-forwardSM	46	Encaminhar SMS do assinante (MO)
reportSM-DeliveryStatus	47	Relatar resultado da entrega de SMS ao HLR

Operação	Opcode	Propósito
sendIdentification	55	Recuperar IMSI/auth do VLR anterior
sendAuthenticationInfo	56	Solicitar vetores de autenticação
restoreData	57	Restaurar dados do assinante após falha do VLR
sendIMSI	58	Resolver MSISDN para IMSI no HLR
processUnstructuredSS-Request	59	Solicitação USSD originada no celular (entrada)
unstructuredSS-Request	60	Solicitação USSD originada na rede (saída)
unstructuredSS-Notify	61	Notificação USSD originada na rede
anyTimeSubscriptionInterrogation	62	Consulta gsmSCF de dados de assinatura
alertServiceCentre	64	Alertar SMSc que um assinante está acessível
anyTimeModification	65	Modificação gsmSCF de dados do assinante
readyForSM	66	Notificar HLR que o assinante está pronto para SMS

Operação	Opcode	Propósito
purgeMS	67	Purgar registro do assinante do VLR/SGSN
provideSubscriberInfo	70	Solicitar estado/localização do assinante
anyTimeInterrogation	71	Consulta gsmSCF de estado/localização do assinante
provideSubscriberLocation	83	Solicitar estimativa de localização do nó de serviço (LCS)
sendRoutingInfoForLCS	85	Consulta GMLC para o nó de serviço do assinante (LCS)
subscriberLocationReport	86	Nó de serviço relata uma localização ao GMLC (LCS)
noteMM-Event	89	Relatar um evento de gerenciamento de mobilidade ao gsmSCF

Tipos de Mensagens TCAP

- **BEGIN** - Iniciar transação
- **CONTINUE** - Meio da transação
- **END** - Resposta final
- **ABORT** - Cancelar transação

Endereçamento SCCP

Formatos de Título Global

- **E.164** - Número de telefone internacional (por exemplo, 447712345678)
- **E.212** - Formato IMSI (por exemplo, 234509876543210)
- **E.214** - Formato de código de ponto

Números de Subsistema (SSN)

- **SSN 6**: HLR
- **SSN 7**: VLR
- **SSN 8**: MSC/SMSC
- **SSN 9**: GMLC
- **SSN 10**: SGSN

SMS TPDU

Tipos de Mensagens

- **SMS-DELIVER** (MT) - Rede para móvel
- **SMS-SUBMIT** (MO) - Móvel para rede
- **SMS-STATUS-REPORT** - Status de entrega
- **SMS-COMMAND** - Comando remoto

Codificações de Caracteres

- **GSM7** - Alfabeto GSM de 7 bits (160 caracteres por SMS)
- **UCS2** - Unicode de 16 bits (70 caracteres por SMS)
- **8-bit** - Dados binários (140 bytes por SMS)

Estados M3UA

- **DOWN** - Sem conexão SCTP
- **CONNECTING** - SCTP conectando
- **ASPUP_SENT** - Aguardando ACK de ASPUP
- **INACTIVE** - ASP ativo, mas não em uso
- **ASPAC_SENT** - Aguardando ACK de ASPAC
- **ACTIVE** - Pronto para tráfego

Códigos de Ponto Comum SS7

Os códigos de ponto são tipicamente valores de 14 bits (ITU) ou 24 bits (ANSI).

Formato de Exemplo (ITU):

- Rede: 3 bits
- Cluster: 8 bits
- Membro: 3 bits

Códigos de Erro SCCP

- **0** - Sem tradução para o endereço
- **1** - Sem tradução para endereço específico
- **2** - Congestionamento do subsistema
- **3** - Falha do subsistema
- **4** - Usuário não equipado
- **5** - Falha do MTP
- **6** - Congestionamento da rede
- **7** - Não qualificado
- **8** - Erro no transporte da mensagem

Códigos de Erro MAP

Código	Erro	Descrição
1	unknownSubscriber	Assinante não encontrado no HLR
27	absentSubscriber	Assinante não acessível
34	systemFailure	Falha na rede
35	dataMissing	Dados necessários não disponíveis
36	unexpectedDataValue	Valor de parâmetro inválido

Documentação Relacionada

- ← Voltar à Documentação Principal
- Guia STP
- Guia do Cliente MAP
- Guia do Centro de SMS
- Guia do HLR
- Recursos Comuns

Guia de Configuração do Gateway CAMEL

Visão Geral

O modo **Gateway CAMEL (CAMELGW)** transforma o OmniSS7 em uma plataforma de Rede Inteligente (IN) que fornece controle de chamadas e cobrança em tempo real usando o protocolo CAMEL Application Part (CAP). O OmniSS7 atua como o **gsmSCF** (Função de Controle de Serviço), recebendo gatilhos do **gsmSSF** no MSC/VLR/GMSC e tomando decisões de cobrança através do CGrateS.

O que é CAMEL?

CAMEL (Aplicações Personalizadas para Lógica Aprimorada de Rede Móvel) é um conjunto de padrões que funcionam em uma rede central GSM ou em uma

rede UMTS. Ele permite que os operadores forneçam serviços que requerem controle em tempo real das chamadas, como:

- **Chamadas pré-pagas** - Verificação e cobrança de saldo em tempo real
- **Serviços de tarifa premium** - Cobrança especial para serviços de valor agregado
- **Controle de roteamento de chamadas** - Roteamento dinâmico de destinos com base em tempo/localização
- **Redes privadas virtuais** - Planos de numeração corporativa
- **Filtragem de chamadas** - Permitir/bloquear chamadas com base em critérios

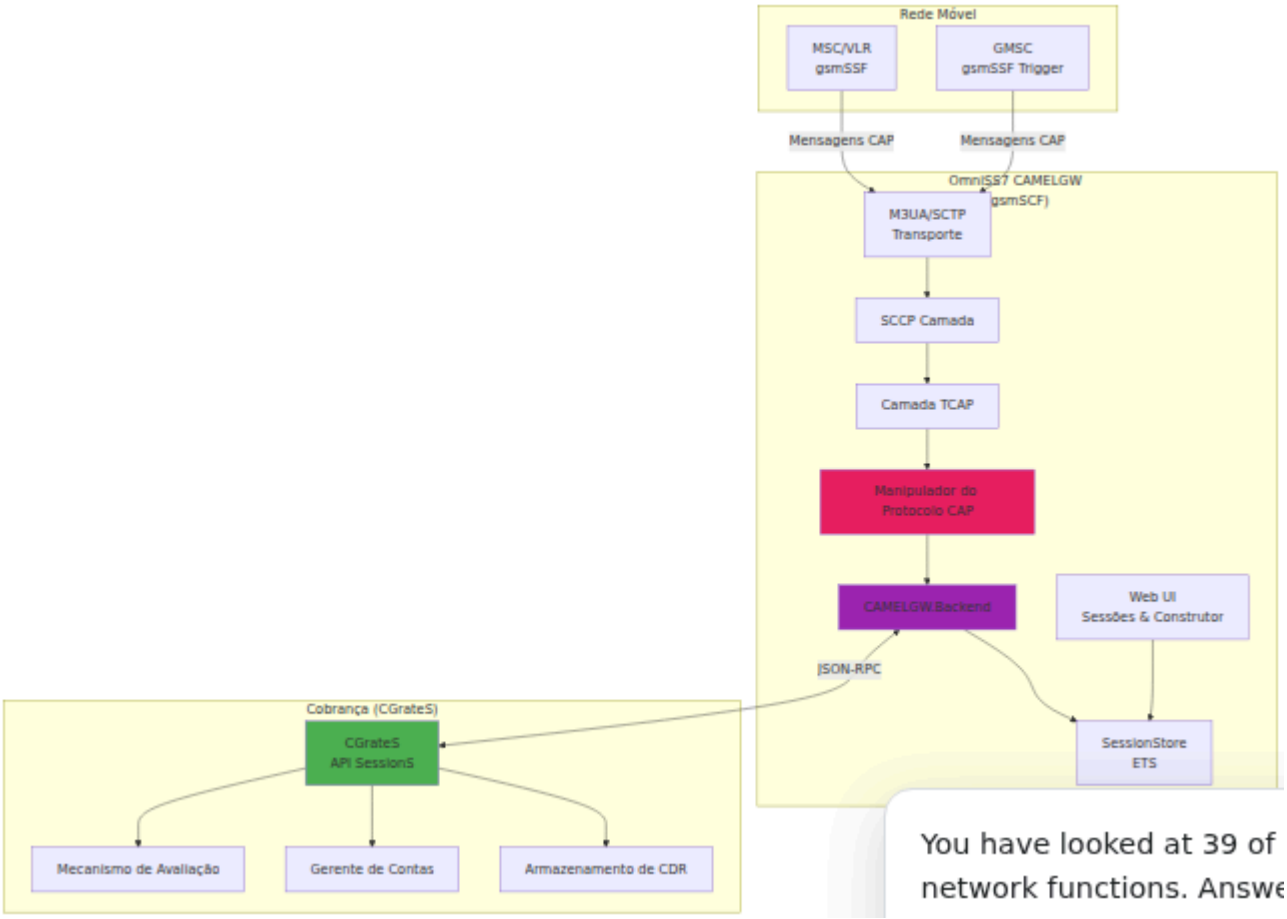
Versões do Protocolo CAP

O OmniSS7 CAMELGW suporta várias versões do CAP. A versão controla o OID do contexto de aplicação transportado na parte do diálogo TCAP:

Versão	Fase	OID do Contexto de Aplicação
CAP v1	Fase CAMEL 1	`0.4.0.0.1.0.50.0`
CAP v2	Fase CAMEL 2	`0.4.0.0.1.0.50.1`
CAP v3	Fase CAMEL 3	`0.4.0.0.1.21.3.4`
CAP v4	Fase CAMEL 4	`0.4.0.0.1.23.3.4`

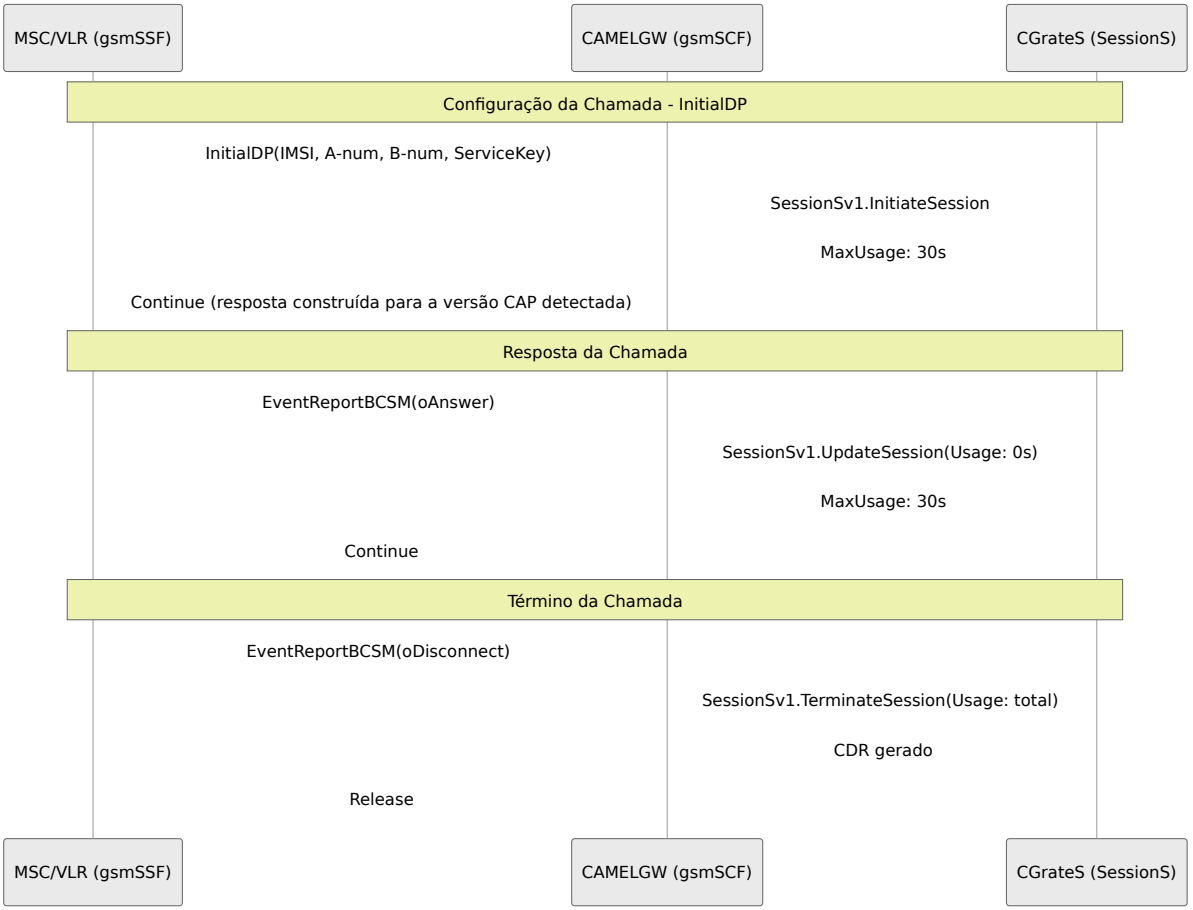
Padrão: CAP v2 (mais amplamente implantado). A versão de entrada é detectada automaticamente a partir da parte do diálogo InitialDP e reutilizada ao gerar a resposta, para que as respostas correspondam à versão CAP de cada MSC.

Arquitetura



You have looked at 39 of
network functions. Answe

Exemplo de Fluxo de Chamadas



Configuração

Pré-requisitos

- OmniSS7 instalado e em execução
- Conectividade M3UA com MSC/GMSC (gsmSSF)
- Uma instância do CGrateS acessível via JSON-RPC (opcional, para cobrança em tempo real)

Habilitar o Modo Gateway CAMEL

O processamento CAMEL é controlado por duas flags independentes. `cap_client_enabled` ativa o cliente CAP/M3UA e os componentes de sessão

CAMEL; `cgrates_enabled` ativa a integração de cobrança do CGrateS por trás do backend. Configure estas opções em `config/runtime.exs`:

```
config :omniss7,
  # Ativa o cliente CAP/CAMEL (M3UA ASP, SessionStore, manipulação
de solicitações)
  cap_client_enabled: true,

  # Outros modos independentes (defina conforme necessário para
sua implantação)
  map_client_enabled: false,
  smsc_mode_enabled: false,

  # Versão CAP usada para solicitações / diálogo de saída (:v1 |
:v2 | :v3 | :v4)
  cap_version: :v2,

  # Integração de cobrança do CGrateS
  cgrates_enabled: true,

  # Conexão M3UA para CAP (conectar como ASP em direção ao
MSC/GMSC gsmSSF)
  cap_client_m3ua: %{
    routing_context: 1,
    opc: 5013,
    dpc: 5011
  }
```

> **Nota sobre `cap_client_m3ua`:** o cliente CAMEL lê apenas `:routing_context`, `:opc` e `:dpc` deste mapa. Quaisquer chaves adicionais são ignoradas pelo cliente CAP. O nome do processo ASP M3UA é fixo internamente como `:camelgw_client_asp`.

> Não existe a flag `camelgw_mode_enabled`, `hlr_mode_enabled` ou `smsc_mode_enabled`-for-CAMEL. As únicas flags que controlam o comportamento CAMEL são `cap_client_enabled` e `cgrates_enabled`.

Flags de Modo

Chave	Padrão	Propósito
`cap_client_enabled`	`false`	Inicia o cliente CAP/CAMEL (M3UA ASP, SessionStore, manipulação de solicitações CAP)
`cgrates_enabled`	`false`	Habilita a integração de cobrança do CGrateS no `CAMELGW.Backend`
`cap_version`	`:v2`	Versão CAP padrão / OID de diálogo para solicitações de saída
`map_client_enabled`	`false`	Modo de cliente MAP independente
`smsc_mode_enabled`	`false`	Modo SMSc independente (requer cliente MAP)

Parâmetros de Cobrança do CGrateS

Chave	Padrão	Pr
`cgrates_enabled`	`false`	Interrup para int CGrate
`cgrates_url`	`http://localhost:2080/jsonrpc`	Endpoi do CGra
`cgrates_tenant`	`cgrates.org`	Inquilin cada ev Sessior
`cgrates_request_type`	`*prepaid`	Tipo de do CGra exempl `*postp`
`cgrates_timeout`	`5000`	Tempo solicita milisse
`cgrates_voice_usage_seconds`	*Obrigatório*	Tamanh de voz segund InitialD ApplyC reserva o Apply conced mesmo interva TAS Ro

Configurar Páginas da Web UI

A Web UI expõe as páginas CAMEL através do painel de controle. Estas são configuradas em `config/config.exs`:

```
config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "Pares"},
    {SS7.Web.CAMELSessionsLive, "/camel_sessions", "Sessões
CAMEL"},
    {SS7.Web.CAMELRequestLive, "/camel_request", "Construtor de
Solicitações CAMEL"}
  ],
  page_order: [
    "/events", "/m3ua", "/camel_sessions", "/camel_request",
    "/application", "/configuration"
  ]
```

Rótulo da Página	Rota	Módulo
Pares	`/m3ua`	`SS7.Web.M3UAStatusLive`
Sessões CAMEL	`/camel_sessions`	`SS7.Web.CAMELSessionsLive`
Construtor de Solicitações CAMEL	`/camel_request`	`SS7.Web.CAMELRequestLive`

Operações CAP Suportadas

As operações CAP são identificadas por nome e direção. Observe que alguns números de opcode CAP são internamente inconsistentes (o mapa de opcode do log de eventos e o gerador de solicitações discordam em algumas operações), portanto, as operações abaixo são descritas por **nome e endpoint**. Onde um opcode é citado, é aquele que o gerador de solicitações realmente emite.

Operações de Entrada (gsmSSF → gsmSCF)

Desencadeadas no opcode do componente TCAP decodificado:

Operação	Notas
InitialDP	Notificação de configuração de chamada; aciona `InitiateSession` do CGrateS. Aceito no opcode 0, e também no opcode 60 (tratado como InitialDP/InitialDPSMS)
AssistRequestInstructions	Reconhecido; desencadeado no opcode 1 em um TCAP Begin
EventReportBCSM	Eventos BCSM. oAnswer é registrado (sem resposta); oDisconnect / oAbandon / tDisconnect / tAbandon / busy / no-answer terminam a sessão do CGrateS e encerram o diálogo
ApplyChargingReport	Opcode 36. Chamada ativa: relata o período ao CGrateS e concede o próximo bloco. Chamada encerrada: termina a sessão do CGrateS com o uso total
Continue	Reconhecido
ConnectToResource	Reconhecido
RequestReportBCSMEvent	Reconhecido
ApplyCharging	Parâmetros decodificados em BER e reconhecidos
Connect	Parâmetros decodificados em BER e reconhecidos
ReleaseCall	Reconhecido

Opcodes não tratados são respondidos com um erro TCAP (Facilidade Não Suportada / Valor de Dados Inesperado).

Operações de Saída (gsmSCF → gsmSSF)

Enviadas em direção ao gsmSSF. Os opcodes abaixo são os valores que o OmniSS7 emite:

Operação	Opcode
InitialDP	0
Connect	20
ReleaseCall	22
RequestReportBCSMEvent	23
Continue	31
ApplyCharging	35
InitialDPSMS	60
NoteMM-Event	89 (MAP)

> **InitialDPSMS** (opcode 60) é para controle CAMEL de MO-SMS: enviado pelo smsSSF para o gsmSCF quando um SMS-CSI / mo-sms-CSI está armado, antes que a mensagem curta seja retransmitida para o SMSC. O arg é codificado com o módulo CAP ASN.1 gerado e o diálogo carrega o contexto de aplicação `id-ac-cap3-sms-AC`.
>
> **NoteMM-Event** é uma operação MAP (opcode 89, não CAP) usada para notificações de mobilidade CAMEL M-CSI do VLR para o gsmSCF. É codificado com o módulo MAP ASN.1 (`:TCAPMessages.encode(:MapSpecificPDUs, …)`) e carrega o contexto de aplicação `mm-EventReporting-v3`.

Todas as mensagens TCAP CAP são codificadas com `:GenericTCAP.encode(:GenericPDUs, …)`. Apenas o NoteMM-Event baseado em MAP usa `:TCAPMessages`.

Recursos da Web UI

Página de Sessões CAMEL

Título: Sessões CAMEL · **Rota:** `/camel_sessions` (HTTPS, porta 8087)

Monitoramento em tempo real de sessões de chamadas CAMEL ativas:

- **Lista de sessões ao vivo** com atualização automática
- **Detalhes da sessão** - OTID, ID da Chamada, Estado, Duração
- **Versão CAP** detectada a partir do InitialDP e armazenada por sessão
- **Informações da chamada** - IMSI, A-number, B-number, Chave de Serviço
- **Rastreamento de estado** - `initiated`, `answered`, `terminated`

Colunas da Tabela: ID da Chamada, Estado, Versão, IMSI, Número Chamador, Número Chamado, Chave de Serviço, Duração, Hora de Início, OTID

Os mesmos dados da sessão estão disponíveis através da API REST em `/camel-sessions` (veja Monitoramento & Operações).

Construtor de Solicitações CAMEL

Título: Construtor de Solicitações CAMEL · **Rota:** `/camel_request` (HTTPS, porta 8087)

Ferramenta interativa para construir e enviar solicitações CAP para testes.

Tipos de solicitação oferecidos pelo formulário: InitialDP, Connect, ReleaseCall, RequestReportBCSMEvent, ApplyCharging, Continue.

Selector de versão CAP: o formulário possui um dropdown de versão CAP por solicitação (v1/v2/v3/v4 com seus OIDs). A seleção substitui temporariamente o padrão configurado para aquela solicitação e persiste entre as alterações do formulário; o histórico de solicitações mostra a versão usada em uma coluna **CAP Ver**.

Opções avançadas SCCP/M3UA:

Campo	Padrão	Notas
GT da Parte Chamado	`99988415011`	Título Global de Destino
GT da Parte Chamadora	(configurável)	Título Global de Origem
SSN Chamado	`146`	gsmSSF
SSN Chamador	`146`	gsmSSF
OPC	`5013`	Código de Ponto de Origem
DPC	`5011`	Código de Ponto de Destino

Integração com CGrateS

O OmniSS7 cobra chamadas através da API JSON-RPC **SessionS** do CGrateS. `CGrateS.SessionsClient` posta para o `cgrates_url` configurado e mapeia operações CAP para métodos SessionS:

Gatilho CAP	Método SessionS
InitialDP	`SessionSv1.InitiateSession` (reservar um bloco)
ApplyChargingReport, chamada ativa	`SessionSv1.UpdateSession` (`LastUsed` = o período, reservar o próximo bloco)
ApplyChargingReport, chamada encerrada; EventReportBCSM desconectar / abandonar; TCAP End ou Abort do MSC	`SessionSv1.TerminateSession` (uso total), então `SessionSv1.ProcessCDR`

A cobrança de voz segue o modelo TAS Ro: o mesmo inquilino (`cgrates_tenant`), `ToR *voice`, `RequestType` `cgrates_request_type`, `Category call`, um bloco fixo (`cgrates_voice_usage_seconds`), e `GetAttributes` em cada solicitação para que os perfis AttributeS do inquilino mapeiem o assinante na conta de cobrança. A `Account` é o **IMSI** do assinante atendido a partir do InitialDP (um InitialDP sem IMSI é liberado). `Destination` é o número chamado. `OriginID` é `CAMEL-<hex OTID>`, `OriginHost` é `CAMELGW`.

Ciclo de Vida da Chamada com Cobrança

1. Iniciação da Chamada (InitialDP)

Quando o MSC envia InitialDP, o CAMELGW:

1. **Detecta a versão CAP** a partir da parte do diálogo
2. **Decodifica o InitialDP** (IMSI, número chamado/chamador ISUP, CalledPartyBCDNumber, chave de serviço, tipo de evento)
3. **Chama o CGrateS** `SessionSv1.InitiateSession` para um bloco
4. **Responde ao MSC** em um TCAP Continue com RequestReportBCSMEvent (O-BCSM ou T-BCSM resposta, desconectar perna 1 e 2, abandonar; todos notifyAndContinue), ApplyCharging e Continue

`MaxUsage` dimensiona a `maxCallPeriodDuration` do ApplyCharging. Um bloco completo é concedido sem `releaseIfDurationExceeded`; um restante mais curto é concedido com ele, para que o MSC libere a chamada quando o último crédito for usado. Sem crédito, ou um erro do CGrateS, responde com **ReleaseCall** (causa 31) em um TCAP End.

2. Resposta da Chamada (EventReportBCSM - oAnswer / tAnswer)

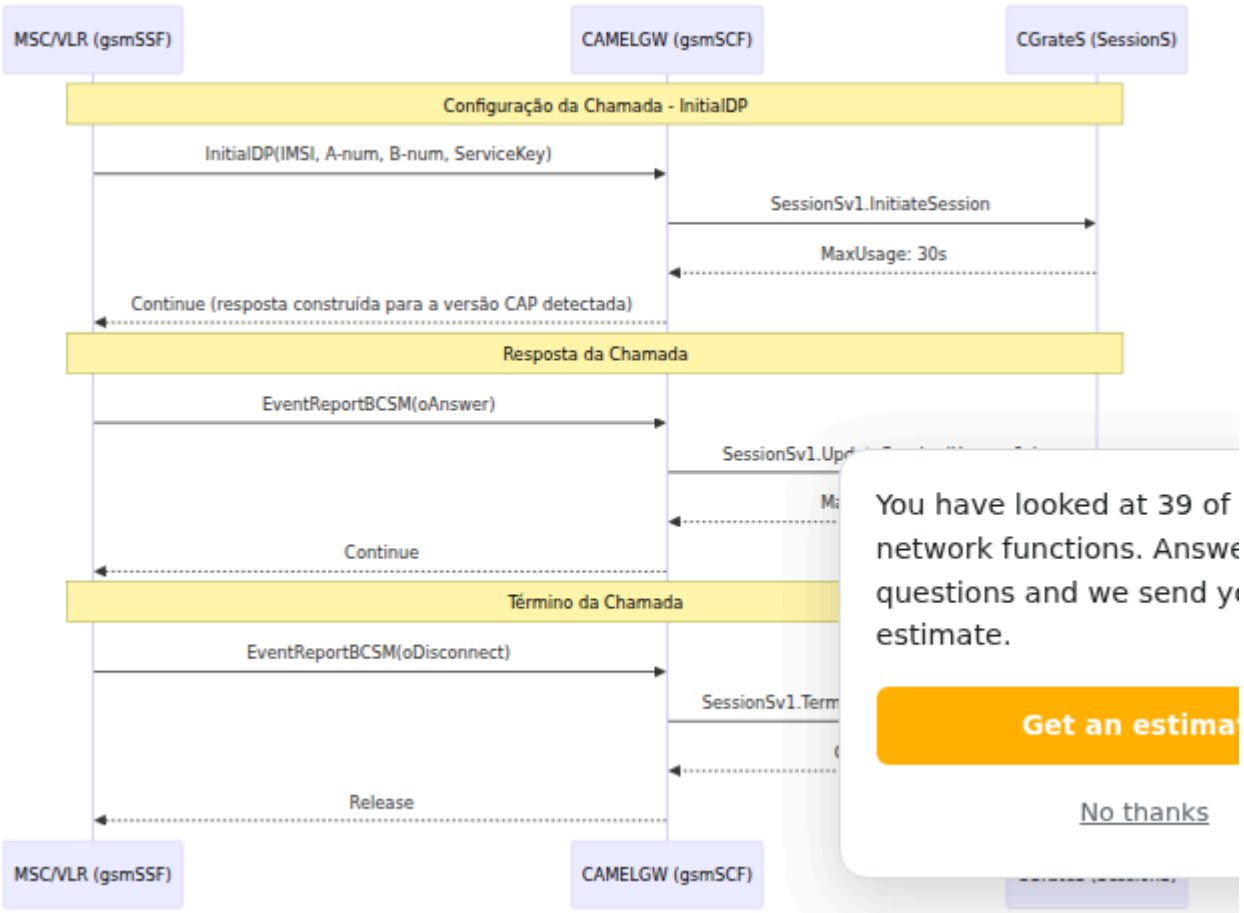
A sessão é marcada como respondida. Nada é enviado de volta; o diálogo permanece aberto.

3. Expiração do Período (ApplyChargingReport)

Cada vez que o período concedido termina, o MSC relata o tempo usado. O gateway o relata ao CGrateS e concede o próximo bloco, ou envia ReleaseCall quando o crédito é esgotado.

4. Término da Chamada

O ApplyChargingReport final (chamada não mais ativa), um EventReportBCSM de desconexão / abandono, ou o MSC fechando o diálogo termina a sessão do CGrateS com o uso total, escreve o CDR (`ProcessCDR`) e exclui a sessão.



Campos CDR do CAMEL

Os CDRs são produzidos pelo CGrateS e normalmente incluem:

- `Account` - IMSI (ou número chamador como fallback)
- `Destination` - Número da parte chamada
- `OriginID` - Identificador da chamada (`CAMEL-<hex OTID>`)
- `Usage` - Duração total da chamada (segundos)
- `IMSI`, `CallingPartyNumber`, `CalledPartyNumber`
- `MSCAddress` - MSC atendente
- `ServiceKey` - Chave de serviço CAMEL

Testes

Testes Manuais com o Construtor de Solicitações

1. Abra `https://seu-servidor:8087/camel_request`
2. Selecione **InitialDP**, defina Chave de Serviço / Números Chamados / Chamadores, opcionalmente escolha uma versão CAP e envie. Observe o OTID gerado.
3. Abra `https://seu-servidor:8087/camel_sessions` para ver a sessão no estado `initiated`.
4. Envie **EventReportBCSM** com `oAnswer` para mover a sessão para `answered`.
5. Envie **ReleaseCall** com causa `16` (Normal) para encerrar.

Testando com um MSC Real

Configure o serviço CAMEL em seu MSC/VLR (Chave de Serviço, Título Global gsmSCF) para que ele acione o InitialDP em direção ao OmniSS7. Exemplo para um MSC Huawei:

```
ADD CAMELSERVICE:
  SERVICEID=1,
  SERVICEKEY=100,
  GSMSCFADDR="55512341234",   # Título Global CAMELGW
  DEFAULTCALLHANDLING=CONTINUE;

ADD CAMELSUBSCRIBER:
  IMSI="310150123456789",
  SERVICEID=1,
  TRIGGERTYPE=TERMCALL;
```

Monitorar Logs

O OmniSS7 emite logs JSON estruturados (um objeto por linha via `OmniLogger.JsonFormatter`), para que você possa filtrar em campos de metadados:

```
# Tail e filtrar eventos CAP pelo campo map_event
tail -f /var/log/omniss7/omniss7.log | jq 'select(.map_event |
tostring | startswith("CAP:"))'
```

Monitoramento & Operações

Métricas do Prometheus

O CAMELGW expõe métricas em `http://localhost:9090/metrics`.

Métricas CAP:

- `cap_requests_total{operation}` - Total de solicitações CAP por operação. Os valores dos rótulos são snake_case, por exemplo, `initial_dp`, `connect`, `release_call`, `continue`, `apply_charging`.
- `cap_request_errors_total{operation}` - Erros de geração/envio de solicitações CAP por operação.

Métricas MAP / API:

- `map_requests_total{operation}` - Total de solicitações MAP por operação
- `map_pending_requests` - Número de aguardando TID MAP

Métricas STP M3UA (se o modo STP estiver habilitado):

- `m3ua_stp_routing_failures_total{reason}` - Falhas de roteamento por motivo

```
# Todos os contadores de solicitações CAP
curl http://localhost:9090/metrics | grep cap_requests_total

# Apenas InitialDP (note o rótulo snake_case)
curl http://localhost:9090/metrics | grep
'cap_requests_total{operation="initial_dp"}'
```

Verificações de Saúde

```
# Status do par M3UA (API TLS na porta 8445)
curl -k https://localhost:8445/m3ua-status

# Sessões CAMEL ativas (retorna session_count + detalhes por
sessão)
curl -k https://localhost:8445/camel-sessions
```

Não existe o endpoint `/api/ocs-status`, `/api/events`, ou `/api/camel/sessions/count`. As contagens de sessão vêm de `/camel-sessions`.

Configuração de Logs

Ajuste o nível de log em `config/runtime.exs`:

```
config :logger, level: :info   # :debug | :info | :warning |
:error
```

O formatador é definido como JSON estruturado em `config/config.exs`:

```
config :logger, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all,
  truncate: :infinity
```

Resolução de Problemas

Problema: Nenhuma mensagem CAP recebida

Sintomas: O Construtor de Solicitações funciona, mas o MSC nunca envia InitialDP.

Verifique:

1. Status do par M3UA: `curl -k https://localhost:8445/m3ua-status`
2. Configuração do serviço CAMEL do MSC (Chave de Serviço, endereço gsmSCF)
3. Roteamento SCCP (o Título Global gsmSCF deve rotear para o CAMELGW)
4. Regras de firewall (permitir a porta SCTP 2905)

Problema: Erros do CGrateS

Sintomas: `not_configured`, `http_error`, ou `cgrates_error` nos logs.

Verifique:

1. `cgrates_enabled: true` e `cgrates_url` apontam para um endpoint JSON-RPC acessível
2. A conta (IMSI ou número chamador) existe e tem saldo no CGrateS
3. `cgrates_tenant` e `cgrates_request_type` correspondem à sua configuração de avaliação do CGrateS
4. `cgrates_timeout` é grande o suficiente para sua latência do CGrateS

Em uma falha de cobrança durante uma chamada, o CAMELGW continua a chamada em vez de encerrá-la; o erro é registrado.

Problema: Sessão não encontrada

Sintomas: EventReportBCSM registra "Sessão não encontrada".

Causa: Desvio de OTID ou a sessão já foi encerrada.

Verifique: Verifique o OTID nos logs e assegure-se de que o DTID/OTID correlacionem-se através do diálogo.

Problema: Erros de decodificação

Sintomas: "Falha ao decodificar InitialDP" nos logs.

Causa: Desvio de versão CAP ou mensagem malformada.

Verifique: Confirme se a configuração da versão CAP corresponde ao MSC e capture um PCAP para análise:

```
tcpdump -i eth0 -w cap_trace.pcap sctp port 2905
```

Resumo

O modo Gateway CAMEL permite que o OmniSS7 atue como um gsmSCF com:

- **Suporte ao protocolo CAP** (v1/v2/v3/v4) com substituição de versão por solicitação
- **Cobrança em tempo real** via API JSON-RPC SessionS do CGrateS
- **Operações de controle de chamadas** (Connect, Release, Continue, ApplyCharging)
- **Gerenciamento de sessões** em ETS, indexadas por OTID
- **Teste interativo** via o Construtor de Solicitações da Web UI
- **Monitoramento ao vivo** de sessões ativas (Web UI e API `/camel-sessions`)
- **Geração de CDR** pelo CGrateS para cobrança e análises

Para mais detalhes:

- Referência Técnica - Operações CAP

Produto: OmniSS7 CAMEL Gateway
Versão da Documentação: 1.1

Guia de Recursos Comuns

← Voltar para a Documentação Principal

Este guia cobre recursos comuns a todos os modos de operação do OmniSS7.

Índice

1. Visão Geral da Interface Web
2. Documentação da API
3. Monitoramento e Métricas
4. Melhores Práticas
5. Multihoming SCTP para Redundância de Rede

Visão Geral da Interface Web

A Interface Web é servida sobre HTTPS na porta `8087` (TLS é aplicado via `force_ssl`), por exemplo, `https://localhost:8087/`.

Navegação Principal

O menu é construído a partir das páginas habilitadas em `config/config.exs` e ordenadas por `page_order`. Entradas comuns (rótulo do menu em negrito):

- **Recursos** (`/application`) - Status da aplicação e informações de tempo de execução
- **Configuração** (`/configuration`) - Visualizador de configuração do sistema
- **Eventos SS7** (`/events`) - Eventos de sinalização SS7 em tempo real e logs de mensagens
- **Logs do Sistema** (`/logs`) - Visualizador de logs da aplicação
- **Cliente SS7** (`/client`) - Cliente de teste interativo de requisição MAP
- **Pares** (`/m3ua`) - Conexões de pares M3UA/M2PA
- **Roteamento** (`/routing`) - Códigos de ponto e rotas de Título Global
- **Teste de Roteamento** (`/routing_test`) - Ferramenta de teste de resolução de rotas
- **Links SMSc** (`/smsc_links`) - Links de backend SMS (SMSc)
- **Assinantes SMSc** (`/smsc_subscribers`) - Assinantes SMS rastreados (SMSc)

- **Links HLR** (`/hlr_links`) - Links de backend HLR (HLR)
- **Assinantes Ativos** (`/subscribers`) - Assinantes UpdateLocation rastreados
- **Sessões CAMEL** (`/camel_sessions`) - Sessões de chamada CAMEL ao vivo (CAMEL GW)
- **Construtor de Requisições CAMEL** (`/camel_request`) - Construtor de requisições CAP interativo (CAMEL GW)
- **Teste MSISDN/IMSI** (`/msisdn_imsi_test`) - Ferramenta de teste de tradução de números

Acessando a Interface Web

1. Abra seu navegador web
2. Navegue até o host configurado na porta 8087 (por exemplo, `https://localhost:8087/`)
3. Veja o painel de status do sistema

Documentação da API Swagger

A documentação interativa da API é servida pela API de operação MAP na porta HTTP 9090:

```
http://your-server:9090/swagger
```

Configuração da Interface Web

O endpoint da interface web é configurado em `config :control_panel, ControlPanelWeb.Endpoint` em `config/config.exs`, e o menu de navegação em `config :control_panel`:

```
config :control_panel, ControlPanelWeb.Endpoint,
  force_ssl: [rewrite_on: [:x_forwarded_proto]],
  server: true,
  url: [host: "0.0.0.0", path: "/"],
  https: [
    port: 8087,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ]

config :control_panel,
  page_order: [
    "/application", "/configuration", "/events", "/logs",
"/client",
    "/m3ua", "/routing", "/routing_test", "/smsc_links",
"/smsc_subscribers",
    "/hlr_links", "/subscribers", "/camel_sessions",
"/camel_request",
    "/msisdn_imsi_test"
  ]
```

Parâmetros de Configuração:

Parâmetro	Tipo	Padrão
`ControlPanelWeb.Endpoint.url.host`	String	`"0.0.0.0"`
`ControlPanelWeb.Endpoint.https.port`	Integer	`8087`
`ControlPanelWeb.Endpoint.https.keyfile`	String	`"priv/cert/omnit`
`ControlPanelWeb.Endpoint.https.certfile`	String	`"priv/cert/omnit`
`ControlPanelWeb.Endpoint.force_ssl`	Keyword	`[rewrite_on: [:x_forwarded_pr`
`page_order`	Lista de Strings	veja o exemplo aci

> Veja a Referência de Configuração para o esquema completo do Painel de Controle.

Configuração do Logger

Os logs são emitidos como JSON estruturado (um objeto por linha) pelo `OmniLogger.JsonFormatter`, que inclui todos os `Logger.metadata`, de modo que o contexto por processo (roteamento STP OPC/DPC/GTs, gateway SMS de/para MSISDNs, transação otid/operação) é transportado automaticamente. Este formato é configurado em `config/config.exs`:

```
config :logger, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all,
  truncate: :infinity
```

O nível de log é definido separadamente:

```
config :logger,
  level: :debug  # Opções: :debug, :info, :warning, :error
```

Parâmetros de Configuração:

Parâmetro	Tipo	Padrão
`:logger.level`	Atom	`:debug`
`:logger.default_formatter.format`	Tuple	`{OmniLogger.JsonF` `:format}`
`:logger.default_formatter.metadata`	Atom/List	`:all`
`:logger.default_formatter.truncate`	Integer/Atom	`:infinity`

Documentação da API

OmniSS7 expõe duas superfícies HTTP:

Superfície	Porta	Protocolo	Conteúdos
API de operação MAP	`9090`	HTTP	Operações MAP `/api/*`, `/swagger`, `/swagger.json`, `/metrics`
API de status JSON (`api_ex`)	`8445`	HTTPS	Status somente leitura (`/api/status`, `/api/m3ua-status`, `/api/routing`, `/api/subscribers`, `/api/camel-sessions`, ...)

URL Base da API MAP

```
http://your-server:9090/api
```

Códigos de Resposta

- **200** - Sucesso
- **400** - Solicitação Inválida
- **504** - Tempo Limite do Gateway

Especificação OpenAPI

```
http://your-server:9090/swagger.json
```

Monitoramento e Métricas

Endpoint de Métricas Prometheus

```
http://your-server:9090/metrics
```

Principais Categorias de Métricas

Métricas M3UA/SCTP:

- Mudanças de estado da associação SCTP
- Transições de estado ASP M3UA
- Unidades de dados de protocolo enviadas/recebidas

Métricas M2PA:

- Transições de estado do link (DOWN → ALIGNMENT → PROVING → READY)
- Mensagens e bytes enviados/recebidos por link
- Erros específicos do link (decodificar, codificar, SCTP)

Métricas STP:

- Mensagens recebidas/enviadas por par
- Falhas de roteamento por motivo
- Distribuição de tráfego entre pares

Métricas do Cliente MAP:

- Requisições MAP por tipo de operação
- Histogramas de duração de requisições
- Medidor de transações pendentes

Métricas CAP:

- Requisições CAP por tipo de operação
- Operações do gateway CAMEL

Métricas SMSc:

- Profundidade da fila
- Taxas de entrega
- Mensagens falhadas

Integração com Grafana

As métricas do OmniSS7 são compatíveis com Prometheus e Grafana.

Melhores Práticas

Recomendações de Segurança

1. **Isolamento de Rede**
    - Implantar em VLAN dedicada
    - Regras de firewall para restringir acesso
    - Permitir SCTP apenas de endereços conhecidos
2. **Segurança da Interface Web**
    - Habilitar TLS para produção
    - Usar proxy reverso com autenticação
    - Restringir a IPs de gerenciamento
3. **Segurança da API**
    - Implementar limitação de taxa
    - Usar chaves de API ou OAuth
    - Registrar todas as solicitações para auditoria

Ajuste de Desempenho

1. **Limites de TPS**
    - Configurar TPS apropriado
    - Monitorar carga do sistema
    - Ajustar buffers SCTP
2. **Ajuste do Backend SMS**

- Manter a API do backend SMS responsiva (as requisições expiram após 5 segundos)
- Ajustar `auto_flush_tps` e `auto_flush_interval` para corresponder à capacidade do backend
- Monitorar profundidade da fila e taxas de entrega

3. **Ajuste do M3UA**
   - Ajustar intervalos de heartbeat SCTP
   - Configurar valores de tempo limite
   - Usar múltiplos links para redundância

Multihoming SCTP para Redundância de Rede

O que é Multihoming SCTP?

Multihoming SCTP é um recurso embutido do protocolo SCTP que permite que uma única conexão M3UA se vincule a múltiplos endereços IP na mesma interface de rede ou em diferentes interfaces de rede. Isso proporciona failover automático e redundância na camada de transporte.

Principais Benefícios:

- **Failover Automático**: Se um caminho de rede falhar, o SCTP muda automaticamente para um caminho alternativo sem interromper a conexão
- **Failover Sem Configuração**: Nenhuma lógica em nível de aplicação necessária - o SCTP gerencia o monitoramento de caminhos e o failover
- **Confiabilidade Melhorada**: Sobrevive a falhas de rede, falhas de switch ou falhas de NIC
- **Balanceamento de Carga**: O SCTP pode distribuir tráfego entre múltiplos caminhos (dependente da implementação)

Como Funciona

Quando você configura múltiplos endereços IP para uma conexão M3UA, o SCTP:

1. **Vincula a todos os IPs**: O socket se vincula a todos os endereços IP configurados simultaneamente
2. **Monitora caminhos**: O SCTP envia continuamente pacotes de heartbeat em todos os caminhos para monitorar sua saúde
3. **Detecta falhas**: Se os heartbeats falharem no caminho primário, o SCTP o marca como inacessível
4. **Failover automático**: O tráfego muda imediatamente para um caminho de backup sem intervenção da aplicação
5. **Recuperação de caminho**: Quando o caminho falhado se recupera, o SCTP o detecta e o marca como disponível novamente

Configuração

O multihoming SCTP é configurado fornecendo uma **lista de endereços IP** em vez de uma única tupla IP.

IP Único (Tradicional)

```
# IP único - sem multihoming
local_ip: {10, 179, 4, 10}
```

Múltiplos IPs (Multihoming Habilitado)

```
# Múltiplos IPs - multihoming habilitado
# Primeiro IP é primário, IPs subsequentes são caminhos de backup
local_ip: [{10, 179, 4, 10}, {10, 179, 4, 11}]
```

Exemplos de Configuração

> **Nota Importante para o Papel do Servidor (Conexões de Entrada):**

> Ao configurar pares com `role: :server` (aceitando conexões de entrada de pares multihomed), você deve especificar `remote_ip` como uma **única tupla** - o endereço IP que o par remoto usa para iniciar a conexão SCTP (SCTP INIT). Não use uma lista.
>
> **Por quê?** O STP combina conexões de entrada com base no IP de origem do pacote SCTP INIT. O SCTP descobrirá automaticamente os outros endereços IP multihomed do par durante o handshake de associação. O formato de lista para `remote_ip` é válido apenas para `role: :client` (conexões de saída).
>
> **Exemplos:**
>
> ```
> # ✓ CORRETO - Papel de servidor com remote_ip único
> %{
>   role: :server,
>   remote_ip: {10, 0, 2, 100},  # Tupla única
>   # ...
> }
>
> # ✗ ERRADO - Papel de servidor com lista NÃO combinará conexões
> de entrada
> %{
>   role: :server,
>   remote_ip: [{10, 0, 2, 100}, {10, 0, 2,
> 101}],  # Lista não funcionará
>   # ...
> }
> ```

Exemplo 1: Par STP com Multihoming (Papel de Cliente - Saída)

```
# Configuração de par em modo STP (conexão de SAÍDA)
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "Partner_STP_Redundant",
      role: :client,  # Saída - nós iniciamos a conexão
      # Multihoming: vincular a dois IPs locais para redundância
      local_ip: [{213, 57, 23, 200}, {213, 57, 23, 201}],
      local_port: 0,
      # Par remoto também suporta multihoming - lista está OK para
papel de cliente
      remote_ip: [{213, 57, 23, 100}, {213, 57, 23, 101}],
      remote_port: 2905,
      routing_context: 1,
      point_code: 100,
      network_indicator: :international
    }
  ]
```

Exemplo 2: Cliente MAP com Multihoming

```
# Modo cliente MAP com multihoming
config :omniss7,
  map_client_enabled: true,
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    # Multihoming: dois IPs locais para failover
    local_ip: [{10, 0, 0, 100}, {10, 0, 0, 101}],
    local_port: 2905,
    # STP remoto com suporte a multihoming
    remote_ip: [{10, 0, 0, 1}, {10, 0, 0, 2}],
    remote_port: 2905,
    routing_context: 1
  }
```

Exemplo 3: Listener STP com Multihoming

```
# Servidor STP autônomo com multihoming
config :omniss7,
  sctp_handler: %{
    enabled: true,
    # Ouvir em múltiplos IPs para conexões de entrada
    local_ip: [{172, 16, 0, 10}, {172, 16, 0, 11}],
    local_port: 2905,
    point_code: 100
  }
```

Exemplo 4: Papel de Servidor - Aceitando Entrada de Par Multihomed

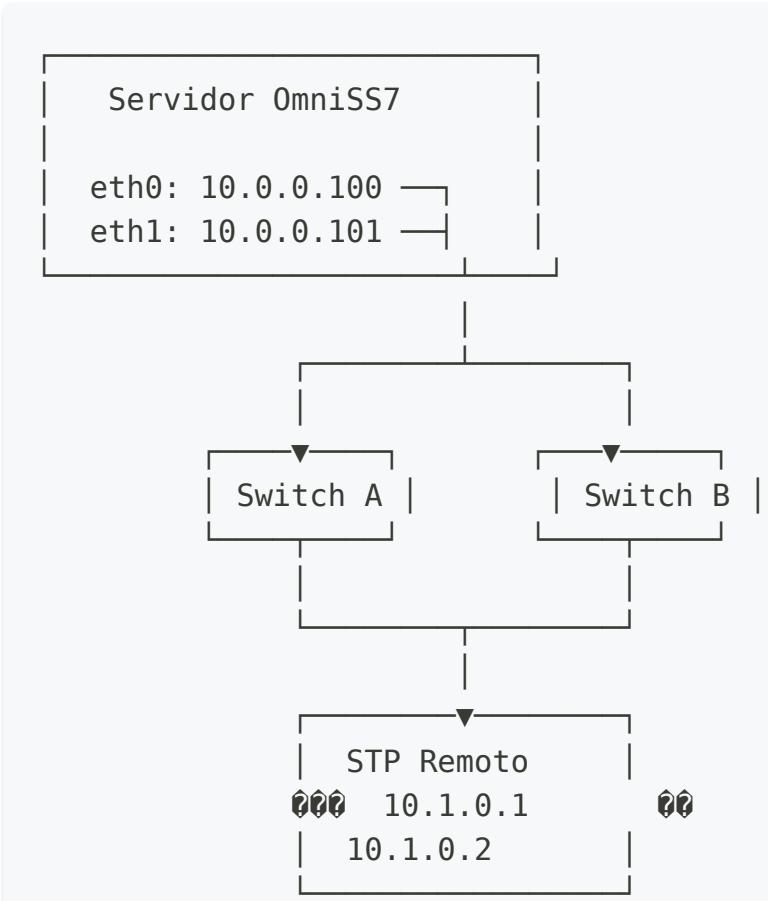
```
# Aceitando conexão de entrada de um HLR multihomed
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "HLR",
      role: :server,  # ENTRADA - HLR se conecta a nós
      # Multihoming: nossos IPs locais (lista está OK)
      local_ip: [{10, 0, 1, 10}, {10, 0, 1, 11}],
      local_port: 2905,
      # IMPORTANTE: Apenas tupla única - o IP que o HLR usa para
iniciar SCTP INIT
      # O SCTP descobrirá automaticamente os outros IPs do HLR
durante o handshake
      remote_ip: {10, 0, 2, 100},  # IP primário apenas (NÃO uma
lista!)
      remote_port: 0,  # Aceitar de qualquer porta de origem
      routing_context: 1,
      point_code: 100,
      network_indicator: :international
    }
  ]
```

Exemplo 5: Configuração Mista (Compatível com Versões Anteriores)

```
# Mistura de pares de IP único e multihomed
config :omniss7,
  m3ua_peers: [
    # Par legado - IP único
    %{
      peer_id: 1,
      name: "Legacy_STP",
      role: :client,
      local_ip: {10, 0, 0, 1},      # Tupla de IP único
      local_port: 0,
      remote_ip: {10, 0, 0, 10},
      remote_port: 2905,
      routing_context: 1,
      point_code: 100
    },
    # Novo par - multihoming
    %{
      peer_id: 2,
      name: "Redundant_STP",
      role: :client,
      local_ip: [{10, 0, 0, 2}, {10, 0, 0, 3}],  # Lista de IPs
      local_port: 0,
      remote_ip: [{10, 0, 0, 20}, {10, 0, 0, 21}],
      remote_port: 2905,
      routing_context: 2,
      point_code: 200
    }
  ]
```

Cenários de Topologia de Rede

Cenário 1: Dual NICs (Implantação Comum)



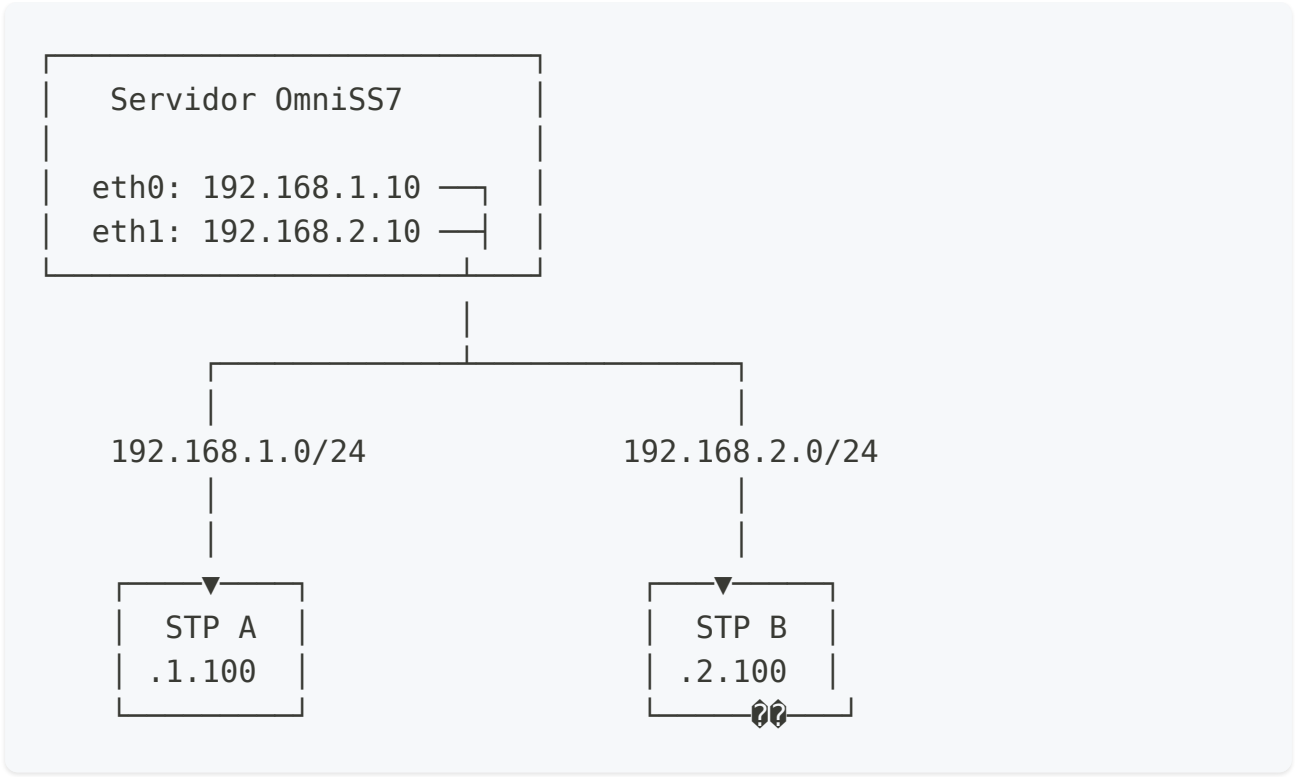
Configuração:

```
local_ip: [{10, 0, 0, 100}, {10, 0, 0, 101}]   # Ambas as NICs
remote_ip: [{10, 1, 0, 1}, {10, 1, 0, 2}]      # Par remoto
```

Benefícios:

- Sobrevive à falha de uma NIC
- Sobrevive à falha de um switch
- Failover automático em <1 segundo

Cenário 2: Múltiplas Sub-redes



Configuração:

```
local_ip: [{192, 168, 1, 10}, {192, 168, 2, 10}]
remote_ip: [{192, 168, 1, 100}, {192, 168, 2, 100}]
```

Benefícios:

- Sobrevive à falha de sub-rede
- Redundância geográfica possível
- Caminhos de roteamento independentes

Monitoramento e Registro

Quando o multihoming está habilitado, você verá mensagens de log que indicam a configuração:

Multihoming Bem-Sucedido

```
[info] Cliente SCTP multihoming: vinculado a 2 IPs locais
[info] Listener STP multihoming habilitado: 2 IPs locais
vinculados
```

Eventos de Failover de Caminho

```
[warning] [MULTIHOMING] Caminho 10.0.0.100 está INACESSÍVEL para o
par Partner_STP (assoc_id=1)
[info] [MULTIHOMING] Caminho 10.0.0.101 agora é o PRINCIPAL para o
par Partner_STP (assoc_id=1)
[info] [MULTIHOMING] Caminho 10.0.0.100 agora está DISPONÍVEL para
o par Partner_STP (assoc_id=1)
```

Exibição na Interface Web

A Interface Web exibe automaticamente informações de multihoming:

Página de Status M3UA:

- **IP Único**: Mostra como `10.0.0.100`
- **Múltiplos IPs**: Mostra como `10.0.0.100 (+1)` ou `10.0.0.100 (+2)`
- **Visualização de Detalhes**: Mostra todos os IPs com rótulos primário/backup

Melhores Práticas

1. Design de Rede

- **Use NICs diferentes** para m��xima redundância
- **Diferentes switches** para sobreviver a falhas de switch
- **Diferentes sub-redes** se possível para diversidade de roteamento
- **Mesmo datacenter inicialmente** - teste antes da separação geográfica

2. Planejamento de Endereço IP

- **Primeiro IP é primário** - certifique-se de que está no caminho mais confiável
- **A ordem importa** - liste os IPs na ordem de preferência
- **Endereçamento consistente** - use esquemas de endereçamento semelhantes para solução de problemas

3. Testando o Failover

```
# Desativar interface primária para testar failover
sudo ip link set eth0 down

# Monitorar logs para failover
tail -f /var/log/omniss7.log | grep MULTIHOMING

# Reativar interface
sudo ip link set eth0 up
```

4. Ambos os Lados Devem Suportar Multihoming

- **Ótimo**: Tanto o local quanto o remoto usam múltiplos IPs
- **Aceitável**: Apenas um lado usa multihoming
- **Nota**: A redundância é melhor quando ambos os pontos finais a suportam

5. Configuração do Firewall

```
# Permitir SCTP em todos os IPs de multihoming
iptables -A INPUT -p sctp --dport 2905 -s 10.0.0.0/24 -j ACCEPT
iptables -A INPUT -p sctp --dport 2905 -s 10.1.0.0/24 -j ACCEPT
```

Solução de Problemas

Problema: Multihoming Não Funciona

Sintomas: Apenas o IP primário é usado, sem failover

Verificações:

1. Verifique o suporte SCTP do Erlang: `erl -eval 'gen_sctp:open(9999, [binary, {ip, {127,0,0,1}}]).'`
2. Verifique o módulo SCTP do kernel: `lsmod | grep sctp`
3. Carregue o SCTP se necessário: `sudo modprobe sctp`
4. Verifique se ambos os IPs estão configurados no sistema: `ip addr show`

Problema: Caminho Não Falhando

Sintomas: Caminho primário marcado como inativo, mas o tráfego não muda

Verificações:

1. Verifique as configurações de heartbeat SCTP
2. Verifique se a tabela de roteamento tem rotas para todos os caminhos
3. Verifique se o firewall permite SCTP em todos os IPs
4. Revise os logs de monitoramento de caminhos SCTP

Problema: Flutuação Frequente de Caminhos

Sintomas: Caminhos constantemente alternando entre ATIVO e INATIVO

Verificações:

1. Instabilidade na rede - verifique os links físicos
2. Heartbeat SCTP muito agressivo - pode precisar de ajuste
3. Firewall derrubando heartbeats SCTP
4. Problemas de MTU em um dos caminhos

Considerações de Desempenho

- **Sobrecarga mínima**: Heartbeats SCTP são pequenos e infrequentes
- **Sem mudanças na aplicação**: O multihoming é transparente para a camada de aplicação
- **Failover rápido**: Tipicamente <1 segundo de detecção e failover
- **Recuperaç��o automática**: Nenhuma intervenção manual necessária

Compatibilidade

- **Compatível com versões anteriores**: O formato de tupla de IP único ainda funciona
- **Implantações mistas**: Pode misturar pares de IP único e multihomed
- **Todos os modos suportados**: Funciona nos modos STP, HLR, SMSc e Cliente MAP
- **Requisito Erlang**: Requer Erlang com suporte SCTP compilado

Monitoramento e Alerta

Métricas Chave:

- Estado da conexão M3UA
- Taxa de sucesso de requisições MAP
- Tempos de resposta da API
- Profundidade da fila de mensagens

Limiares de Alerta:

- M3UA inativo > 1 minuto
- Taxa de tempo limite MAP > 10%
- Profundidade da fila > 1000
- Taxa de erro da API > 5%

Referência Completa de Configuração

Todos os Parâmetros de Configuração

Esta seção fornece uma referência completa de todos os parâmetros de configuração disponíveis em todos os modos de operação.

Configuração do Logger (`:logger`)

```
config :logger, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all,
  truncate: :infinity

config :logger,
  level: :debug  # :debug | :info | :warning | :error
```

Parâmetro	Tipo	Obrigatório	
`:logger.level`	Atom	Não	`:de`
`:logger.default_formatter.format`	Tuple	Não	`{Om` `:fo`
`:logger.default_formatter.metadata`	Atom/List	Não	`:al`
`:logger.default_formatter.truncate`	Integer/Atom	Não	`:in`

Configuração da Interface Web (`:control_panel`)

```
config :control_panel, ControlPanelWeb.Endpoint,
  force_ssl: [rewrite_on: [:x_forwarded_proto]],
  server: true,
  url: [host: "0.0.0.0", path: "/"],
  https: [
    port: 8087,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ]

config :control_panel,
  page_order: [
    "/application", "/configuration", "/events", "/logs",
"/client",
    "/m3ua", "/routing", "/routing_test", "/smsc_links",
"/smsc_subscribers",
    "/hlr_links", "/subscribers", "/camel_sessions",
"/camel_request",
    "/msisdn_imsi_test"
  ]
```

Parâmetro	Tipo	Obrigatório	
`ControlPanelWeb.Endpoint.url.host`	String	Não	`"0`
`ControlPanelWeb.Endpoint.https.port`	Integer	Não	`80`
`ControlPanelWeb.Endpoint.https.keyfile`	String	Não	`"p`
`ControlPanelWeb.Endpoint.https.certfile`	String	Não	`"p`
`ControlPanelWeb.Endpoint.force_ssl`	Keyword	Não	`[r` `[:`
`page_order`	Lista de Strings	Não	ve

Configuração do SocketHandler SCTP (`:omniss7`)

```
config :omniss7,
  sctp_handler: %{
    enabled: false,
    local_ip: {127, 0, 0, 1},
    local_port: 2905
  },
  enable_gt_routing: true,
  m3ua_peers: [...],
  m3ua_routes: [...],
  m3ua_gt_routes: [...]
```

Parâmetro	Tipo	Obrigatório	Padrão	Descriçã
`sctp_handler.enabled`	Boolean	Sim	`false`	Habilitar modo STP na inicializaçã
`sctp_handler.local_ip`	Tuple	Sim	`{127, 0, 0, 1}`	IP para vincular para M3UA de entrada
`sctp_handler.local_port`	Integer	Sim	`2905`	Porta SCTP para M3UA
`enable_gt_routing`	Boolean	Não	`false`	Habilitar roteament de Título Global

Parâmetros de Par M3UA:

Parâmetro	Tipo	Obrigatório	Descrição
`peer_id`	Integer	Sim	Identificador único do par
`name`	String	Sim	Nome descritivo do par
`role`	Atom	Sim	`:client` ou `:server`
`local_ip`	Tuple ou Lista	Se `:client`	IP(s) local(is) para vincular. Único: `{10, 0, 0, 1}` ou Lista: `[{10, 0, 0, 1}, {10, 0, 0, 2}]`
`local_port`	Integer	Se `:client`	Porta local (0 para dinâmica)
`remote_ip`	Tuple ou Lista	Sim	IP(s) do par remoto. Único: `{10, 0, 0, 10}` ou Lista: `[{10, 0, 0, 10}, {10, 0, 0, 11}]`
`remote_port`	Integer	Se `:client`	Porta do par remoto
`routing_context`	Integer	Sim	Contexto de roteamento M3UA
`point_code`	Integer	Sim	Código de ponto SS7
`network_indicator`	Atom	Não	`:international` ou `:national`

Parâmetros de Rota M3UA:

Parâmetro	Tipo	Obrigatório	Descrição
`dest_pc`	Integer	Sim	Código de ponto de destino
`peer_id`	Integer	Sim	Par pelo qual roteá-lo
`priority`	Integer	Sim	Prioridade da rota (menor = maior prioridade)
`network_indicator`	Atom	Não	`:international` ou `:national`

Parâmetros de Rota GT M3UA:

Parâmetro	Tipo	Obrigatório	Descrição
`gt_prefix`	String	Sim	Prefixo de Título Global a ser correspondido
`peer_id`	Integer	Sim	Par de destino
`priority`	Integer	Sim	Prioridade da rota
`description`	String	Não	Descrição da rota para registro
`source_ssn`	Integer	Não	Correspondência apenas se o SSN de origem corresponder
`dest_ssn`	Integer	Não	Reescrever o SSN de destino para este valor

Configuração do Cliente MAP (`:omniss7`)

```
config :omniss7,
  map_client_enabled: false,
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :map_client_asp,
    local_ip: {10, 0, 0, 100},
    local_port: 2905,
    remote_ip: {10, 0, 0, 1},
    remote_port: 2905,
    routing_context: 1
  }
```

Parâmetro	Tipo	Obrigatório	Padrã
`map_client_enabled`	Boolean	Sim	`false`
`map_client_m3ua.mode`	String	Sim	`"ASP"`
`map_client_m3ua.callback`	Tuple	Sim	`{MapClient :handle_pay []}`
`map_client_m3ua.process_name`	Atom	Sim	`:map_client`
`map_client_m3ua.local_ip`	Tuple	Sim	-
`map_client_m3ua.local_port`	Integer	Sim	`2905`
`map_client_m3ua.remote_ip`	Tuple	Sim	-
`map_client_m3ua.remote_port`	Integer	Sim	`2905`
`map_client_m3ua.routing_context`	Integer	Sim	-

Configuração do Centro de SMS (`:omniss7`)

```
config :omniss7,
  auto_flush_enabled: false,
  auto_flush_interval: 10_000,
  auto_flush_dest_smsc: nil,
  auto_flush_tps: 10
```

Parâmetro	Tipo	Obrigatório	Padrão	Descriçã
`auto_flush_enabled`	Boolean	Não	`false`	Habilitar aut flush da fila SMS
`auto_flush_interval`	Integer	Não	`10000`	Intervalo de polling da fil (milissegund
`auto_flush_dest_smsc`	String/nil	Não	`nil`	Filtrar por SMSC de destino (nil = todos)
`auto_flush_tps`	Integer	Não	`10`	Máximo de transações p segundo

Configuração da API HTTP (`:omniss7`)

O backend de SMS agora usa a API HTTP em vez de conexões diretas ao banco de dados.

```
config :omniss7,
  smsc_api_base_url: "https://omnimessage.example.com:8443",
  smsc_name: "omni-smsc01"  # Opcional: padrão para nome derivado
do hostname
```

Parâmetros da API:

Parâmetro	Tipo	Obrigatório	Padrão
`smsc_api_base_url`	String	Sim	`"https://omnimessage.examp`
`smsc_name`	String	Não	nome derivado do hostname

Endpoints da API Usados:

- `POST /api/frontends` - Registrar esta instância de frontend com o backend
- `POST /api/messages_raw` - Inserir novas mensagens SMS
- `GET /api/messages` - Recuperar fila de mensagens (com cabeçalho `smsc`)
- `PATCH /api/messages/{id}` - Marcar mensagem como entregue
- `PUT /api/messages/{id}` - Atualizar status da mensagem
- `POST /api/events` - Adicionar rastreamento de eventos
- `GET /api/status` - Endpoint de verificação de saúde

Notas de Configuração:

- A verificação SSL é desabilitada por padrão para certificados autoassinados
- As requisições HTTP expiram após 5 segundos

- Todos os timestamps estão no formato ISO 8601
- A API usa JSON para corpos de requisição/resposta

Documentação Relacionada

- ← Voltar para a Documentação Principal
- Guia STP
- Guia do Cliente MAP
- Guia do Centro de SMS
- Guia HLR

OmniSS7 por Omnitouch Network Services

Referência de Configuração

← Voltar à Documentação Principal

Este documento fornece uma referência abrangente para todos os parâmetros de configuração do OmniSS7.

Índice

1. Visão Geral
2. Onde a Configuração Está
3. Flags de Modo Operacional
4. Parâmetros do Modo HLR
5. Parâmetros do Modo SMSc
6. Parâmetros do Modo STP
7. Parâmetros do Modo CAMEL Gateway
8. Parâmetros de NAT de Título Global
9. Parâmetros de Conexão M3UA
10. Status do Subsistema SCCP
11. Filtragem de Gateway
12. Parâmetros do Gateway USSD
13. Flags de Tempo de Execução / Supervisão
14. Parâmetros de Infraestrutura
15. Parâmetros de Banco de Dados
16. Valores Codificados

Visão Geral

O OmniSS7 suporta quatro modos operacionais, selecionados inteiramente através da configuração:

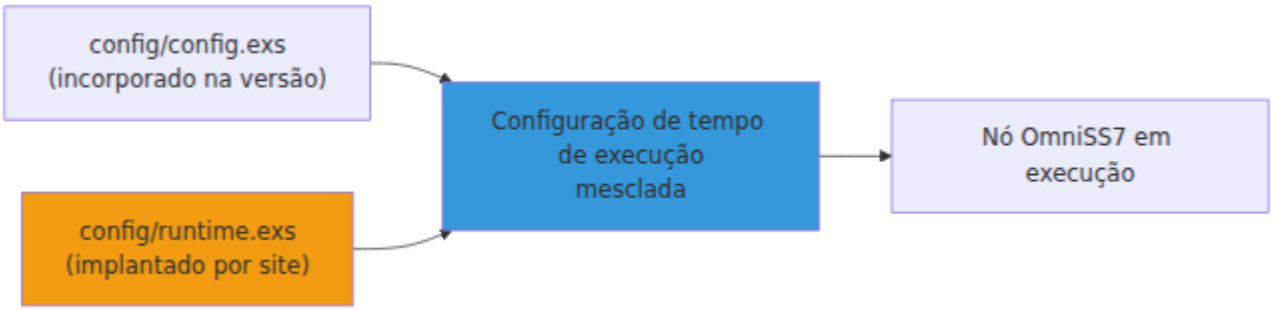
- **Modo STP** - Ponto de Transferência de Sinal para roteamento
- **Modo HLR** - Registro de Localização do Usuário para gerenciamento de assinantes
- **Modo SMSc** - Centro de SMS para entrega de mensagens
- **Modo CAMEL GW** - Gateway CAMEL para controle inteligente de chamadas

Um único nó pode combinar funções (por exemplo, HLR + SMSc) ativando as Flags de Modo Operacional relevantes.

Onde a Configuração Está

O OmniSS7 lê sua configuração de duas camadas. Entender qual arquivo possui um valor determina onde você deve alterá-lo em um sistema implantado.

Arquivo	Função	Editar na implantação?
`config/config.exs`	**Configuração padrão / em tempo de compilação** enviada com a versão. Define as páginas do painel de controle, os controladores REST `:api_ex`, o formatador de log e todos os padrões básicos.	Não: parte da construção.
`config/runtime.exs` (no repositório)	Um **stub de teste de 4 linhas** apenas. Na árvore de origem, não configura nada além do ambiente de teste.	Não.
`config/runtime.exs` (implantado)	A **configuração de tempo de execução por implantação**, implantada separadamente em cada servidor (tipicamente renderizada a partir de um template Ansible/Jinja). É aqui que as flags de modo, GTs, códigos de ponto, pares M3UA e backends de API são definidos para um determinado site.	**Sim. Este é o arquivo voltado para o operador.**



> **Em resumo:** exemplos neste documento usam `config :omniss7, ...`. Em um nó ativo, esses vão para o **implantado** `config/runtime.exs`. O `config/runtime.exs` comprometido no repositório é intencionalmente um stub usado apenas pelo conjunto de testes automatizados. Não confunda isso com a configuração de produção.

Todas as chaves abaixo são lidas via `Application.get_env(:omniss7, ...)` (ou a aplicação relevante) em tempo de execução, então os padrões mostrados são os valores usados quando uma chave é omitida.

Flags de Modo Operacional

Controle quais recursos estão habilitados.

Parâmetro	Tipo	Padrão	Descrição	Modos
`map_client_enabled`	Booleano	`false`	Habilitar conectividade do cliente MAP e M3UA	Todos
`hlr_mode_enabled`	Booleano	`false`	Habilitar recursos específicos do HLR	HLR
`smsc_mode_enabled`	Booleano	`false`	Habilitar recursos específicos do SMSc	SMSc
`cap_client_enabled`	Booleano	`false`	Habilitar cliente CAP para operações CAMEL	CAMEL GW
`camelgw_mode_enabled`	Booleano	`false`	Habilitar recursos do Gateway CAMEL	CAMEL GW
`ussd_gateway_enabled`	Booleano	`false`	Habilitar Gateway USSD (ponte HTTP/JSON)	USSD GW

Exemplo:

```
config :omniss7,
  map_client_enabled: true,
  hlr_mode_enabled: true,
  smsc_mode_enabled: false
```

Parâmetros do Modo HLR

Configuração para o modo HLR (Registro de Localização do Usuário).

Configuração da API HLR

Parâmetro	Tipo	Padrão	Obrigatório
`hlr_api_base_url`	String	-	**Sim**
`hlr_api_key`	String	-	**Sim**
`hlr_api_status_path`	String	`"/api/status"`	Não

Parâmetro	Tipo	Padrão	Obrigatório
`hlr_service_center_gt_address`	String	-	**Sim**
`smsc_service_center_gt_address`	String	-	**Sim**

> **Verificação SSL:** Não há chave `hlr_api_verify_ssl`. Ela não é lida em nenhum lugar no código. A verificação do certificado TLS para chamadas de API de backend é regida pelo campo `verify_ssl` do mapa genérico `:api` (veja Verificação SSL do Backend da API).

Exemplo:

```
config :omniss7,
  hlr_api_base_url: "https://omnihss.example.com:8443",
  hlr_api_key: "<Chave da API OmniHSS com o papel
hlr_interworking>",
  hlr_api_status_path: "/api/status",
  hlr_service_center_gt_address: "55512341111",
  smsc_service_center_gt_address: "55512341112"
```

Configuração do AlertServiceCenter

Quando um assinante realiza UpdateLocation, o HLR envia mensagens alertServiceCenter para GTs SMSc configurados para indicar que o assinante agora está acessível.

Parâmetro	Tipo	Padrão	Obrigatório
`hlr_smsc_alert_gts`	Lista de Strings	`[]`	Não
`hlr_alert_location_expiry_seconds`	Inteiro	`172800`	Não

Exemplo:

```
config :omniss7,
  hlr_smsc_alert_gts: [
    "15559876543",
    "15559876544"
  ],
  hlr_alert_location_expiry_seconds: 172800  # 48 horas
```

Mapeamento MSISDN ↔ IMSI

Configuração para geração sintética de IMSI a partir de MSISDNs. Para uma explicação técnica detalhada do algoritmo de mapeamento, veja Mapeamento MSISDN ↔ IMSI no Guia HLR.

Parâmetro	Tipo	Padrão	Obrigatório	Descriçã
`hlr_imsi_plmn_prefix`	String	`"50557"`	Não	Prefixo PLM (MCC+MNC para geraçã sintética de IMSI
`hlr_msisdn_country_code`	String	`"61"`	Não	Prefixo do código do país para mapeamen reverso IMSI→MSISD
`hlr_msisdn_nsn_offset`	Inteiro	`0`	Não	Deslocame no MSISDN onde o NSN começa (tipicament o comprimen do código d país)
`hlr_msisdn_nsn_length`	Inteiro	`9`	Não	Comprimen do Número Nacional do Assinante a ser extraíd do MSISDN

Exemplo (código do país de 2 dígitos):

```
config :omniss7,
  hlr_imsi_plmn_prefix: "50557",      # MCC 505 + MNC 57
  hlr_msisdn_country_code: "99",      # Exemplo de código do país
de 2 dígitos
  hlr_msisdn_nsn_offset: 2,           # Ignorar código do país de
2 dígitos
  hlr_msisdn_nsn_length: 9            # Extrair NSN de 9 dígitos
```

Exemplo (código do país de 3 dígitos):

```
config :omniss7,
  hlr_imsi_plmn_prefix: "50557",      # MCC 505 + MNC 57
  hlr_msisdn_country_code: "999",     # Exemplo de código do país
de 3 dígitos
  hlr_msisdn_nsn_offset: 3,           # Ignorar código do país de
3 dígitos
  hlr_msisdn_nsn_length: 8            # Extrair NSN de 8 dígitos
```

Importante: Defina `nsn_offset` como o comprimento do seu código do país para extrair corretamente o NSN. Por exemplo:

- Código do país "9" (1 dígito) → `nsn_offset: 1`
- Código do país "99" (2 dígitos) → `nsn_offset: 2`
- Código do país "999" (3 dígitos) → `nsn_offset: 3`

Configuração do InsertSubscriberData (ISD)

Configuração para dados de provisionamento de assinantes enviados para VLRs durante UpdateLocation. Para uma explicação detalhada da sequência ISD e do fluxo de mensagens, veja Configuração do InsertSubscriberData no Guia HLR.

Parâmetro	Tipo	Padrão	Obrigatório
`isd_network_access_mode`	Átomo	`:packetAndCircuit`	Não
`isd_send_ss_data`	Booleano	`true`	Não
`isd_send_call_barring`	Booleano	`true`	Não

Exemplo:

```
config :omniss7,
  isd_network_access_mode: :packetAndCircuit,
  isd_send_ss_data: true,
  isd_send_call_barring: true
```

Configuração CAMEL

Configuração para roteamento inteligente de chamadas baseado em CAMEL. Para uma explicação detalhada da integração CAMEL e das chaves de serviço, veja Integração CAMEL no Guia HLR.

Parâmetro	Tipo	Padrão	Ob
`camel_service_key`	Inteiro	`11_110`	Não
`camel_trigger_detection_point`	Átomo	`:termAttemptAuthorized`	Não
`camel_gsmscf_gt_address`	String	(usa GT chamado)	Não

| `hlr_camel_subscriptions` | Lista | `[]` | Não | O-CSI / T-CSI por prefixo IMSI enviados ao VLR no InsertSubscriberData (veja Assinaturas CAMEL no ISD) |

Exemplo:

```
config :omniss7,
  camel_service_key: 11_110,
  camel_trigger_detection_point: :termAttemptAuthorized,
  hlr_camel_subscriptions: [
    %{
      imsi_prefix: "3133800000009",
      o_csi: %{service_key: 100, gsmscf_address: "14155550300",
default_call_handling: :release_call},
      t_csi: %{service_key: 101, gsmscf_address: "14155550300",
default_call_handling: :continue_call}
    }
  ]
```

Prefixos VLR de Origem

Configuração para distinguir assinantes locais e de roaming. Para uma explicação detalhada da detecção de local/roaming e operações PRN, veja Tratamento de Assinantes de Roaming no Guia HLR.

Parâmetro	Tipo	Padrão	Obrigatório	Descrição
`home_vlr_prefixes`	Lista	`["5551231"]`	Não	Prefixos GT do VLR considerados rede "local"

Exemplo:

```
config :omniss7,
  home_vlr_prefixes: ["5551231", "5551234"]
```

Parâmetros do Modo SMSc

Configuração para o modo Centro de SMS.

Configuração da API SMSc

Parâmetro	Tipo	Padrão
`smsc_api_base_url`	String	`"https://omnimessage.example`
`smsc_api_status_path`	String	`"/api/status"`
`smsc_name`	String	`"{hostname}_SMSc"`
`smsc_service_center_gt_address`	String	-

> **Verificação SSL:** Não há chave `smsc_api_verify_ssl`. Ela não é lida em nenhum lugar no código. A verificação do certificado TLS para chamadas de API de backend é regida pelo campo `verify_ssl` do mapa genérico `:api` (veja Verificação SSL do Backend da API).

Exemplo:

```
config :omniss7,
  smsc_api_base_url: "https://omnimessage.example.com:8443",
  smsc_api_status_path: "/api/status",
  smsc_name: "ipsmgw",
  smsc_service_center_gt_address: "55512341112"
```

Nota: O registro no frontend ocorre a cada **5 minutos** (codificado) via módulo `SMS.FrontendRegistry`.

Configuração de Auto-Flush

Parâmetro	Tipo	Padrão	Obrigatório	Descri
`auto_flush_enabled`	Booleano	`true`	Não	Habilitar processamer automático d SMS
`auto_flush_interval`	Inteiro	`10_000`	Não	Intervalo de processamer fila em miliss
`auto_flush_dest_smsc`	String	-	**Sim**	Nome do SM destino para flush
`auto_flush_tps`	Inteiro	`10`	Não	Taxa de processamer mensagens (transações/

Exemplo:

```
config :omniss7,
  auto_flush_enabled: true,
  auto_flush_interval: 10_000,
  auto_flush_dest_smsc: "ipsmgw",
  auto_flush_tps: 10
```

Parâmetros do Modo STP

Configuração para o modo Ponto de Transferência de Sinal M3UA. Para configuração de roteamento detalhada e exemplos, veja o Guia de Configuração STP.

Servidor STP Autônomo

Parâmetro	Tipo	Padrão	Obrigatório	Descri
`sctp_handler.enabled`	Booleano	`false`	Não	Habilitar servidor SocketHa SCTP autônom
`sctp_handler.local_ip`	Tupla ou Lista	`{127, 0, 0, 1}`	Não	Endereço para escu conexões único: `{1 0, 0, 1}` Múltiplos para multihom SCTP: `[{ 0, 0, 1} {10, 0, 2}]`
`sctp_handler.local_port`	Inteiro	`2905`	Não	Porta par escutar
`sctp_handler.point_code`	Inteiro	-	**Sim** (se habilitado)	Código d ponto SS deste STF

Exemplo (IP Único):

```
config :omniss7,
  sctp_handler: %{
    enabled: true,
    local_ip: {10, 179, 4, 10},
    local_port: 2905,
    point_code: 100
  }
```

Exemplo (Multihoming SCTP):

```
config :omniss7,
  sctp_handler: %{
    enabled: true,
    # Múltiplos IPs para redundância
    local_ip: [{10, 179, 4, 10}, {10, 179, 4, 11}],
    local_port: 2905,
    point_code: 100
  }
```

> **Nota**: Para informações detalhadas sobre configuração e benefícios de multihoming SCTP, veja Multihoming SCTP no Guia Comum.

Roteamento de Título Global

Parâmetro	Tipo	Padrão	Obrigatório	Descrição
`enable_gt_routing`	Booleano	`false`	Não	Habilitar roteamento GT além do roteamento PC

Exemplo:

```
config :omniss7,
  enable_gt_routing: true
```

Configuração de Pares M3UA/M2PA

Os pares são configurados via a lista `m3ua_peers` (suporta os protocolos M3UA e M2PA). Para exemplos completos de configuração, veja o Guia de Configuração STP e Suporte ao Protocolo M2PA.

Parâmetros Comuns de Par:

Parâmetro	Tipo	Padrão	Obrigatório	De
`peer_id`	Inteiro	-	**Sim**	Ident única
`name`	String	-	**Sim**	Nom do pa
`protocol`	Átomo	`:m3ua`	Não	Tipo proto `:m3u`
`role`	Átomo	`:client`	Não	Pape cone `:cli` `:ser` ou `:`
`local_ip`	Tupla ou Lista	-	**Sim**	Ende local vincu
`local_port`	Inteiro	-	**Sim**	Porta (M3U M2PA
`remote_ip`	Tupla ou Lista	-	**Sim**	Ende remo
`remote_port`	Inteiro	-	**Sim**	Porta remo
`routing_context`	Inteiro	-	Não	Ident conte rotea M3UA M3UA

Parâmetro	Tipo	Padrão	Obrigatório	De
`point_code`	Inteiro	-	**Sim**	Códig local
`network_indicator`	Átomo	`:international`	Não	Indic rede: `:int` ou `:n`
`initiate_connection`	Booleano	`true`	Não	Se de cone

Parâmetros Específicos do M2PA:

Parâmetro	Tipo	Padrão	Obrigatório	Descrição
`adjacent_point_code`	Inteiro	-	**Sim** (M2PA)	Código de ponto do par adjacente

Gerenciamento de Socket:

O M2PA usa automaticamente o SCTP.SocketHandler para gerenciamento de socket compartilhado. Todos os pares M2PA usam o socket compartilhado, o que permite que múltiplos pares compartilhem eficientemente a mesma porta SCTP. Para configuração detalhada, veja Requisitos de Socket M2PA.

Exemplo (Par M3UA):

```
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "HLR_East",
      protocol: :m3ua,
      role: :sgp,
      local_ip: {10, 179, 4, 10},
      local_port: 2905,
      remote_ip: {10, 179, 4, 20},
      remote_port: 2905,
      point_code: 100,
      network_indicator: :international
    }
  ]
```

Exemplo (Par M2PA):

```
config :omniss7,
  sctp_handler: %{
    enabled: true,
    local_ip: {10, 179, 4, 10},
    local_port: 3565,
    point_code: 100
  },
  m3ua_peers: [
    %{
      peer_id: 2,
      name: "M2PA_Link_STP_West",
      protocol: :m2pa,
      role: :client,
      local_ip: {10, 179, 4, 10},
      local_port: 3565,
      remote_ip: {10, 179, 4, 30},
      remote_port: 3565,
      point_code: 100,
      adjacent_point_code: 200
    }
  ]
```

Rotas de Código de Ponto M3UA

Configuração de roteamento de Código de Ponto. As rotas definem qual par usar para alcançar códigos de ponto de destino específicos.

Parâmetro	Tipo	Padrão	Obrigatório	Descrição
`m3ua_routes`	Lista de Mapas	`[]`	Não	Lista de rotas de código de ponto. Se não especificado, as rotas são geradas automaticamente a partir dos códigos de ponto dos pares.

Formato da Rota: Cada rota em `m3ua_routes` deve ser um mapa com:

- `dest_pc`: Código de ponto de destino (Inteiro)
- `peer_id`: ID do par para roteamento (Inteiro)
- `priority`: Prioridade da rota - valor mais baixo = maior prioridade (Inteiro)
- `network_indicator`: Indicador de rede (Átomo): `:international` ou `:national`

Exemplo:

```
config :omniss7,
  m3ua_routes: [
    # Rota para PC 100 via par 1 (maior prioridade)
    %{dest_pc: 100, peer_id: 1, priority: 1, network_indicator:
:international},
    # Rota para PC 200 via par 2
    %{dest_pc: 200, peer_id: 2, priority: 1, network_indicator:
:international},
    # Balanceamento de carga: mesmo dest_pc com prioridades
diferentes
    %{dest_pc: 300, peer_id: 3, priority: 1, network_indicator:
:international},
    %{dest_pc: 300, peer_id: 4, priority: 2, network_indicator:
:international}
  ]
```

Rotas de Título Global M3UA

Roteamento baseado em prefixo de Título Global com transformação avançada de parâmetros SCCP. O maior prefixo correspondente é usado primeiro, depois a prioridade.

Parâmetro	Tipo	Padrão	Obrigatório	Descrição
`m3ua_gt_routes`	Lista de Mapas	`[]`	Não	Lista de regras de roteamento de Título Global com transformações opcionais

Formato da Rota: Cada rota em `m3ua_gt_routes` deve ser um mapa com:

Parâmetros Básicos:

- `gt_prefix`: Prefixo de Título Global a ser correspondido (String) - string vazia corresponde a todos
- `peer_id`: ID do par para roteamento (Inteiro) - use 0 para DESCARTAR tráfego

- `priority`: Prioridade da rota - valor mais baixo = maior prioridade (Inteiro)
- `description`: Descrição legível por humanos (String)

Parâmetros Opcionais de Correspondência:

- `source_ssn`: Correspondência do Número do SubSistema de origem (Inteiro)
- `source_tt`: Correspondência do Tipo de Tradução de origem (Inteiro)
- `source_npi`: Correspondência do Indicador do Plano de Numeração de origem (Inteiro)
- `source_nai`: Correspondência do Indicador da Natureza do Endereço de origem (Inteiro)

Parâmetros Opcionais de Transformação:

- `dest_ssn`: Transformar SSN na mensagem encaminhada (Inteiro)
- `dest_tt`: Transformar Tipo de Tradução na mensagem encaminhada (Inteiro)
- `dest_npi`: Transformar Indicador do Plano de Numeração na mensagem encaminhada (Inteiro)
- `dest_nai`: Transformar Indicador da Natureza do Endereço na mensagem encaminhada (Inteiro)
- `route_on_ssn`: Inverter o indicador de roteamento da Parte Chamadora para roteamento por SSN (Booleano, padrão `false`). Deixe `false` para roteamento de GT de trânsito para que o nó downstream execute GTT (ITU-T Q.714 §2.4); defina `true` **somente** quando este STP for o tradutor final entregando a um endpoint apenas SSN. Veja Guia STP → Roteamento por SSN Após GTT.

Valores Comuns:

- **Tipo de Tradução (TT):** 0=Desconhecido, 1=Internacional, 2=Nacional, 3=Específico da Rede
- **Plano de Numeração (NPI):** 0=Desconhecido, 1=ISDN(E.164), 6=Celular(E.212)
- **Natureza do Endereço (NAI):** 0=Desconhecido, 1=Assinante, 3=Nacional, 4=Internacional

- **Número do Subsistema (SSN):** 6=HLR, 7=VLR, 8=MSC, 9=EIR, etc.

Exemplo:

```
config :omniss7,
  m3ua_gt_routes: [
    # Roteamento básico por prefixo
    %{gt_prefix: "1234", peer_id: 1, priority: 1, description:
"Números dos EUA"},
    %{gt_prefix: "44", peer_id: 2, priority: 1, description:
"Números do Reino Unido"},

    # Transformação do Tipo de Tradução
    %{
      gt_prefix: "61",
      peer_id: 3,
      priority: 1,
      description: "Números australianos: transformação TT 0→1",
      source_tt: 0,  # Correspondência TT=0 (Desconhecido)
      dest_tt: 1     # Transformar para TT=1 (Internacional)
    },

    # Transformação NPI
    %{
      gt_prefix: "49",
      peer_id: 1,
      priority: 1,
      description: "Números alemães: conversão NPI de
Celular→ISDN",
      source_npi: 6,  # Correspondência NPI=6 (Celular/E.212)
      dest_npi: 1     # Transformar para NPI=1 (ISDN/E.164)
    },

    # Transformação combinada com roteamento SSN
    %{
      gt_prefix: "86",
      source_ssn: 8,   # Correspondência SSN=8 (MSC)
      peer_id: 3,
      dest_ssn: 6,     # Reescrever para SSN=6 (HLR)
      priority: 1,
      description: "Tráfego chinês: Normalização completa",
      source_tt: 0,
      dest_tt: 2,
      source_npi: 6,
      dest_npi: 1,
      source_nai: 4,
      dest_nai: 3
```

```
    },

    # Rota padrão/alternativa
    %{
      gt_prefix: "",
      peer_id: 1,
      priority: 99,
      description: "Rota padrão de fallback"
    }
  ]
```

Parâmetros do Modo de Gateway CAMEL

Configuração para o modo de Gateway CAMEL (protocolo CAP).

Flags do Modo CAMEL

Ative os recursos CAMEL/CAP (defina `cap_client_enabled: true` e `camelgw_mode_enabled: true` em Flags de Modo Operacional).

Configuração do Protocolo CAP

Parâmetro	Tipo	Padrão	Obrigatório	Descriçã
`cap_version`	Átomo	`:v2`	Não	Versão do protocolo CAP: `:v1`, `:v2`, `:v3` ou `:v4`
`camel_gsmscf_gt_address`	String	(usa GT chamado)	Não	Título Global gsmSCF padrão para respostas CAMEL

Mapeamento da Versão CAP:

- `:v1` → OID do Contexto da Aplicação: 0.4.0.0.1.0.50.0
- `:v2` → OID do Contexto da Aplicação: 0.4.0.0.1.0.50.1 (padrão - mais amplamente suportado)
- `:v3` → OID do Contexto da Aplicação: 0.4.0.0.1.21.3.4
- `:v4` → OID do Contexto da Aplicação: 0.4.0.0.1.23.3.4

Nota: As solicitações recebidas são detectadas automaticamente a partir de seu OID de contexto de aplicação e as respostas correspondem à versão da solicitação.

Exemplo:

```
config :omniss7,
  cap_client_enabled: true,
  camelgw_mode_enabled: true,
  cap_version: :v2,
  camel_gsmscf_gt_address: "99988411553"
```

Integração CGrateS

Integração de cobrança em tempo real com CGrateS para faturamento pré-pago/pós-pago.

Parâmetro	Tipo	Padrão	Obrigatório	D
`cgrates_enabled`	Booleano	`false`	Não	Ativa integ CGra
`cgrates_url`	String	-	**Sim** (se ativado)	URL JSON CGra
`cgrates_tenant`	String	`"cgrates.org"`	Não	Ident locat CGra
`cgrates_request_type`	String	`"*prepaid"`	Não	Tipo `"*pr` `"*po` `"*ps`
`cgrates_timeout`	Inteiro	`5000`	Não	Temp solici CGra milis

Exemplo:

```
config :omniss7,
  cgrates_enabled: true,
  cgrates_url: "http://localhost:2080/jsonrpc",
  cgrates_tenant: "cgrates.org",
  cgrates_request_type: "*prepaid",
  cgrates_timeout: 5000
```

Conexão CAP M3UA

O Gateway CAMEL usa uma conexão M3UA separada para operações CAP.

Parâmetro	Tipo	Padrão	Obrigatório	Descrição
`cap_client_m3ua`	Mapa	-	**Sim**	Configuração da conexão CAP M3UA (mesma estrutura que `map_client_m3ua`, incluindo parâmetros opcionais `opc` e `dpc`)

Exemplo:

```
config :omniss7,
  cap_client_m3ua: %{
    mode: "ASP",
    callback: {CapClient, :handle_payload, []},
    process_name: :camelgw_client_asp,
    local_ip: {10, 5, 198, 200},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 4,
    opc: 5013,    # Código de Ponto de Origem
    dpc: 5011     # Código de Ponto de Destino
  }
```

Parâmetros de NAT de Título Global

A Tradução de Endereço de Rede de Título Global permite diferentes GTs de resposta com base no prefixo da parte chamadora, prefixo da parte chamada ou ambos. As regras são correspondidas por peso (menor = maior prioridade), depois pela especificidade do prefixo (prefixo combinado mais longo = mais específico). Para explicação detalhada e exemplos, consulte o Guia de NAT de Título Global.

Parâmetro	Tipo	Padrão	Obrigatório	Descrição
`gt_nat_enabled`	Booleano	`false`	Não	Ativar/desativar o recurso GT NAT
`gt_nat_rules`	Lista de Mapas	`[]`	**Sim** (se ativado)	Lista de regras de NAT GT com correspondência de prefixo

Formato da Regra: Cada regra em `gt_nat_rules` deve ser um mapa com:

- `calling_prefix`: Prefixo de string para corresponder ao GT chamador (opcional)
- `called_prefix`: Prefixo de string para corresponder ao GT chamado (opcional)
- `weight`: Prioridade inteira (menor = maior prioridade) - padrão: 100
- `response_gt`: Título Global a ser usado nas respostas (obrigatório)

Prioridade de Correspondência:

1. As regras são correspondidas pelo `weight` (valor menor = maior prioridade)
2. Se os pesos forem iguais, o comprimento do prefixo combinado mais longo vence

3. Tanto `calling_prefix` quanto `called_prefix` podem ser usados juntos para correspondência precisa

Exemplo:

```
config :omniss7,
  gt_nat_enabled: true,
  gt_nat_rules: [
    # Alta prioridade: Correspondência de ambos chamando de "8772"
E chamado para "555"
    %{calling_prefix: "8772", called_prefix: "555", weight: 1,
response_gt: "111111"},

    # Média prioridade: Correspondência apenas chamando de "8772"
    %{calling_prefix: "8772", weight: 10, response_gt:
"99988411553"},

    # Média prioridade: Correspondência apenas chamado para "555"
    %{called_prefix: "555", weight: 10, response_gt:
"99988411554"},

    # Correspondência apenas chamando de "8773"
    %{calling_prefix: "8773", weight: 10, response_gt:
"99988411554"},

    # Regra de fallback curinga (corresponde a tudo, maior peso)
    %{weight: 100, response_gt: "99988411555"}
  ]
```

Veja Também: Guia de GT NAT para uso detalhado e exemplos.

Parâmetros de Conexão M3UA

Configuração de conexão M3UA para modo cliente MAP. Para uso detalhado e exemplos, consulte o Guia do Cliente MAP.

Parâmetro	Tipo	Padrão	Obrigató
`map_client_m3ua.mode`	String	-	**Sim**
`map_client_m3ua.callback`	Tupla	-	**Sim**
`map_client_m3ua.process_name`	Átomo	-	**Sim**
`map_client_m3ua.local_ip`	Tupla ou Lista	-	**Sim**
`map_client_m3ua.local_port`	Inteiro	`2905`	**Sim**
`map_client_m3ua.remote_ip`	Tupla ou Lista	-	**Sim**
`map_client_m3ua.remote_port`	Inteiro	`2905`	**Sim**
`map_client_m3ua.routing_context`	Inteiro	-	**Sim**
`map_client_m3ua.opc`	Inteiro	`5013`	Não

Parâmetro	Tipo	Padrão	Obrigató
`map_client_m3ua.dpc`	Inteiro	`5011`	Não
`map_client_m3ua.receive_watchdog`	Booleano	`true`	Não
`map_client_m3ua.receive_watchdog_idle`	Inteiro	`15`	Não

Exemplo (IP Único):

```
config :omniss7,
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1,
    opc: 5013,    # Código de Ponto de Origem (2-114-5)
    dpc: 5011     # Código de Ponto de Destino (2-114-3)
  }
```

> **Formato do Código de Ponto**: Códigos de ponto no formato `X-Y-Z` convertem-se em inteiros como: `(X * 2048) + (Y * 8) + Z`. Por exemplo, `2-114-5` = `(2 * 2048) + (114 * 8) + 5` = `5013`.

Exemplo (Multihoming SCTP):

```
config :omniss7,
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    # Múltiplos IPs locais para redundância
    local_ip: [{10, 179, 4, 11}, {10, 179, 4, 12}],
    local_port: 2905,
    # Múltiplos IPs remotos para redundância STP
    remote_ip: [{10, 179, 4, 10}, {10, 179, 4, 20}],
    remote_port: 2905,
    routing_context: 1
  }
```

> **Nota**: Para informações detalhadas sobre configuração e benefícios de multihoming SCTP, consulte Multihoming SCTP no Guia Comum.

Watchdog de Recebimento

O **watchdog de recebimento** monitora conexões SCTP em busca de sockets zumbis. Um socket zumbi é uma associação que permanece em estado ESTABELECIDO no nível do SO, mas onde a extremidade remota parou silenciosamente de enviar dados. Sem o watchdog, o nó pode não detectar uma conexão morta até que a próxima tentativa de envio falhe.

Qualquer payload SCTP recebido redefine o temporizador ocioso, incluindo respostas M3UA BEAT, mensagens NOTIFY e dados de aplicação. De acordo com RFC 4666 §3.8, M3UA BEAT é opcional. SCTP já realiza heartbeat obrigatório na camada de transporte. Quando o SG remoto não envia tráfego periódico na camada de aplicação (e não envia M3UA BEATs), desative o watchdog para evitar ciclos de reconexão desnecessários.

Parâmetro	Tipo	Padrão	Descrição
`receive_watchdog`	Booleano	`true`	Ativar ou desativar o watchdog de recebimento. Quando `false`, conexões ociosas nunca são encerradas pelo watchdog; o heartbeat da camada de transporte SCTP ainda opera normalmente.
`receive_watchdog_idle`	Inteiro	`15`	Segundos de inatividade antes que o watchdog encerre a conexão e acione uma reconexão. Não tem efeito quando `receive_watchdog: false`.

Exemplo: desativar o watchdog para um cliente ASP:

```
config :omniss7,
  map_client_m3ua: %{
    mode: "ASP",
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1,
    receive_watchdog: false        # Desativar - SG remoto não
envia tráfego periódico
  }
```

Exemplo: personalizar o limite ocioso:

```
config :omniss7,
  map_client_m3ua: %{
    mode: "ASP",
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1,
    receive_watchdog: true,
    receive_watchdog_idle: 30      # Encerrar após 30 s de
silêncio (padrão: 15 s)
  }
```

Status do Subsistema SCCP

OmniSS7 responde a Testes de Status do Subsistema de gerenciamento SCCP (SST) a partir do status real do subsistema local consultado (ITU-T Q.714 §5.3.4). A chave `local_subsystems` declara quais Números de Subsistema (SSNs) este nó realmente hospeda. Um SST para um SSN listado é respondido como `Subsystem-Allowed` (SSA); qualquer outro SSN é respondido como `Subsystem-Prohibited` (SSP).

Parâmetro	Tipo	Padrão	Obrigatório	Descrição
`local_subsystems`	Lista de Inteiros ou `:all`	`:all`	Não	SSNs SCCP locais em serviço usados para responder ao SST. `:all` (o padrão) trata cada subsistema como disponível. Isso se adequa a um relay puro que não hospeda subsistemas. Operadores que terminam SSNs específicos devem listá-los para que o SST reflita a realidade.

SSNs Comuns: 6 = HLR, 7 = VLR, 8 = MSC, 9 = EIR, 146 = gsmSCF/CAMEL.

Exemplo:

```
config :omniss7,
  local_subsystems: [6, 7, 8, 146]
```

Filtragem de Gateway

A filtragem de gateway (o firewall SCCP) permite ou nega tráfego de trânsito antes que ele seja roteado. As regras correspondem ao peer de origem, OPC,

DPC, Indicador de Serviço, prefixo GTA da Parte Chamadora/Chamado SCCP e SSN; a primeira regra habilitada correspondente (prioridade mais baixa) decide `allow`/`deny`. Quando nenhuma regra corresponde, a ação padrão abaixo se aplica. Mensagens negadas são descartadas e contadas como `m3ua_stp_routing_failures_total{reason="screened"}`. Consulte o Guia STP para o modelo completo.

Parâmetro	Tipo	Padrão	Obrigatório	Descriç
`screening_default_action`	`:allow` ou `:deny`	`:allow`	Não	Ação aplicada a uma mensager de trânsit quando nenhuma regra de filtragem correspon `:allow` (padrão) é uma postu padrão de permissão `:deny` é padrão de negação (lista de permissão Definir `:deny` tor a filtrager ativa mes sem regra configura

Exemplo:

```
config :omniss7,
  # Padrão de negação: apenas o tráfego que corresponde a uma
regra de permissão explícita é encaminhado.
  screening_default_action: :deny
```

As regras em si são mantidas na tabela Mnesia `ss7_screen_rule` e gerenciadas programaticamente; apenas a ação padrão é definida via configuração.

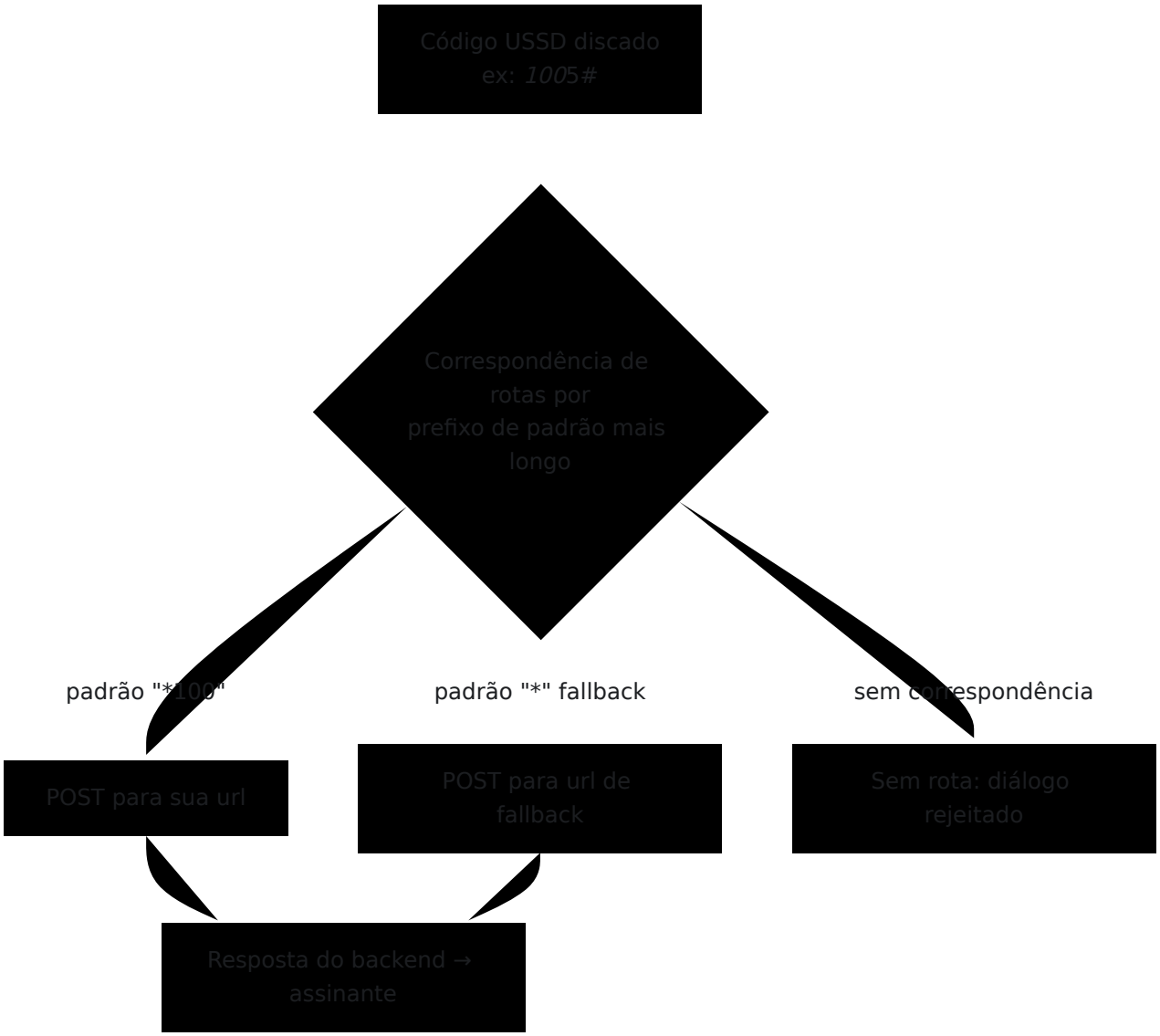
Parâmetros do Gateway USSD

O Gateway USSD conecta USSD iniciado pela rede e USSD iniciado pelo celular a um backend HTTP/JSON. Ative-o com `ussd_gateway_enabled: true` (veja Flags de Modo Operacional), em seguida, configure o mapa `ussd_gateway`. Para o protocolo de callback completo e ciclo de vida da sessão, consulte o Guia do Gateway USSD.

Parâmetro	Tipo	Padrão	Obrigatório	
`ussd_gateway.routes`	Lista de Mapas	`[]`	**Sim** (se ativado)	Re ro sh de er `{p` `<p` `<c` **O lo co ve** pa es fa
`ussd_gateway.session_timeout_ms`	Inteiro	`180_000`	Não	Te to se m m é ex
`ussd_gateway.turn_timeout_ms`	Inteiro	`30_000`	Não	Te (m ur as de m
`ussd_gateway.http_timeout_ms`	Inteiro	`5_000`	Não	Te so HT

Parâmetro	Tipo	Padrão	Obrigatório	
				ca ap
`ussd_gateway.max_text_length`	Inteiro	`182`	Não	Ma ca st US Te sã ur re

Roteamento: Quando um código USSD chega, o gateway classifica as rotas configuradas por comprimento de `pattern` em ordem decrescente. O gateway então seleciona a primeira rota cujo `pattern` é um prefixo do código discado. Isso fornece correspondência determinística de prefixo mais longo.



Exemplo:

```
config :omniss7,
  ussd_gateway_enabled: true,
  ussd_gateway: %{
    routes: [
      %{pattern: "*100", url: "http://balance-app:9000/ussd"},
      %{pattern: "*200", url: "http://topup-app:9000/ussd"},
      %{pattern: "*",    url: "http://default-app:9000/ussd"}
    ],
    session_timeout_ms: 180_000,
    turn_timeout_ms: 30_000,
    http_timeout_ms: 5_000,
    max_text_length: 182
  }
```

Flags de Tempo de Execução / Supervisão

Essas flags controlam quanto da árvore de supervisão é iniciada. Elas são úteis para executar o OmniSS7 sem interface (como uma dependência de biblioteca) ou para remover a camada web em implantações restritas.

Parâmetro	Tipo	Padrão	Obrigatório	Des
`start_web_interfaces`	Booleano	`true`	Não	Iniciar a int controle, o REST `:api_` logger de e Quando `fa` o logger de (`SS7.Event` Pode ser su inicializaçã variável de `ENABLE_WEB` (`true`/`1` ou
`start_http_server`	Booleano	`true`	Não	Iniciar o ser Plug.Cowbo `api_port` (`/api/*`, `/m` `/swagger`). Servidor AP
`api_port`	Inteiro	`9090`	Não	Porta TCP p servidor HT Plug.Cowbo `/api/*`, `/m` `/swagger`). alvo de cole Prometheus alterá-lo.
`headless`	Booleano	`false`	Não	Supervisão apenas de Quando `tru` OmniSS7 in os supervis codec/trans

Parâmetro	Tipo	Padrão	Obrigatório	Des
				(registros M USSD e sup dinâmicos): sem cliente sem manipu sem servido Destinado a codecs Omn uma depen
`vlr_cache_ttl_seconds`	Inteiro	`3600`	Não	Tempo de v (segundos) VLR em cac caminho de SMSc.

Exemplo (modo biblioteca sem interface):

```
config :omniss7,
  headless: true
```

Exemplo (sem camada web, apenas API REST):

```
config :omniss7,
  start_web_interfaces: false,
  start_http_server: true
```

Parâmetros de Infraestrutura

Configuração para componentes de infraestrutura do sistema, incluindo licenciamento, interface web, servidor API e logging.

Configuração de Licença

Parâmetro	Tipo	Padrão	Obrigatór
`license_client.license_server_api_urls`	Lista de Strings	-	**Sim**
`license_client.licensee`	String	-	**Sim**

Exemplo:

```
config :license_client,
  license_server_api_urls: ["https://localhost:10443/api"],
  licensee: "Omnitouch Network Services Pty. Ltd."
```

Interface Web do Painel de Controle

Parâmetro	Tipo	P
`control_panel.parent_application_readable_name`	String	`"OmniSS7 St`
`control_panel.use_additional_pages`	Lista de Tuplas	`[]`
`control_panel.page_order`	Lista de Strings	`[]`
`ControlPanelWeb.Endpoint.url.host`	String	`"0.0.0.0"`
`ControlPanelWeb.Endpoint.https.port`	Inteiro	`8087`

Parâmetro	Tipo	P
`ControlPanelWeb.Endpoint.https.keyfile`	String	`"priv/cert/`
`ControlPanelWeb.Endpoint.https.certfile`	String	`"priv/cert/`

Exemplo:

```
config :control_panel, ControlPanelWeb.Endpoint,
  url: [host: "0.0.0.0", path: "/"],
  https: [
    port: 8087,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ],
  parent_application_readable_name: "OmniSS7 Stack STP"

config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "Pares"}
  ],
  page_order: ["/events", "/m3ua", "/application",
"/configuration"]
```

Servidor API REST

Parâmetro	Tipo	Padrão	Ob
`start_http_server`	Booleano	`true`	Nã
`api_ex.api.port`	Inteiro	`8445`	Nã
`api_ex.api.listen_ip`	String	`"0.0.0.0"`	Nã
`api_ex.api.product_name`	String	`"OmniSS7"`	Nã
`api_ex.api.title`	String	`"API - OmniSS7"`	Nã
`api_ex.api.hostname`	String	`"localhost"`	Nã
`api_ex.api.enable_tls`	Booleano	`true`	Nã
`api_ex.api.tls_cert_path`	String	`"priv/cert/omnitouch.crt"`	Nã
`api_ex.api.tls_key_path`	String	`"priv/cert/omnitouch.pem"`	Nã

Exemplo:

```
config :omniss7,
  start_http_server: true

config :api_ex,
  api: %{
    port: 8445,
    listen_ip: "0.0.0.0",
    product_name: "OmniSS7",
    title: "API - OmniSS7",
    hostname: "localhost",
    enable_tls: true,
    tls_cert_path: "priv/cert/omnitouch.crt",
    tls_key_path: "priv/cert/omnitouch.pem"
  }
```

Portas e transportes em um relance:

Porta	Transporte	Servidor	Propósito
9090	HTTP (Plug.Cowboy)	`APIhandler`	MAP/CAP REST `/ap` `/metrics`, `/swagg` `/swagger.json`. Alternado por `start_http_serve`
8087	HTTPS (Bandit/Phoenix)	`ControlPanelWeb.Endpoint`	UI do painel de co da web. Parte da c web (`start_web_inter`
8445	HTTPS / TLS	`:api_ex` controladores REST	Status, CRUD de roteamento, links, assinantes, visuali CAMEL/SCTP/M3UA da camada web.

Endpoints da API:

- API REST MAP/CAP + Swagger: `http://[server-ip]:9090/...` (veja Guia da API)
- Controladores REST `:api_ex`: `https://[server-ip]:8445/api/*`
- Painel de controle da web: `https://[server-ip]:8087/`
- Métricas do Prometheus: `http://[server-ip]:9090/metrics`

Verificação SSL do Backend da API

Chamadas de saída para APIs HLR/SMSc do backend **não** usam chaves `*_verify_ssl` por alvo. A verificação de certificado é controlada pelo mapa genérico `:api`:

Parâmetro	Tipo	Padrão	Descrição
`api.base_url`	String	recai em `smsc_api_base_url`	URL base para o cliente API genérico.
`api.timeout`	Inteiro	`5000`	Tempo limite da solicitação em milissegundos.
`api.verify_ssl`	Booleano	`false`	Se deve verificar os certificados TLS do backend. **Este é o único interruptor de verificação SSL.** Não há `hlr_api_verify_ssl` ou `smsc_api_verify_ssl`.

Exemplo:

```
config :omniss7,
  api: %{
    base_url: "https://backend.example.com:8443",
    timeout: 5000,
    verify_ssl: false
  }
```

Configuração de Registro

Parâmetro	Tipo	Padrão	Obr
`logger.level`	Átomo	`:debug`	Não
`logger.backends`	Lista	`[:console]`	Não
`logger.default_formatter.format`	String	-	Não
`logger.default_formatter.metadata`	Lista	`[]`	Não
`logger.default_formatter.truncate`	Átomo/Integiro	`:infinity`	Não

Exemplo:

```
config :logger,
  level: :debug,
  backends: [:console, SS7.Web.LoggerBackend]

config :logger, :default_formatter,
  format: "[$date] [$time] [$level] $message\n",
  metadata: [:error_code, :file],
  truncate: :infinity
```

Parâmetros do Banco de Dados

Configuração para persistência do banco de dados Mnesia.

Parâmetro	Tipo	Padrão	Obrigatório	Descri
`mnesia_storage_type`	Átomo	`:disc_copies`	Não	Tipo de armazena Mnesia: `:disc_cop` ou `:ram_c`

Exemplo:

```
config :omniss7,
  mnesia_storage_type: :disc_copies  # Produção
  # mnesia_storage_type: :ram_copies  # Apenas para teste
```

Tipos de Armazenamento:

- `:disc_copies` - Armazenamento em disco persistente (sobrevive a reinicializações) - **Recomendado para produção**
- `:ram_copies` - Apenas em memória (perdido na reinicialização) - Apenas para teste

Tabelas Mnesia:

- `m3ua_peer` - Conexões de pares M3UA
- `m3ua_route` - Rotas de Código de Ponto
- `m3ua_gt_route` - Rotas de Título Global

Localização: diretório `Mnesia.{node_name}/`

Valores Codificados

Os seguintes valores estão **codificados no código-fonte** e não podem ser alterados via configuração.

Timeouts

Valor	Impacto	Solução alternativa
Timeout de solicitação MAP: **10 segundos**	Todas as operações MAP expiram após 10s	Modificar o código-fonte
Timeout ISD: **10 segundos**	Cada mensagem ISD expira após 10s	Modificar o código-fonte

Servidor HTTP

Valor	Impacto	Solução alternativa
IP HTTP: **0.0.0.0**	O servidor de métricas/Swagger escuta em todas as interfaces	Modificar o código-fonte
Porta HTTP: **9090**	O endpoint de métricas/Swagger roda na porta 9090	Definir `config :omniss7, api_port: <port>` (padrão `9090`)

Intervalos de Registro

Valor	Impacto	Solução alternativa
Registro do frontend: **5 minutos**	SMSc registra com o backend a cada 5 min	Modificar o código-fonte

Atualização Automática da UI da Web

Página	Intervalo
Gerenciamento de Roteamento	5 segundos
Assinantes Ativos	2 segundos

Exemplos de Configuração

Configuração Mínima do HLR

```
config :omniss7,
  map_client_enabled: true,
  hlr_mode_enabled: true,
  smsc_mode_enabled: false,

  hlr_api_base_url: "https://omnihss.example.com:8443",
  hlr_api_key: "<Chave da API OmniHSS com o papel
hlr_interworking>",
  hlr_service_center_gt_address: "55512341111",
  smsc_service_center_gt_address: "55512341112",

  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }
```

Configuração Mínima do SMSc

```
config :omniss7,
  map_client_enabled: true,
  hlr_mode_enabled: false,
  smsc_mode_enabled: true,

  smsc_api_base_url: "https://omnimessage.example.com:8443",
  smsc_name: "ipsmgw",
  smsc_service_center_gt_address: "55512341112",

  auto_flush_enabled: true,
  auto_flush_interval: 10_000,
  auto_flush_dest_smsc: "ipsmgw",
  auto_flush_tps: 10,

  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :stp_client_asp,
    local_ip: {10, 179, 4, 12},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }
```

STP com Servidor Autônomo

```
config :omniss7,
  map_client_enabled: true,
  hlr_mode_enabled: false,
  smsc_mode_enabled: false,

  enable_gt_routing: true,
  mnesia_storage_type: :disc_copies,

  sctp_handler: %{
    enabled: true,
    local_ip: {10, 179, 4, 10},
    local_port: 2905,
    point_code: 100
  },

  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :stp_client_asp,
    local_ip: {10, 179, 4, 10},
    local_port: 2906,
    remote_ip: {10, 179, 4, 11},
    remote_port: 2905,
    routing_context: 1
  }
```

Resumo

Total de Parâmetros de Configuração: 85+

Por Categoria:

- Modo Operacional: 6 parâmetros
- Modo HLR: 18 parâmetros
- Modo SMSc: 8 parâmetros

- Modo STP: 5+ parâmetros (mais listas de m3ua_peers, m3ua_routes, m3ua_gt_routes)
- Modo CAMEL GW: 14 parâmetros
- NAT de Título Global: 2 parâmetros
- Conexão M3UA: 8 parâmetros
- Status do Subsistema SCCP: 1 parâmetro
- Gateway USSD: 5 parâmetros
- Tempo de Execução / Supervisão: 4 parâmetros
- Infraestrutura (Licença, Web, API, Registro): 23 parâmetros
- Banco de Dados: 1 parâmetro

Parâmetros Obrigatórios por Modo:

Modo HLR:

- `hlr_api_base_url`
- `hlr_api_key`
- `hlr_service_center_gt_address`
- `smsc_service_center_gt_address`
- Todos os parâmetros `map_client_m3ua.*` (8)

Modo SMSc:

- `smsc_api_base_url`
- `smsc_service_center_gt_address`
- `auto_flush_dest_smsc` (se auto-flush ativado)
- Todos os parâmetros `map_client_m3ua.*` (8)

Modo STP:

- `sctp_handler.point_code` (se o manipulador SCTP ativado)
- `sctp_handler.local_ip`
- `sctp_handler.local_port`

Modo CAMEL GW:

- `cgrates_url` (se CGrateS ativado)

- Todos os parâmetros `cap_client_m3ua.*` (8)

Infraestrutura:

- `license_client.license_server_api_urls`
- `license_client.licensee`

Documentação Relacionada

- **Guia HLR** - Configuração específica do HLR
- **Guia SMSc** - Configuração específica do SMSc
- **Guia STP** - Configuração de roteamento STP
- **Guia da API** - Referência da API REST
- **Guia do Gateway USSD** - Configuração do Gateway USSD e protocolo de callback HTTP
- **Guia da UI da Web** - Documentação da interface da web

Guia de NAT de Título Global

Visão Geral

A Tradução de Endereço de Título Global (GT NAT) é um recurso que permite que o OmniSS7 responda com diferentes endereços de Título Global com base no prefixo GT da parte chamadora, no prefixo GT da parte chamada ou em uma combinação de ambos. Isso é essencial ao operar com múltiplos Títulos Globais e precisar garantir que as respostas utilizem o GT correto com base em qual rede ou par está chamando e/ou qual GT foi chamado.

O Que Há de Novo (GT NAT Aprimorado)

O recurso GT NAT foi aprimorado com novas capacidades poderosas:

Novos Recursos

1. **Correspondência de Prefixo da Parte Chamado**: As regras agora podem corresponder ao `called_prefix` além do `calling_prefix`
2. **Correspondência Combinada**: As regras podem corresponder a prefixos de chamada E de chamada simultaneamente
3. **Priorização Baseada em Peso**: As regras agora usam um campo `weight` (menor = maior prioridade) em vez de apenas o comprimento do prefixo
4. **Correspondência Flexível**: Agora você pode criar regras com:
   - Apenas prefixo de chamada
   - Apenas prefixo de chamada
   - Ambos os prefixos de chamada e chamada
   - Nenhum (regra de curinga/recuo)

Novo Formato de Regra

Campos obrigatórios:

- `weight`: Prioridade inteira (menor = maior prioridade)
- `response_gt`: O GT a ser respondido

Campos opcionais (pelo menos um recomendado para correspondência específica):

- `calling_prefix`: Correspondência no prefixo GT da parte chamadora
- `called_prefix`: Correspondência no prefixo GT da parte chamada

Exemplo:

```
gt_nat_rules: [
  # Regra específica com ambos os prefixos - maior prioridade
  %{calling_prefix: "8772", called_prefix: "555", weight: 1,
response_gt: "111111"},

  # Regras específicas - prioridade média
  %{calling_prefix: "8772", weight: 10, response_gt: "222222"},
  %{called_prefix: "555", weight: 10, response_gt: "333333"},

  # Curva de recuo - menor prioridade
  %{weight: 100, response_gt: "999999"}
]
```

Casos de Uso

Operação Multi-Rede

Quando você tem várias redes parceiras e cada uma espera respostas de um GT específico:

- **Rede A** chama seu GT `111111` e espera respostas de `111111`
- **Rede B** chama seu GT `222222` e espera respostas de `222222`

Sem o GT NAT, você precisaria de instâncias separadas ou roteamento complexo. Com o GT NAT, uma única instância do OmniSS7 pode lidar com isso de forma inteligente.

Cenários de Roaming

Ao operar como um HLR ou SMSc com acordos de roaming:

- Assinantes da **rede doméstica** usam GT `555000`
- **Parceiro de roaming 1** usa GT `555001`
- **Parceiro de roaming 2** usa GT `555002`

O GT NAT garante que cada parceiro receba respostas do GT correto que eles estão configurados para rotear.

Testes e Migração

Durante migrações de rede ou testes:

- Migre gradualmente o tráfego do GT antigo para o GT novo
- Mantenha ambos os GTs durante o período de transição
- Roteie respostas com base em qual GT o chamador usou

Como Funciona

Fluxo de Tradução de Endereço

1. **Solicitação de Entrada**: O OmniSS7 recebe uma mensagem SCCP com:
    - GT da Parte Chamado: `55512341112` (seu GT)
    - GT da Parte Chamadora: `877234567` (GT deles)
2. **Pesquisa GT NAT**: O sistema verifica o GT chamador `877234567` em relação às regras de prefixo configuradas
3. **Correspondência de Prefixo**: Encontra o prefixo correspondente mais longo (por exemplo, `8772` corresponde a `877234567`)

4. **Seleção do GT de Resposta**: Usa `response_gt` da regra correspondente (por exemplo, `55512341112`)

5. **Resposta Enviada**: A resposta SCCP usa:

    - GT da Parte Chamado: `877234567` (invertido - GT deles)
    - GT da Parte Chamadora: `55512341112` (GT NAT'd)

Tipos de Resposta Afetados

O GT NAT se aplica a várias camadas da pilha SS7:

Camada SCCP (Todas as Respostas)

- Endereços GT Chamado/Chamadora SCCP em todas as mensagens de resposta
- Reconhecimentos de ISD (InsertSubscriberData)
- Respostas de UpdateLocation
- Respostas de erro

Camada MAP (Operação-Específica)

- **Respostas SRI-for-SM**: `networkNode-Number` (endereço GT do SMSc)
- **UpdateLocation**: `hlr-Number` nas respostas
- **InsertSubscriberData**: GT HLR em mensagens ISD

Configuração

Configuração Básica

Adicione ao `config/runtime.exs`:

```
config :omniss7,
  # Habilitar GT NAT
  gt_nat_enabled: true,

  # Definir regras de GT NAT
  gt_nat_rules: [
    # Regra 1: Chamadas do prefixo "8772" recebem resposta do
"55512341112"
    %{calling_prefix: "8772", response_gt: "55512341112"},

    # Regra 2: Chamadas do prefixo "8773" recebem resposta do
"55512341111"
    %{calling_prefix: "8773", response_gt: "55512341111"},

    # Regra padrão (prefixo vazio corresponde a tudo)
    %{calling_prefix: "", response_gt: "55512311555"}
  ]
```

Parâmetros de Configuração

Para referência completa de configuração, veja Parâmetros de NAT de Título Global na Referência de Configuração.

Parâmetro	Tipo	Obrigatório	Descrição
`gt_nat_enabled`	Booleano	Sim	Habilitar/desabilitar recurso GT NAT
`gt_nat_rules`	Lista de Mapas	Sim (se habilitado)	Lista de regras de correspondência de prefixo

Formato da Regra

Cada regra é um mapa com as seguintes chaves:

```
%{
  calling_prefix: "8772",      # (Opcional) Prefixo para
corresponder ao GT chamador
  called_prefix: "555",        # (Opcional) Prefixo para
corresponder ao GT chamado
  weight: 10,                  # (Obrigatório) Valor de prioridade
(menor = maior prioridade)
  response_gt: "55512341112"   # (Obrigatório) GT a ser usado nas
respostas
}
```

Campos da Regra:

- **`calling_prefix`** (Opcional): Prefixo de string para corresponder ao GT chamador de entrada
  - A correspondência é feita como uma correspondência de prefixo de string
  - String vazia `""` ou `nil` atua como curinga (corresponde a qualquer GT chamador)
  - Pode ser omitido para corresponder a qualquer GT chamador
- **`called_prefix`** (Opcional): Prefixo de string para corresponder ao GT chamado de entrada
  - A correspondência é feita como uma correspondência de prefixo de string
  - String vazia `""` ou `nil` atua como curinga (corresponde a qualquer GT chamado)
  - Pode ser omitido para corresponder a qualquer GT chamado
- **`weight`** (Obrigatório): Valor de prioridade inteiro
  - Peso menor = maior prioridade (processado primeiro)
  - Deve ser >= 0
  - Usado como critério de ordenação primário para regras de correspondência

- **`response_gt`** (Obrigatório): O endereço de Título Global a ser usado nas respostas
    - Deve ser uma string de número E.164 válida
    - Deve corresponder a um dos seus GTs configurados

Pelo menos um dos `calling_prefix` ou `called_prefix` deve ser especificado para roteamento específico. Ambos podem ser omitidos para uma regra de curinga/recuo.

Lógica de Correspondência de Regras

As regras são avaliadas por **peso primeiro (crescente), depois pela especificidade combinada do prefixo**:

Algoritmo de Correspondência:

1. Filtrar regras onde todos os prefixos especificados correspondem
    - Se `calling_prefix` estiver definido, deve corresponder ao GT chamador
    - Se `called_prefix` estiver definido, deve corresponder ao GT chamado
    - Se ambos estiverem definidos, ambos devem corresponder
    - Se nenhum estiver definido, a regra atua como um curinga
2. Ordenar regras correspondentes por:
    - **Primário**: Peso (crescente - valores menores primeiro)
    - **Secundário**: Comprimento combinado do prefixo (decrescente - mais longo = mais específico)
3. Retornar a primeira regra correspondente

Exemplos:

```
# Regras de exemplo
gt_nat_rules: [
  # Peso 1: Maior prioridade - corresponde a ambos os prefixos
  %{calling_prefix: "8772", called_prefix: "555", weight: 1,
response_gt: "111111"},

  # Peso 10: Prioridade média - regras específicas
  %{calling_prefix: "8772", weight: 10, response_gt: "222222"},  #
Apenas chamada
  %{called_prefix: "555", weight: 10, response_gt: "333333"},    #
Apenas chamada

  # Peso 100: Menor prioridade - curinga de recuo
  %{weight: 100, response_gt: "444444"}  # Corresponde a tudo
]

# Exemplos de correspondência:
# Chamando: "877234567", Chamado: "555123" -> "111111" (peso 1,
ambos correspondem)
# Chamando: "877234567", Chamado: "999999" -> "222222" (peso 10,
apenas chamada)
# Chamando: "999999999", Chamado: "555123" -> "333333" (peso 10,
apenas chamada)
# Chamando: "999999999", Chamado: "888888" -> "444444" (peso 100,
curinga)
```

Exemplos

Exemplo 1: Dois Parceiros de Rede

Cenário: Você opera um SMSc com dois parceiros de rede. Cada um espera respostas de um GT diferente.

```
config :omniss7,
  gt_nat_enabled: true,

  # GT padrão do SMSc (usado quando o GT NAT está desativado ou
nenhuma regra corresponde)
  smsc_service_center_gt_address: "5551000",

  # Regras de GT NAT para parceiros
  gt_nat_rules: [
    # Parceiro A (prefixo 4412) espera respostas do GT 5551001
    %{calling_prefix: "4412", weight: 10, response_gt: "5551001"},

    # Parceiro B (prefixo 4413) espera respostas do GT 5551002
    %{calling_prefix: "4413", weight: 10, response_gt: "5551002"},

    # Padrão: usar GT padrão do SMSc (curinga de recuo)
    %{weight: 100, response_gt: "5551000"}
  ]
```

Fluxo de Tráfego:

```
SRI-for-SM de entrada de 44121234567:
  GT Chamado: 5551001 (seu GT que o Parceiro A usa)
  GT Chamador: 44121234567 (GT do Parceiro A)

Pesquisa GT NAT:
  "44121234567" corresponde ao prefixo "4412"
  GT de resposta selecionado: "5551001"

Resposta SRI-for-SM para 44121234567:
  GT Chamado: 44121234567 (invertido)
  GT Chamador: 5551001 (NAT'd)
  networkNode-Number: 5551001 (na resposta MAP)
```

Exemplo 2: HLR com GTs Regionais

Cenário: HLR nacional com diferentes GTs por região.

```
config :omniss7,
  gt_nat_enabled: true,
  hlr_service_center_gt_address: "555000",  # GT padrão do HLR

  gt_nat_rules: [
    # VLRs da região norte (prefixo 5551)
    %{calling_prefix: "5551", weight: 10, response_gt: "555100"},

    # VLRs da região sul (prefixo 5552)
    %{calling_prefix: "5552", weight: 10, response_gt: "555200"},

    # VLRs da região oeste (prefixo 5553)
    %{calling_prefix: "5553", weight: 10, response_gt: "555300"},

    # Padrão para outras regiões (curinga)
    %{weight: 100, response_gt: "555000"}
  ]
```

Exemplo 3: Cenário de Migração

Cenário: Migrando gradualmente do GT antigo para o novo GT.

```
config :omniss7,
  gt_nat_enabled: true,
  hlr_service_center_gt_address: "123456789",  # GT antigo
(padrão)

  gt_nat_rules: [
    # Redes migradas (já atualizaram suas configurações)
    %{calling_prefix: "555", weight: 10, response_gt:
"987654321"},  # Novo GT
    %{calling_prefix: "666", weight: 10, response_gt:
"987654321"},  # Novo GT

    # Todos os outros ainda usam o GT antigo (curinga)
    %{weight: 100, response_gt: "123456789"}  # GT antigo
  ]
```

Exemplo 4: Correspondência de Prefixo da Parte Chamado (NOVO)

Cenário: Você tem múltiplos GTs para diferentes serviços e deseja responder com o GT correto com base em qual GT foi chamado.

```
config :omniss7,
  gt_nat_enabled: true,

  gt_nat_rules: [
    # Quando chamam seu GT de SMS (5551xxx), responda com esse GT
    %{called_prefix: "5551", weight: 10, response_gt: "555100"},

    # Quando chamam seu GT de Voz (5552xxx), responda com esse GT
    %{called_prefix: "5552", weight: 10, response_gt: "555200"},

    # Quando chamam seu GT de Dados (5553xxx), responda com esse
GT
    %{called_prefix: "5553", weight: 10, response_gt: "555300"},

    # Recuo padrão
    %{weight: 100, response_gt: "555000"}
  ]
```

Fluxo de Tráfego:

```
Solicitação de entrada para GT Chamado: 555100 (seu GT de SMS)
GT Chamador: 441234567 (qualquer chamador)

Pesquisa GT NAT:
  GT Chamado "555100" corresponde ao prefixo "5551"
  GT de resposta selecionado: "555100"

A resposta usa GT Chamador: 555100 (corresponde ao que eles
chamaram)
```

Exemplo 5: Correspondência Combinada de Prefixo Chamador + Chamado (AVANÇADO)

Cenário: Diferentes parceiros chamam diferentes GTs, e você deseja controle detalhado.

```
config :omniss7,
  gt_nat_enabled: true,

  gt_nat_rules: [
    # Parceiro A chamando seu GT de SMS - maior prioridade (peso
1)
    %{calling_prefix: "4412", called_prefix: "5551", weight: 1,
response_gt: "555101"},

    # Parceiro B chamando seu GT de SMS - maior prioridade (peso
1)
    %{calling_prefix: "4413", called_prefix: "5551", weight: 1,
response_gt: "555102"},

    # Qualquer um chamando seu GT de SMS - prioridade média (peso
10)
    %{called_prefix: "5551", weight: 10, response_gt: "555100"},

    # Parceiro A chamando qualquer GT - prioridade média (peso 10)
    %{calling_prefix: "4412", weight: 10, response_gt: "555200"},

    # Recuo padrão - baixa prioridade (peso 100)
    %{weight: 100, response_gt: "555000"}
  ]
```

Exemplos de Correspondência:

```
# Parceiro A chama GT de SMS
Chamando: "441234567", Chamado: "555100"
→ Corresponde à regra de peso 1 (ambos os prefixos) → "555101"

# Parceiro A chama GT de Voz
Chamando: "441234567", Chamado: "555200"
→ Corresponde à regra de peso 10 (apenas chamada) → "555200"

# Chamador desconhecido chama GT de SMS
Chamando: "999999999", Chamado: "555100"
→ Corresponde à regra de peso 10 (apenas chamada) → "555100"

# Chamador desconhecido chama GT de Voz
Chamando: "999999999", Chamado: "555200"
→ Corresponde ao curinga de peso 100 → "555000"
```

Modos Operacionais

O GT NAT funciona em todos os modos operacionais do OmniSS7:

Modo HLR

O GT NAT afeta:

- Respostas de UpdateLocation (GT HLR na resposta)
- Mensagens InsertSubscriberData (GT HLR como parte chamadora)
- Respostas de SendAuthenticationInfo
- Respostas de Cancel Location

Para mais informações sobre operações HLR, veja o Guia de Configuração HLR.

Configuração:

```
config :omniss7,
  hlr_mode_enabled: true,
  hlr_service_center_gt_address: "5551234567",  # GT padrão HLR

  gt_nat_enabled: true,
  gt_nat_rules: [
    %{calling_prefix: "331", weight: 10, response_gt:
"5551234568"},  # França
    %{calling_prefix: "44", weight: 10, response_gt:
"5551234569"},   # Reino Unido
    %{weight: 100, response_gt: "5551234567"}  # Curinga padrão
  ]
```

Modo SMSc

O GT NAT afeta:

- Respostas SRI-for-SM (campo `networkNode-Number`) - veja Detalhes SRI-for-SM
- Reconhecimentos de MT-ForwardSM

Para mais informações sobre operações SMSc, veja o Guia de Configuração SMSc.

Configuração:

```
config :omniss7,
  smsc_mode_enabled: true,
  smsc_service_center_gt_address: "5559999",  # GT padrão SMSc

  gt_nat_enabled: true,
  gt_nat_rules: [
    %{calling_prefix: "1", weight: 10, response_gt: "5559991"},
# América do Norte
    %{calling_prefix: "44", weight: 10, response_gt: "5559992"},
# Reino Unido
    %{calling_prefix: "86", weight: 10, response_gt: "5559993"},
# China
    %{weight: 100, response_gt: "5559999"}  # Curinga padrão
  ]
```

Modo Gateway CAMEL

O GT NAT afeta:

- Todas as respostas em nível SCCP (GT gsmSCF como Parte Chamadora)
- Respostas de operações CAMEL/CAP (InitialDP, EventReportBCSM, etc.)
- Reconhecimentos de RequestReportBCSMEvent
- Respostas de ApplyCharging
- Respostas de Continue

Configuração:

```
config :omniss7,
  camelgw_mode_enabled: true,
  camel_gsmscf_gt_address: "55512341112",  # GT padrão gsmSCF

  gt_nat_enabled: true,
  gt_nat_rules: [
    %{calling_prefix: "555", weight: 10, response_gt:
"55512341111"},  # Rede A
    %{calling_prefix: "666", weight: 10, response_gt:
"55512311555"},  # Rede B
    %{weight: 100, response_gt: "55512341112"}  # Curinga padrão
  ]
```

Caso de Uso: Ao operar como um gsmSCF (Função de Controle de Serviço) para várias redes, cada gsmSSF da rede pode esperar respostas de um GT gsmSCF específico. O GT NAT garante que o GT correto seja usado com base em qual gsmSSF está chamando.

Registro e Depuração

Habilitar Registro de GT NAT

O GT NAT inclui registro automático de todas as traduções:

```
# Nos logs, você verá:
[info] GT NAT [resposta SRI-for-SM]: GT Chamador 877234567 -> GT
de Resposta 55512341112
[info] GT NAT [UpdateLocation ISD]: GT Chamador 331234567 -> GT de
Resposta 55512341111
[info] GT NAT [resposta MAP BEGIN]: GT Chamador 441234567 -> GT de
Resposta 55512311555
```

O campo de contexto mostra onde o NAT foi aplicado:

- `"resposta SRI-for-SM"` - No manipulador SRI-for-SM
- `"UpdateLocation ISD"` - Em mensagens InsertSubscriberData
- `"UpdateLocation END"` - Na resposta UpdateLocation END
- `"resposta MAP BEGIN"` - Respostas MAP BEGIN genéricas
- `"ISD ACK"` - Reconhecimento ISD
- `"resposta de erro HLR"` - Resposta de erro do HLR
- `"resposta CAMEL"` - Respostas de operações CAMEL/CAP (gsmSCF)

Validação

O sistema valida a configuração do GT NAT na inicialização:

```
# Verificar configuração do GT NAT
iex> GtNat.validate_config()
{:ok, [
  %{calling_prefix: "8772", weight: 10, response_gt:
"55512341112"},
  %{calling_prefix: "8773", weight: 10, response_gt:
"55512341111"}
]}

# Verificar se está habilitado
iex> GtNat.enabled?()
true

# Obter todas as regras
iex> GtNat.get_rules()
[
  %{calling_prefix: "8772", weight: 10, response_gt:
"55512341112"},
  %{calling_prefix: "8773", weight: 10, response_gt:
"55512341111"}
]
```

Testando o GT NAT

Teste a lógica do GT NAT programaticamente:

```
# Testar tradução com GT chamador apenas (called_gt é nil)
iex> GtNat.translate_response_gt("877234567", nil, "default_gt")
"55512341112"

# Testar tradu��ão com ambos os GTs chamador e chamado
iex> GtNat.translate_response_gt("877234567", "555123",
"default_gt")
"55512341112"

# Testar com registro (GT chamado nil)
iex> GtNat.translate_response_gt_with_logging("877234567", nil,
"default_gt", "test")
# Logs: GT NAT [test]: GT Chamador 877234567 -> GT de Resposta
55512341112
"55512341112"

# Testar com registro (ambos os GTs)
iex> GtNat.translate_response_gt_with_logging("877234567",
"555123", "default_gt", "test")
# Logs: GT NAT [test]: GT Chamador 877234567, GT Chamado 555123 ->
GT de Resposta 55512341112
"55512341112"

# Testar sem correspondência (retorna padrão)
iex> GtNat.translate_response_gt("999999999", "888888",
"default_gt")
"default_gt"
```

Solução de Problemas

Problema: GT NAT Não Funciona

Verifique 1: Está habilitado?

```
iex> Application.get_env(:omniss7, :gt_nat_enabled)
true  # Deve ser true
```

Verifique 2: As regras estão configuradas?

```
iex> Application.get_env(:omniss7, :gt_nat_rules)
[%{calling_prefix: "8772", response_gt: "55512341112"}, ...]  #
Deve retornar lista
```

Verifique 3: Verifique os logs Procure por "GT NAT" nos logs para ver se as traduções estão acontecendo.

Problema: GT Errado nas Respostas

Sintoma: Respostas usam endereço GT inesperado

Causa: A correspondência de prefixo da regra pode ser muito ampla ou a regra padrão está capturando tráfego

Solução: Revise os pesos e prefixos das regras:

```
# RUIM: Curinga com peso baixo (captura tudo primeiro)
gt_nat_rules: [
  %{weight: 1, response_gt: "111111"},               # Isso
corresponde a tudo primeiro!
  %{calling_prefix: "8772", weight: 10, response_gt: "222222"}  #
Nunca alcançado
]

# BOM: Regras específicas com peso menor, curinga com peso maior
gt_nat_rules: [
  %{calling_prefix: "8772", weight: 10, response_gt: "222222"},  #
Específico, peso baixo
  %{weight: 100, response_gt: "111111"}  # Curinga, peso alto
(recuo)
]
```

Problema: GT NAT Não Aplicado a Tipo de Mensagem Específico

Sintoma: Algumas respostas usam GT NAT'd, outras não

Cobertura Atual:

- □ GT Chamador SCCP (todas as respostas)
- □ Respostas SRI-for-SM (networkNode-Number)
- □ Mensagens ISD de UpdateLocation (GT HLR)
- □ Respostas de UpdateLocation END
- □ Reconhecimentos ISD
- □ Respostas MAP BEGIN

Se um tipo específico de mensagem não usar GT NAT, pode não ter sido implementado ainda. Verifique o código-fonte ou entre em contato com o suporte.

Considerações de Desempenho

Desempenho de Pesquisa

O GT NAT usa correspondência de prefixo simples com complexidade O(n), onde n é o número de regras.

Dicas de desempenho:

- Mantenha a contagem de regras abaixo de 100 para melhor desempenho
- Use prefixos específicos para reduzir a contagem de regras
- A regra padrão (prefixo vazio) deve ser a última

Benchmark (sistema típico):

- 10 regras: < 1µs por pesquisa
- 50 regras: < 5µs por pesquisa
- 100 regras: < 10µs por pesquisa

Uso de Memória

Cada regra requer ~100 bytes de memória:

- 10 regras ≈ 1 KB
- 100 regras ≈ 10 KB

Melhores Práticas

1. Sempre Inclua uma Regra de Recuo Curinga

```
gt_nat_rules: [
  %{calling_prefix: "8772", weight: 10, response_gt: "111111"},
  %{calling_prefix: "8773", weight: 10, response_gt: "222222"},
  %{weight: 100, response_gt: "default_gt"}  # Sempre tenha um
curinga com peso alto
]
```

2. Use Prefixos e Pesos Significativos

```
# BOM: Prefixos claros e específicos com pesos apropriados
%{calling_prefix: "331", weight: 10, response_gt: "..."}  # França
%{calling_prefix: "44", weight: 10, response_gt: "..."}   # Reino
Unido

# RUIM: Prefixos excessivamente amplos ou pesos confusos
%{calling_prefix: "3", weight: 5, response_gt: "..."}     # Muitos
países
%{calling_prefix: "331", weight: 100, response_gt: "..."}  # Peso
deve ser menor para regras específicas
```

3. Documente Suas Regras

```
gt_nat_rules: [
  # Parceiro XYZ - rede do Reino Unido (faixa de GT: 4412xxxxxxx)
  # Peso 10: Prioridade padrão do parceiro
  %{calling_prefix: "4412", weight: 10, response_gt: "5551001"},

  # Parceiro ABC - rede da França (faixa de GT: 33123xxxxxx)
  # Peso 10: Prioridade padrão do parceiro
  %{calling_prefix: "33123", weight: 10, response_gt: "5551002"}
]
```

4. Teste Antes da Implantação

```
# Teste no iex antes de implantar
iex> GtNat.translate_response_gt("44121234567", nil, "default")
"5551001"  # Resultado esperado

# Teste com GT chamado
iex> GtNat.translate_response_gt("44121234567", "555123",
"default")
"5551001"  # Resultado esperado
```

5. Monitore os Logs

Habilite o registro em nível INFO para ver todas as traduções de GT NAT em produção.

Integração com Outros Recursos

Modo STP

O GT NAT funciona independentemente do roteamento STP. O STP roteia com base em códigos de ponto e GTs de destino, enquanto o GT NAT lida com endereçamento de resposta.

Para mais informações sobre roteamento STP, veja o Guia de Configuração STP.

Integração CAMEL

O GT NAT é **totalmente integrado** com operações CAMEL/CAP:

Camada SCCP:

- GT da Parte Chamadora em todas as respostas CAMEL
- Aplicado automaticamente com base no GT gsmSSF de entrada

Configuração:

- `camel_gsmscf_gt_address` - GT padrão gsmSCF (opcional)
- Se não configurado, usa o GT da Parte Chamado da solicitação de entrada
- As regras de GT NAT substituem o padrão com base no prefixo da parte chamadora

Exemplo:

```
# Quando gsmSSF 555123456 chama seu gsmSCF
# Entrada: Chamado=55512341112, Chamando=555123456
# Pesquisa GT NAT: "555" -> response_gt="55512341111"
# Resposta: Chamado=555123456, Chamando=55512341111
```

Balanceamento de Carga

O GT NAT pode ser combinado com balanceamento de carga M3UA para gerenciamento avançado de tráfego.

Guia de Migração

Habilitando GT NAT em Sistema Existente

1. **Preparar Configuração**

   ```
   # Adicione ao runtime.exs (mantenha desativado inicialmente)
   config :omniss7,
     gt_nat_enabled: false,  # Começar desativado
     gt_nat_rules: [
       # Suas regras aqui com pesos
       %{calling_prefix: "877", weight: 10, response_gt:
   "111111"},
       %{weight: 100, response_gt: "999999"}  # Curinga de recuo
     ]
   ```

2. **Testar Configuração**

```
# Validar configuração compila
mix compile

# Testar no iex
iex -S mix
iex> GtNat.validate_config()
```

3. **Habilitar em Staging**

```
gt_nat_enabled: true  # Mudar para true
```

4. **Monitorar Logs**

```
tail -f log/omniss7.log | grep "GT NAT"
```

5. **Implantar em Produção**

    - Implantar durante a janela de manutenção
    - Monitorar as primeiras 24 horas de perto
    - Ter um plano de reversão pronto (definir `gt_nat_enabled: false`)

Suporte

Para problemas ou perguntas:

- Verifique os logs para mensagens "GT NAT"
- Valide a configuração com `GtNat.validate_config()`
- Revise a seção de solução de problemas deste guia
- Entre em contato com o suporte OmniSS7 com trechos de log

Guia de Configuração do HLR

← Voltar para a Documentação Principal

Este guia fornece configuração para usar o OmniSS7 como um **Home Location Register (HLR/HSS)** com **OmniHSS** como o banco de dados de assinantes backend.

Integração do OmniHSS

O modo HLR do OmniSS7 funciona como uma interface de sinalização SS7 que se conecta ao **OmniHSS**, um servidor de assinantes doméstico (HSS) completo. Esta arquitetura separa as preocupações:

- **OmniSS7 (Frontend HLR)**: Lida com toda a sinalização do protocolo SS7/MAP, roteamento SCCP e comunicação de rede
- **OmniHSS (Backend HSS)**: Gerencia dados de assinantes, autenticação, provisionamento e recursos avançados

Por que OmniHSS?

O OmniHSS fornece gerenciamento de assinantes de nível de operadora com recursos incluindo:

- **Suporte Multi-IMSI**: Cada assinante pode ter múltiplos IMSIs associados a um único MSISDN para roaming internacional, troca de rede e provisionamento de eSIM
- **Autenticação Flexível**: Suporte para algoritmos de autenticação Milenage (3G/4G/5G) e COMP128 (2G)
- **Rastreamento de Sessão de Circuito e Pacote**: Rastreamento independente de registros de rede CS (circuito comutado) e PS (pacote comutado)

- **Provisionamento Avançado**: Perfis de serviço personalizáveis, serviços suplementares e dados de assinatura CAMEL
- **Design API-First**: API RESTful HTTP para integração com sistemas de cobrança, CRM e provisionamento
- **Atualizações em Tempo Real**: Rastreamento de localização, gerenciamento de sessão e geração de vetores de autenticação

Todos os dados de assinantes, credenciais de autenticação e configurações de serviço são armazenados e gerenciados no OmniHSS. O OmniSS7 consulta o OmniHSS através de chamadas de API HTTPS para responder a operações MAP como UpdateLocation, SendAuthenticationInfo e SendRoutingInfo.

Importante: O modo HLR do OmniSS7 é um **frontend de sinalização apenas**. Toda a lógica de gerenciamento de assinantes, algoritmos de autenticação, regras de provisionamento e operações de banco de dados são tratadas pelo OmniHSS. Este guia cobre a configuração do protocolo SS7/MAP no OmniSS7. Para informações sobre provisionamento de assinantes, configuração de autenticação, perfis de serviço e operações administrativas, **consulte a documentação do OmniHSS**.

Suporte Multi-IMSI

O OmniHSS suporta nativamente configurações Multi-IMSI, permitindo que um único assinante (identificado por MSISDN) tenha múltiplos IMSIs. Isso possibilita:

- **Perfis de Roaming Internacional**: Diferentes IMSIs para diferentes regiões para reduzir custos de roaming
- **eSIM Multi-Perfil**: Múltiplos perfis de rede em um único dispositivo compatível com eSIM
- **Troca de Rede**: Troca sem interrupções entre redes sem alterar o MSISDN
- **Coordenação de Dual SIM**: Coordenação entre múltiplos SIMs físicos ou virtuais
- **Teste e Desenvolvimento**: Múltiplos IMSIs de teste apontando para o mesmo assinante

Como funciona:

- Cada IMSI tem suas próprias credenciais de autenticação (Ki, OPc, algoritmo)
- Cada IMSI pode ter registros de sessão de circuito e pacote independentes
- Serviços e perfis de assinantes podem ser compartilhados ou personalizados por IMSI
- O OmniSS7 consulta o OmniHSS por IMSI, e o OmniHSS retorna os dados apropriados do assinante
- Sistemas de cobrança podem rastrear o uso por IMSI enquanto associam todos os IMSIs a uma única conta

Exemplo de cenário Multi-IMSI:

```
Assinante MSISDN: +1-555-123-4567
├─ IMSI 1: 310260123456789 (Rede Doméstica dos EUA - autenticação
Milenage)
├─ IMSI 2: 208011234567890 (Perfil de Roaming da França -
autenticação Milenage)
└─ IMSI 3: 440201234567891 (Perfil de Roaming do Reino Unido -
autenticação COMP128)
```

Todos os três IMSIs podem ser usados independentemente para registro na rede, mas todos pertencem à mesma conta de assinante. O OmniHSS gerencia o mapeamento IMSI-para-assinante e garante a autenticação e provisionamento adequados para cada IMSI.

Índice

1. Integração do OmniHSS
2. Suporte Multi-IMSI
3. O que é o Modo HLR?
4. Habilitando o Modo HLR
5. Banco de Dados de Assinantes
6. Vetores de Autenticação
7. Atualizações de Localização
8. Integração CAMEL
9. Tratamento de Assinantes em Roaming
10. Operações HLR

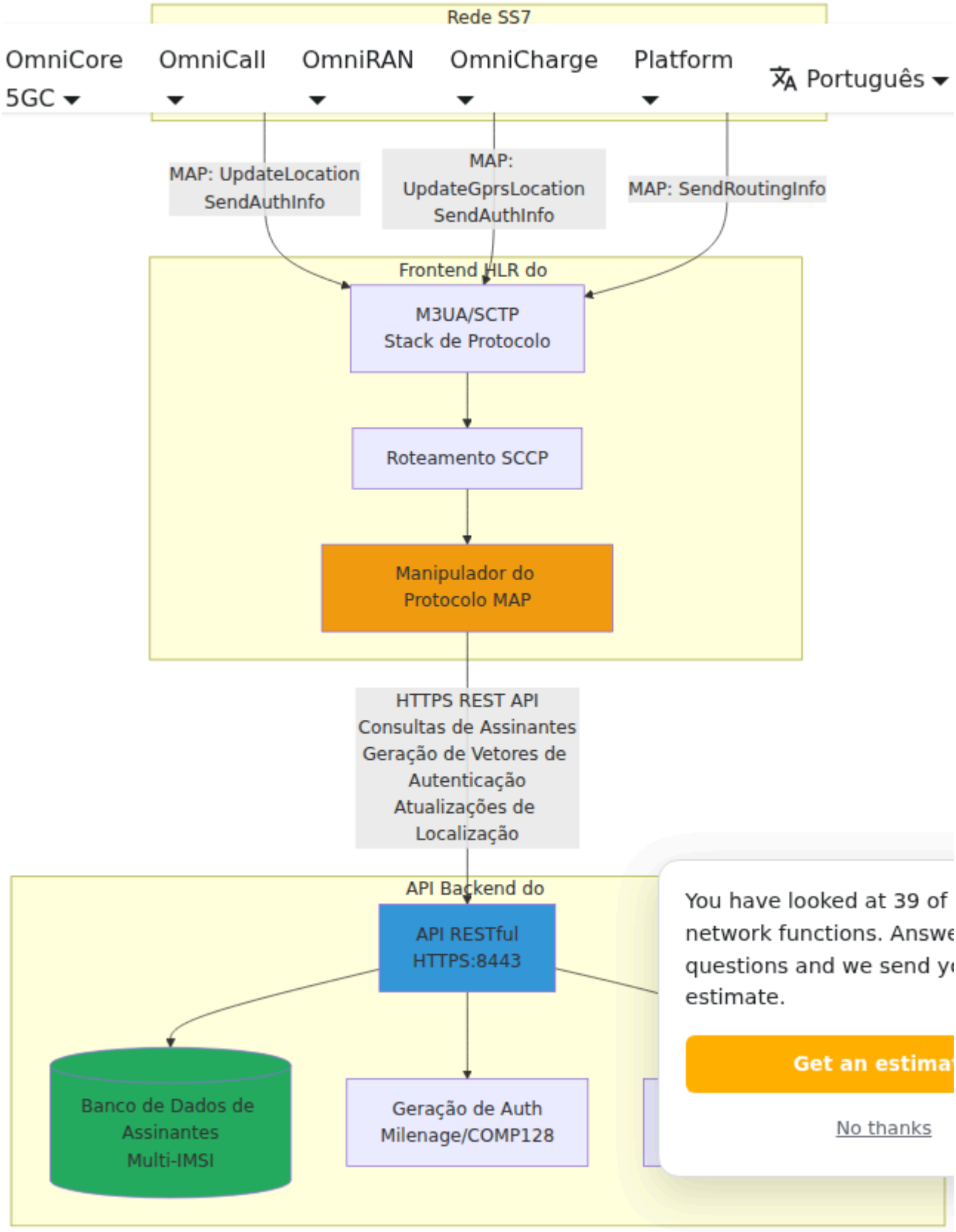
- Mapeamento de Campos de Resposta
  - SendRoutingInfo (SRI)
  - UpdateLocation / ISD
  - SendRoutingInfoForSM
- Resumo da Fonte de Campo

O que é o Modo HLR?

Modo HLR permite que o OmniSS7 funcione como um Home Location Register para:

- **Gerenciamento de Assinantes**: Armazenar e gerenciar dados de assinantes
- **Autenticação**: Gerar vetores de autenticação para acesso à rede
- **Rastreamento de Localização**: Processar atualizações de localização de VLRs
- **Informações de Roteamento**: Fornecer informações de roteamento para chamadas e SMS

Arquitetura HLR



Habilitando o Modo HLR

O OmniSS7 pode operar em diferentes modos (STP, HLR, SMSc). O modo é selecionado através de um conjunto de flags de modo e a configuração associada em **`config/config.exs`**, o arquivo de configuração padrão da aplicação.

Mudando para o Modo HLR

Para executar o OmniSS7 como um HLR, defina as flags de modo para que os recursos HLR sejam habilitados e os recursos SMSc sejam desabilitados, em seguida, adicione a configuração específica do HLR:

1. **Abra** `config/config.exs`
2. **Defina** as flags de modo sob `config :omniss7`:
   - `map_client_enabled: true`: necessário para enviar e receber operações MAP
   - `hlr_mode_enabled: true`: habilita o tratamento específico do HLR
   - `smsc_mode_enabled: false`: desabilita recursos SMSc
3. **Adicione** os parâmetros de configuração do HLR (endpoint da API, endereços GT, configurações ISD/CAMEL, conexão M3UA) mostrados abaixo
4. **Personalize** os parâmetros para sua implantação
5. **Reinicie** a aplicação para que as alterações tenham efeito

> **Nota**: `config/runtime.exs` é um stub mínimo usado apenas para testes automatizados. Ele **não** contém definições de modo operacional. Todas as configurações de modo e HLR estão em `config/config.exs` (com substituições específicas do ambiente em arquivos como `config/test.exs`). Implantações de produção fornecem seu próprio `config/config.exs`.

Configuração do Modo HLR

A configuração completa do HLR (em `config/config.exs`) é a seguinte:

```
config :omniss7,
  # Flags de Modo - Habilitar apenas recursos HLR
  map_client_enabled: true,
  hlr_mode_enabled: true,
  smsc_mode_enabled: false,

  # Configuração da API Backend do OmniHSS
  hlr_api_base_url: "https://omnihss.example.com:8443",
  hlr_api_key: "<Chave da API OmniHSS com o papel
hlr_interworking>",

  # Endereço GT do Centro de Serviço HLR para operações SMS
  hlr_service_center_gt_address: "1234567890",

  # Configuração de Mapeamento MSISDN ↔ IMSI
  # Veja: Seção de Mapeamento MSISDN ↔ IMSI para detalhes
  hlr_imsi_plmn_prefix: "50557",
  hlr_msisdn_country_code: "61",
  hlr_msisdn_nsn_offset: 0,
  hlr_msisdn_nsn_length: 9,

  # Configuração do InsertSubscriberData
  # Modo de Acesso à Rede: :packetAndCircuit, :packetOnly, ou
:circuitOnly
  isd_network_access_mode: :packetAndCircuit,

  # Enviar ISD #2 (dados de Serviços Suplementares)
  isd_send_ss_data: true,

  # Enviar ISD #3 (dados de Bloqueio de Chamadas)
  isd_send_call_barring: true,

  # Configuração CAMEL (para respostas SendRoutingInfo)
  # Chave de Serviço para iniciação de serviço CAMEL
  camel_service_key: 11_110,

  # Ponto de Detecção de Gatilho CAMEL
  # Opções: :termAttemptAuthorized, :tBusy, :tNoAnswer, :tAnswer
  camel_trigger_detection_point: :termAttemptAuthorized,

  # Prefixos VLR Domésticos
  # Lista de prefixos de endereço VLR que são considerados "rede
doméstica"
```

```
  # Se o VLR do assinante começar com um desses prefixos, use a
resposta SRI padrão
  # Caso contrário, o assinante está em roaming e precisamos
enviar PRN para obter MSRN
  # Padrão é [] (sem prefixos = todos os VLRs tratados como
roaming) se omitido
  home_vlr_prefixes: ["555123"],

  # Configuração de Conexão M3UA
  # Conectar como ASP para receber operações MAP (UpdateLocation,
SendAuthInfo, etc.)
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    # Endpoint local (sistema HLR)
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    # Endpoint remoto STP
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }
```

Parâmetros de Configuração para Personalizar

Para uma referência completa de todos os parâmetros de configuração, consulte a Referência de Configuração.

Parâmetro	Tipo	Padrão
`hlr_api_base_url`	String	*Obrigatório*
`hlr_api_key`	String	*Obrigatório*
`hlr_service_center_gt_address`	String	*Obrigatório*
`smsc_service_center_gt_address`	String	*Obrigatório*
`hlr_smsc_alert_gts`	Lista	`[]`
`hlr_alert_location_expiry_seconds`	Inteiro	`172800`

Parâmetro	Tipo	Padrão
`hlr_imsi_plmn_prefix`	String	`"50557"`
`hlr_msisdn_country_code`	String	`"61"`
`hlr_msisdn_nsn_offset`	Inteiro	`0`
`hlr_msisdn_nsn_length`	Inteiro	`9`
`isd_network_access_mode`	Átomo	`:packetAndCircuit`
`isd_send_ss_data`	Booleano	`true`

Parâmetro	Tipo	Padrão
`isd_send_call_barring`	Booleano	`true`
`camel_service_key`	Inteiro	`11_110`
`camel_trigger_detection_point`	Átomo	`:termAttemptAuthorized`
`hlr_camel_subscriptions`	Lista	`[]`
`home_vlr_prefixes`	Lista	`[]`
`local_ip`	Tupla	*Obrigatório*
`local_port`	Inteiro	`2905`
`remote_ip`	Tupla	*Obrigatório*

Parâmetro	Tipo	Padrão
`remote_port`	Inteiro	`2905`
`routing_context`	Inteiro	`1`

O que Acontece Quando o Modo HLR é Habilitado

Quando `hlr_mode_enabled: true`, a interface web mostrará:

- **Eventos SS7** - Registro de eventos
- **Cliente SS7** - Teste de operações MAP
- **Pares** - Status da conexão (pares M3UA/SCTP)
- **Links HLR** - Status da API HLR + gerenciamento de assinantes ← *específico do HLR*
- **Recursos** - Monitoramento do sistema
- **Configuração** - Visualizador de configuração

As abas **Roteamento**, **Teste de Roteamento** e **Links SMSc** serão ocultadas.

Notas Importantes

- **Configuração Obrigatória**: O parâmetro `hlr_service_center_gt_address` é **mandatório**. A aplicação falhará ao iniciar se não estiver configurado.
- **Backend OmniHSS**: A API OmniHSS backend deve ser acessível na `hlr_api_base_url` configurada
- **Chave da API OmniHSS**: `hlr_api_key` é obrigatória. O OmniSS7 a envia no cabeçalho `x-omnihss-key` em cada solicitação ao OmniHSS, e também na verificação de status dos Links HLR. Se a chave estiver faltando ou vazia, cada solicitação ao OmniHSS falha com um erro que menciona `hlr_api_key`. O OmniSS7 não envia uma solicitação sem a chave. Se a chave estiver incorreta, o OmniHSS retorna HTTP 401.

- **Timeout de Solicitação da API**: Todas as solicitações da API OmniHSS têm um **timeout fixo de 5 segundos**
- **Timeout de Solicitação MAP**: Todas as solicitações MAP (SRI, UpdateLocation, SendAuthInfo, etc.) têm um **timeout fixo de 10 segundos**
- **Timeout ISD**: Cada mensagem InsertSubscriberData (ISD) em uma sequência UpdateLocation tem um **timeout fixo de 10 segundos**
- A conexão M3UA com o STP é necessária para receber operações MAP
- Após mudar de modos, você deve reiniciar a aplicação para que as alterações tenham efeito
- **Interface Web**: Consulte o Guia da Interface Web para informações sobre como usar a interface web
- **Acesso à API**: Consulte o Guia da API para documentação da API REST e acesso ao Swagger UI

Banco de Dados de Assinantes

O OmniHSS gerencia todos os dados de assinantes incluindo identidades, credenciais de autenticação, perfis de serviço e informações de localização. O OmniSS7 recupera esses dados através de chamadas de API RESTful.

Modelo de Assinante do OmniHSS

O OmniHSS armazena informações abrangentes sobre assinantes:

- **Múltiplos IMSIs por assinante**: Suporte para configurações Multi-IMSI (eSIM, perfis de roaming, troca de rede)
- **Credenciais de autenticação**: Ki, OPc e seleção de algoritmo (Milenage ou COMP128)
- **Perfis de serviço**: Categoria de assinante, serviços permitidos, parâmetros de QoS
- **Rastreamento de localização**: Rastreamento independente do VLR/MSC atual (sessão de circuito) e SGSN/GGSN (sessão de pacote)

- **Dados de assinatura CAMEL**: Chaves de serviço, pontos de gatilho e endereços gsmSCF
- **Serviços suplementares**: Encaminhamento de chamadas, bloqueio, espera, configurações CLIP/CLIR
- **Estado administrativo**: Habilitado/desabilitado, restrições de serviço, datas de expiração

---

Vetores de Autenticação

Gerar Vetores de Autenticação

O OmniHSS gera vetores de autenticação usando os algoritmos Milenage ou COMP128 com base no método de autenticação configurado de cada assinante. Quando o OmniSS7 recebe solicitações MAP **sendAuthenticationInfo**:

1. O OmniSS7 extrai o IMSI da solicitação MAP
2. O OmniSS7 chama a API do OmniHSS para gerar vetores de autenticação
3. O OmniHSS recupera as credenciais Ki e OPc do assinante
4. O OmniHSS gera o número solicitado de vetores (RAND, XRES, CK, IK, AUTN)
5. O OmniSS7 codifica os vetores no formato MAP e os retorna para o VLR/SGSN solicitante

Integração da API OmniHSS

O OmniSS7 se comunica com o OmniHSS via API REST HTTPS para recuperar informações de assinantes, atualizar dados de localização e gerar vetores de autenticação:

```
config :omniss7,
  hlr_api_base_url: "https://omnihss-server:8443",
  hlr_api_key: "<Chave da API OmniHSS com o papel
hlr_interworking>"
```

Cada solicitação ao OmniHSS contém o cabeçalho `x-omnihss-key` com o valor de `hlr_api_key`. O OmniHSS usa a chave para encontrar o papel do cliente. A chave HLR deve ter o papel `hlr_interworking`. Este papel dá acesso à pesquisa de assinantes, configuração de chaves, vetor de autenticação e endpoints de sessão de circuito que o HLR usa.

Quando o OmniSS7 recebe operações MAP da rede SS7, ele consulta o OmniHSS para:

- **Recuperar dados do assinante** por IMSI ou MSISDN
- **Gerar vetores de autenticação** usando credenciais Ki/OPc armazenadas
- **Atualizar a localização da sessão de circuito** quando os assinantes realizam UpdateLocation
- **Verificar o status do assinante** e direitos de serviço

Atualizações de Localização

Processamento de Atualização de Localização

Ao receber solicitações MAP **updateLocation**, o OmniSS7 coordena com o OmniHSS para registrar o assinante em um novo VLR:

1. **Extrair informações de localização** da solicitação UpdateLocation (IMSI, novo GT VLR, novo GT MSC)
2. **Consultar o OmniHSS** para verificar se o assinante existe e está habilitado
3. **Atualizar a sessão de circuito** no OmniHSS com a nova localização VLR/MSC
4. **Enviar mensagens InsertSubscriberData (ISD)** para provisionar o assinante no novo VLR
5. **Retornar a resposta UpdateLocation** para o VLR (inclui GT HLR de `hlr_service_center_gt_address`)
6. **Enviar alertServiceCenter** para os GTs do SMSc configurados (se `hlr_smsc_alert_gts` estiver preenchido)

Nota: O parâmetro de configuração `hlr_service_center_gt_address` especifica o Título Global do HLR que é retornado nas respostas UpdateLocation. Isso permite que o VLR/MSC identifique e roteie mensagens de volta para este HLR.

Integração do Centro de Serviço de Alerta

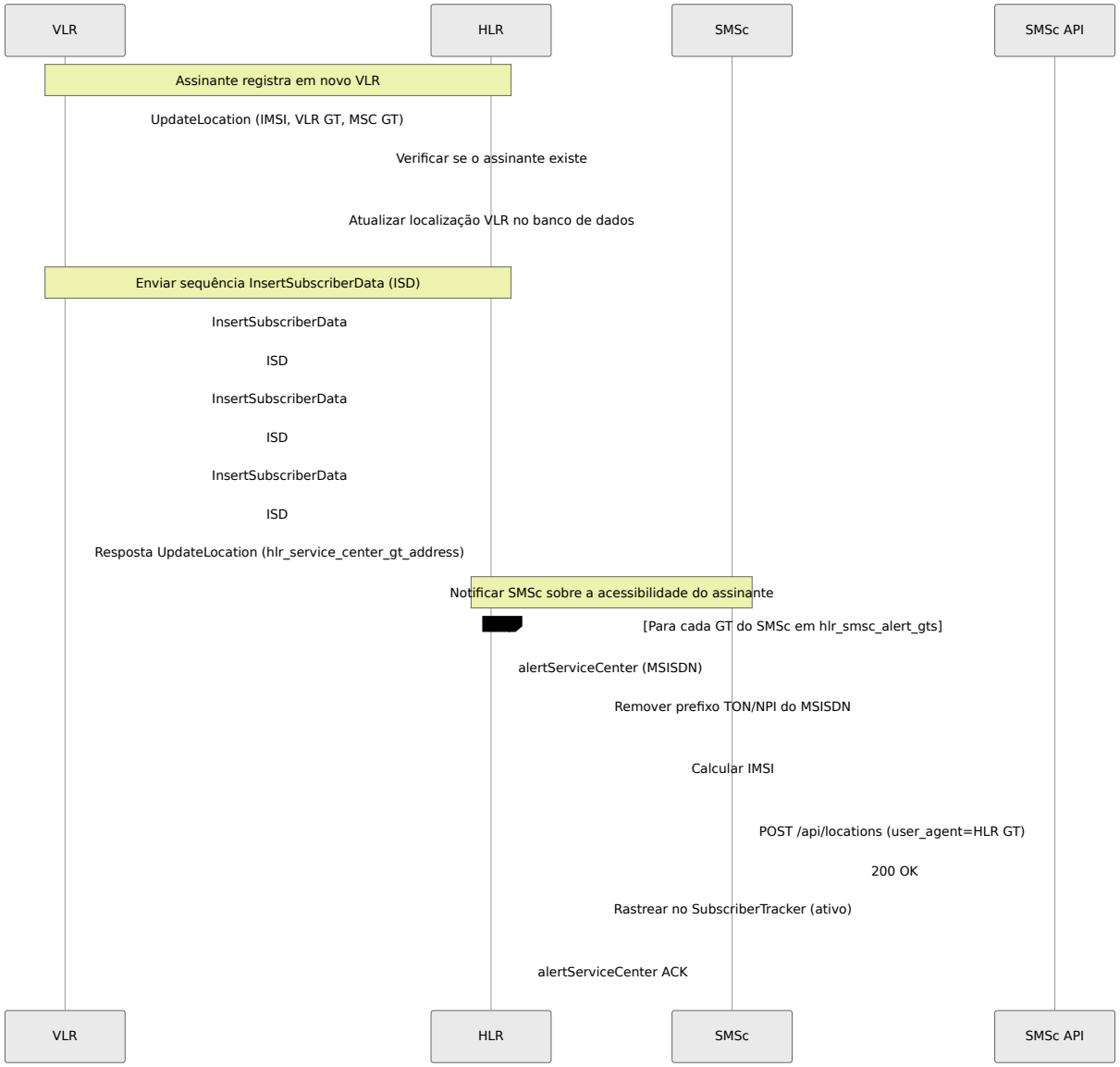
Após uma atualização de localização bem-sucedida, o HLR pode notificar automaticamente os sistemas SMSc que um assinante agora está acessível, enviando mensagens **alertServiceCenter** (código de operação MAP 64). Para informações sobre como o SMSc lida com esses alertas, consulte Tratamento do Centro de Serviço de Alerta no Guia SMSc.

Configuração

Configure a lista de Títulos Globais do SMSc a serem notificados:

```
config :omniss7,
  # Lista de GTs do SMSc para enviar alertServiceCenter após
UpdateLocation
  hlr_smsc_alert_gts: [
    "15559876543",
    "15559876544"
  ],

  # Tempo de expiração da localização quando o SMSc recebe
alertServiceCenter (padrão: 48 horas)
  hlr_alert_location_expiry_seconds: 172800
```

Diagrama de Fluxo



Comportamento

Quando um assinante realiza UpdateLocation:

1. O HLR envia alertServiceCenter para **cada** GT do SMSc na lista `hlr_smsc_alert_gts`
2. A mensagem inclui o MSISDN do assinante
3. O HLR usa `hlr_service_center_gt_address` como o GT da parte chamadora
4. Endereçamento SCCP: SSN chamador=6 (HLR), SSN chamado=8 (SMSc)

O SMSc recebe o alerta e:

- **Remove o prefixo TON/NPI** do MSISDN (por exemplo, "19123123213" → "123123213")
- Marca o assinante como acessível em seu banco de dados de localização (via POST para /api/locations)
- **Define o campo** `user_agent` para o GT do HLR ao chamar a API (para rastrear qual HLR enviou o alerta)
- Define o tempo de expiração da localização com base em `hlr_alert_location_expiry_seconds`
- Rastreia o assinante no SMSc Subscriber Tracker para monitoramento

Testando

Use a página **Assinantes Ativos** na interface da Web para enviar manualmente mensagens alertServiceCenter para teste:

1. Navegue até a aba "Assinantes Ativos"
2. Encontre a seção "Testar Alert Service Center"
3. Insira MSISDN, SMSc GT e HLR GT (os padrões são preenchidos automaticamente a partir da configuração)
   - SMSc GT padrão é a primeira entrada em `hlr_smsc_alert_gts`
   - HLR GT padrão é `hlr_service_center_gt_address`
4. Clique em "Enviar alertServiceCenter"

Isso é útil para testar o manuseio de alertas SMSc sem exigir um fluxo completo de UpdateLocation. O formulário utiliza validação `phx-blur` para evitar mostrar erros enquanto digita.

Configuração do InsertSubscriberData (ISD)

Após um UpdateLocation bem-sucedido, o HLR envia dados de provisionamento do assinante para o VLR usando mensagens **InsertSubscriberData** (ISD). A configuração do ISD permite personalizar quais dados são enviados e como.

Para referência de parâmetros de configuração, veja Configuração do ISD na Referência de Configuração.

Sequência ISD

O HLR pode enviar até 3 mensagens ISD sequenciais:

1. **ISD #1** (Sempre enviado) - Dados básicos do assinante:
    - IMSI
    - MSISDN
    - Categoria do assinante
    - Status do assinante (serviceGranted)
    - Lista de serviços de bearer
    - Lista de teleserviços
    - Modo de acesso à rede
2. **ISD #2** (Opcional) - Dados de Serviços Suplementares (SS):
    - Configurações de encaminhamento de chamadas (incondicional, ocupado, sem resposta, não acessível)
    - Espera de chamada
    - Retenção de chamada
    - Serviço de múltiplas partes
    - Status e recursos do serviço suplementar
3. **ISD #3** (Opcional) - Dados de Bloqueio de Chamadas:
    - Bloqueio de todas as chamadas de saída (BAOC)
    - Bloqueio de chamadas internacionais de saída (BOIC)
    - Dados de restrição de acesso

Opções de Configuração

```
# Configuração do InsertSubscriberData
# Modo de Acesso à Rede: :packetAndCircuit, :packetOnly, ou
:circuitOnly
isd_network_access_mode: :packetAndCircuit,

# Enviar ISD #2 (dados de Serviços Suplementares)
isd_send_ss_data: true,

# Enviar ISD #3 (dados de Bloqueio de Chamadas)
isd_send_call_barring: true,
```

Modo de Acesso à Rede

O parâmetro `isd_network_access_mode` controla que tipo de acesso à rede o assinante está permitido:

Valor	Descrição	Caso de Uso
`:packetAndCircuit`	Tanto comutado por pacotes (GPRS/LTE) quanto comutado por circuitos (voz)	Padrão - Assinantes de serviço completo
`:packetOnly`	Apenas comutado por pacotes (dados/LTE)	Cartões SIM apenas para dados, dispositivos IoT
`:circuitOnly`	Apenas comutado por circuitos (voz/SMS)	Dispositivos legados, planos apenas de voz

Controlando Mensagens ISD

Você pode controlar quais mensagens ISD são enviadas com base nas suas necessidades de rede:

Enviar todos os ISDs (Padrão - Conjunto completo de recursos):

```
isd_send_ss_data: true,
isd_send_call_barring: true,
```

Enviar apenas dados básicos do assinante (Provisionamento mínimo):

```
isd_send_ss_data: false,
isd_send_call_barring: false,
```

Enviar dados básicos + serviços suplementares (Sem bloqueio de chamadas):

```
isd_send_ss_data: true,
isd_send_call_barring: false,
```

Exemplo de Fluxo ISD

Quando UpdateLocation é recebido:

```
VLR → HLR: UpdateLocation (INÍCIO)
HLR → VLR: InsertSubscriberData #1 (CONTINUAR) - Dados básicos
VLR → HLR: ISD #1 ACK (CONTINUAR)
HLR → VLR: InsertSubscriberData #2 (CONTINUAR) - Dados de SS [se
habilitado]
VLR → HLR: ISD #2 ACK (CONTINUAR)
HLR ⟲ VLR: InsertSubscriberData #3 (CONTINUAR) - Bloqueio de
chamadas [se habilitado]
VLR → HLR: ISD #3 ACK (CONTINUAR)
HLR → VLR: Resposta UpdateLocation (FIM)
```

Se `isd_send_ss_data` ou `isd_send_call_barring` estiverem definidos como `false`, essas mensagens ISD são puladas, e o UpdateLocation FIM é enviado mais cedo.

Melhores Práticas

- **Configuração Padrão**: Use `:packetAndCircuit` e habilite todos os ISDs para máxima compatibilidade

- **IoT/M2M**: Use `:packetOnly` e desabilite dados de SS/bloqueio de chamadas para dispositivos apenas de dados
- **Interoperabilidade**: Alguns VLRs mais antigos podem não suportar todos os serviços suplementares - desabilite `isd_send_ss_data` se encontrar problemas
- **Desempenho**: Desabilitar ISDs não utilizados reduz a sobrecarga de mensagens e acelera as atualizações de localização

---

Integração CAMEL

Configuração CAMEL para SendRoutingInfo

Ao responder a solicitações de **SendRoutingInfo** (SRI) de um GMSC (Gateway MSC), o HLR pode instruir o GMSC a invocar serviços CAMEL para roteamento inteligente de chamadas e controle de serviços.

Para referência de parâmetros de configuração, veja Configuração CAMEL na Referência de Configuração.

O que é CAMEL?

CAMEL (Aplicações Personalizadas para Lógica Aprimorada de Rede Móvel) é um protocolo que permite serviços de rede inteligentes em redes GSM/UMTS. Ele permite que os operadores de rede implementem serviços de valor agregado, como:

- Cobrança pré-paga
- Filtragem e bloqueio de chamadas
- Redes Privadas Virtuais (VPN)
- Serviços de tarifa premium
- Encaminhamento de chamadas com lógica personalizada
- Serviços baseados em localização

Opções de Configuração

```
# Configuração CAMEL (para respostas SendRoutingInfo)
# Chave de Serviço para iniciação do serviço CAMEL
camel_service_key: 11_110,

# Ponto de Detecção de Gatilho CAMEL
# Opções: :termAttemptAuthorized, :tBusy, :tNoAnswer, :tAnswer
camel_trigger_detection_point: :termAttemptAuthorized,
```

Chave de Serviço

A `camel_service_key` identifica qual serviço CAMEL deve ser invocado no gsmSCF (Função de Controle de Serviço). Este é um identificador numérico configurado em sua rede:

Chave de Serviço	Caso de Uso Típico
`11_110`	Controle de chamada pré-paga (padrão)
`100`	Serviço pré-pago de origem
`200`	Encaminhamento de chamadas com lógica personalizada
`300`	Rede Privada Virtual (VPN)
Personalizado	Serviços específicos do operador

Exemplo de Configuração:

```
# Para controle de chamada pré-paga
camel_service_key: 11_110,

# Para serviço VPN
camel_service_key: 300,
```

Ponto de Detecção de Gatilho

O `camel_trigger_detection_point` especifica quando o serviço CAMEL deve ser acionado durante a configuração da chamada:

Ponto de Detecção	Descrição	Quando Acionado
`:termAttemptAuthorized`	Tentativa de chamada autorizada (padrão)	Antes da chamada ser roteada para o assinante
`:tBusy`	Ocupado na terminação	Quando o assinante está ocupado
`:tNoAnswer`	Sem resposta na terminação	Quando o assinante não responde
`:tAnswer`	Resposta na terminação	Quando o assinante atende a chamada

Exemplos de Configuração:

Controle padrão pré-pago (acionar antes do roteamento):

```
camel_trigger_detection_point: :termAttemptAuthorized,
```

Tratamento personalizado para ocupado (acionar quando ocupado):

```
camel_trigger_detection_point: :tBusy,
```

Cobrança baseada em resposta (acionar na resposta):

```
camel_trigger_detection_point: :tAnswer,
```

Resposta SRI com CAMEL

Quando configurado, as respostas SendRoutingInfo incluem informações de assinatura CAMEL:

```
GMSC → HLR: SendRoutingInfo (INÍCIO)
HLR → GMSC: Resposta SRI (FIM) com:
  - IMSI
  - Número do VLR
  - Estado do assinante
  - Informações de roteamento CAMEL:
    * Chave de Serviço: 11_110
    * Endereço gsmSCF: <endereço configurado>
    * Ponto de Detecção de Gatilho: termAttemptAuthorized
    * Tratamento de Chamada Padrão: continueCall

GMSC contata gsmSCF no ponto de gatilho para executar o serviço
CAMEL
```

Melhores Práticas

- **Redes de Produção**: Use chaves de serviço padronizadas acordadas com seu provedor gsmSCF
- **Teste**: Use `:termAttemptAuthorized` para testes mais abrangentes
- **Serviços Pré-pagos**: A chave de serviço `11_110` é um padrão comum da indústria para chamadas pré-pagas de terminação
- **Tratamento de Fallback**: `defaultCallHandling: :continueCall` garante que as chamadas prossigam se o gsmSCF não estiver acessível

Assinaturas CAMEL no InsertSubscriberData

Durante o UpdateLocation, o HLR envia um InsertSubscriberData adicional contendo `vlrCamelSubscriptionInfo` quando o IMSI do assinante corresponde a uma regra em `hlr_camel_subscriptions`. A primeira regra correspondente vence. O MSC usa o O-CSI para chamadas MO (DP collectedInfo) e o VT-CSI para chamadas MT (DP termAttemptAuthorized), e abre o diálogo CAP em direção ao título global `gsmscf_address`, de modo que o gsmSCF seja escolhido pelo HLR, não pela configuração do MSC.

```
config :omniss7,
  hlr_camel_subscriptions: [
    %{
      imsi_prefix: "3133800000009",
      o_csi: %{service_key: 100, gsmscf_address: "14155550300",
default_call_handling: :release_call},
      t_csi: %{service_key: 101, gsmscf_address: "14155550300",
default_call_handling: :continue_call}
    }
  ]
```

Chave	Descrição
`imsi_prefix`	Prefixo IMSI ao qual a regra se aplica
`o_csi` / `t_csi`	Pelo menos um. `service_key` (InitialDP serviceKey), `gsmscf_address` (gsmSCF E.164 GT), `default_call_handling` (`:release_call` ou `:continue_call`, aplicado pelo MSC quando o gsmSCF não responde)

Os CSIs são enviados com o manuseio de capacidade CAMEL 2 (CAP v2). Uma regra com uma chave ausente gera um erro nomeando a chave.

Tratamento de Assinantes em Roaming

Detecção de VLR Local vs VLR em Roaming

Quando o HLR recebe uma solicitação de **SendRoutingInfo** (SRI), ele precisa determinar se o assinante está em um VLR "local" (dentro da sua rede) ou em um VLR em roaming (visitando outra rede). O comportamento difere com base nessa determinação:

Para referência de parâmetros de configuração, veja Prefixos de VLR Local na Referência de Configuração.

- **VLR Local**: Retornar resposta SRI padrão com informações de roteamento CAMEL
- **VLR em Roaming**: Enviar uma solicitação Provide Roaming Number (PRN) para obter um MSRN, e então retorná-lo na resposta SRI

Configuração

```
# Prefixos de VLR Local
# Lista de prefixos de endereço VLR que são considerados rede
"local"
# Se o endereço VLR do assinante começa com um desses prefixos,
use a resposta SRI padrão
# Caso contrário, o assinante está em roaming e precisamos enviar
PRN para obter MSRN
home_vlr_prefixes: ["555123"],
```

Exemplo de Configuração:

```
# Rede local única
home_vlr_prefixes: ["555123"],

# Múltiplas redes locais (por exemplo, diferentes regiões ou
subsidiárias)
home_vlr_prefixes: ["555123", "555124", "555125"],
```

Como Funciona

1. Fluxo de Assinante Local (Padrão)

Quando o endereço VLR do assinante começa com um prefixo local configurado:

```
GMSC → HLR: SendRoutingInfo (MSISDN: "1234567890")
HLR consulta a API de backend para dados do assinante
HLR verifica o endereço VLR: "5551234567"
HLR determina: VLR começa com "555123" → Rede local
HLR → GMSC: Resposta SRI com informações de roteamento CAMEL:
  - IMSI
  - Número do VLR: "5551234567"
  - Endereço gsmSCF (MSC): "5551234501"
  - Chave de serviço CAMEL: 11_110
  - Ponto de detecção de gatilho: termAttemptAuthorized
```

2. Fluxo de Assinante em Roaming (PRN Necessário)

Quando o endereço VLR do assinante NÃO corresponde a nenhum prefixo local:

```
GMSC → HLR: SendRoutingInfo (MSISDN: "1234567890")
HLR consulta a API de backend para dados do assinante
HLR verifica o endereço VLR: "49170123456"
HLR determina: VLR não começa com "555123" → Roaming
HLR → MSC: ProvideRoamingNumber (PRN):
  - MSISDN: "1234567890"
  - IMSI: "999999876543210"
  - Número do MSC: "49170123456"
  - Endereço GMSC: "5551234501"
MSC → HLR: Resposta PRN com MSRN: "49170999888777"
HLR → GMSC: Resposta SRI com informações de roteamento:
  - IMSI
  - Número do VLR: "49170123456"
  - Número de Roaming (MSRN): "49170999888777"
```

Diferenças na Estrutura de Resposta

Resposta SRI de Assinante Local

```
%{
  imsi: "999999876543210",
  extendedRoutingInfo: {
    :camelRoutingInfo, %{
      gmscCamelSubscriptionInfo: %{
        "t-CSI": %{
          serviceKey: 11_110,
          "gsmSCF-Address": "5551234501",
          defaultCallHandling: :continueCall,
          "t-BcsmTriggerDetectionPoint": :termAttemptAuthorized
        }
      }
    }
  },
  subscriberInfo: %{
    locationInformation: %{"vlr-number": "5551234567"},
    subscriberState: {:notProvidedFromVLR, :NULL}
  }
}
```

Resposta SRI de Assinante em Roaming

```
%{
  imsi: "999999876543210",
  extendedRoutingInfo: {
    :routingInfo, %{
      roamingNumber: "49170999888777"  # MSRN do PRN
    }
  },
  subscriberInfo: %{
    locationInformation: %{"vlr-number": "49170123456"},
    subscriberState: {:notProvidedFromVLR, :NULL}
  }
}
```

Operação Provide Roaming Number (PRN)

Estrutura da Solicitação PRN

A solicitação PRN enviada ao MSC/VLR contém:

Campo	Fonte	Descrição
MSISDN	Solicitação SRI	Número de telefone do assinante
IMSI	API HLR	IMSI do assinante
Número do MSC	API HLR	MSC servindo o assinante em roaming (`serving_msc`)
Endereço GMSC	Solicitação SRI	GMSC fazendo a solicitação SRI original
Número de Referência da Chamada	Estático	Identificador de referência da chamada
Fases CAMEL Suportadas	Estático	Fases CAMEL suportadas pelo GMSC

Tratamento da Resposta PRN

O HLR espera uma resposta PRN contendo:

- **MSRN** (Número de Roaming da Estação Móvel): Um número temporário alocado pela rede visitada para roteamento da chamada

Tratamento de Erros:

- Se o PRN expirar → Retorna erro 27 (Assinante Ausente) na resposta SRI
- Se o PRN falhar → Retorna erro 27 (Assinante Ausente) na resposta SRI
- Se o MSRN não puder ser extraído → Retorna erro 27 (Assinante Ausente) na resposta SRI

Exemplos de Configuração

Operador de Rede Local Única

```
# Todos os endereços VLR que começam com "555123" são considerados
locais
home_vlr_prefixes: ["555123"],
```

- VLR `5551234567` → Local (resposta CAMEL)
- VLR `5551235001` → Local (resposta CAMEL)
- VLR `49170123456` → Roaming (PRN + resposta MSRN)

Operador Multi-Região

```
# Múltiplas redes locais em diferentes regiões
home_vlr_prefixes: ["555123", "555124", "555125"],
```

- VLR `5551234567` → Local (região 1)
- VLR `5552341234` → Local (região 2)
- VLR `5553411111` → Local (região 3)
- VLR `44201234567` → Roaming (internacional)

Configuração de Teste

Para testar a funcionalidade PRN, defina uma lista vazia para tratar todos os VLRs como em roaming:

```
# Todos os VLRs são tratados como em roaming (para testar o fluxo
PRN)
home_vlr_prefixes: [],
```

Melhores Práticas

- **Seleção de Prefixo**: Use o prefixo único mais curto que identifique os VLRs da sua rede (por exemplo, código do país + código da rede)
- **Múltiplos Prefixos**: Inclua todos os prefixos de VLR em sua rede, incluindo diferentes regiões e subsidiárias
- **Acordos de Roaming**: Certifique-se de que o PRN seja devidamente suportado pelas redes parceiras de roaming

- **Teste**: Teste minuciosamente os cenários locais e em roaming antes da implantação em produção
- **Monitoramento**: Monitore as taxas de tempo limite do PRN para identificar problemas de conectividade com parceiros de roaming

Solução de Problemas

Sintoma: Todos os assinantes tratados como em roaming

- **Causa**: `home_vlr_prefixes` não configurado ou prefixos não correspondem aos endereços VLR
- **Solução**: Verifique os endereços VLR em seu banco de dados e atualize os prefixos conforme necessário

Sintoma: Solicitações PRN expirando

- **Causa**: Problemas de conectividade de rede com o MSC/VLR parceiro de roaming
- **Solução**: Verifique o roteamento M3UA/SCCP para endereços remotos do MSC

Sintoma: MSRN inválido na resposta SRI

- **Causa**: Formato de resposta PRN do parceiro de roaming não corresponde à estrutura esperada
- **Solução**: Revise os logs de resposta PRN e ajuste `extract_msrn_from_prn/1` se necessário

---

Operações HLR

Operações MAP Suportadas

OmniSS7 despacha operações MAP de entrada por opcode (independente da fase TCAP). As seguintes operações de entrada são tratadas:

Opcode	Operação	Propósito
2	`updateLocation`	Registrar localização VLR (aciona sequência ISD + alertServiceCenter opcional)
23	`updateGprsLocation`	Registrar localização SGSN (comutada por pacotes)
3	`cancelLocation`	Desregistrar do antigo VLR
7	`insertSubscriberData`	Receber perfil do assinante (por exemplo, ao atuar como lado VLR)
22	`sendRoutingInfo` (SRI)	Fornecer MSRN/roteamento CAMEL para chamadas (local vs roaming)
45	`sendRoutingInfoForSM` (SRI-for-SM)	Fornecer IMSI + SMSc GT para roteamento de SMS
44	`MT-Forward-SM`	Entrega de SMS móvel-terminada
46	`MO-Forward-SM`	Recepção de SMS móvel-originada
47	`reportSM-DeliveryStatus`	Relatório de status de entrega de SMS-GMSC
64	`alertServiceCenter`	Alerta de assinante acessível para SMSc
66	`readyForSM`	MSC/SGSN sinaliza que o assinante está pronto para SMS
56	`sendAuthenticationInfo`	Gerar vetores de autenticação

Opcode	Operação	Propósito
57	`restoreData`	Restauração de dados do assinante VLR
67	`purgeMS`	Purge um registro de assinante no HLR
59	`processUnstructuredSS-Request`	USSD (tratado via gateway USSD quando habilitado)
71	`anyTimeInterrogation` (ATI)	Interrogação de localização/estado do assinante
10	`registerSS`	Registro de serviço suplementar (por exemplo, Encaminhamento de Chamadas Incondicional)

OpCodes não tratados são respondidos com um erro `facilityNotSupported` (21).

Mapeamento de Campos de Resposta

Esta seção detalha de onde cada campo nas respostas HLR vem.

Resposta SendRoutingInfo (SRI)

Propósito: Fornece informações de roteamento para chamadas recebidas a um assinante.

O HLR fornece dois tipos diferentes de resposta com base em se o assinante está em um VLR local ou em roaming:

Resposta de Assinante Local (Roteamento CAMEL)

Usado quando o endereço VLR do assinante começa com um valor configurado em `home_vlr_prefixes`.

Estrutura da Resposta:

Campo	Fonte	Descrição
IMSI	API OmniHSS	IMSI do assinante do banco de dados OmniHSS
Número do VLR	API OmniHSS	VLR atual servindo o assinante (`circuit_session.assigned_vlr`)
Estado do Assinante	Estático	Sempre `notProvidedFromVLR`
extendedRoutingInfo	-	Tipo: `camelRoutingInfo`
Endereço gsmSCF	API OmniHSS	MSC servindo o assinante (`circuit_session.assigned_msc`)
Chave de Serviço	config.exs	Identificador de serviço CAMEL (`camel_service_key`)
Ponto de Detecção de Gatilho	config.exs	Quando acionar CAMEL (`camel_trigger_detection_point`)
Manuseio de Capacidade CAMEL	Estático	Nível de suporte à fase CAMEL
Tratamento de Chamada Padrão	Estático	Fallback se gsmSCF não estiver acessível

Resposta de Assinante em Roaming (Roteamento MSRN)

Usado quando o endereço VLR do assinante NÃO corresponde a nenhum valor configurado em `home_vlr_prefixes`.

Estrutura da Resposta:

Campo	Fonte	Descrição	
IMSI	API OmniHSS	IMSI do assinante do banco de dados OmniHSS	`"9`
Número do VLR	API OmniHSS	VLR atual servindo o assinante (`circuit_session.assigned_vlr`)	`"4`
Estado do Assinante	Estático	Sempre `notProvidedFromVLR`	`:n`
extendedRoutingInfo	-	Tipo: `routingInfo`	-
Número de Roaming (MSRN)	Resposta PRN	MSRN obtido da solicitação ProvideRoamingNumber	`"4`

Lógica de Decisão de Roteamento:

```
1. OmniSS7 recebe solicitação SendRoutingInfo
2. OmniSS7 consulta dados do assinante na API OmniHSS
3. OmniSS7 verifica o endereço VLR em relação aos
home_vlr_prefixes:

   Se VLR começa com prefixo local:
     → Retornar informações de roteamento CAMEL (fluxo de
assinante local)

   Se VLR NÃO corresponder a nenhum prefixo local:
     → Enviar ProvideRoamingNumber (PRN) para MSC
     → Extrair MSRN da resposta PRN
     → Retornar informações de roteamento com MSRN (fluxo de
assinante em roaming)
```

Fluxo de Dados:

- OmniSS7 consulta OmniHSS para informações do assinante
- OmniHSS retorna IMSI, localização atual do VLR/MSC e estado do assinante
- OmniSS7 usa esses dados para construir a resposta MAP

Requisitos de Configuração:

```
# Em config/config.exs
home_vlr_prefixes: ["555123"],  # Lista de prefixos de VLR locais
```

Respostas de Erro:

- Se `serving_vlr` e `serving_msc` forem `null`: Retorna erro 27 (Assinante Ausente)
- Se assinante não encontrado: Retorna erro 1 (Assinante Desconhecido)
- Se a solicitação PRN expirar (caso de roaming): Retorna erro 27 (Assinante Ausente)
- Se a resposta PRN for inválida (caso de roaming): Retorna erro 27 (Assinante Ausente)

Resposta UpdateLocation com InsertSubscriberData

Propósito: Registra o assinante no novo VLR e provisiona os dados do assinante.

Resposta UpdateLocation END

Campo	Fonte	Descrição	Exemplo
Número HLR	config.exs	Este Título Global do HLR (`hlr_service_center_gt_address`)	`"555123456`
Tipo de Mensagem TCAP	Estático	Resposta final após todas as ISDs	`END`

InsertSubscriberData #1 (Dados Básicos do Assinante)

Campo	Fonte	Descrição	Exemplo
IMSI	Requisição	Da requisição UpdateLocation	`"9999998765432`
MSISDN	OmniHSS API	Número de telefone do assinante do OmniHSS	`"555123456"`
Categoria	Estático	Categoria do assinante	`"\n"` (0x0A)
Status do Assinante	Estático	Status do serviço	`:serviceGrante`
Lista de Serviços de Transporte	Estático	Serviços de transporte suportados	`[<<31>>]`
Lista de Teleserviços	Estático	Teleserviços suportados	`[<<17>>, "!", "\""]`
Modo de Acesso à Rede	config.exs	Acesso por pacote/circuito (`isd_network_access_mode`)	`:packetAndCirc`

InsertSubscriberData #2 (Serviços Suplementares) - Opcional

Campo	Fonte	Descrição	Controlado Por
SS Provisionado	Estático	Dados de serviços suplementares	`isd_send_ss_data: true`
Encaminhamento de Chamadas	Estático	Configurações de encaminhamento (incondicional, ocupado, sem resposta, não acessível)	Config habilitado
Esperando Chamadas	Estático	Status do serviço de espera de chamadas	Config habilitado
Serviço Multi-participante	Estático	Suporte a chamadas em conferência	Config habilitado

ISD #2 inclui:

- Encaminhamento de chamadas incondicional (código SS 21)
- Encaminhamento de chamadas em ocupado (código SS 41)
- Encaminhamento de chamadas sem resposta (código SS 42)
- Encaminhamento de chamadas quando não acessível (código SS 62)
- Espera de chamadas (código SS 43)
- Serviço multi-participante (código SS 51)
- Serviços CLIP/CLIR

InsertSubscriberData #3 (Bloqueio de Chamadas) - Opcional

Campo	Fonte	Descrição	Controlado Por
Informações de Bloqueio de Chamadas	Estático	Configurações de bloqueio de chamadas	`isd_send_call_barring: true`
BAOC	Estático	Bloqueio de Todas as Chamadas de Saída (código SS 146)	Config habilitado
BOIC	Estático	Bloqueio de Chamadas Internacionais de Saída (código SS 147)	Config habilitado
Dados de Restrição de Acesso	Estático	Restrições de acesso à rede	Config habilitado

Controle de Sequência ISD:

- ISD #1: **Sempre enviado** - Contém dados essenciais do assinante
- ISD #2: Enviado apenas se `isd_send_ss_data: true` em config/config.exs
- ISD #3: Enviado apenas se `isd_send_call_barring: true` em config/config.exs

Resposta SendRoutingInfoForSM (SRI-for-SM)

Propósito: Fornece informações de roteamento MSC/SMSC para entrega de SMS. Quando um SMSc precisa entregar um SMS a um assinante, ele envia uma requisição SRI-for-SM ao HLR para determinar onde rotear a mensagem.

O HLR responde em um dos dois modos, decidido pelo **GT da parte chamadora SCCP** da requisição:

1. Roteamento doméstico (padrão: todo originador externo). Este modo retorna um IMSI sintético mais o próprio GT do IP-SM-GW como o número do nó da rede. O HLR nunca expõe IMSIs reais e nós de serviço a operadores externos. O MT-ForwardSM retorna ao IP-SM-GW, que então lida com a entrega. Este é o comportamento correto para tráfego de interconexão e roaming.

Campo	Fonte	Descrição	Como Gerado	
IMSI	Calculado	IMSI sintético derivado do MSISDN	`PLMN_PREFIX + zero_padded_MSISDN`	
Número do Nó da Rede	config.exs	Endereço GT do SMSC para roteamento de SMS	`smsc_service_center_gt_address`	

2. Roteamento direto na rede (apenas quando a requisição vem de um dos nossos próprios nós SMSc / IP-SM-GW). Quando o GT chamador corresponde à lista configurada de `home_smsc_gts`, o HLR procura o assinante em seu backend e retorna o **IMSI real** e o **MSC de serviço real** como o número do nó da rede, para que nosso SMSc possa enviar o MT-ForwardSM diretamente ao MSC. Se nenhum MSC de serviço estiver disponível para esse assinante, ele recua para o roteamento doméstico.

Campo	Fonte	Descrição	Exemplo
IMSI	Backend HLR	IMSI real do assinante	`"505570123456789"`
Número do Nó da Rede	Backend HLR	`serving_msc` real da última atualização de Location	`"5559876543"`

Parâmetros de Configuração (de `config/config.exs`):

```
# Endereço GT do Centro de Serviço (retornado nas respostas SRI-for-SM roteadas para casa)
# Isso informa ao SMSc solicitante onde enviar mensagens MT-ForwardSM
smsc_service_center_gt_address: "5551234567",  # Obrigatório

# Roteamento direto na rede: GTs chamadores SCCP de nossos próprios nós SMSc / IP-SM-GW.
# Um SRI-for-SM de qualquer um destes obtém o MSC de serviço REAL em vez do roteamento doméstico.
# Todos os outros são roteados para casa. Deixe vazio ([]) para roteá-los todos para casa.
home_smsc_gts: ["5551234567"],                # Opcional (padrão [])

# Configuração de Mapeamento MSISDN ↔ IMSI
# Prefixo PLMN: MCC (001 = Rede de Teste) + MNC (01 = Operador de Teste)
hlr_imsi_plmn_prefix: "001001",                 # Apenas o parâmetro de configuração necessário!
```

> **Nota:** O roteamento direto na rede só funciona se o HLR realmente possui o `serving_msc` do assinante. Ou seja, o MSC de serviço realizou um MAP `updateLocation` para este HLR. Se o MSC se registrar por outro mecanismo (por exemplo, GSUP para um HLR separado), o backend não terá a localização e a consulta será roteada para casa.

Mapeamento MSISDN ↔ IMSI

Parâmetros de Configuração:

Esses parâmetros controlam como o OmniSS7 gera IMSIs sintéticos a partir de MSISDNs para respostas SRI-for-SM:

- **`hlr_imsi_plmn_prefix`**: O prefixo MCC+MNC a ser usado ao construir IMSIs sintéticos (por exemplo, "50557" para MCC=505, MNC=57)
- **`hlr_msisdn_country_code`**: Código do país a ser precedido ao fazer o mapeamento reverso IMSI→MSISDN (por exemplo, "61" para Austrália, "1"

para EUA/Canadá)

- `hlr_msisdn_nsn_offset`: Posição do caractere onde o Número Nacional do Assinante (NSN) começa dentro do MSISDN (tipicamente 0 se o MSISDN não incluir o código do país, ou o comprimento do código do país se incluir)
- `hlr_msisdn_nsn_length`: Número de dígitos a serem extraídos do MSISDN como o NSN

Para detalhes adicionais de configuração, veja Mapeamento MSISDN ↔ IMSI na Referência de Configuração.

Por que o Mapeamento de MSISDN para IMSI é Necessário?

O protocolo MAP para **SendRoutingInfoForSM** (SRI-for-SM) requer que o HLR retorne um **IMSI** (Identidade Internacional do Assinante Móvel) em sua resposta. No entanto, o SMSc solicitante só conhece o **MSISDN** do assinante (número de telefone).

Em uma rede tradicional:

- O SMSc envia SRI-for-SM com o MSISDN de destino (por exemplo, "5551234567")
- O HLR deve procurar o assinante em seu banco de dados para encontrar seu IMSI
- O HLR retorna o IMSI na resposta SRI-for-SM
- O SMSc então usa este IMSI ao enviar MT-ForwardSM para o MSC/VLR

Abordagem do OmniSS7 - IMSIs Sintéticos:

Em vez de manter um banco de dados completo de assinantes com mapeamentos de MSISDN para IMSI, o OmniSS7 usa um esquema de codificação simples para **calcular** IMSIs sintéticos diretamente a partir do MSISDN. Esta abordagem oferece dois benefícios principais:

1. **Privacidade**: IMSIs reais de assinantes armazenados no banco de dados HLR nunca são expostos em respostas SRI-for-SM enviadas pela rede SS7
2. **Simplicidade**: Não há necessidade de consultar o banco de dados HLR para buscas de IMSI durante operações SRI-for-SM - o IMSI é calculado em tempo real a partir do MSISDN

Como Funciona:

MSISDNs são codificados diretamente na parte do assinante do IMSI (os dígitos após MCC+MNC):

```
IMSI = PLMN_PREFIX + zero_padded_MSISDN
```

Onde:

- **PLMN_PREFIX**: MCC + MNC (por exemplo, "001001" para Rede de Teste)
- **MSISDN**: Todos os dígitos numéricos do número de telefone
- **Zero Padding**: Preenchido à esquerda com zeros para preencher o IMSI exatamente com 15 dígitos

Exemplo Passo a Passo:

```
# Configuração
plmn_prefix = "001001"  # MCC 001 + MNC 01

# Entrada: MSISDN da requisição SRI-for-SM (decodificado TBCD)
msisdn = "555123456"    # 9 dígitos

# Passo 1: Calcular espaço disponível para o número do assinante
subscriber_digits = 15 - String.length("001001")  # = 9 dígitos

# Passo 2: Preencher à esquerda MSISDN com zeros para preencher a
parte do assinante
padded_msisdn = String.pad_leading("555123456", 9, "0")  # =
"555123456" (nenhum preenchimento necessário)

# Passo 3: Concatenar prefixo PLMN + MSISDN preenchido
imsi = "001001" <> "555123456"  # = "001001555123456" (exatamente
15 dígitos)
```

Exemplos Completos:

MSISDN de Entrada	Prefixo PLMN	Dígitos do Assinante Disponíveis	MSISDN Preenchido	IMSI Fina
`"555123456"`	`"001001"` (6)	9	`"555123456"`	`"001001555123`
`"99"`	`"001001"` (6)	9	`"000000099"`	`"001001000000`
`"999999999"`	`"001001"` (6)	9	`"999999999"`	`"001001999999`
`"91123456789"`	`"001001"` (6)	9	`"555123456"`	`"001001555123`

Tratamento de Casos Limite:

- **MSISDNs Curtos**: Preenchidos à esquerda com zeros (por exemplo, `"99"` → `"000000099"`)
- **MSISDNs Longos**: Mantidos os dígitos mais à direita, truncados os dígitos mais à esquerda (por exemplo, `"91123456789"` → `"555123456"`)
- **Comprimento do IMSI**: Sempre exatamente 15 dígitos

Mapeamento Reverso (IMSI → MSISDN):

O SMSc pode reverter este mapeamento para converter IMSIs de volta em MSISDNs:

```
# Entrada: IMSI da resposta SRI-for-SM
imsi = "001001555123456"

# Passo 1: Remover prefixo PLMN
plmn_prefix = "001001"
subscriber_portion = String.slice(imsi, 6, 9)  # = "555123456"

# Passo 2: Remover zeros à esquerda para obter o MSISDN real
msisdn = String.replace_leading(subscriber_portion, "0", "")  # =
"555123456"
```

Exemplos de Mapeamento Reverso:

IMSI de Entrada	Prefixo PLMN	Porção do Assinante	Remover Zeros à Esquerda	MSISDI Final
`"001001555123456"`	`"001001"`	`"555123456"`	`"555123456"`	`"5551234`
`"001001000000099"`	`"001001"`	`"000000099"`	`"99"`	`"99"`
`"001001999999999"`	`"001001"`	`"999999999"`	`"999999999"`	`"9999999`

Propriedades deste Mapeamento:

- **Determinístico**: O mesmo MSISDN sempre produz o mesmo IMSI
- **Reversível**: Pode converter de volta de IMSI para MSISDN
- **Configuração Mínima**: Apenas requer `hlr_imsi_plmn_prefix`
- **Preservação da Privacidade**: IMSIs reais nunca expostos
- **Sem Consulta ao Banco de Dados**: Cálculo rápido, nenhuma chamada de API necessária
- **Sempre 15 Dígitos**: O IMSI é sempre exatamente 15 dígitos

Tratamento de Entrada MSISDN:

Quando o HLR recebe uma requisição SRI-for-SM, o MSISDN passa por decodificação TBCD:

1. **Decodificação TBCD**: Converter TBCD binário para string (pode incluir prefixo TON/NPI como "91")
2. **Extrair Dígitos**: Manter apenas dígitos numéricos, remover quaisquer caracteres não numéricos
3. **Normalizar**: Se mais longo que o espaço disponível, pegar os dígitos mais à direita; se mais curto, preencher à esquerda com zeros
4. **Codificar**: Concatenar prefixo PLMN + MSISDN normalizado

Considerações de Segurança:

Os IMSIs sintéticos retornados nas respostas SRI-for-SM são puramente para fins de roteamento. Eles NÃO são os IMSIs reais armazenados no banco de dados de assinantes do HLR. Isso fornece uma camada adicional de proteção à privacidade, pois os IMSIs reais dos assinantes são expostos apenas quando absolutamente necessário (por exemplo, durante operações de UpdateLocation ou SendAuthenticationInfo que requerem vetores de autenticação reais).

Fluxo de Resposta:

```
1. SMSc → HLR: Requisição SRI-for-SM
   - MSISDN (TBCD): "91123456789" (inclui TON/NPI)

2. Processamento do HLR:
   - Decodificação TBCD: "91123456789"
   - Extrair dígitos: "91123456789" (11 dígitos)
   - Ajustar para 9 dígitos: "555123456" (9 mais à direita)
   - Adicionar PLMN: "001001" + "555123456" = "001001555123456"
   - Obter GT do SMSC da configuração: "5551234567"

3. HLR → SMSc: Resposta SRI-for-SM
   - IMSI: "001001555123456" (sintético, sempre 15 dígitos)
   - Número do Nó da Rede: "5551234567" (onde enviar MT-ForwardSM)

4. SMSc envia MT-ForwardSM para "5551234567" com IMSI
"001001555123456"
```

Configuração:

Os seguintes parâmetros são usados em `config/config.exs`:

```
# Prefixo PLMN: MCC (001 = Rede de Teste) + MNC (01 = Operador de
Teste)
hlr_imsi_plmn_prefix: "001001",

# Extração NSN (se MSISDNs incluírem código do país)
hlr_msisdn_country_code: "1",      # Usado para mapeamento reverso
(IMSI→MSISDN)
hlr_msisdn_nsn_offset: 1,          # Ignorar 1 dígito do código do
país
hlr_msisdn_nsn_length: 10          # Extrair NSN de 10 dígitos
```

Configuração de Extração NSN:

Se seus MSISDNs incluírem o código do país (por exemplo, `"99988000088"` em vez de apenas `"88000088"`), você deve configurar a extração do NSN:

- **`hlr_msisdn_nsn_offset`**: Posição onde o NSN começa (tipicamente o comprimento do seu código do país)
- **`hlr_msisdn_nsn_length`**: Número de dígitos no NSN

Exemplos:

Exemplo	Código do País	Exemplo de MSISDN	nsn_offset	nsn_length	N Extr
CC de 1 dígito	`"9"`	`"95551234567"`	`1`	`10`	`"5551`
CC de 2 dígitos	`"99"`	`"99412345678"`	`2`	`9`	`"4123`
CC de 3 dígitos	`"999"`	`"99988000088"`	`3`	`8`	`"8800`

Como Funciona:

1. **MSISDN → IMSI**: Extrair NSN do MSISDN, preencher com zeros à esquerda, concatenar com o prefixo PLMN

```
MSISDN: "99988000088"
NSN: String.slice("99988000088", 3, 8) = "88000088"
NSN Preenchido: "088000088" (9 dígitos)
IMSI: "547050" + "088000088" = "547050088000088"
```

2. **IMSI → MSISDN**: Remover prefixo PLMN, remover zeros à esquerda, preceder código do país

```
IMSI: "547050088000088"
Porção do Assinante: "088000088"
Remover zeros: "88000088"
MSISDN: "+999" + "88000088" = "+99988000088"
```

Requisitos da API: Nenhum - SRI-for-SM usa valores calculados e configuração apenas. Nenhuma

chamada de API ao backend é necessária.

Resumo da Fonte dos Campos

Tipo de Fonte	Descrição	Exemplos
OmniHSS API	Dados dinâmicos do banco de dados de assinantes do OmniHSS	IMSI, MSISDN, VLR/MSC de serviço da circuit_session
config.exs	Parâmetros de configuração do OmniSS7	`smsc_service_center_gt_address`, `camel_service_key`, `isd_network_access_mode`
Estático	Valores codificados no gerador de respostas	Status do assinante, serviços de transporte, códigos SS
Requisição	Campos extraídos da requisição MAP recebida	IMSI da UpdateLocation, MSISDN da SRI
Calculado	Valores derivados usando lógica	IMSI sintético na SRI-for-SM (hlr_imsi_prefix + NSN)

Dependências de Configuração

Obrigatório em config/config.exs:

- `hlr_service_center_gt_address` - Usado nas respostas UpdateLocation
- `smsc_service_center_gt_address` - Usado nas respostas SRI-for-SM (onde o MT-ForwardSM deve ser roteado)

Opcional em config/config.exs (com padrões):

- `camel_service_key` - Padrão: `11_110`
- `camel_trigger_detection_point` - Padrão: `:termAttemptAuthorized`
- `isd_network_access_mode` - Padrão: `:packetAndCircuit`
- `isd_send_ss_data` - Padrão: `true`
- `isd_send_call_barring` - Padrão: `true`
- `hlr_imsi_plmn_prefix` - Padrão: `"001001"` (prefixo PLMN para mapeamento MSISDN↔IMSI)

Obrigatório do OmniHSS:

OmniHSS deve fornecer endpoints de API REST para:

- Consulta de assinantes por IMSI e MSISDN
- Atualizações de localização de sessão de circuito (atribuição de VLR/MSC)
- Geração de vetores de autenticação
- Consultas de status de assinante e perfil de serviço

Documentação Relacionada

Documentação OmniSS7:

- ← Voltar à Documentação Principal
- Guia de Recursos Comuns
- Guia do Cliente MAP
- Referência Técnica
- Referência de Configuração

Documentação OmniHSS: Para gerenciamento de assinantes, provisionamento, configuração de autenticação e operações administrativas,

consulte a **documentação do produto OmniHSS**. O OmniHSS contém toda a lógica do banco de dados de assinantes, algoritmos de autenticação, regras de provisionamento de serviços e capacidades de gerenciamento de Multi-IMSI.

OmniSS7 por Omnitouch Network Services

Guia de Configuração do Cliente MAP

← Voltar para a Documentação Principal

Este guia cobre como operar o OmniSS7 como um **Cliente MAP**. Neste papel, o OmniSS7 se conecta a uma rede SS7 como um Processo de Servidor de Aplicação (ASP). Ele emite operações MAP (Mobile Application Part) em direção a HLRs, MSC/VLRs, SGSNs, GMLCs e outros elementos da rede central. O comportamento do protocolo segue 3GPP TS 29.002.

Índice

1. O que é o Modo Cliente MAP?
2. Habilitando o Modo Cliente MAP
3. Referência de Operações MAP
4. Exemplos Comuns de Operações

5. Enviando Solicitações via API
6. Métricas e Monitoramento
7. Solução de Problemas

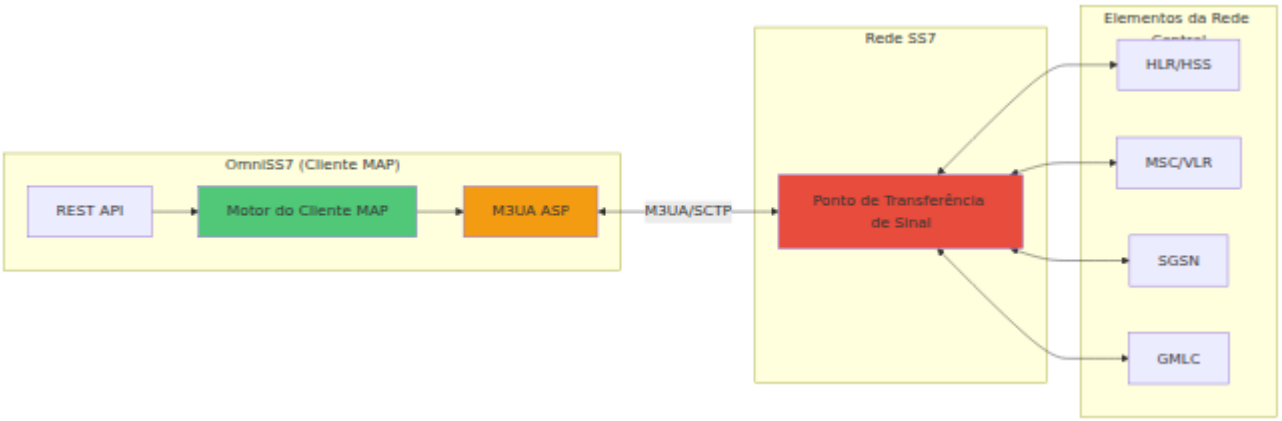
O que é o Modo Cliente MAP?

Modo Cliente MAP permite que o OmniSS7 se conecte como um **Processo de Servidor de Aplicação (ASP)** a um par M3UA (STP ou SGP) via SCTP e troque mensagens **MAP**. Cada operação é exposta como um endpoint HTTP. A API codifica uma solicitação como um diálogo TCAP/MAP e a transporta via SCCP e M3UA para a rede SS7. O OmniSS7 então decodifica a resposta da rede e a retorna de forma síncrona para o chamador.

Usos típicos incluem:

- **Gerenciamento de mobilidade**: Atualizar Localização, Cancelar Localização, Purgar MS, Fornecer Número de Roaming
- **Tratamento de chamadas**: Enviar Informações de Roteamento, Fornecer Informações do Assinante
- **SMS**: SRI-for-SM, MO/MT-ForwardSM, relatório de status de entrega
- **Autenticação**: Enviar Informações de Autenticação
- **Dados do assinante**: Inserir/Excluir Dados do Assinante, Interrogatório a Qualquer Momento
- **Serviços suplementares**: registrar/apagar/ativar/interrogar SS, gerenciamento de senhas
- **Serviços de localização (LCS)** e **roteamento GPRS**

Arquitetura da Rede



Habilitando o Modo Cliente MAP

O modo Cliente MAP é configurado em `config/runtime.exs`. Para a referência completa de conexão M3UA, veja Parâmetros de Conexão M3UA.

```
config :omniss7,
  # Habilitar o modo Cliente MAP
  map_client_enabled: true,

  # Conexão M3UA para o Cliente MAP (conecta como um ASP a um
STP/SGP remoto)
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :map_client_asp,
    local_ip: {10, 0, 0, 100},
    local_port: 2905,
    remote_ip: {10, 0, 0, 1},
    remote_port: 2905,
    routing_context: 1
  }
```

Parâmetros `map_client_m3ua`

Parâmetro	Tipo	Obrigatório	Padrão	Descrição
`mode`	String	Sim	-	Papel M3UA. `"ASP"` para modo cliente (conecta-se a um STP/SGP).
`callback`	Tupla	Sim	-	`{Módulo, Função, Args}` invocado para cada carga M3UA recebida. Para o Cliente MAP, isso é `{MapClient, :handle_payload, []}`.
`process_name`	Átomo	Sim	-	Nome registrado do processo M3UA ASP.
`local_ip`	Tupla	Sim	-	Endereço de ligação SCTP local como uma tupla IP, por exemplo, `{10, 0, 0, 100}`.
`local_port`	Inteiro	Não	2905	Porta SCTP local. 2905 é a porta M3UA registrada na IANA.
`remote_ip`	Tupla	Sim	-	Endereço IP do STP/SGP remoto como uma tupla.

Parâmetro	Tipo	Obrigatório	Padrão	Descrição
`remote_port`	Inteiro	Não	2905	Porta SCTP do STP/SGP remoto.
`routing_context`	Inteiro	Não	-	Valor do Contexto de Roteamento M3UA, quando o par requer um.

> Chaves de nível superior: `map_client_enabled` (Booleano, padrão `false`) ativa o Cliente MAP; `map_client_m3ua` (Mapa) contém a conexão acima.

Você configura o endereçamento SCCP para operações de saída (Títulos Globais de Chamadas/Chamadas, SSNs, códigos de ponto) separadamente. Veja a Referência de Configuração.

Referência de Operações MAP

Cada operação abaixo é acessível como `POST /api/<endpoint>`. Os opcodes são os códigos de operação MAP do 3GPP TS 29.002. As solicitações são síncronas: a API bloqueia até que a resposta SS7 chegue ou a solicitação expire (veja Códigos de Resposta).

Gerenciamento de Mobilidade

Operação	Endpoint	Opcode	Descrição
Atualizar Localização	`/api/updateLocation`	2	Registrar o VLR de atendimento de um assinante com o HLR.
Cancelar Localização	`/api/cancelLocation`	3	Instruir um VLR a excluir um registro de assinante.
Fornecer Número de Roaming	`/api/prn`	4	Obter um MSRN do MSC de atendimento.
Purgar MS	`/api/purgeMS`	67	Marcar um assinante como purgado no HLR.
Enviar Identificação	`/api/sendIdentification`	55	Recuperar dados de IMSI/autenticação do VLR anterior.
Reiniciar	`/api/reset`	37	Reinicialização iniciada pelo HLR em direção a um VLR.
Restaurar Dados	`/api/restoreData`	57	O VLR solicita a restauração dos dados do assinante do HLR.

Tratamento de Chamadas & Dados do

Assinante

Operação	Endpoint	Opcode	
Enviar Informações de Roteamento	`/api/sri`	22	Cons infor rote char
Fornecer Informações do Assinante	`/api/provideSubscriberInfo`	70	Solic esta assi
Excluir Dados do Assinante	`/api/deleteSubscriberData`	8	Rem assi
Interrogatório a Qualquer Momento	`/api/anyTimeInterrogation`	71	gsm infor do a
Interrogatório de Assinatura a Qualquer Momento	`/api/anyTimeSubscriptionInterrogation`	62	Cons assi
Modificação a Qualquer Momento	`/api/anyTimeModification`	65	Mod assi
Notificar Dados do Assinante Modificados	`/api/noteSubscriberDataModified`	5	HLR sobr
Notificar Evento MM	`/api/noteMM-Event`	89	Rela gere mob

SMS

Operação	Endpoint	Opcode	Descrição
Enviar Informações de Roteamento para SM	`/api/sri-for-sm`	45	Consultar o HLR para o nó de atendimento para entrega de SMS.
MT-Forward SM	`/api/MT-forwardSM`	44	Entregar um SM terminado em móvel ao MSC/SGSN de atendimento.
MO-Forward SM	`/api/forwardSM`	46	Submeter um SM originado em móvel ao SMSC.
Enviar SM (PDU)	`/api/sendSM`, `/api/deliverPDU`	44	Construir e entregar um PDU SMS-DELIVER.
Relatar Status de Entrega de SM	`/api/reportSM-DeliveryStatus`	47	Relatar o resultado da entrega de SM ao HLR.
Pronto para SM	`/api/readyForSM`	66	Notificar o HLR que um MS está acessível para SMS.
Alertar Centro de Serviço	`/api/alertServiceCentre`	64	Alertar um SMSC que um

Operação	Endpoint	Opcode	Descrição
			assinante está disponível.

Autenticação & Equipamento

Operação	Endpoint	Opcode	Descrição
Enviar Informações de Autenticação	`/api/send-auth-info`	56	Recuperar vetores de autenticação do HLR.
Enviar IMSI	`/api/sendIMSI`	58	Resolver um MSISDN para seu IMSI no HLR.
Verificar IMEI	`/api/checkIMEI`	43	Consultar status do equipamento no EIR.

Serviços Suplementares

Operação	Endpoint	Opcode	Descrição
Registrar SS	`/api/registerSS`	10	Registrar um serviço suplementar (por exemplo, encaminhamento de chamadas / CFU).
Apagar SS	`/api/eraseSS`	11	Apagar um serviço suplementar.
Ativar SS	`/api/activateSS`	12	Ativar um serviço suplementar.
Desativar SS	`/api/deactivateSS`	13	Desativar um serviço suplementar.
Interrogar SS	`/api/interrogateSS`	14	Interrogar o status de um serviço suplementar.
Registrar Senha	`/api/registerPassword`	17	Registrar uma senha SS.
Obter Senha	`/api/getPassword`	18	Recuperar/verificar uma senha SS.
Encaminhar Verificação de Indicação de SS	`/api/forwardCheckSS-Indication`	38	Verificação não confirmada em direção a um VLR.

Serviços de Localização (LCS) & GPRS

Operação	Endpoint	Opcode	Descrição
Enviar Informações de Roteamento para LCS	`/api/sendRoutingInfoForLCS`	85	GMLC consulta o HLR para roteamento LCS.
Fornecer Localização do Assinante	`/api/provideSubscriberLocation`	83	Solicitar a localização de um assinante do nó de atendimento.
Relatório de Localização do Assinante	`/api/subscriberLocationReport`	86	Relatar a localização de um assinante ao GMLC.
Enviar Informações de Roteamento para GPRS	`/api/sendRoutingInfoForGprs`	24	Consultar o HLR para roteamento GPRS (seleção de GGSN).
Atualizar Localização GPRS	`/api/updateGprsLocation`	23	Registrar o SGSN de atendimento de um assinante no HLR.

> **Escopo LCS:** as operações de Serviços de Localização são **somente de saída (solicitante)**. O Cliente MAP as aciona em direção a um HLR, nó de atendimento ou GMLC. O nó não atua como um *servidor* LCS (não responderá a um ProvideSubscriberLocation recebido ou processará um SubscriberLocationReport recebido). Os codificadores transportam os elementos de informação obrigatórios, além das opções principais mencionadas acima. O conjunto de parâmetros LCS mais amplo (QoS, prioridade, formas GAD, palavra-código, número de referência) ainda não está exposto.

CAP / CAMEL & Injeção Bruta

Operação	Endpoint	Descrição
CAP InitialDP	`/api/cap/initialDP`	Acionar um diálogo CAMEL em direção a um gsmSCF.
CAP Connect / Continue / ReleaseCall / ApplyCharging	`/api/cap/connect`, `/api/cap/continue`, `/api/cap/releaseCall`, `/api/cap/applyCharging`	Operações de controle de chamada CAMEL. Veja o Guia do Gateway CAMEL.
Raw SCCP	`/api/raw/sccp`	Injetar uma carga SCCP construída manualmente (teste/diagnósticos).
Raw M3UA	`/api/raw/m3ua`	Injetar uma carga M3UA construída manualmente (teste/diagnósticos).

Exemplos Comuns de Operações

Enviar Informações de Roteamento para SM (SRI-for-SM)

Consulta o HLR para o nó de atendimento usado para entregar um SMS. Para como o lado do HLR processa isso, veja SRI-for-SM no Guia do HLR.

Endereçamento SCCP (caminho de entrega de SMS do SMSc). Por padrão, o SRI-for-SM é enviado **roteado por GT endereçado ao MSISDN de destino** (chamado SSN 6/HLR), o que requer que o STP mantenha uma entrada GTT para esse intervalo de números. Para assinantes **on-net**, o SMSc endereça o SRI diretamente ao **HLR GT** configurado. Ele então roteia no próprio endereço de par (já roteável) do HLR, de modo que nenhuma GTT por intervalo é necessária, e o HLR retorna o verdadeiro MSC de atendimento. Isso é selecionado por destino:

Configuração (`:omniss7`)	Propósito	Exemplo
`hlr_gt_address`	GT do HLR de origem para endereçar SRIs on-net	`"61455501111"`
`home_msisdn_prefixes`	Prefixos de MSISDN on-net; uma correspondência endereça o HLR GT	`["6777"]`

Um destino que corresponda a uma entrada `home_msisdn_prefixes` (com `hlr_gt_address` definido) é enviado ao HLR GT com SSN chamado 6; tudo o mais mantém o padrão de roteamento por GT por MSISDN. A solicitação SRI-for-SM aceita `sccp_opts` (`:called_party`, `:called_ssn`) para substituir o partido chamado SCCP para qualquer chamador.

Endpoint: `POST /api/sri-for-sm`

Solicitação:

```
{
  "msisdn": "447712345678",
  "serviceCenter": "447999123456"
}
```

Resposta:

```
{
  "result": {
    "imsi": "234509876543210",
    "locationInfoWithLMSI": {
      "networkNode-Number": "447999555111"
    }
  }
}
```

cURL:

```
curl -X POST http://localhost/api/sri-for-sm \
  -H "Content-Type: application/json" \
  -d '{"msisdn": "447712345678", "serviceCenter": "447999123456"}'
```

Enviar Informações de Roteamento (SRI)

Endpoint: `POST /api/sri`

Solicitação:

```
{
  "msisdn": "447712345678",
  "gmsc": "447999123456"
}
```

Fornecer Número de Roaming (PRN)

Endpoint: `POST /api/prn`

Solicitação:

```
{
  "msisdn": "447712345678",
  "gmsc": "447999123456",
  "msc_number": "447999555111",
  "imsi": "234509876543210"
}
```

Enviar Informações de Autenticação

Endpoint: `POST /api/send-auth-info`

Solicitação:

```
{
  "imsi": "234509876543210",
  "vectors": 5
}
```

Resposta:

```
{
  "result": {
    "authenticationSetList": [
      {
        "rand": "0123456789ABCDEF0123456789ABCDEF",
        "xres": "ABCDEF0123456789",
        "ck": "0123456789ABCDEF0123456789ABCDEF",
        "ik": "FEDCBA9876543210FEDCBA9876543210",
        "autn": "0123456789ABCDEF0123456789ABCDEF"
      }
    ]
  }
}
```

Atualizar Localização

Registra o VLR de atendimento de um assinante no HLR, que responde com a sequência de Inserir Dados do Assinante. Veja Atualizações de Localização no Guia do HLR.

Endpoint: `POST /api/updateLocation`

Solicitação:

```
{
  "imsi": "234509876543210",
  "vlr": "447999555111"
}
```

Interrogatório a Qualquer Momento (ATI)

Endpoint: `POST /api/anyTimeInterrogation`

Solicitação:

```
{
  "hlr_gt": "447999123456",
  "msisdn": "447712345678"
}
```

Categorias de Operação



Enviando Solicitações via API

Usando Swagger UI

A interface Swagger UI fornece uma interface interativa para cada endpoint.

1. Navegue até `http://your-server/swagger`.
2. Selecione um endpoint (por exemplo, `/api/sri-for-sm`) e clique em **Experimente**.
3. Preencha o corpo da solicitação e clique em **Executar**.
4. A resposta SS7 decodificada é exibida abaixo.

O documento OpenAPI bruto está disponível em `http://your-server/swagger.json`.

Códigos de Resposta da API

Código	Significado
200	Sucesso: o resultado decodificado está no corpo da resposta.
400	Solicitação inválida: parâmetros inválidos ou ausentes.
504	Tempo limite do gateway: sem resposta SS7 dentro do tempo limite da solicitação (padrão de 10 segundos).

Algumas operações são não confirmadas (fire-and-forget) e retornam imediatamente após o envio da mensagem.

Métricas do Cliente MAP

Todas as métricas são exportadas no endpoint Prometheus (`/metrics`).

Métrica: `map_requests_total` **Tipo:** Contador **Descrição:** Total de solicitações MAP enviadas. **Rótulos:** `operation`: por exemplo `sri`,

`sri_for_sm`, `prn`, `authentication_info`, `update_location`.

Métrica: `map_request_errors_total` **Tipo:** Contador **Descrição:** Total de solicitações MAP que resultaram em erro ou tempo limite. **Rótulos:** `operation`.

Métrica: `map_request_duration_milliseconds` **Tipo:** Histograma **Descrição:** Distribuição dos tempos de ida e volta das solicitações MAP. **Rótulos:** `operation`.

Métrica: `map_pending_requests` **Tipo:** Gauge **Descrição:** Solicitações MAP atualmente aguardando uma resposta.

Exemplos de Consultas Prometheus

```
# Solicitações SRI-for-SM na última hora
increase(map_requests_total{operation="sri_for_sm"}[1h])

# Tempo médio de resposta SRI
rate(map_request_duration_milliseconds_sum{operation="sri"}[5m])
  / rate(map_request_duration_milliseconds_count{operation="sri"}
[5m])

# Taxa de erro por operação
sum(rate(map_request_errors_total[5m])) by (operation)

# Solicitações pendentes atuais
map_pending_requests
```

Solução de Problemas do Cliente MAP

Solicitações Expiram (504)

Sintomas: A API retorna 504; sem resposta do HLR/MSC.

Causas possíveis:

- A associação M3UA ao STP/SGP não está ATIVA.
- O STP não tem rota para o Título Global de destino.
- Endereçamento SCCP incorreto (GT de Chamado/Chamador, SSN) ou Contexto de Roteamento.

Resolução:

1. Confirme o status da associação M3UA no painel ou através das visualizações de status do Guia STP.
2. Verifique a conectividade da rede (SCTP) com o STP.
3. Confirme que o Título Global de destino é roteável e que o subsistema de destino (por exemplo, HLR SSN 6) está em serviço.
4. Revise os logs em busca de mensagens de retorno SCCP (veja abaixo).

Mensagens de Retorno / Erro SCCP

Sintomas: Logs mostram mensagens de retorno do Serviço Unitdata SCCP (UDTS).

Causas comuns de retorno:

- **Sem Tradução para o Endereço**: o Título Global não está na tabela de roteamento do STP.
- **Falha do Subsistema**: o subsistema de destino (por exemplo, HLR SSN 6) está indisponível.
- **Falha / Congestionamento da Rede**: problema transitório na rede.

Resolução:

- Confirme a configuração de roteamento/tradução do STP para o GT de destino.
- Verifique se o subsistema de destino está operacional.
- Tente novamente após a limpeza do congestionamento.

Documentação Relacionada

- ← Voltar para a Documentação Principal
- Referência de Configuração: todos os parâmetros de configuração
- Guia de Recursos Comuns: Interface Web, API, monitoramento
- Guia STP: roteamento e Tradução de Título Global
- Guia HLR: processamento de operações do lado do HLR
- Guia do Centro de SMS: entrega de SMS
- Guia do Gateway CAMEL: operações CAP/CAMEL

OmniSS7 por Omnitouch Network Services

Guia de Configuração do Centro de SMS (SMSc)

← Voltar para a Documentação Principal

Este guia fornece configuração detalhada para usar o OmniSS7 como um **Centro de SMS (SMSc)** na frente com o **OmniMessage** como o armazenamento de mensagens e plataforma de entrega no backend.

Integração com o OmniMessage

O modo SMSc do OmniSS7 funciona como uma frente de sinalização SS7 que se conecta ao **OmniMessage**, uma plataforma de SMS de nível de operadora. Esta arquitetura separa as preocupações:

- **OmniSS7 (Frontend SMSc)**: Gerencia toda a sinalização do protocolo SS7/MAP, roteamento SCCP e comunicação de rede
- **OmniMessage (Backend SMS)**: Gerencia armazenamento de mensagens, enfileiramento, lógica de repetição, rastreamento de entrega e decisões de roteamento

Por que o OmniMessage?

O OmniMessage fornece capacidades de mensagens SMS de nível de operadora com recursos incluindo:

- **Gerenciamento de Fila de Mensagens**: Armazenamento persistente com lógica de repetição configurável e enfileiramento por prioridade
- **Rastreamento de Entrega**: Status de entrega em tempo real, relatórios de entrega (DLR) e rastreamento de razões de falha
- **Suporte a Múltiplos SMSc**: Várias instâncias de frontend podem se conectar a um único backend OmniMessage para balanceamento de carga

e redundância
- **Inteligência de Roteamento**: Regras de roteamento avançadas baseadas em destino, remetente, conteúdo da mensagem e hora do dia
- **Limitação de Taxa**: Controles de TPS (transações por segundo) por rota para evitar congestionamento da rede
- **Design API-First**: API RESTful HTTP para integração com sistemas de cobrança, portais de clientes e aplicações de terceiros
- **Análise e Relatórios**: Estatísticas de volume de mensagens, taxas de sucesso de entrega e métricas de desempenho

Todos os dados de mensagens, estado de entrega e configurações de roteamento são armazenados e gerenciados no OmniMessage. O OmniSS7 consulta o OmniMessage por meio de chamadas de API HTTPS para recuperar mensagens pendentes, atualizar o status de entrega e registrar-se como um frontend ativo.

Importante: O modo SMSc do OmniSS7 é um **frontend de sinalização apenas**. Toda a lógica de roteamento de mensagens, gerenciamento de filas, algoritmos de repetição, rastreamento de entrega e regras de negócios são tratados pelo OmniMessage. Este guia cobre a configuração do protocolo SS7/MAP no OmniSS7. Para informações sobre roteamento de mensagens, configuração de filas, relatórios de entrega, limitação de taxa e análises, **consulte a documentação do OmniMessage**.

Índice

1. Integração com o OmniMessage
2. O que é o Modo Centro de SMS?
3. Habilitando o Modo SMSc
4. Configuração da API HTTP
5. Fluxos de Mensagens SMS
6. Tratamento do Centro de Serviço de Alerta
7. Prevenção de Loop
8. Rastreamento de Assinantes SMSc
9. Cache de Endereço VLR

10. Configuração de Auto-Flush
11. Métricas e Monitoramento
12. Solução de Problemas

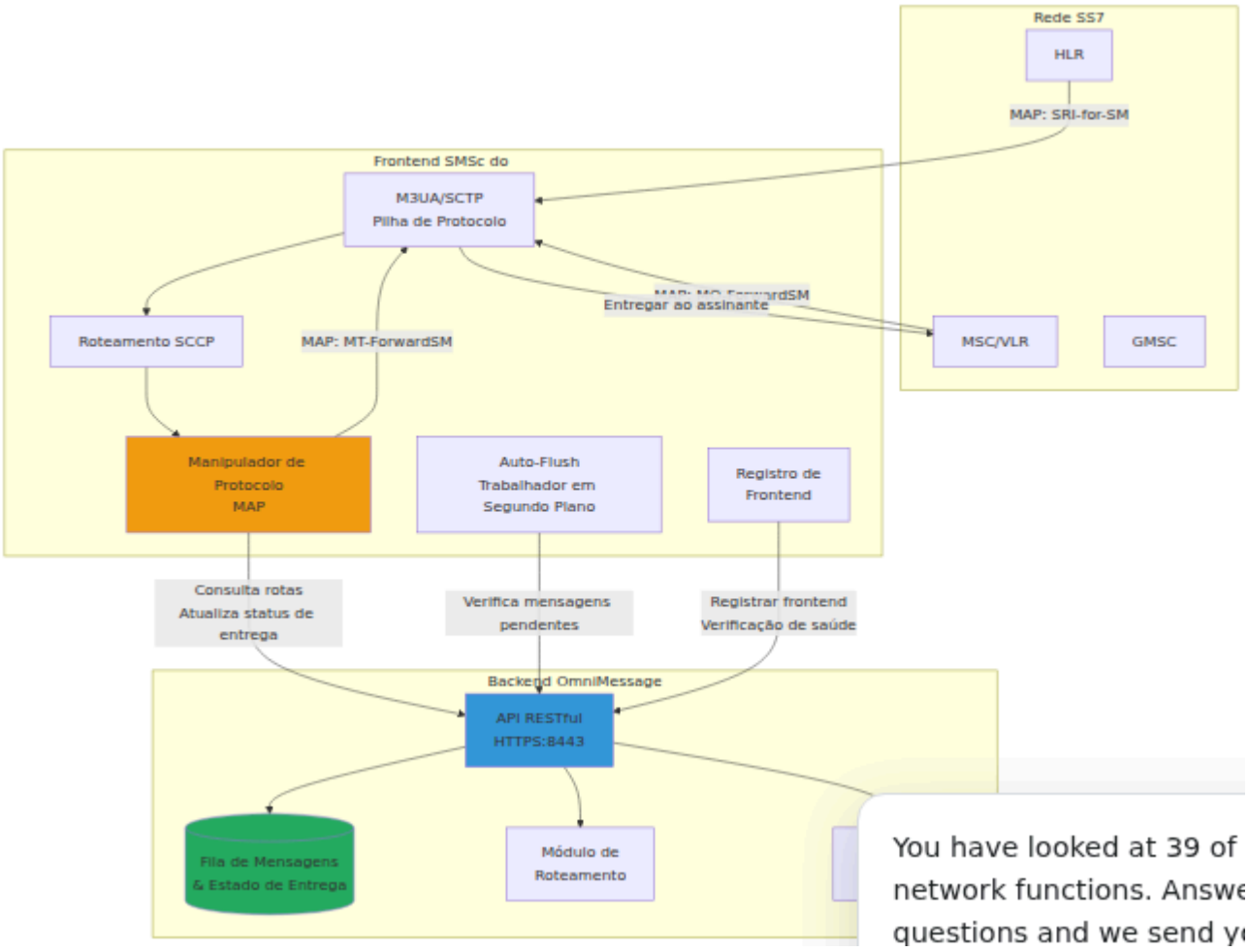
O que é o Modo Centro de SMS?

Nota: Esta seção cobre apenas a configuração de sinalização SS7 do OmniSS7. Para regras de roteamento de mensagens, gerenciamento de filas, rastreamento de entrega e configuração de lógica de negócios, consulte a **documentação do produto OmniMessage**.

Modo Centro de SMS permite que o OmniSS7 funcione como um SMSc para:

- **Entrega de MT-SMS**: Entrega de SMS Terminados em Móvel para assinantes
- **Tratamento de MO-SMS**: Recepção e roteamento de SMS Originados em Móvel
- **Enfileiramento de Mensagens**: Fila de mensagens com suporte a banco de dados e lógica de repetição
- **Auto-Flush**: Entrega automática de SMS da fila
- **Relatórios de Entrega**: Rastrear o status de entrega da mensagem

Arquitetura do Centro de SMS



Habilitando o Modo SMSc

O OmniSS7 pode operar em diferentes modos (STP, HLR, SMSc). O modo é selecionado através de um conjunto de flags de modo e a configuração associada em `config/config.exs`, o arquivo de configuração padrão da aplicação.

Mudando para o Modo SMSc

Para executar o OmniSS7 como um SMSc, defina as flags de modo para que os recursos do SMSc sejam habilitados, e adicione a configuração específica do SMSc:

1. **Abra** `config/config.exs`
2. **Defina** as flags de modo sob `config :omniss7`:
   - `map_client_enabled: true`: necessário para capacidades de roteamento/MAP
   - `smsc_mode_enabled: true`: habilita o tratamento específico do SMSc
   - `hlr_mode_enabled: false`: desabilita recursos do HLR
3. **Adicione** os parâmetros de configuração do SMSc (endpoint da API, nome do SMSc, endereço GT, configurações de auto-flush, conexão M3UA) mostrados abaixo
4. **Personalize** os parâmetros para sua implantação
5. **Reinicie** a aplicação para que as mudanças tenham efeito

> **Nota**: `config/runtime.exs` é um stub mínimo usado apenas para testes automatizados. Ele **não** contém definições de modo operacional. Toda a configuração de modo e SMSc reside em `config/config.exs` (com substituições específicas de ambiente em arquivos como `config/test.exs`). Implantações em produção fornecem seu próprio `config/config.exs`.

Configuração do Modo SMSc

A configuração completa do SMSc (em `config/config.exs`) se parece com isto:

```
config :omniss7,
  # Flags de Modo - Habilitar recursos STP + SMSc
  # Nota: map_client_enabled é true porque SMSc precisa de
capacidades de roteamento
  map_client_enabled: true,
  hlr_mode_enabled: false,
  smsc_mode_enabled: true,

  # Configuração da API do Backend OmniMessage
  smsc_api_base_url: "https://omnimessage.example.com:8443",
  # Identificação do SMSc para registro com o backend
  smsc_name: "ipsmgw",
  # Endereço GT do Centro de Serviço para operações de SMS
  smsc_service_center_gt_address: "5551234567",

  # Configuração de Auto Flush (processamento de fila de SMS em
segundo plano)
  auto_flush_enabled: true,
  auto_flush_interval: 10_000,
  auto_flush_dest_smsc: "ipsmgw",
  auto_flush_tps: 10,

  # Configuração de Conexão M3UA
  # Conectar como ASP para enviar/receber operações MAP SMS
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :stp_client_asp,
    # Endpoint local (sistema SMSc)
    local_ip: {10, 179, 4, 12},
    local_port: 2905,
    # Endpoint remoto STP
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }
```

As páginas da interface web são registradas separadamente sob `config :control_panel` em `config/config.exs`. A lista padrão `use_additional_pages` registra todas as páginas do OmniSS7 (~16 entradas); o runtime filtra páginas que não se aplicam ao modo ativo. Note que a página de status de pares é rotulada como **"Peers"**:

```
config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.TestClientLive, "/client", "Cliente SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "Peers"},
    {SS7.Web.SCTPConnectionsLive, "/sctp", "Conexões SCTP"},
    {SS7.Web.RoutingLive, "/routing", "Roteamento"},
    {SS7.Web.RoutingTestLive, "/routing_test", "Teste de
Roteamento"},
    {SS7.Web.SmscLinksLive, "/smsc_links", "Links SMSc"},
    {SS7.Web.SmscSubscribersLive, "/smsc_subscribers", "Assinantes
SMSc"},
    {SS7.Web.HlrLinksLive, "/hlr_links", "Links HLR"},
    # ... páginas adicionais (Assinantes Ativos, CAMEL, Logs,
Teste MSISDN/IMSI, etc.)
  ],
  page_order: [
    "/application", "/configuration", "/events", "/logs",
"/client", "/m3ua",
    "/routing", "/routing_test", "/smsc_links",
"/smsc_subscribers", "/hlr_links",
    # ... caminhos restantes ordenados
  ]
```

Parâmetros de Configuração para Personalizar

Para uma referência completa de todos os parâmetros de configuração, consulte a Referência de Configuração.

Parâmetro	Tipo	Padrão	De
`smsc_api_base_url`	String	*Obrigatório*	Endpoin **OmniM** backend
`smsc_name`	String	`"{hostname}_SMSc"`	Seu ide SMSc pa
`smsc_service_center_gt_address`	String	*Obrigatório*	Título G Centro d
`auto_flush_enabled`	Boolean	`true`	Habilita process automá
`auto_flush_interval`	Inteiro	`10_000`	Interval process fila em
`auto_flush_dest_smsc`	String	`nil`	Filtro de SMSc de auto-flu process mensag pendent
`auto_flush_tps`	Inteiro	`10`	Taxa de process mensag (transa
`local_ip`	Tupla	*Obrigatório*	Endereç sistema
`local_port`	Inteiro	`2905`	Porta SC

Parâmetro	Tipo	Padrão	De
`remote_ip`	Tupla	*Obrigatório*	Endereç para co SS7
`remote_port`	Inteiro	`2905`	Porta SC
`routing_context`	Inteiro	`1`	ID do co roteame

O que Acontece Quando o Modo SMSc é Habilitado

Quando `smsc_mode_enabled: true` e `map_client_enabled: true`, a interface web mostrará:

- **Eventos SS7** - Registro de eventos
- **Cliente SS7** - Teste de operações MAP
- **Peers** - Status de conexão (pares M3UA/SCTP)
- **Roteamento** - Gerenciamento da tabela de rotas (STP habilitado)
- **Teste de Roteamento** - Teste de rotas (STP habilitado)
- **Links SMSc** - Status da API SMSc + gerenciamento de fila de SMS ← *Específico do SMSc*
- **Recursos** - Monitoramento do sistema
- **Configuração** - Visualizador de configuração

A aba **Links HLR** será oculta.

Notas Importantes

- O modo SMSc requer `map_client_enabled: true` para capacidades de roteamento
- **Backend OmniMessage**: A API do OmniMessage deve ser acessível no `smsc_api_base_url` configurado

- **Registro do Frontend**: O sistema registra automaticamente com o OmniMessage a cada **~85 segundos** (`@registration_interval` = 85.000 ms) através do módulo `SMS.FrontendRegistry`
- **Tempo Limite de Requisição da API**: Todas as requisições da API do OmniMessage têm um **tempo limite fixo de 5 segundos**
- **Tempo Limite de Requisição MAP**: SRI-for-SM tem um tempo limite de diálogo de 10 segundos. MT-ForwardSM tem um **tempo limite de diálogo de 30 segundos** para acomodar respostas lentas do VLR em redes 2G/CS
- O auto-flush processa automaticamente a fila de SMS em segundo plano
- A conexão M3UA com o STP é necessária para enviar/receber operações MAP SMS
- Após mudar de modos, você deve reiniciar a aplicação para que as mudanças tenham efeito
- **Interface Web**: Consulte o Guia da Interface Web para informações sobre como usar a interface web
- **Acesso à API**: Consulte o Guia da API para documentação da API REST e acesso à Swagger UI

Configuração da API HTTP

Configuração do Backend OmniMessage

O OmniSS7 se comunica com o OmniMessage via API REST HTTPS para gerenciar a entrega de mensagens, rastrear o estado do assinante e registrar-se como um frontend ativo:

```
config :omniss7,
  # URL base da API OmniMessage
  smsc_api_base_url: "https://omnimessage.example.com:8443",
  # Identificador do nome do SMSc para registro (padrão para
hostname_SMSc se vazio)
  smsc_name: "omni-smsc01",
  # Endereço GT do Centro de Serviço para operações de SMS
  smsc_service_center_gt_address: "5551234567"
```

Parâmetros de Configuração:

Parâmetro	Tipo	Obrigatório	Padr
`smsc_api_base_url`	String	Sim	`"https://loca`
`smsc_name`	String	Não	`""` (usa `"{hostname}_SMS`
`smsc_service_center_gt_address`	String	Não	`"5551234567"`

Registro do Frontend

O sistema registra automaticamente a si mesmo com o OmniMessage na inicialização e **re-registra a cada ~85 segundos** através do módulo `SMS.FrontendRegistry`. Isso permite que o OmniMessage:

- Rastreie frontends ativos para balanceamento de carga
- Monitore tempo de atividade e status de saúde
- Colete informações de configuração
- Gerencie roteamento de SMS distribuído entre múltiplos frontends

Detalhes da Implementação:

- **Intervalo de Registro**: ~85 segundos (`@registration_interval 85_000`, fixo)
- **Processo**: Iniciado automaticamente quando `smsc_mode_enabled: true`

Payload de Registro:

```
{
  "frontend_name": "omni-smsc01",
  "configuration": "{...}",
  "frontend_type": "SS7",
  "hostname": "smsc-server01",
  "uptime_seconds": 12345
}
```

Nota: O nome do frontend é retirado do parâmetro de configuração `smsc_name`. Se não estiver definido, o padrão é `"{hostname}_SMSc"`.

Comunicação com a API do OmniMessage

Quando o OmniSS7 recebe operações MAP da rede SS7 ou processa a fila de mensagens, ele se comunica com o OmniMessage para:

- **Registrar-se como um frontend ativo** e relatar status de saúde
- **Submeter mensagens originadas em móvel (MO)** recebidas de assinantes
- **Recuperar mensagens terminadas em móvel (MT)** da fila para entrega
- **Atualizar status de entrega** com relatórios de sucesso/falha
- **Consultar informações de roteamento** para encaminhamento de mensagens

Endpoint	Método	Propósito	Corpo da R
`/api/frontends`	POST	Registrar instância de frontend	`{"frontend_nar` `"configuratior` `"frontend_type` `"hostname": ".` `"uptime_second`
`/api/messages_raw`	POST	Inserir nova mensagem SMS (TPDU bruto)	`{"rp_originato` `"...",` `"rp_destinatio` `"...", "direct` `"message_body"` `"send_time": "` `"smsc_node_nam` `"source_smsc":` `"source_type":`
`/api/messages`	GET	Obter fila de mensagens	Cabeçalho: `sms` `<smsc_name>`
`/api/messages/{id}`	PATCH	Marcar mensagem como entregue	`{"deliver_time` `"dest_smsc": "`
`/api/messages/{id}`	PUT	Atualizar status da mensagem	`{"dest_smsc":`
`/api/locations`	POST	Inserir/atualizar localização do assinante	`{"msisdn": "."` `"...", "locati` `"ims_capable":` `"csfb": false,` `"...", "user_a` `"...", "ran_lo` `"...", "imei":` `"registered":`

Endpoint	Método	Propósito	Corpo da R
`/api/events`	POST	Adicionar rastreamento de eventos	`{"message_id"` `"name": "...",` `"description":`
`/api/status`	GET	Verificação de saúde (o caminho é configurável via `smsc_api_status_path`, padrão `/api/status`; consultado pela página de Links SMSc, não pelo `SMSc.APIClient`)	-

Formato de Resposta da API

Todas as respostas da API usam o formato JSON com as seguintes convenções:

- **Respostas de sucesso**: HTTP 200-201 com corpo JSON contendo dados de resultado
- **Respostas de erro**: HTTP 4xx/5xx com detalhes de erro no corpo da resposta
- **Carimbos de data/hora**: formato ISO 8601 (por exemplo, `"2025-10-21T12:34:56Z"`)
- **IDs de mensagem**: identificadores inteiros ou de string

Módulos do Cliente da API

O sistema SMS consiste em três módulos principais:

1. SMSc.APIClient

Módulo principal do cliente da API que fornece toda a comunicação HTTP API com o OmniMessage:

- `frontend_register/4` - Registrar frontend com o OmniMessage

- `insert_message/1` - Inserir mensagem SMS bruta (recebe um único mapa de mensagem; publica em `/api/messages_raw`)
- `insert_location/9` - Inserir/atualizar dados de localização do assinante
- `get_message_queue/2` - Recuperar mensagens pendentes da fila
- `mark_dest_smsc/3` - Marcar mensagem como entregue ou falhada
- `add_event/3` - Adicionar rastreamento de eventos para mensagens
- `flush_queue/2` - Processar mensagens pendentes (SRI-for-SM + MT-forwardSM)
- `auto_flush/2` - Loop de processamento contínuo da fila

2. SMS.FrontendRegistry

Gerencia o registro periódico do frontend com o backend:

- Registra automaticamente na inicialização
- Re-registra a cada ~85 segundos
- Usa `smsc_name` da configuração (reverte para hostname)
- Coleta informações de configuração e tempo de atividade do sistema

3. SMS.Utils

Funções utilitárias para operações de SMS:

- `generate_tp_scts/0` - Gerar timestamp de SMS no formato TPDU

Fluxos de Mensagens SMS

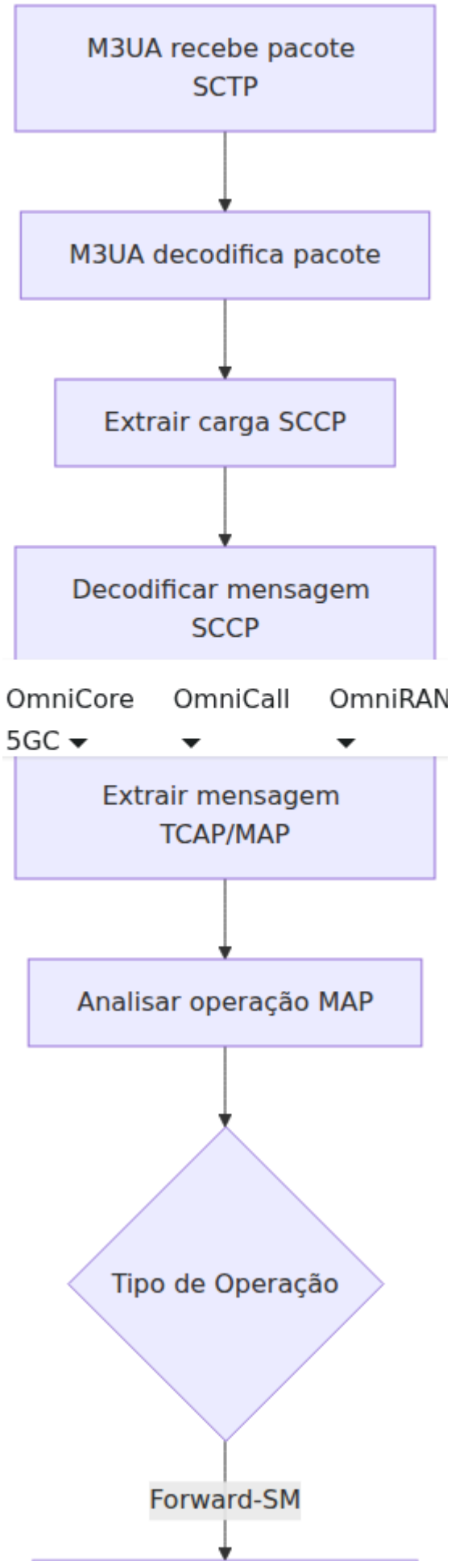
Fluxo de SMS de Entrada (Originado em Móvel)

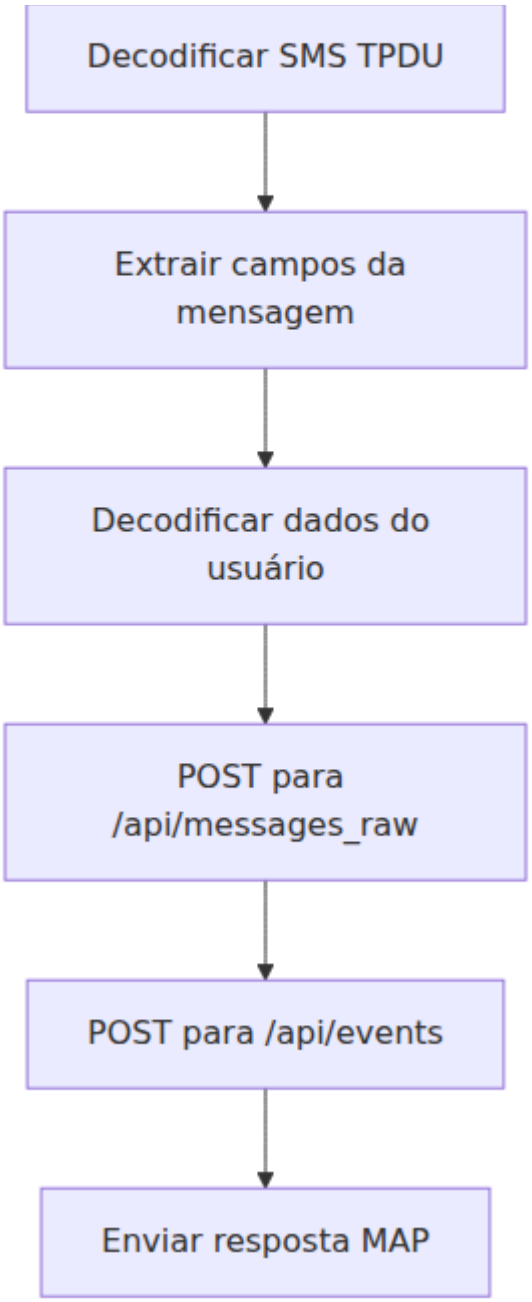

M3UA recebe pacote SCTP
M3UA decodifica pacote
Extrair carga SCCP
Decodificar mensagem SCCP
Extrair mensagem TCAP/MAP
Analisar operação MAP
Tipo de Operação
Forward-SM



Fluxo de SMS de Saída (Terminado em Móvel)

Quando uma API HSS/HLR está configurada (`hlr_api_base_url`), o IP-SM-GW pode entregar MT-SMS para **assinantes on-net** sem uma ida e volta SRI-for-SM, consultando diretamente a API HSS para o IMSI real do assinante e servindo o VLR. Um **cache VLR** no Rastreador de Assinantes evita consultas repetidas ao HSS para o mesmo assinante.


M3UA recebe pacote SCTP
M3UA decodifica pacote
Extrair carga SCCP
Decodificar mensagem SCCP
OmniCore 5GC
OmniCall
OmniRAN
Extrair mensagem TCAP/MAP
Analisar operação MAP
Tipo de Operação
Forward-SM



Principais Etapas Explicadas:

- **Verificação de Cache VLR**: Antes de consultar a API HSS, o sistema verifica o cache VLR do Rastreador de Assinantes para uma entrada válida (IMSI + endereço VLR dentro do TTL). Um hit de cache ignora completamente a API HSS. Consulte Cache de Endereço VLR para detalhes.
- **Entrega Direta da API HSS (on-net)**: Quando `hlr_api_base_url` está configurado, o IP-SM-GW consulta a API HSS com o MSISDN de destino para obter o **IMSI real** do assinante e o **endereço VLR de serviço**. Se ambos estiverem disponíveis, o IP-SM-GW envia o MT-ForwardSM diretamente para o VLR. Isso contorna completamente o SRI-for-SM.

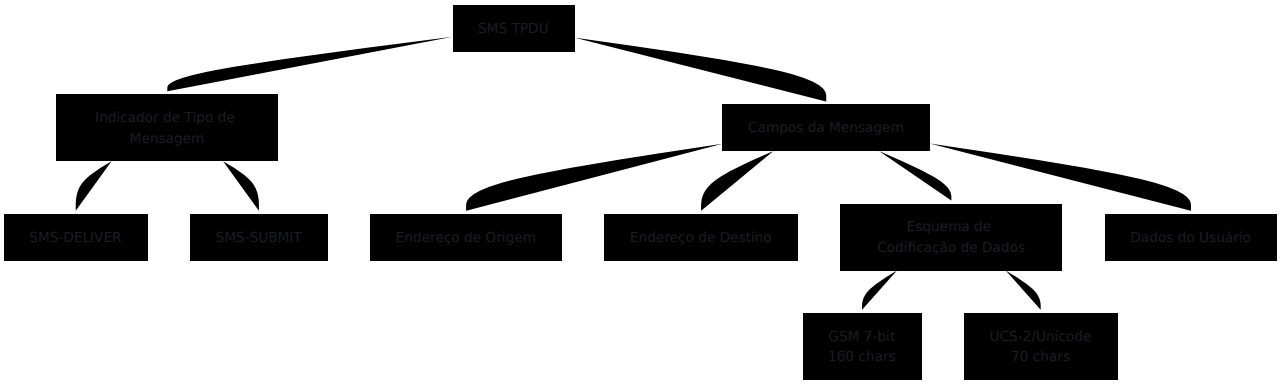
- **Assinante Ausente (on-net)**: Se a API HSS encontra o assinante, mas nenhum VLR de serviço está atribuído, a mensagem é marcada como falhada com uma entrada de localização expirada. As tentativas de entrega são feitas quando o HLR envia um alertServiceCenter após o assinante re-registrar.

- **Fallback SRI-for-SM (off-net)**: Quando a API HSS não está configurada ou o assinante não é encontrado no HSS, o fluxo padrão SRI-for-SM é utilizado. O HLR responde com um IMSI sintético e número de nó de rede. Consulte o Guia SRI-for-SM no HLR.

- **VLR Limpar em Caso de Falha**: Quando a entrega do MT-ForwardSM falha (timeout, erro SCCP, etc.), o endereço VLR em cache é limpo para que a próxima tentativa reconsulte a API HSS por um novo VLR.

- **Timeout do MT-ForwardSM**: O diálogo do MT-ForwardSM tem um **timeout de 30 segundos** para acomodar respostas mais lentas do VLR 2G/CS.

Estrutura do SMS TPDU



Tratamento do Centro de Serviço de Alerta

O SMSc pode receber mensagens **alertServiceCenter** do HLR para rastrear o status de acessibilidade do assinante.

Para informações sobre como o HLR envia mensagens alertServiceCenter, veja Integraç��o do Centro de Serviço de Alerta no Guia HLR.

O que é alertServiceCenter?

Quando um assinante realiza uma Atualização de Localização no HLR (ou seja, registra-se com um novo VLR/MSC), o HLR pode notificar os sistemas SMSc de que o assinante agora está acessível, enviando uma mensagem **alertServiceCenter** (código de operação MAP 64).

Configuração

O tempo de expiração da localização é configurado no HLR:

```
config :omniss7,
  # Tempo de expiração da localização quando o SMSc recebe
alertServiceCenter (padrão: 48 horas)
  hlr_alert_location_expiry_seconds: 172800
```

Comportamento

Quando o SMSc recebe uma mensagem alertServiceCenter:

1. **Decodificar MSISDN**: Extrair o MSISDN do assinante da mensagem (formato TBCD)
2. **Remover prefixo TON/NPI**: Remover prefixos comuns como "19", "11", "91" (por exemplo, "19123123213" → "123123213")
3. **Resolver IMSI**: Se `hlr_api_base_url` estiver configurado, consultar a API HSS para resolver o **IMSI real** do assinante. Se a consulta ao HSS falhar ou nenhuma API estiver configurada, voltar a gerar um IMSI sintético usando o mesmo mapeamento que SRI-for-SM. O IMSI real é crítico para a entrega MT na rede para assinantes 2G/CS. O MSC/VLR requer o IMSI correto no MT-ForwardSM.
4. **POST para /api/locations**: Atualizar o banco de dados de localização com:
   - `msisdn`: Número de telefone do assinante (limpo)
   - `imsi`: IMSI sintético
   - `location`: Nome do SMSc (por exemplo, "ipsmgw")
   - `expires`: Hora atual + `hlr_alert_location_expiry_seconds`
   - `csfb`: true (assinante acessível via Circuit-Switched Fallback)
   - `ims_capable`: false (isto é registro 2G/3G CS, não IMS/VoLTE)
   - `user_agent`: HLR GT que enviou o alerta (para rastreamento)
   - `ran_location`: "SS7"
5. **Rastrear no Rastreador de Assinantes do SMSc**: Registrar o assinante com HLR GT, status=ativo, contadores de mensagens em 0
6. **Enviar ACK**: Responder ao HLR com reconhecimento de alertServiceCenter

Tratamento de Assinante Ausente

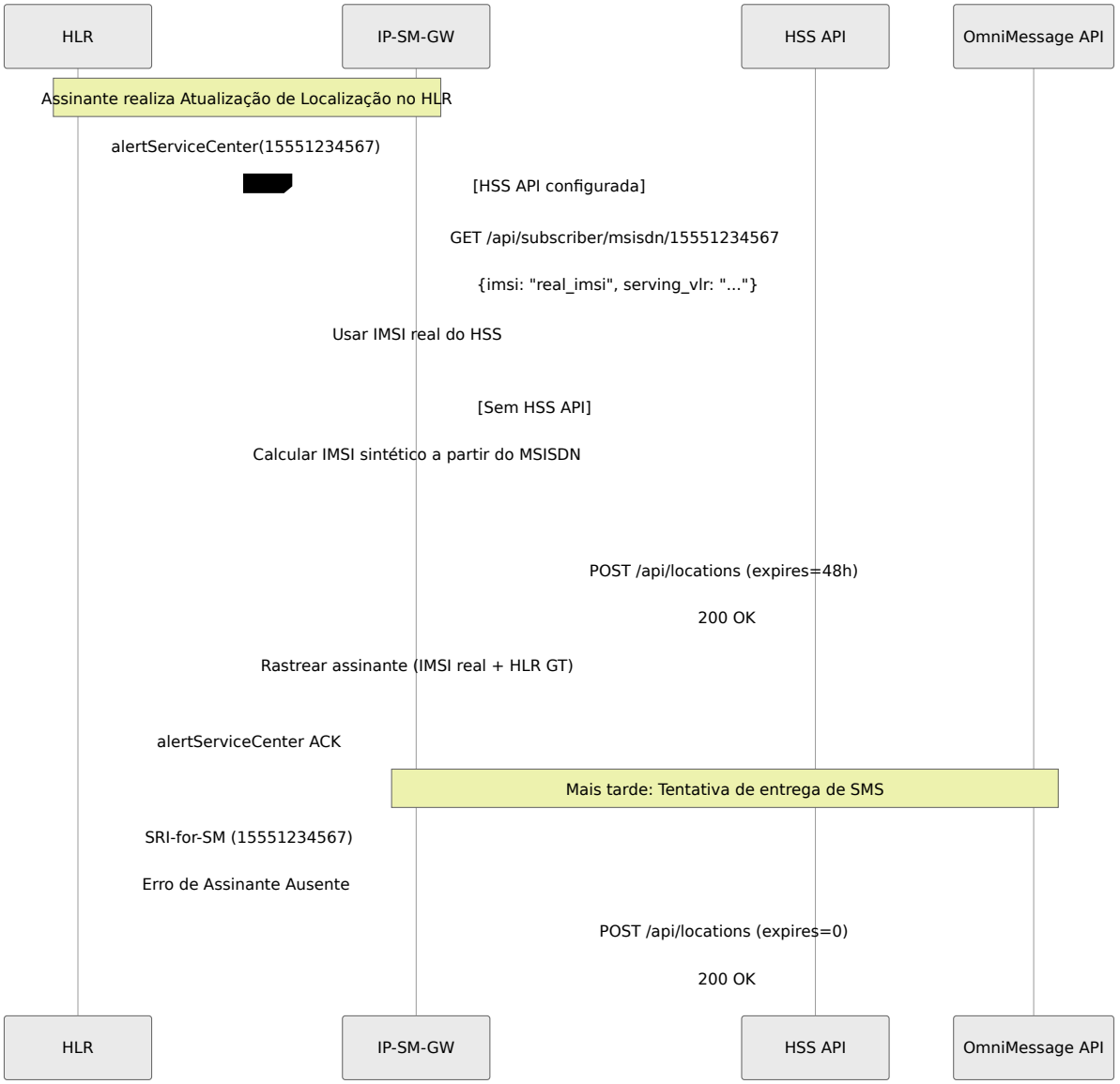
Quando o SMSc tenta entregar uma mensagem e recebe um erro de "assinante ausente" durante SRI-for-SM (para mais informações sobre SRI-for-SM, veja SRI-for-SM no Guia HLR):

1. **Detectar ausência**: SRI-for-SM retorna erro `absentSubscriberDiagnosticSM`

2. **Expirar localização**: POST para /api/locations com `expires=0` para marcar o assinante como inacessível
3. **Agente do usuário**: Definido como "SS7_AbsentSubscriber" para identificar a origem
4. **Atualizar rastreador**: Marcar o assinante como `failed` no Rastreador de Assinantes do SMSc

Isso garante que o banco de dados de localização e o rastreador reflitam com precisão o status de acessibilidade do assinante.

Diagrama de Fluxo



Endpoint da API

POST /api/locations

```
{
  "msisdn": "15551234567",
  "imsi": "001010123456789",
  "location": "ipsmgw",
  "ims_capable": false,
  "csfb": true,
  "expires": "2025-11-01T12:00:00Z",
  "user_agent": "15551111111",
  "ran_location": "SS7",
  "imei": "",
  "registered": "2025-10-30T12:00:00Z"
}
```

Nota: O campo `user_agent` contém o HLR GT que enviou o alertServiceCenter, permitindo que o SMSc rastreie qual HLR está fornecendo atualizações de localização.

Para assinantes ausentes, `expires` é definido para o horário atual (expiração imediata).

Prevenção de Loop

O SMSc implementa **prevenção automática de loop** para evitar loops infinitos de roteamento de mensagens quando as mensagens se originam de redes SS7, enquanto ainda permite a entrega legítima de MO→MT entre assinantes na mesma rede.

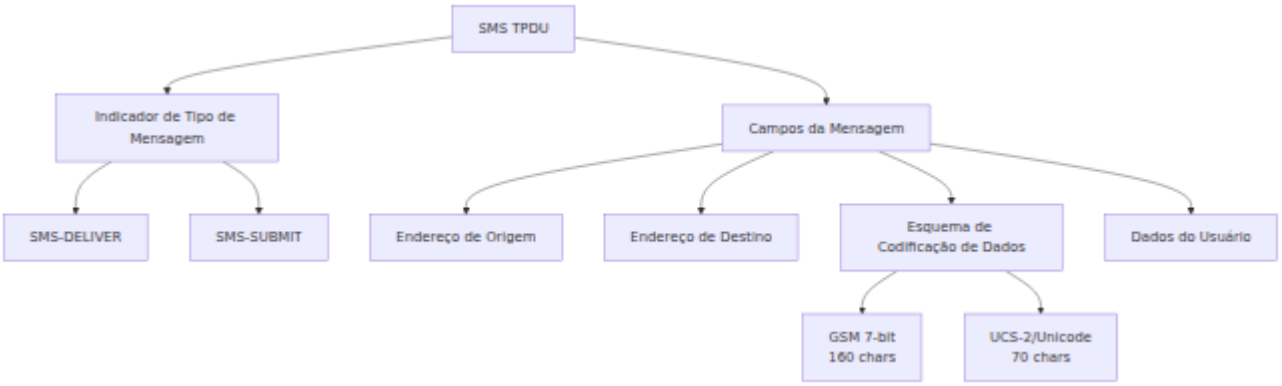
Por que a Prevenção de Loop é Importante

Quando o SMSc recebe mensagens SMS originadas de dispositivos móveis (MO) da rede SS7, ele as insere na fila de mensagens com um campo `source_smsc` identificando sua origem (por exemplo, `"SS7_MO_15551234567"`). Sem a prevenção de loop, essas mensagens poderiam ser:

1. Recebidas da rede SS7 → Enfileiradas com `source_smsc` contendo "SS7"
2. Recuperadas da fila → Processadas para entrega
3. Enviadas de volta para a rede SS7 → Criando um loop

Como Funciona

A lógica de prevenção de loop distingue entre destinos **na rede** e **fora da rede**. O SMSc bloqueia mensagens da SS7 apenas quando o assinante de destino **não está na rede**. Isso evita loops de roteamento de trânsito. Ele ainda permite a entrega legítima de MO→MT entre assinantes na mesma rede (por exemplo, MSC para MSC via o IP-SM-GW).



Detecção na Rede

Quando uma mensagem se origina da SS7, o sistema verifica se o assinante de destino está na rede antes de aplicar a prevenção de loop:

1. **Cache VLR** (mais rápido): Verificar se o assinante tem um endereço VLR em cache com IMSI válido no Rastreador de Assinantes
2. **API HSS** (autoritativa): Se `hlr_api_base_url` estiver configurado e o cache falhar, consultar a API HSS pelo MSISDN. Se o assinante existir no HSS, ele está na rede.

Se nenhuma das verificações encontrar o assinante, o destino é considerado **fora da rede** e a mensagem é bloqueada.

Implementação

Ao processar mensagens da fila, o SMSc verifica o campo `source_smsc`:

- **Se `source_smsc` contém "SS7" e o destino está fora da rede**:
  - Mensagem é ignorada
  - Evento adicionado: `"Prevenção de Loop"` com descrição indicando que o destino está fora da rede
  - Mensagem marcada como falhada via solicitação PUT
  - Registrada com nível de aviso
- **Se `source_smsc` contém "SS7" e o destino está na rede**:
  - Mensagem processada normalmente. Este é um fluxo legítimo de MO→MT.
  - A entrega prossegue via a API HSS / caminho de cache VLR
- **Se `source_smsc` não contém "SS7"**:
  - Mensagem processada normalmente (por exemplo, mensagens da API web, SMPP ou outras fontes não-SS7)

Valores de SMSC de Origem

Mensagens podem ter vários valores de `source_smsc`:

Fonte	Valor de Exemplo	Ação
Rede SS7 (MO-FSM), destino fora da rede	`"SS7_MO_15551234567"`	**Ignorado** - Prevenção de loop
Rede SS7 (MO-FSM), destino na rede	`"SS7_MO_15551234567"`	**Permitido** - Entrega MO→MT na rede
API Externa/SMPP	`"ipsmgw"` ou `"api_gateway"`	Processado normalmente
Outro SMSc	`"smsc-node-01"`	Processado normalmente

Rastreamento de Eventos

Quando uma mensagem é ignorada devido à prevenção de loop, um evento é registrado:

```
{
  "message_id": 12345,
  "name": "Prevenção de Loop",
  "description": "Mensagem ignorada - source_smsc
'SS7_MO_15551234567' contém 'SS7' e o destino está fora da rede,
prevenindo loop de mensagem"
}
```

Este evento é visível em:

- **Interface Web**: Página de Eventos SS7 (`/events`)
- **Banco de Dados**: Tabela `events` via API
- **Logs**: Entradas de log de nível de aviso

Configuração

A prevenção de loop está **sempre habilitada** e não pode ser desativada. Esta é uma característica crítica de segurança para evitar interrupções na rede devido a loops de mensagens. O bypass na rede requer tanto `smsc_mode_enabled: true` quanto o cache VLR ou `hlr_api_base_url` configurados.

Cenários de Exemplo

Cenário 1: MO-SMS de SS7 para assinante fora da rede (bloqueado)

```
1. Telefone móvel → MSC/VLR → IP-SM-GW (via MO-ForwardSM)
2. IP-SM-GW recebe MO-FSM, insere na fila: source_smsc =
"SS7_MO_9991900"
3. Auto-flush recupera mensagem da fila
4. Destino não encontrado no cache VLR ou HSS → fora da rede
5. IP-SM-GW detecta "SS7" em source_smsc + destino fora da rede →
IGNORAR
6. Evento registrado: "Prevenção de Loop"
7. Nenhum SRI-for-SM ou MT-ForwardSM enviado (loop prevenido)
```

Cenário 2: MO-SMS de SS7 para assinante na rede (permitido)

```
1. Telefone móvel → MSC/VLR → IP-SM-GW (via MO-ForwardSM)
2. IP-SM-GW recebe MO-FSM, insere na fila: source_smsc =
"SS7_MO_9991900"
3. Auto-flush recupera mensagem da fila
4. Destino encontrado na API HSS → assinante na rede
5. IP-SM-GW permite entrega apesar da origem SS7
6. HSS retorna IMSI real + VLR → MT-ForwardSM enviado diretamente
para VLR
```

Rastreamento de Assinantes do SMSc

O SMSc inclui um **Rastreador de Assinantes** GenServer que mantém o estado em tempo real para assinantes com base em mensagens alertServiceCenter e tentativas de entrega de mensagens.

Propósito

O rastreador fornece:

- **Monitoramento de acessibilidade**: Quais assinantes estão atualmente acessíveis

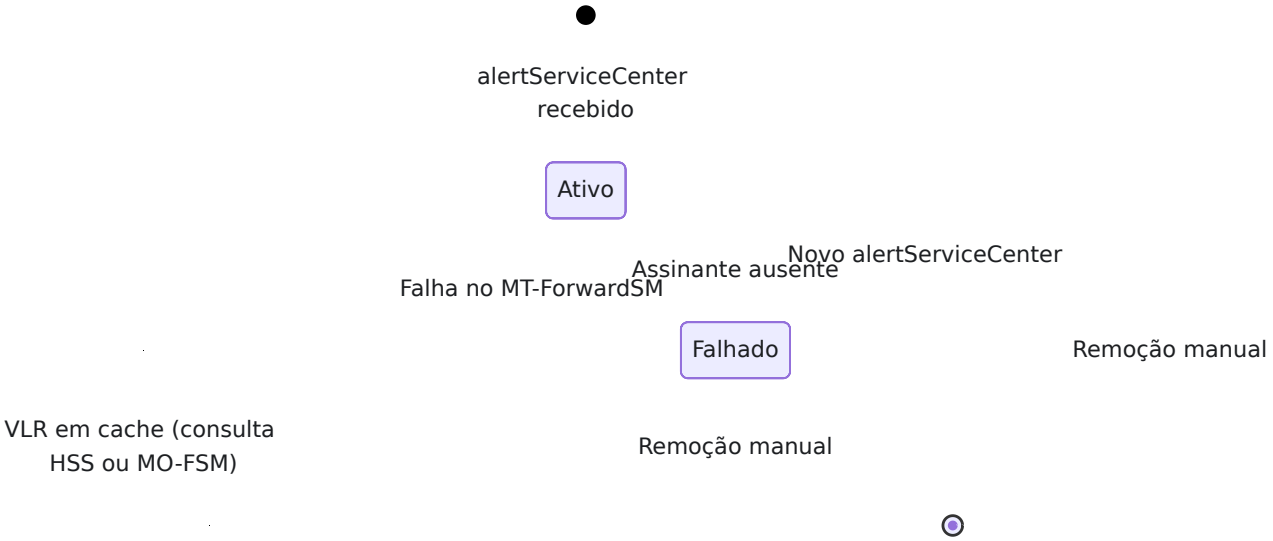
- **Rastreamento de HLR**: Qual HLR enviou o alertServiceCenter para cada assinante
- **Contadores de mensagens**: Número de mensagens enviadas/recebidas por assinante
- **Rastreamento de falhas**: Marcar assinantes como falhados quando tentativas de entrega falham
- **Visibilidade na Interface Web**: Painel em tempo real mostrando todos os assinantes rastreados

Informações Rastreada

Para cada assinante, o rastreador armazena:

Campo	Descrição	Exemplo
`msisdn`	Número de telefone do assinante (chave)	"15551234567"
`imsi`	IMSI do assinante (real do HSS se disponível)	"001010123456789"
`hlr_gt`	HLR GT que enviou alertServiceCenter	"15551111111"
`messages_sent`	Contagem de mensagens MT-FSM enviadas	5
`messages_received`	Contagem de mensagens MO-FSM recebidas	2
`status`	`:active` ou `:failed`	`:active`
`updated_at`	Timestamp Unix da última atualização	1730246400
`vlr_address`	Endereço VLR GT em cache	"14155550100"
`vlr_cached_at`	Timestamp Unix quando o VLR foi cacheado pela última vez	1730246400

Transições de Estado



Comportamento

Quando alertServiceCenter é recebido:

- Criar ou atualizar entrada do assinante
- Definir `status = :active`
- Registrar HLR GT
- Resolver IMSI real da API HSS (se configurada), caso contrário, usar IMSI sintético
- Preservar cache VLR existente e contadores de mensagens

Quando a API HSS retorna assinante com VLR:

- Cache o endereço VLR e IMSI real no rastreador
- Entregas subsequentes usam o VLR em cache sem reconsultar o HSS

Quando MO-ForwardSM é recebido:

- Cache o GT chamador SCCP como o endereço VLR do assinante (o GT chamador em um MO-ForwardSM é o VLR que retransmitiu a mensagem)
- Cria uma entrada no rastreador se uma não existir

Quando a entrega do MT-ForwardSM falha:

- Limpar o endereço VLR em cache para que a próxima tentativa de entrega reconsulte a API HSS

Quando SRI-for-SM é bem-sucedido:

- Incrementar o contador `messages_sent`
- Atualizar o timestamp `updated_at`

Quando SRI-for-SM falha:

- Definir `status = :failed`
- Manter no rastreador para monitoramento

Quando o assinante é removido:

- Excluir da tabela ETS
- Não aparece mais na Interface Web

Interface Web - Página de Assinantes do SMSc

Caminho: `/smsc_subscribers` **Atualização automática**: A cada 2 segundos

Nota: Esta página está disponível apenas quando executada no modo SMSc. Após habilitar a configuração do SMSc em `config/config.exs`, você deve reiniciar a aplicação para que a rota se torne disponível.

A página **Assinantes do SMSc** fornece monitoramento em tempo real de todos os assinantes rastreados:

Recursos

1. **Tabela de Assinantes**
    - MSISDN, IMSI, HLR GT, VLR GT (endereço VLR em cache)
    - Contadores de mensagens enviadas/recebidas
    - Badge de status (Ativo/Falhado) com codificação de cores
    - Timestamp da última atualização e duração
    - Botão de remoção para assinantes individuais

2. **Estatísticas Resumidas**
    - Total de assinantes rastreados
    - Contagem de assinantes ativos
    - Contagem de assinantes falhados
    - Número de HLRs únicos
3. **Ações**
    - Limpar Tudo: Remover todos os assinantes rastreados
    - Remover: Remover assinante individual

Exemplo de Visualização

```
┌──────────────────────────────────────────────────────────────────────────────
│ Assinantes Rastreados do SMSc                                    Total:
├──────────────────────────────────────────────────────────────────────────────
│ MSISDN          IMSI              HLR GT         VLR GT         Msg
│                                                                  S/
├──────────────────────────────────────────────────────────────────────────────
│ 15551234567     001010123456789   15551111111    14155550100    5/2
│ 15559876543     001010987654321   15551111111    -              0/0
│ 15551112222     001010111222233   15552222222    14155550200    3/1
└──────────────────────────────────────────────────────────────────────────────

Resumo:  Total: 3  |  Ativo: 2  |  Falhado: 1  |  HLRs Únicos: 2
```

Funções da API

O rastreador expõe estas funções para acesso programático:

```
# Chamado quando alertServiceCenter é recebido
SMSc.SubscriberTracker.alert_received(msisdn, imsi, hlr_gt)

# Incrementar contadores de mensagens
SMSc.SubscriberTracker.message_sent(msisdn)
SMSc.SubscriberTracker.message_received(msisdn)

# Marcar como falhado (falha no SRI-for-SM)
SMSc.SubscriberTracker.mark_failed(msisdn)

# Remover do rastreamento
SMSc.SubscriberTracker.remove_subscriber(msisdn)

# Funções de consulta
SMSc.SubscriberTracker.get_active_subscribers()
SMSc.SubscriberTracker.get_subscriber(msisdn)
SMSc.SubscriberTracker.count_subscribers()
SMSc.SubscriberTracker.clear_all()
```

Integração

O rastreador é integrado automaticamente com:

- **Manipulador de alertServiceCenter**: Chama `alert_received/3` na atualização de localização bem-sucedida
- **Manipulador de SRI-for-SM**: Incrementa `messages_sent` na roteamento bem-sucedido
- **Manipulador de assinante ausente**: Chama `mark_failed/1` quando o assinante está ausente
- **Erros de assinante desconhecido**: Chama `mark_failed/1` quando SRI-for-SM falha

---

Cache de Endereço VLR

Quando opera como um IP-SM-GW com uma API HSS configurada, o Rastreador de Assinantes cacheia endereços VLR para evitar consultar o HSS em cada tentativa de entrega MT.

Como Funciona

O cache VLR é populado a partir de duas fontes:

1. **Consulta à API HSS**: Quando o pipeline de entrega consulta a API HSS e obtém um assinante com um VLR em serviço, o IMSI e o endereço VLR são armazenados em cache no Rastreador de Assinantes.
2. **MO-ForwardSM**: Quando um MO-ForwardSM chega de um assinante, o GT da parte chamadora SCCP (que é o VLR/MSC que retransmitiu a mensagem) é armazenado como o endereço VLR do assinante.

Em tentativas subsequentes de entrega MT, o cache é verificado primeiro. Se uma entrada válida existir (tanto IMSI quanto VLR presentes, dentro do TTL), a API HSS é completamente ignorada.

Invalidação do Cache

O cache VLR é limpo quando:

- **A entrega do MT-ForwardSM falha** (timeout, erro SCCP, rejeição do VLR): o VLR em cache pode estar desatualizado
- **O TTL expira**: o TTL é configurável via `vlr_cache_ttl_seconds`. O cache trata entradas mais antigas que o TTL como falhas de cache.

O cache **não** é limpo em alertServiceCenter. A entrada VLR existente é preservada porque o assinante ainda pode estar acessível no mesmo VLR.

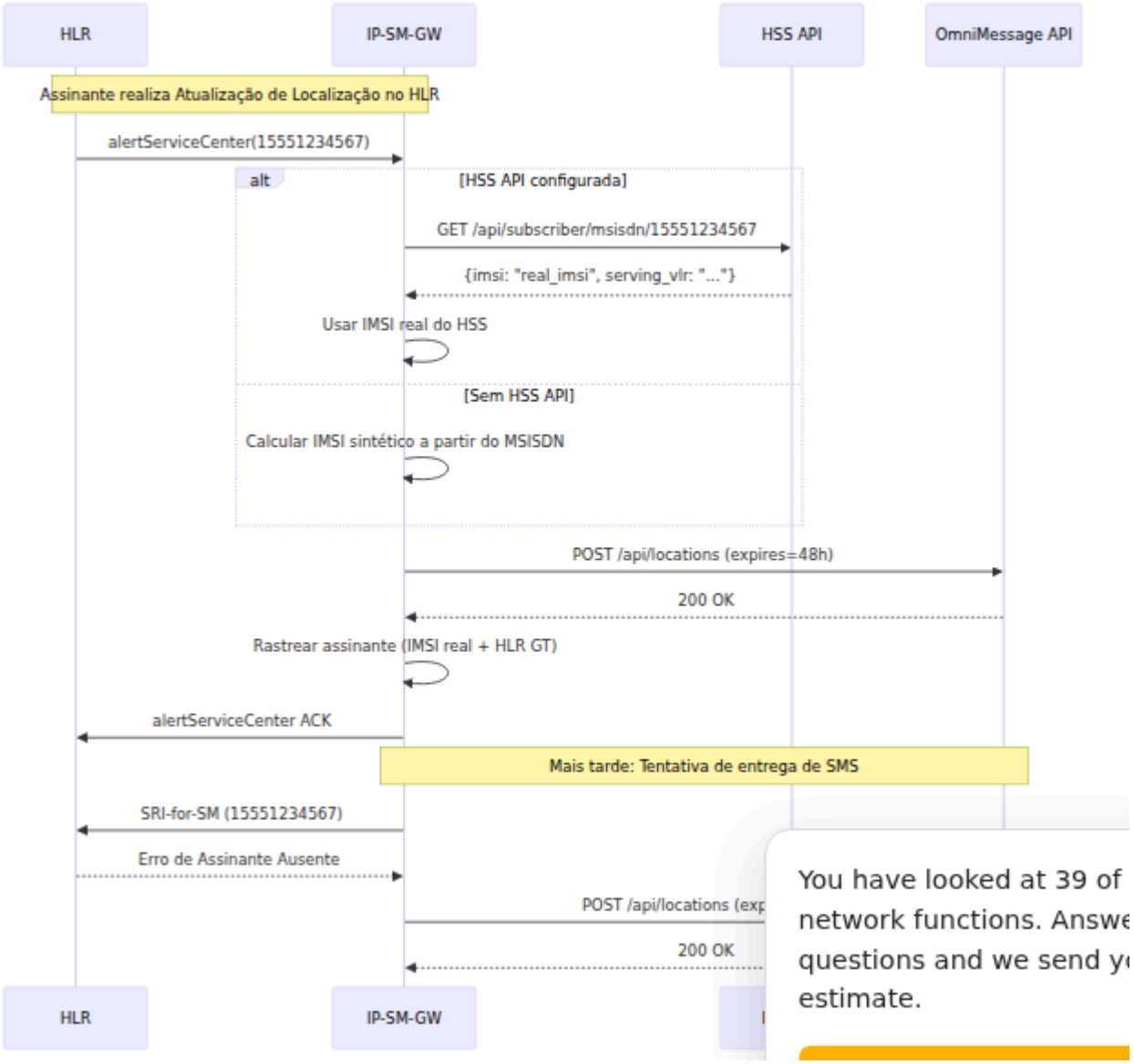
Configuração

```
config :omniss7,
  # Endpoint da API HSS/HLR (necessário para cache de VLR e
entrega na rede)
  hlr_api_base_url: "https://omnihss.example.com:8443",
  # Chave da API OmniHSS (necessária quando hlr_api_base_url está
configurado)
  hlr_api_key: "<Chave da API OmniHSS com o papel
hlr_interworking>",

  # TTL do cache VLR em segundos (padrão: 3600 = 1 hora)
  vlr_cache_ttl_seconds: 3600
```

Parâmetro	Tipo	Necessário	Padrão	Des
`hlr_api_base_url`	String	Não	`nil`	URL bas HSS/HLR configur habilita de assin rede, ca e entreg direta. Q `nil`, tod entregas fluxo SR
`hlr_api_key`	String	Quando `hlr_api_base_url` está configurado	`nil`	Chave d OmniHS OmniSS no cabe `omnihss` cada sol API HSS. a chave OmniHS ao papel `hlr_int` Se `hlr_api` estiver c e a chav ausente cada sol API HSS um erro nomeia `hlr_api`

Parâmetro	Tipo	Necessário	Padrão	Des
`vlr_cache_ttl_seconds`	Inteiro	Não	`3600`	Idade m segundo uma ent em cach consider Após ess a próxim tentativa entrega a API HS mais bai aumenta tráfego mas mel precisão assinant

Diagrama de Sequência



Fila de SMS Auto-Flush

O serviço **Auto-Flush** processa automaticamente mensagens SMS pendentes.

Para referência de parâmetros de configuração, veja Configuração do Auto-Flush no Referência de Configuração.

Configuração

```
config :omniss7,
  auto_flush_enabled: true,           # Habilitar/desabilitar
auto-flush
  auto_flush_interval: 10_000,        # Intervalo de polling em
milissegundos
  auto_flush_dest_smsc: nil,          # Filtro: nil = todos
  auto_flush_tps: 10                  # Máximo de transações por
segundo
```

Como Funciona

1. **Polling**: A cada `auto_flush_interval` milissegundos, consulta a API por mensagens pendentes
2. **Filtragem**: Opcionalmente filtrar por `auto_flush_dest_smsc`
3. **Limitação de Taxa**: Processar até `auto_flush_tps` mensagens por ciclo
4. **Entrega**: Para cada mensagem:
   - Enviar **SRI-for-SM** (Enviar Informações de Roteamento para Mensagem Curta) para o HLR para obter informações de roteamento
     - O HLR retorna um IMSI sintético calculado a partir do MSISDN
     - O HLR retorna o endereço GT do SMSC onde o MT-ForwardSM deve ser enviado
     - Veja Detalhes do SRI-for-SM no Guia HLR para documentação completa
   - Em caso de sucesso, enviar **MT-forwardSM** para MSC/VLR
   - Atualizar o status da mensagem via API (entregue/falhado)
   - Adicionar rastreamento de eventos via API

> **Mergulho Técnico**: Para uma explicação completa de como o SRI-for-SM funciona, incluindo mapeamento de MSISDN para IMSI, configuração do endereço GT do centro de serviços e geração de IMSI sintético que preserva a privacidade, consulte a seção SRI-for-SM no Guia de Configuração do HLR.

Métricas do SMSc

OmniSS7 expõe métricas do Prometheus. As métricas MAP compartilhadas rastreiam a atividade de SMS. Não há contadores de fila/entrega específicos do SMSc. As métricas relevantes para operações de SMS são:

Métrica	Tipo	Rótulos	Relevâ S
`map_requests_total`	Contador	`operation`	Conta op MAP. As relevant SMS incl `sri_for` `forward` `MTforwa` `prn`.
`map_request_duration_milliseconds`	Histograma	`operation`	Duração de solicitaç MAP, inc for-SM e ForwardS

O conjunto completo de métricas declaradas (nem todas específicas de SMS) é: `map_requests_total`, `cap_requests_total`, `ussd_requests_total`, `ussd_active_sessions`, `map_pending_requests`, `omniss7_license_status` e `map_request_duration_milliseconds`.

Exemplos de Consultas:

```
# Taxa de operações MT-ForwardSM nos últimos 5 minutos
rate(map_requests_total{operation="MTforward_sm"}[5m])

# Taxa de operações SRI-for-SM nos últimos 5 minutos
rate(map_requests_total{operation="sri_for_sm"}[5m])

# Duração média do MT-ForwardSM (últimos 5 minutos)
rate(map_request_duration_milliseconds_sum{operation="MTforward_sm"}
[5m]) /
rate(map_request_duration_milliseconds_count{operation="MTforward_sm"
[5m])
```

Solução de Problemas do SMSc

Problema: Mensagens Não Entregues

Verificações:

1. Verifique se o auto-flush está habilitado
2. Verifique a conexão com o banco de dados
3. Monitore os logs em busca de erros
4. Verifique se a conexão M3UA está ATIVA
5. Verifique os limites de TPS

Problema: Alta Profundidade de Fila

Causas Possíveis:

- Limite de TPS muito baixo
- Problemas de tempo limite do HLR
- Problemas de conectividade de rede
- Números de destino inválidos

Soluções:

- Aumente `auto_flush_tps`
- Verifique a disponibilidade do HLR
- Revise os logs de mensagens falhadas

API MT-forwardSM

Enviar SMS via API

Endpoint da API: `POST /api/MT-forwardSM`

Todos os quatro campos são obrigatórios. Os nomes dos campos estão em snake_case (note que `destination_service_centre` usa a grafia britânica). `smsPDU` é o SMS TPDU codificado em hex.

Requisição:

```
{
  "imsi": "234509876543210",
  "destination_service_centre": "447999555111",
  "originating_service_center": "447999123456",
  "smsPDU":
"040B917477218345F60000157030185714 0C0BD4F29C0E9281C4E1F11A"
}
```

Resposta:

Em caso de sucesso (HTTP 200), a resposta contém uma única chave `result` que contém o resultado TCAP formatado do diálogo MT-ForwardSM (não há `message_id`, e o valor não é a string literal `"success"`):

```
{
  "result": "<formatted TCAP result>"
}
```

Se o diálogo MAP expirar, o endpoint retorna HTTP 504 com `{"error": "timeout"}`. Uma requisição malformada/faltando campos retorna HTTP 400 com `{"error": "invalid request"}`.

Documentação Relacionada

Documentação do OmniSS7:

- ← Voltar para a Documentação Principal
- Guia de Configuração do HLR - Configuração e operações do modo HLR
  - **Detalhes Técnicos do SRI-for-SM** - Documentação completa sobre mapeamento de MSISDN para IMSI e configuração do centro de serviços
- Guia de Recursos Comuns - UI da Web, API, Monitoramento
- Guia do Cliente MAP - Operações MAP
- Referência Técnica - Especificações de protocolo

Documentação do OmniMessage: Para configuração de roteamento de mensagens, gerenciamento de filas, rastreamento de entrega, limitação de taxa e análises, consulte a **documentação do produto OmniMessage**. O OmniMessage contém toda a lógica de roteamento de mensagens, algoritmos de retry de fila, manipulação de relatórios de entrega e motor de regras de negócios.

Guia de Configuração do M3UA & M2PA STP

← Voltar para a Documentação Principal

Este guia fornece configuração detalhada para usar o OmniSS7 como um **Ponto de Transferência de Sinalização (STP)**.

Índice

1. O que é um STP?
2. Funções de Rede do STP
3. Interfacing com Redes TDM
4. Habilitando o Modo STP
5. Configurando Pares
6. Suporte ao Protocolo M2PA
    - M3UA vs M2PA
    - Configurando Pares M2PA
    - Configuração SLTM
    - Gerenciando M2PA via Interface Web
    - Métricas M2PA
7. Roteamento de Código de Ponto
8. Roteamento de Título Global
9. Tratamento de Trânsito SCCP
    - Roteamento em SSN Após GTT
    - Aplicação de Contador de Saltos
    - Retorno de Mensagem em Caso de Falha
10. Estado de Destino (Gerenciamento de Roteamento de Sinalização)
11. Filtragem de Gateway (Firewall SCCP)
12. Recursos de Gerenciamento de Roteamento
    - Desabilitando Rotas

    - ROTAS DROP - Prevenindo Loops de Roteamento
13. Roteamento Avançado
14. Testando Configuração
15. Métricas e Monitoramento
16. Monitoramento de Pares M3UA
    - Tratamento de Contexto de Roteamento M3UA
    - Desambiguação de Múltiplos Pares com o Mesmo IP

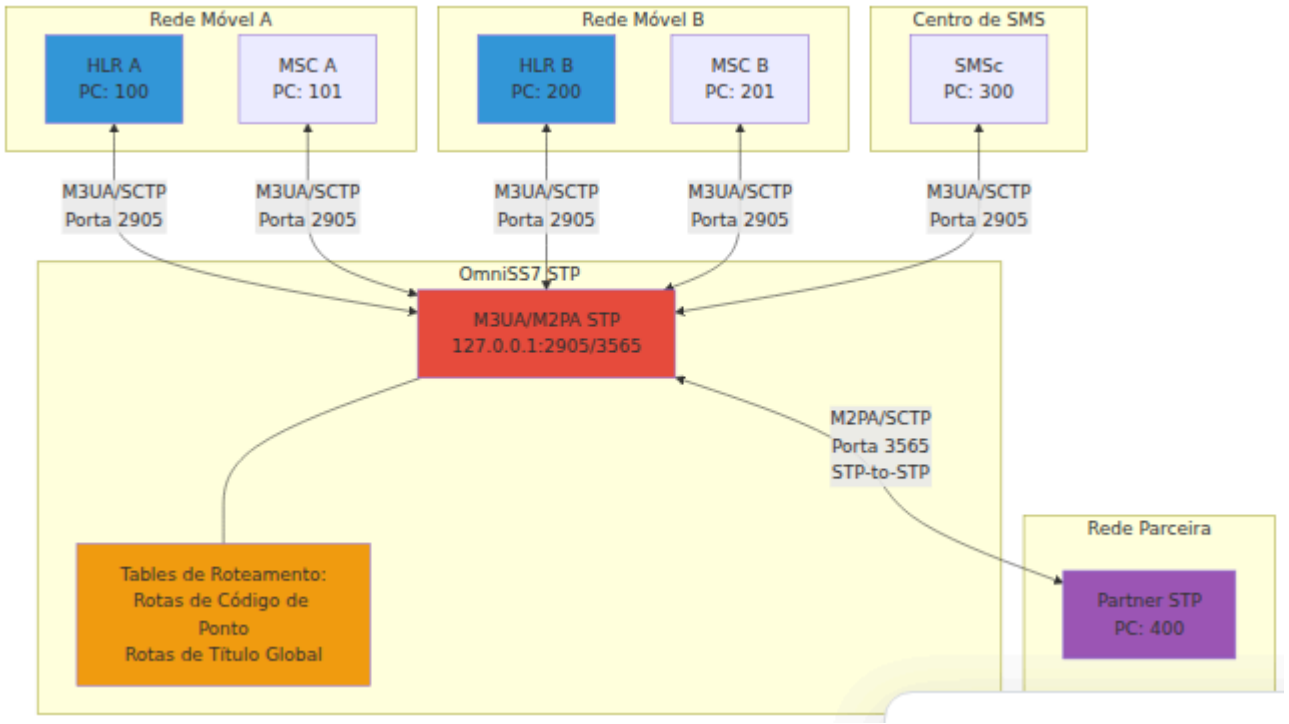
O que é um Ponto de Transferência de Sinalização (STP)?

Um **Ponto de Transferência de Sinalização (STP)** é um elemento crítico da rede em redes de sinalização SS7 e baseadas em IP que roteia mensagens de sinalização entre nós da rede.

Funções do STP

- **Roteamento de Mensagens**: Roteia o tráfego de sinalização SS7 com base no Código de Ponto (PC) de destino ou Título Global (GT)
- **Tradução de Protocolo**: Conecta redes SS7 tradicionais com redes M3UA/SCTP baseadas em IP
- **Distribuição de Carga**: Distribui o tráfego entre múltiplos destinos usando roteamento baseado em prioridade
- **Gateway de Rede**: Conecta diferentes redes de sinalização e provedores de serviços
- **Ocultação de Topologia**: Pode reescrever endereços para ocultar a topologia interna da rede

Diagrama de Rede STP



Funções de Rede do STP Explicadas

ASP (Processo de Servidor de Aplicação)

- **Função**: Cliente conectando-se a um SGP/STP remoto
- **Direção**: Conexão de saída
- **Caso de Uso**: Seu STP conecta-se ao STP de uma rede parceira

SGP (Processo de Gateway de Sinalização)

- **Função**: Servidor aceitando conexões de ASPs
- **Direção**: Conexão de entrada
- **Caso de Uso**: Redes parceiras conectam-se ao seu STP

AS (Servidor de Aplicação)

- **Definição**: Agrupamento lógico de um ou mais ASPs
- **Propósito**: Fornece redundância e compartilhamento de carga
- **Caso de Uso**: Múltiplos ASPs atendendo ao mesmo destino

Interfacing com Redes TDM via Gateways de Sinalização

OmniSS7 é uma plataforma de sinalização baseada em IP que utiliza protocolos SIGTRAN (M3UA/M2PA sobre SCTP). Para trocar sinalização com redes SS7 **baseadas em TDM** legadas, você precisa de um **Gateway de Sinalização (SGW)** para conectar os dois mundos.

O que é um Gateway de Sinalização?

Um **Gateway de Sinalização (SGW)** é um elemento de rede que converte sinalização SS7 entre:

- **Lado TDM**: SS7 tradicional usando MTP1/MTP2/MTP3 sobre links E1/T1
- **Lado IP**: SIGTRAN usando M3UA ou M2PA sobre SCTP/IP

O SGW atua como um tradutor de protocolo. Ele permite que aplicações baseadas em IP, como o OmniSS7, se comuniquem com equipamentos legados de TDM (switches, STPs, HLRs) que não suportam sinalização IP.

Arquitetura TDM para IP



Comparação da Pilha de Protocolos

Camada	TDM SS7	Gateway de Sinalização	IP (SIGTRAN)
Parte do Usuário	SCCP/TCAP/MAP	← Transparente →	SCCP/TCAP/MAP
Rede	MTP3	Conversão	M3UA/M2PA
Link	MTP2	Terminação	SCTP
Física	MTP1 (E1/T1)	Terminação	IP/Ethernet

O SGW termina MTP1/MTP2 no lado TDM e apresenta os dados do usuário MTP3 sobre M3UA ou M2PA no lado IP. As camadas superiores (SCCP, TCAP, MAP, CAP) passam de forma transparente.

Conectando o OmniSS7 a um Gateway de Sinalização

OmniSS7 conecta-se a um Gateway de Sinalização como um **ASP (Processo de Servidor de Aplicação)**, enquanto o SGW atua como um **SGP (Processo de Gateway de Sinalização)**.

Exemplo de Configuração

```
config :omniss7,
  # Conectar ao Gateway de Sinalização como ASP
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :sgw_connection,
    # Endpoint local (OmniSS7)
    local_ip: {10, 0, 0, 1},
    local_port: 0,                    # Porta local dinâmica
    # Endpoint do Gateway de Sinalização
    remote_ip: {10, 0, 0, 100},       # Endereço IP do SGW
    remote_port: 2905,                # Porta M3UA do SGW
    routing_context: 1                # Conforme atribuído pelo
SGW
  },

  # Ou configure como um par para roteamento STP
  peers: [
    %{
      peer_id: 1,
      name: "TDM_Gateway",
      role: :client,                  # OmniSS7 inicia a conexão
      local_ip: {10, 0, 0, 1},
      local_port: 0,
      remote_ip: {10, 0, 0, 100},     # Endereço IP do SGW
      remote_port: 2905,
      routing_context: 1,
      point_code: 100,                # Faixa de código de ponto
atrás do SGW
      network_indicator: :international
    }
  ],

  # Roteie códigos de ponto da rede TDM através do gateway
  m3ua_routes: [
    %{
      dest_pc: 100,                   # Código de ponto do MSC TDM
      peer_id: 1,                     # Roteie via par SGW
      priority: 1,
      network_indicator: :international
    },
    %{
      dest_pc: 200,                   # Código de ponto do HLR TDM
```

```
      peer_id: 1,
      priority: 1,
      network_indicator: :international
    }
  ]
```

Considerações de Configuração do SGW

Ao configurar seu Gateway de Sinalização para trabalhar com o OmniSS7:

Parâmetro	Descrição	Valor Típico
IP Remoto	Endereço IP do OmniSS7	IP do seu servidor
Porta Remota	Porta SCTP do OmniSS7	2905
Contexto de Roteamento	Identificador AS no SGW	Atribuído pelo administrador do SGW
Códigos de Ponto	Códigos de ponto da rede TDM acessíveis via SGW	Específico da rede
Modo de Tráfego	Modo de manuseio de tráfego ASP	Substituir ou Compartilhar carga

Cenários de Implantação

Cenário 1: Aplicação IP Acessando Rede TDM

OmniSS7 envia consultas MAP (SRI-SM, PRN) para HLRs baseados em TDM via SGW:

```
OmniSS7 (ASP) → SGW (SGP) → Rede TDM → HLR
     M3UA          MTP2           MTP3
```

Cenário 2: OmniSS7 como STP Entre IP e TDM

OmniSS7 roteia tráfego entre elementos de rede baseados em IP e redes TDM:

```
SMSC IP → OmniSS7 STP → SGW → HLR TDM
                ↓
          HLR baseado em IP
```

Cenário 3: SGW Duplo para Redundância

Conecte-se a dois Gateways de Sinalização para alta disponibilidade:

```
peers: [
  %{
    peer_id: 1,
    name: "SGW_Primary",
    role: :client,
    remote_ip: {10, 0, 0, 100},
    remote_port: 2905,
    point_code: 100,
    # ... outras configurações
  },
  %{
    peer_id: 2,
    name: "SGW_Backup",
    role: :client,
    remote_ip: {10, 0, 0, 101},
    remote_port: 2905,
    point_code: 100,
    # ... outras configurações
  }
],

m3ua_routes: [
  # Rota primária via SGW_Primary
  %{dest_pc: 100, peer_id: 1, priority: 1, network_indicator:
:international},
  # Rota de backup via SGW_Backup
  %{dest_pc: 100, peer_id: 2, priority: 2, network_indicator:
:international}
]
```

Habilitando o Modo STP M3UA

OmniSS7 pode operar em diferentes modos. Para usá-lo como um STP, você fornece configuração de roteamento STP sob a chave de aplicação `:omniss7`.

Onde a Configuração Está

Arquivo	Propósito
`config/config.exs`	Configuração padrão, sempre carregada (interface web, ouvinte de API na porta 8445, registro)
`config/stp_example_runtime.exs`	Um **exemplo de STP** completo e pronto para edição (`sctp_handler`, pares, rotas de PC, rotas de GT)
`config/runtime.exs`	Sobrescritas de tempo de execução por implantação. Neste repositório, é um stub mínimo; servidores de produção implantam seus próprios

> **Nota**: Lançamentos anteriores enviaram três blocos de modo comentados dentro de `config/runtime.exs`. Esse layout não existe mais. O exemplo de STP agora está em seu próprio arquivo, `config/stp_example_runtime.exs`.

Mudando para o Modo STP

Para configurar um STP, copie a configuração de exemplo para a configuração que sua implantação carrega em tempo de execução:

1. **Abra** `config/stp_example_runtime.exs` e revise o exemplo de STP (manipulador SCTP, pares, rotas de PC, rotas de GT).
2. **Copie** o bloco `config :omniss7, ...` para o `config/runtime.exs` da sua implantação (ou mescle as chaves relevantes).

3. **Personalize** os endereços IP, portas, códigos de ponto, pares e rotas para sua rede.
4. **Reinicie** a aplicação: `iex -S mix`

Configuração do Modo STP

O exemplo de STP em `config/stp_example_runtime.exs` configura um ouvinte STP de entrada (`sctp_handler`), habilita o roteamento de GT e define pares, além de rotas de PC e GT:

```
config :omniss7,
  # Ouvinte SCTP (servidor STP de entrada) - veja o Modo STP
Autônomo abaixo
  sctp_handler: %{
    enabled: true,
    local_ip: {127, 0, 0, 1},
    local_port: 2905,
    point_code: 101            # Código de ponto deste STP
  },

  # Habilitar roteamento de Título Global (além do roteamento de
PC)
  enable_gt_routing: true,

  # Pares M3UA, rotas de PC e rotas de GT seguem
  m3ua_peers: [ %{peer_id: 1, name: "STP_West", role: :client,
...} ],
  m3ua_routes: [ %{dest_pc: 100, peer_id: 1, priority: 1,
network_indicator: :international} ],
  m3ua_gt_routes: [ %{gt_prefix: "44", peer_id: 2, priority: 1,
description: "Números do Reino Unido"} ]
```

Veja Configurando Pares M3UA, Roteamento de Código de Ponto e Roteamento de Título Global abaixo para o conjunto completo de parâmetros.

Parâmetros de Configuração a Personalizar

Para uma referência completa de todos os parâmetros de configuração, veja a Referência de Configuração.

Estas são as chaves STP de nível superior sob `config :omniss7`:

Parâmetro	Tipo	Padrão	Descrição	Exemplo
`sctp_handler`	Mapa	*Opcional*	Configuração do ouvinte STP de entrada (veja Modo STP Autônomo)	`%{enabled: true, local_port: 2905, point_code: 101}`
`enable_gt_routing`	Booleano	`false`	Habilitar roteamento de Título Global (além do roteamento de PC)	`true`
`m3ua_peers`	Lista	`[]`	Definições de pares M3UA / M2PA (veja Configurando Pares M3UA)	Nenhum
`m3ua_routes`	Lista	`[]`	Rotas de Código de Ponto (veja Roteamento de Código de Ponto)	Nenhum
`m3ua_gt_routes`	Lista	`[]`	Rotas de Título Global (veja Roteamento de Título Global)	Nenhum

> **Dica**: Use multihoming SCTP fornecendo uma lista de endereços IP para `local_ip` e/ou `remote_ip` para habilitar failover automático. Veja o Guia de Multihoming SCTP.

O que Acontece Quando o Modo STP é Habilitado

Uma vez que pares e rotas estão configurados, a interface web mostra:

- **Eventos SS7** - Registro de eventos
- **M3UA** - Status da conexão
- **Roteamento** - Gerenciamento da tabela de rotas ← *Específico do STP*
- **Teste de Roteamento** - Teste de rotas ← *Específico do STP*
- **Recursos** - Monitoramento do sistema
- **Configuração** - Visualizador de configuração

Notas Importantes

- O protocolo SCTP (protocolo IP 132) deve ser permitido através dos firewalls
- A porta M3UA padrão é 2905 (padrão da indústria)
- Certifique-se de que há recursos de sistema suficientes para lidar com o tráfego de roteamento
- **Persistência de Roteamento**: Todas as rotas configuradas via interface web ou API são armazenadas no **banco de dados Mnesia** e **sobrevivem a reinicializações**
- **Mesclagem de Configuração**: Rotas de `runtime.exs` são carregadas na inicialização e mescladas com rotas Mnesia
- Após alterar a configuração, você deve reiniciar a aplicação para que as alterações tenham efeito
- **Interface Web**: Veja o Guia da Interface Web para gerenciar rotas via interface web
- **Acesso à API**: Veja o Guia da API para documentação da API REST e acesso à interface Swagger

---

Modo STP Autônomo

Você pode executar um **servidor STP M3UA autônomo** que escuta por conexões de entrada. Este é o modo usado pelo exemplo em `config/stp_example_runtime.exs`.

Habilitando o STP Autônomo

Adicione esta configuração sob `config :omniss7` em sua configuração de tempo de execução:

```
config :omniss7,
  sctp_handler: %{
    enabled: true,
    local_ip: {127, 0, 0, 1},    # Endereço IP para escutar
    local_port: 2905,             # Porta para escutar
    point_code: 100               # Código de ponto deste STP
  }
```

Parâmetros de Configuração do STP

Parâmetro	Tipo	Padrão	Descrição	Exemplo
`enabled`	Booleano	`false`	Habilitar servidor STP autônomo	`true`
`local_ip`	Tupla	`{127, 0, 0, 1}`	Endereço IP para escutar conexões	`{0, 0, 0, 0}`
`local_port`	Inteiro	`2905`	Porta para escutar	`2905`
`point_code`	Inteiro	*Obrigatório*	Código de ponto SS7 deste STP	`100`

Quando Usar o STP Autônomo

- **Roteamento Puro**: Quando você só precisa de roteamento M3UA sem funcionalidade de cliente MAP
- **STP Central**: Para criar um roteador de sinalização central para múltiplos elementos de rede
- **Arquitetura Hub**: Conectar múltiplos HLRs, MSCs e SMSCs através de um STP central

Nota: Você pode habilitar tanto `map_client_m3ua` quanto `sctp_handler` simultaneamente se precisar de conexões de saída e funcionalidade de STP de entrada.

Persistência da Tabela de Roteamento (Mnesia)

Todas as tabelas de roteamento (pares, rotas de Código de Ponto e rotas de Título Global) são armazenadas em um **banco de dados Mnesia** para persistência.

Como o Roteamento Funciona

1. **Rotas de runtime.exs**: Rotas definidas em `config/runtime.exs` sob `peers` (ou legado `m3ua_peers`), `m3ua_routes`, e `m3ua_gt_routes` são carregadas na inicialização da aplicação
2. **Rotas da Interface Web**: Rotas adicionadas via a página de Roteamento da Interface Web são armazenadas no Mnesia
3. **Mesclagem de Rotas**: Na reinicialização, as rotas de runtime.exs são mescladas com as rotas existentes do Mnesia (sem duplicatas)
4. **Persistência**: Todas as rotas configuradas via Interface Web **sobrevivem a reinicializações da aplicação**

Tipo de Armazenamento do Mnesia

Controle como as tabelas de roteamento são armazenadas. Para mais detalhes sobre a configuração do banco de dados, veja Parâmetros do Banco de Dados na Referência de Configuração.

```
config :omniss7,
  mnesia_storage_type: :disc_copies  # ou :ram_copies para testes
```

Tipo de Armazenamento	Descrição	Persistência	Caso de Uso
`:disc_copies`	Armazenamento baseado em disco (padrão)	**Sobrevive a reinicializações**	Ambientes de produção
`:ram_copies`	Apenas em memória	Perdido na reinicialização	Testes, desenvolvimento

Padrão: `:disc_copies`

Localização do Banco de Dados Mnesia

O Mnesia armazena tabelas de roteamento no diretório Mnesia da aplicação:

- **Localização**: `Mnesia.{node_name}/` (por exemplo, `Mnesia.nonode@nohost/`)
- **Tabelas** (todas criadas em `M3UARouting.init_database`):

Tabela	Prop��sito
`m3ua_peer`	Endpoints de conexão M3UA / M2PA
`m3ua_route`	Rotas de Código de Ponto (DPC → par, com suporte a máscara)
`m3ua_gt_route`	Rotas de Título Global (prefixo → par, com correspondência e reescritas de SSN/TT/NPI/NAI)
`m3ua_dest_state`	Estado de disponibilidade por destino controlado pelo gerenciamento de roteamento de sinalização MTP3 (TFP/TFA/TFR, Q.704 §13)
`ss7_screen_rule`	Regras de permissão/negação de filtragem de gateway (firewall SCCP)

Gerenciando Rotas

Você tem três opções para gerenciar rotas:

1. **runtime.exs** - Configuração estática carregada na inicialização
2. **Interface Web** - Gerenciamento interativo de rotas (veja Guia da Interface Web)
3. **API REST** - Gerenciamento programático de rotas (veja Guia da API)

Melhor Prática: Use `runtime.exs` para configuração base e a Interface Web para alterações dinâmicas de rotas durante a operação.

Configurando Pares M3UA

Os pares representam endpoints de conexão M3UA (outros STPs, HLRs, MSCs, SMSCs). Adicione pares ao `config/runtime.exs`.

Exemplo de Configuração de Par

> **Nota**: A chave de configuração `peers` é o padrão atual. A chave legado `m3ua_peers` ainda é suportada para compatibilidade retroativa.

```
config :omniss7,
  peers: [
    # Conexão de saída para o STP Parceiro (papel: :client)
    %{
      peer_id: 1,                           # Identificador único
      name: "Partner_STP_West",             # Nome descritivo
      role: :client,                        # :client para saída,
:server para entrada
      local_ip: {10, 0, 0, 1},             # IP local para
vincular
      local_port: 0,                        # 0 = atribuição de
porta dinâmica
      remote_ip: {10, 0, 0, 10},           # IP do par remoto
      remote_port: 2905,                    # Porta do par remoto
      routing_context: 1,                   # Contexto de
roteamento M3UA
      point_code: 100,                      # Código de ponto
deste par
      network_indicator: :international     # :international ou
:national
    },

    # Conexão com HLR Local (papel: :client)
    %{
      peer_id: 2,
      name: "Local_HLR",
      role: :client,
      local_ip: {10, 0, 0, 1},
      local_port: 0,
      remote_ip: {10, 0, 0, 20},
      remote_port: 2905,
      routing_context: 2,
      point_code: 200,
      network_indicator: :international
    },

    # Conexão de entrada do MSC Remoto (papel: :server)
    # Para papel :server, o STP aguarda conexão de entrada
    %{
      peer_id: 3,
      name: "Remote_MSC",
      role: :server,                        # Aceitar conexão de
entrada
```

```
      remote_ip: {10, 0, 0, 30},           # IP de origem esperado
      remote_port: 2905,                    # Porta de origem
esperada (0 = aceitar de qualquer porta)
      routing_context: 3,
      point_code: 300,
      network_indicator: :international
    },

    # Conexão de entrada com porta de origem dinâmica (sem
filtragem de porta)
    %{
      peer_id: 4,
      name: "Dynamic_Client",
      role: :server,
      remote_ip: {10, 0, 0, 40},           # IP de origem esperado
      remote_port: 0,                       # 0 = aceitar conexões
de qualquer porta de origem
      routing_context: 4,
      point_code: 400,
      network_indicator: :international
    }
  ]
```

Parâmetros de Configuração do Peer

Parâmetro	Tipo	Necessário	Descrição
`peer_id`	Inteiro	Sim	Identificador numérico único para o peer
`name`	String	Sim	Nome legível por humanos para logs e monitoramento
`role`	Atom	Sim	`:client` (saída) ou `:server` (entrada)
`local_ip`	Tupla ou Lista	Sim (cliente)	Endereço(s) IP local(is) para vincular. Único: `{10, 0, 0, 1}` ou Múltiplos para multihoming SCTP: `[{10, 0, 0, 1}, {10, 0, 0, 2}]`
`local_port`	Inteiro	Sim (cliente)	Porta local (0 para dinâmica)
`remote_ip`	Tupla ou Lista	Sim	**Função do cliente**: Tupla ou lista única para peer remoto multihomed. **Função do servidor**: Apenas tupla única - o IP do qual o peer remoto se conecta (veja a nota abaixo)
`remote_port`	Inteiro	Sim	Porta do peer remoto (0 para entrada = aceitar qualquer porta de origem)
`routing_context`	Inteiro	Sim	Identificador de contexto de roteamento M3UA

Parâmetro	Tipo	Necessário	Descrição
`point_code`	Inteiro	Sim	Código de ponto SS7 deste peer
`network_indicator`	Atom	Não	`:international` ou `:national`

> **Multihoming SCTP para Função de Servidor (Conexões de Entrada)**: Ao aceitar conexões de entrada de um peer remoto multihomed, você só precisa especificar o **único endereço IP** que o peer remoto usa para iniciar a conexão SCTP (SCTP INIT). Isso deve ser uma **tupla única**, não uma lista. O SCTP descobrirá automaticamente os outros endereços IP multihomed do peer durante o handshake de associação. O formato de lista para `remote_ip` é usado apenas para **função de cliente** ao se conectar a um peer remoto multihomed.
>
> **Multihoming SCTP para Função de Cliente (Conexões de Saída)**: Para redundância de rede ao se conectar a peers remotos, você pode configurar múltiplos endereços IP para `local_ip` e `remote_ip` usando listas. Isso permite a troca automática se um caminho de rede falhar. Veja o Guia de Multihoming SCTP para exemplos de configuração detalhados e melhores práticas.

Filtragem de Porta de Origem para Conexões de Entrada

Para **conexões de entrada** (função: `:server`), o parâmetro `remote_port` controla a filtragem da porta de origem:

- **Porta Específica** (por exemplo, `remote_port: 2905`): Aceitar apenas conexões dessa porta de origem exata
  - Fornece segurança adicional validando a porta de origem
  - Use quando o peer remoto usar uma porta de origem fixa
- **Qualquer Porta** (`remote_port: 0`): Aceitar conexões de qualquer porta de origem

- Útil quando o peer remoto usa portas de origem dinâmicas/ephemerais
- Apenas valida o endereço IP de origem
- Mais flexível, mas ligeiramente menos seguro

Exemplo:

```
# Aceitar apenas de 192.0.2.10:2905 (porta específica)
%{
  peer_id: 1,
  name: "Strict_Peer",
  role: :server,
  remote_ip: {10, 5, 198, 200},
  remote_port: 2905,
  # ... outra configuração
}

# Aceitar de 192.0.2.10 com qualquer porta de origem
%{
  peer_id: 2,
  name: "Flexible_Peer",
  role: :server,
  remote_ip: {10, 5, 198, 200},
  remote_port: 0,  # Aceitar de qualquer porta de origem
  # ... outra configuração
}
```

Suporte ao Protocolo M2PA

OmniSS7 suporta os protocolos **M3UA** e **M2PA** para transporte de sinalização SS7.

O que é M2PA?

M2PA (Camada de Adaptação Peer-to-Peer do Usuário MTP2) é um protocolo padronizado pelo IETF (RFC 4165) para transportar mensagens MTP3 SS7 sobre redes IP usando SCTP.

M3UA vs M2PA: Principais Diferenças

Recurso	M3UA	M2PA
Arquitetura	Cliente/Servidor (ASP/SGW)	Peer-to-Peer
Caso de Uso	Gateway entre SS7 e IP	Links diretos ponto a ponto
Gerenciamento de Estado de Link	Nível de aplicação (ASPUP/ASPAC)	Estilo MTP2 (Alinhamento, Prova, Pronto)
Números de Sequência	Sem sequenciamento inerente	BSN/FSN de 24 bits para entrega ordenada
Implantação Típica	Gateway SS7 para IP, STP	Links de sinalização diretos entre nós
RFC	RFC 4666	RFC 4165

Orientação sobre Seleção de Protocolo

Recomendação: Use M3UA por padrão. Use M2PA apenas quando especificamente necessário.

Quando Usar M3UA (Recomendado)

M3UA é o protocolo recomendado para a maioria das implantações:

- **Implantações STP**: Implementações padrão de ponto de transferência de sinalização
- **Funções de Gateway**: Conectando redes SS7 com sinalização baseada em IP

- **Conexões de Elementos de Rede**: Conectando HLRs, MSCs, SMSCs e outros elementos de rede ao seu STP
- **Gateway de Sinalização (SGW)**: Gateway central aceitando conexões de vários Servidores de Aplicação
- **Topologias Flexíveis**: Arquiteturas cliente/servidor com controle centralizado
- **Redes Multi-vendor**: Padrão da indústria amplamente suportado (RFC 4666)

Use M3UA para conectar elementos de rede (HLR, MSC, SMSC, VLR, etc.) ao seu STP.

Quando Usar M2PA (Apenas Casos Especiais)

M2PA deve ser usado apenas em cenários específicos:

- **Links STP para STP**: Conexões diretas ponto a ponto entre Pontos de Transferência de Sinalização em uma rede multi-STP
- **Substituição de TDM Legado**: Substituindo links tradicionais SS7 TDM quando o sistema remoto exige especificamente M2PA
- **Compatibilidade MTP2 Necessária**: Ao conectar a sistemas legados que exigem gerenciamento de estado de link no estilo MTP2
- **Requisito de Parceiro**: Quando um parceiro ou interconexão exige especificamente o protocolo M2PA

Importante: Não use M2PA para conectar elementos de rede (HLR, MSC, SMSC) ao seu STP - use M3UA em vez disso. M2PA é projetado para interconexões STP para STP onde ambos os lados operam como nós de roteamento.

Configurando Peers M2PA

Peers M2PA são configurados da mesma forma que peers M3UA, com um parâmetro `protocol` adicional.

Configuração do Peer M2PA

Adicione peers M2PA à sua configuração `peers` em `config/runtime.exs` (tanto peers M3UA quanto M2PA compartilham a mesma seção de configuração, diferenciados pelo parâmetro `protocol`):

Parâmetros Chave para M2PA:

Parâmetro	Valor	Descrição
`protocol`	`:m2pa`	Especifica o protocolo M2PA (padrão é `:m3ua` se omitido)
`role`	`:client` ou `:server`	Direção da conexão
`local_port`	Inteiro	Porta SCTP local (a porta padrão do M2PA é **3565**)
`remote_port`	Inteiro	Porta SCTP remota (a porta padrão do M2PA é **3565**)
`point_code`	Inteiro	Seu código de ponto
`adjacent_point_code`	Inteiro	Código de ponto do peer remoto (específico do M2PA)
`send_sltm`	Booleano	Controla o comportamento do SLTM (veja Configuração SLTM abaixo)
`network_indicator`	Atom	`:international` ou `:national` - deve corresponder ao peer remoto

Nota: M2PA usa **porta 3565** como padrão da indústria (diferente da porta 2905 do M3UA).

Configuração SLTM

SLTM (Mensagem de Teste de Link de Sinalização) e **SLTA (Reconhecimento de Teste de Link de Sinalização)** são mensagens de manutenção MTP3 usadas para verificar a conectividade de ponta a ponta após um link M2PA atingir o estado PRONTO. De acordo com a ITU-T Q.707, um lado envia SLTM e o outro responde com SLTA para confirmar que o link está operacional.

Comportamento SLTM

O parâmetro `send_sltm` controla qual lado inicia o teste SLTM:

Valor `send_sltm`	Comportamento
`true`	Enviamos SLTM quando o link se torna PRONTO, aguardamos SLTA
`false`	Aguardamos o peer enviar SLTM, respondemos com SLTA
Não definido (padrão)	Segue Q.707: o respondedor SCTP envia SLTM, o iniciador SCTP aguarda

Comportamento Padrão (Conforme Q.707):

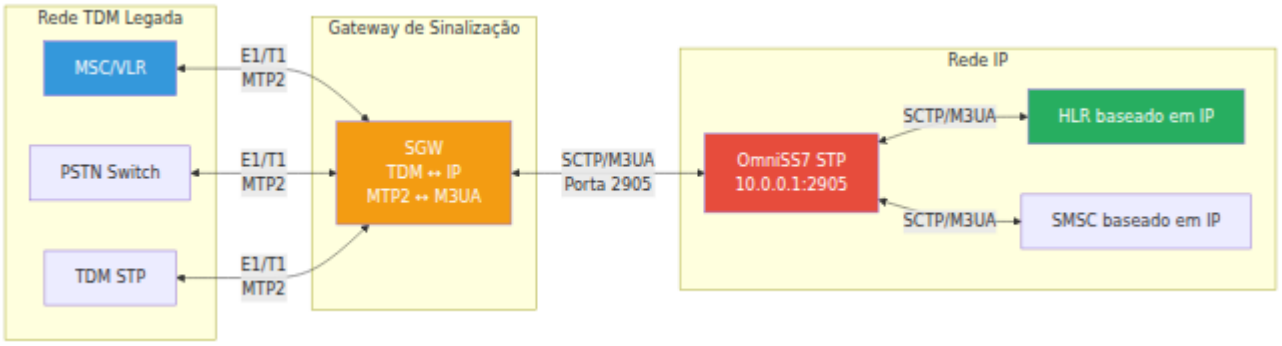
- Se `initiate_connection: true` (iniciador/client SCTP) → Aguardamos o peer enviar SLTM
- Se `initiate_connection: false` (respondedor/servidor SCTP) → Enviamos SLTM

Quando Substituir os Padrões SLTM

Substitua o comportamento padrão ao se conectar a equipamentos que não seguem as convenções Q.707:

```
# Exemplo: Forçar nosso lado a enviar SLTM independentemente do
papel SCTP
%{
  peer_id: 100,
  name: "Partner_STP",
  protocol: :m2pa,
  role: :client,
  local_ip: {10, 0, 0, 1},
  local_port: 3565,
  remote_ip: {10, 0, 0, 2},
  remote_port: 3565,
  point_code: 7415,
  adjacent_point_code: 15528,
  network_indicator: :international,
  send_sltm: true  # Substituir: Enviamos SLTM mesmo sendo o
iniciador SCTP
}
```

Fluxo de Mensagem SLTM



Resolução de Problemas de SLTM

Sintoma: Link atinge PRONTO, mas nenhum tráfego flui

Ambos os lados podem estar aguardando que o outro envie SLTM. Verifique os logs por:

```
"aguardando o peer enviar SLTM"
```

Resolução: Configure `send_sltm: true` em um lado para quebrar o impasse.

Sintoma: SLTM enviado, mas nenhum SLTA recebido

Causas possíveis:

1. `adjacent_point_code` está incorreto (SLTM enviado com DPC errado)
2. Desvio de indicador de rede (`:international` vs `:national`)
3. Peer remoto não configurado para responder ao nosso código de ponto

Resolução: Verifique se `adjacent_point_code` corresponde ao código de ponto do STP (configurado em `sctp_handler.point_code`).

Estados do Link M2PA

Links M2PA progridem através de vários estados durante a inicialização:

1. **Desligado** - Nenhuma conexão estabelecida
2. **Alinhamento** - Fase de sincronização inicial (~1 segundo)
3. **Prova** - Verificação da qualidade do link (~2 segundos)
4. **Pronto** - Link ativo e pronto para tráfego

A progressão do estado do link garante sinalização confiável antes que o tráfego seja trocado.

Gerenciando Peers M2PA via UI Web

A página **Roteamento** na UI Web fornece suporte total para gerenciar peers M2PA:

1. **Navegue** até a página de Roteamento
2. **Selecione** a aba "Peers"
3. **Clique** em "Adicionar Novo Peer"
4. **Escolha** "M2PA (RFC 4165)" no menu suspenso de Protocolo
5. **Preencha** a configuração do peer:
   - Nome do Peer (identificador descritivo)
   - Protocolo: M2PA
   - Função: cliente ou servidor
   - Código de Ponto (seu PC)

    - Endereços IP Locais/Remotos
    - Portas Locais/Remotas (tipicamente 3565 para M2PA)
    - Indicador de Rede (internacional ou nacional)
6. **Clique** em "Salvar Peer"

A tabela de peers exibe o tipo de protocolo com codificação de cores:

- **Azul** - peers M3UA
- **Verde** - peers M2PA

Comportamento de Roteamento M2PA

Peers M2PA se integram perfeitamente ao sistema de roteamento do OmniSS7:

- **Rotas de Código de Ponto**: Funcionam de forma idêntica para M2PA e M3UA
- **Rotas de Título Global**: Totalmente suportadas em links M2PA
- **Prioridade de Rota**: Peers M2PA e M3UA podem ser misturados nas mesmas tabelas de roteamento
- **Revezamento de Mensagens**: Mensagens podem chegar em M2PA e ser roteadas para M3UA, e vice-versa

Métricas M2PA

M2PA fornece métricas abrangentes do Prometheus para monitorar a saúde do link e o tráfego:

Métricas de Tráfego:

- `m2pa_messages_sent_total` - Total de mensagens MTP3 enviadas por link
- `m2pa_messages_received_total` - Total de mensagens MTP3 recebidas por link
- `m2pa_bytes_sent_total` - Total de bytes enviados sobre M2PA
- `m2pa_bytes_received_total` - Total de bytes recebidos sobre M2PA

Todas as métricas de tráfego são rotuladas por: `link_name`, `point_code`, `adjacent_pc`

Métricas de Estado do Link:

- `m2pa_link_state_changes_total` - Transições de estado do link (DESCONECTADO → ALINHAMENTO → PROVA → PRONTO)
  - Rótulos: `link_name`, `from_state`, `to_state`

Métricas de Erro:

- `m2pa_errors_total` - Total de erros por tipo
  - `decode_error` - Falhas de decodificação de mensagens M2PA
  - `encode_error` - Falhas de codificação de mensagens M2PA
  - `sctp_send_error` - Falhas de transmissão SCTP
  - Rótulos: `link_name`, `error_type`

Métricas de Acesso:

- Endpoint do Prometheus: `http://your-server:9090/metrics`
- Métricas se registram automaticamente na inicialização da aplicação

Melhores Práticas M2PA

1. **Seleção de Porta**: Use a porta 3565 para M2PA (padrão da indústria)
2. **Monitoramento de Link**: Monitore mudanças de estado do link via métricas
3. **Regras de Firewall**: Certifique-se de que SCTP (protocolo IP 132) está permitido
4. **Códigos de Ponto**: Certifique-se de que os códigos de ponto adjacentes estão configurados corretamente em ambos os lados
5. **Indicador de Rede**: Deve corresponder entre peers (internacional ou nacional)
6. **Teste**: Use a página de Teste de Roteamento para verificar a conectividade após a configuração

Requisitos de Socket M2PA

M2PA usa sockets compartilhados SCTP.SocketHandler. Todos os peers M2PA usam automaticamente o SCTP.SocketHandler para gerenciamento de

sockets, o que permite que múltiplos peers compartilhem eficientemente a mesma porta SCTP.

Requisitos:

- SCTP.SocketHandler deve estar habilitado e em execução
- O socket compartilhado deve ser configurado antes que os peers M2PA comecem

Exemplo:

```
# Habilitar SCTP.SocketHandler
sctp_handler: %{
  enabled: true,
  local_ip: {10, 179, 4, 10},
  local_port: 3565,
  point_code: 100
}

# Peers M2PA usam automaticamente o socket compartilhado
peers: [
  %{
    peer_id: 1,
    name: "M2PA_Link_1",
    protocol: :m2pa,
    role: :client,
    local_ip: {10, 179, 4, 10},
    local_port: 3565,
    remote_ip: {10, 179, 4, 20},
    remote_port: 3565,
    point_code: 100,
    adjacent_point_code: 200
  },
  %{
    peer_id: 2,
    name: "M2PA_Link_2",
    protocol: :m2pa,
    role: :client,
    local_ip: {10, 179, 4, 10},
    local_port: 3565,           # Mesma porta compartilhada entre
peers
    remote_ip: {10, 179, 4, 30},
    remote_port: 3565,
    point_code: 100,
    adjacent_point_code: 300
  }
]
```

Requisitos SCTP:

- Entrega ordenada (sem a flag `unordered`)
- PPID 5 (identificador M2PA conforme RFC 4165)

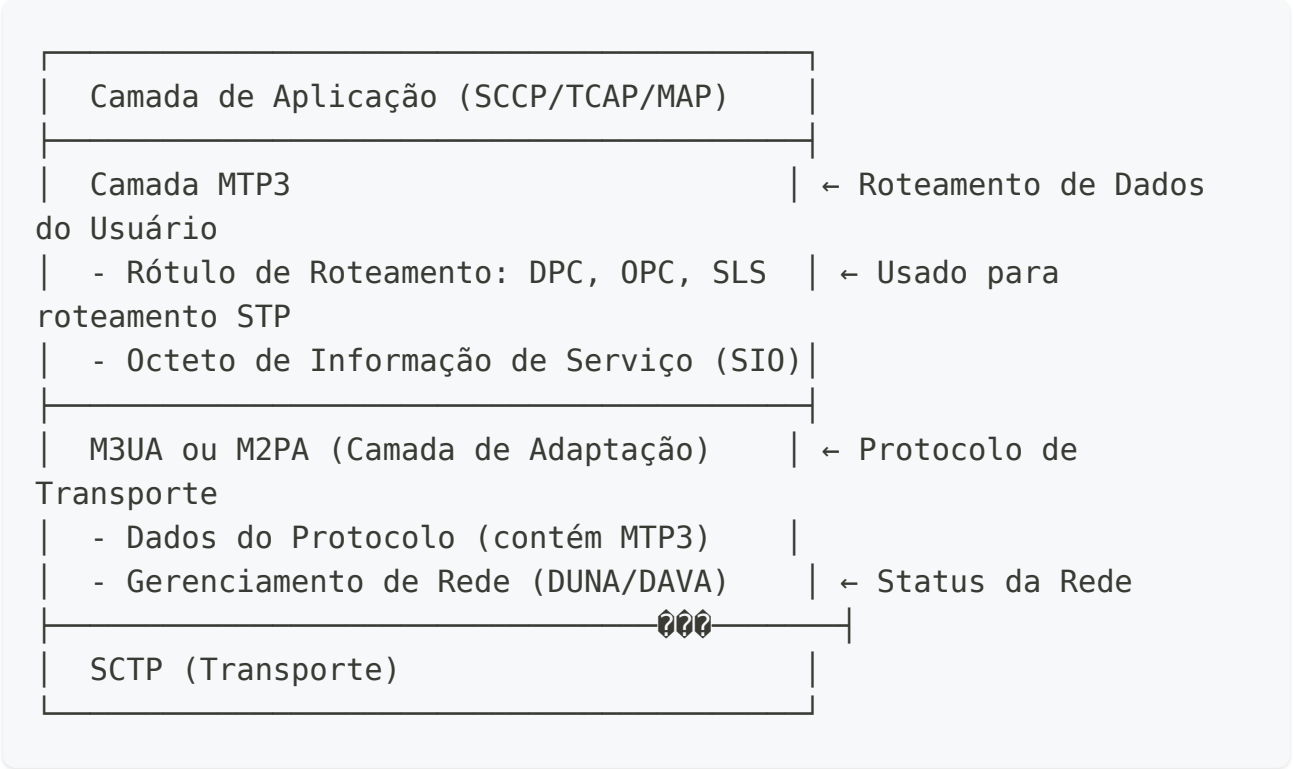
Configurando Roteamento de Código de Ponto

O roteamento de Código de Ponto direciona mensagens com base no **Código de Ponto de Destino (DPC)** no cabeçalho MTP3.

Compreendendo os Códigos de Ponto na Pilha de Protocólios SS7

Códigos de ponto existem em diferentes camadas da pilha de protocolos SS7. Compreender essa distinção é importante:

Camadas da Pilha de Protocólios:



Dois Tipos de Códigos de Ponto:

1. **Códigos de Ponto da Camada MTP3** (Usados para Roteamento):
    - Localizados no rótulo de roteamento MTP3 (DPC, OPC)
    - Presentes no parâmetro de Dados do Protocolo M3UA (tag 528)
    - Presentes nas mensagens de Dados do Usuário M2PA

    - **O STP usa esses valores DPC para decisões de roteamento**
    - Esses determinam onde a mensagem é finalmente entregue

2. **Códigos de Ponto da Camada M3UA** (Usados para Gerenciamento de Rede):

    - Presentes em mensagens de gerenciamento M3UA (DUNA, DAVA, SCON, DUPU)
    - Indicam códigos de ponto afetados para status da rede
    - Informam aos peers quais destinos estão disponíveis/in disponíveis
    - Não usados para roteamento de dados do usuário

Como Funciona o Roteamento STP:

- **Para mensagens M3UA DATA**: O STP extrai a mensagem MTP3 do parâmetro de Dados do Protocolo (tag 528), que contém o rótulo de roteamento MTP3 (DPC, OPC, SLS). O DPC da camada MTP3 é usado para procurar rotas.
- **Para mensagens de Dados do Usuário M2PA**: O STP extrai a mensagem MTP3 do campo de dados do usuário M2PA e, em seguida, lê o DPC do rótulo de roteamento MTP3.
- **Mensagens de gerenciamento M3UA**: Mensagens de gerenciamento de rede (DUNA, DAVA, SCON) contêm códigos de ponto afetados na camada M3UA para sinalizar o status da rede entre peers.

Rotas Básicas de Código de Ponto

Adicione rotas a `config/runtime.exs`:

```
config :omniss7,
  m3ua_routes: [
    # Roteie todo o tráfego para PC 100 para o peer 1 (Partner
STP)
    %{
      dest_pc: 100,                    # Código de ponto de
destino
      peer_id: 1,                      # Peer para roteamento
      priority: 1,                     # Prioridade (menor = maior
prioridade)
      network_indicator: :international
      # mask: 14                       # Opcional: padrão é 14
(correspondência exata)
    },

    # Roteie todo o tráfego para PC 200 para o peer 2 (HLR Local)
    %{
      dest_pc: 200,
      peer_id: 2,
      priority: 1,
      network_indicator: :international
    },

    # Exemplo de balanceamento de carga: PC 300 com rotas primária
e de backup
    %{
      dest_pc: 300,
      peer_id: 3,                      # Rota primária
      priority: 1,
      network_indicator: :international
    },
    %{
      dest_pc: 300,
      peer_id: 4,                      # Rota de backup (número de
prioridade mais alto)
      priority: 2,
      network_indicator: :international
    }
  ]
```

Nota: O campo `mask` é opcional e padrão para `14` (correspondência exata de código de ponto). Especifique `mask` apenas quando precisar de roteamento

baseado em intervalo (veja a seção Máscaras de Código de Ponto abaixo).

Lógica de Roteamento

1. O STP recebe mensagens M3UA DATA ou mensagens de Dados do Usuário M2PA
2. O STP extrai a **mensagem MTP3** do campo de Dados do Protocolo (M3UA) ou do campo de Dados do Usuário (M2PA)
3. O STP lê o **Código de Ponto de Destino (DPC)** do rótulo de roteamento MTP3
4. Procura na tabela de roteamento por DPC correspondente (considerando máscaras)
5. Se várias rotas existirem, seleciona a rota com **máscara mais específica** (valor de máscara mais alto), então **número de prioridade mais baixo**
6. Envolve a mensagem MTP3 em M3UA DATA ou Dados do Usuário M2PA para o peer de destino
7. Roteia a mensagem para o peer correspondente
8. Se o peer selecionado estiver desligado, tenta a próxima rota de prioridade mais alta

Máscaras de Código de Ponto

Códigos de ponto são valores de 14 bits (intervalo de 0-16383). Por padrão, rotas correspondem a um único código de ponto exatamente (máscara `/14`).

No entanto, você pode usar **máscaras de código de ponto** para criar rotas que correspondam a **intervalos** de códigos de ponto.

Compreendendo Máscaras

A máscara especifica quantos **bits mais significativos** devem corresponder entre o PC de destino da rota e o DPC da mensagem de entrada. Os bits restantes podem ser qualquer valor, criando um intervalo de códigos de ponto correspondentes.

Tabela de Referência de Máscaras:

Máscara	Códigos de Ponto Correspondidos	Caso de Uso
`/14`	1 PC (correspondência exata)	Destino único (padrão)
`/13`	2 PCs	Pequeno intervalo
`/12`	4 PCs	Pequeno intervalo
`/11`	8 PCs	Pequeno intervalo
`/10`	16 PCs	Intervalo médio
`/9`	32 PCs	Intervalo médio
`/8`	64 PCs	Intervalo médio
`/7`	128 PCs	Intervalo médio-grande
`/6`	256 PCs	Grande intervalo
`/5`	512 PCs	Grande intervalo
`/4`	1.024 PCs	Intervalo muito grande
`/3`	2.048 PCs	Intervalo muito grande
`/2`	4.096 PCs	Intervalo extremamente grande
`/1`	8.192 PCs	Metade de todos os PCs
`/0`	16.384 PCs	Todos os PCs (rota padrão/catch-all)

Exemplos de Máscaras de Código de Ponto

Nota: O campo `mask` é **opcional** em todos os exemplos. Se omitido, o padrão é `14` (correspondência exata).

Exemplo 1: Código de Ponto Único (Comportamento Padrão)

```
# Sem campo de máscara (recomendado para um único PC)
%{
  dest_pc: 1000,
  peer_id: 1,
  priority: 1,
  network_indicator: :international
}
# Máscara padrão é 14 - Corresponde: Apenas PC 1000

# Máscara explícita (mesmo resultado)
%{
  dest_pc: 1000,
  peer_id: 1,
  priority: 1,
  mask: 14,                              # Correspondência exata
explícita
  network_indicator: :international
}
# Corresponde: Apenas PC 1000
```

Exemplo 2: Pequeno Intervalo

```
%{
  dest_pc: 1000,
  peer_id: 2,
  priority: 1,
  mask: 12,                              # Corresponde a 4 PCs
  network_indicator: :international
}
# Corresponde: PC 1000, 1001, 1002, 1003
```

Exemplo 3: Intervalo Médio

```
%{
  dest_pc: 1000,
  peer_id: 3,
  priority: 1,
  mask: 8,                                # Corresponde a 64 PCs
  network_indicator: :international
}
# Corresponde: PC 1000-1063 (64 códigos de ponto consecutivos)
```

Exemplo 4: Rota Padrão/Catch-All

```
%{
  dest_pc: 0,
  peer_id: 4,
  priority: 10,                           # Baixa prioridade
(número alto)
  mask: 0,                                # Corresponde a todos os
PCs
  network_indicator: :international
}
# Corresponde: Todos os códigos de ponto (0-16383)
# Use como uma rota catch-all/padrão com baixa prioridade
```

Combinando Rotas Específicas e Máscaras

Você pode combinar rotas específicas com rotas mascaradas para um roteamento flexível:

```
config :omniss7,
  m3ua_routes: [
    # Rota específica para PC 1000 (tem precedência)
    %{
      dest_pc: 1000,
      peer_id: 1,
      priority: 1,
      network_indicator: :international
      # a máscara padrão é 14 (correspondência exata)
    },

    # Rota de intervalo para PCs 1000-1063
    %{
      dest_pc: 1000,
      peer_id: 2,
      priority: 1,
      mask: 8,                          # Corresponde a 64 PCs
      network_indicator: :international
    },

    # Rota padrão/de fallback para todos os outros PCs
    %{
      dest_pc: 0,
      peer_id: 3,
      priority: 10,                     # Baixa prioridade
      mask: 0,                          # Corresponde a todos os
PCs
      network_indicator: :international
    }
  ]
```

Decisão de Roteamento para DPC 1000:

1. Corresponde à máscara `/14` (PC 1000 exatamente) - **Selecionado** (mais específico)
2. Também corresponde à máscara `/8` (intervalo de PC 1000-1063) - Ignorado (menos específico)
3. Também corresponde à máscara `/0` (todos os PCs) - Ignorado (menos específico)

Decisão de Roteamento para DPC 1015:

1. Não corresponde à máscara `/14` (apenas PC 1000)
2. Corresponde à máscara `/8` (intervalo de PC 1000-1063) - **Selecionado** (correspondência mais específica)
3. Também corresponde à máscara `/0` (todos os PCs) - Ignorado (menos específico)

Decisão de Roteamento para DPC 5000:

1. Não corresponde à máscara `/14`
2. Não corresponde à máscara `/8`
3. Corresponde à máscara `/0` (todos os PCs) - **Selecionado** (única correspondência, rota de fallback)

Melhores Práticas

1. **Omitir `mask` para Destinações Únicas**: Para correspondências exatas de código de ponto, omita completamente o campo `mask` (padrão é `/14`)
2. **Usar `/14` Explicitamente Somente Quando Necessário**: Especifique `mask: 14` apenas quando precisar deixar claro na documentação ou ao misturar com rotas de intervalo
3. **Usar Máscaras de Intervalo para Blocos de Rede**: Roteie segmentos inteiros de rede para pares específicos com máscaras de `/0` a `/13`
4. **Usar `/0` como Fallback**: Crie uma rota padrão com baixa prioridade para capturar tráfego não correspondido
5. **O Mais Específico Vence**: O mecanismo de roteamento sempre seleciona a rota correspondente mais específica (maior valor de máscara) primeiro
6. **Prioridade como Desempate**: Se várias rotas tiverem a mesma máscara, o número de prioridade mais baixo vence

---

Configurando Roteamento de Título Global (GT)

O roteamento de Título Global permite **roteamento baseado em conteúdo** usando números de telefone ou valores IMSI em vez de códigos de ponto. Para

tradução avançada de endereço de Título Global com base na parte chamadora/chamada, consulte o Guia de NAT de Título Global.

Pré-requisitos

- Habilitar roteamento GT: `enable_gt_routing: true` em `config/runtime.exs`

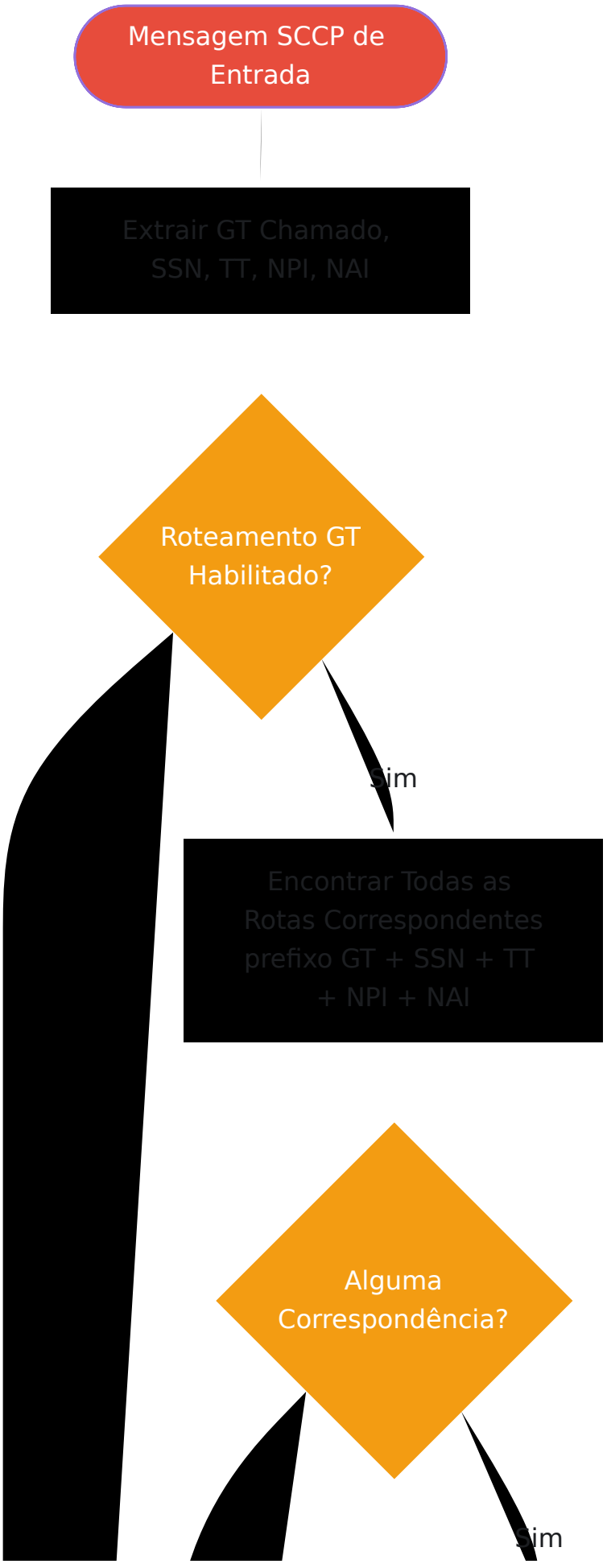
Configuração de Rota GT

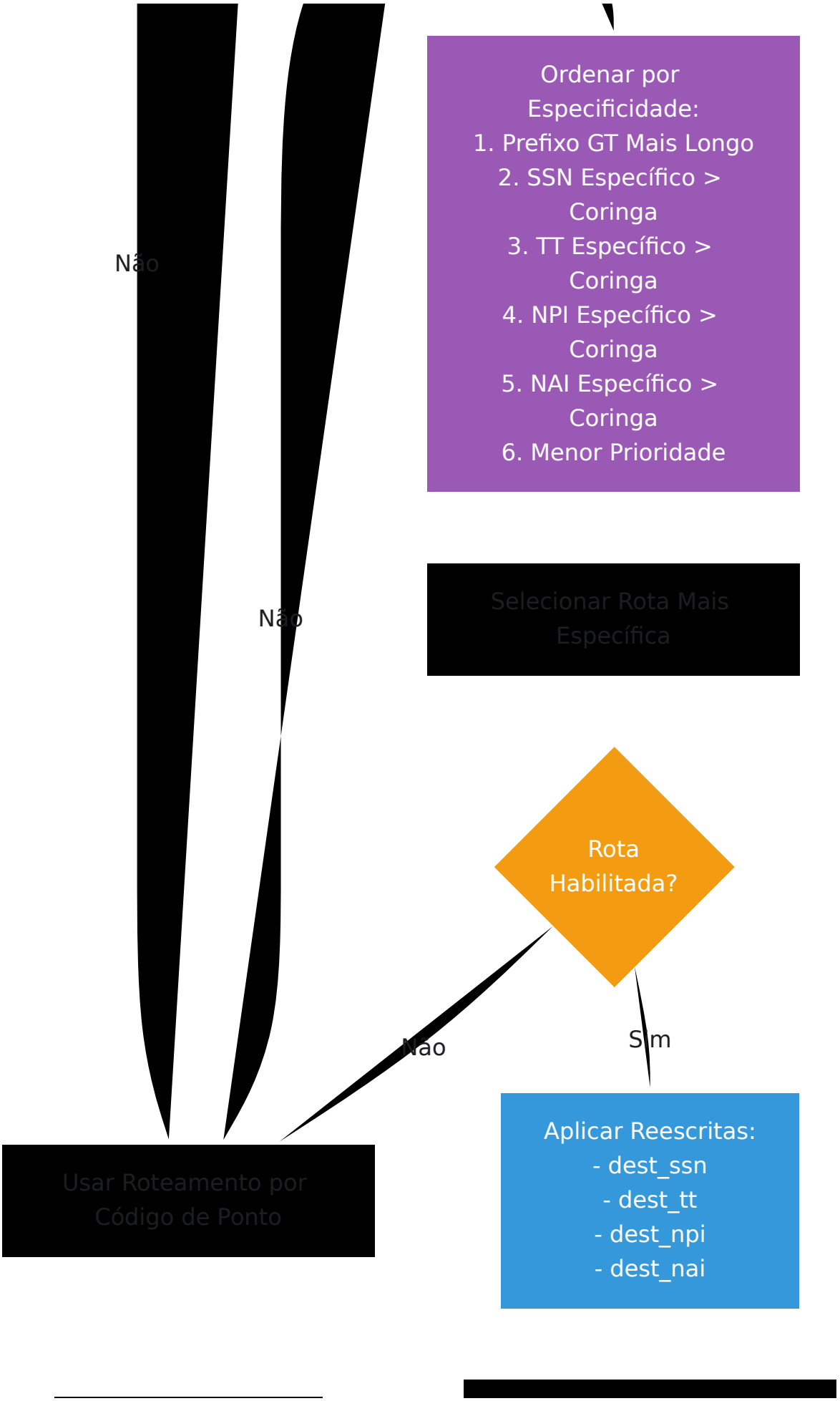
```
config :omniss7,
  # Habilitar roteamento GT
  enable_gt_routing: true,

  m3ua_gt_routes: [
    # Roteie todos os números do Reino Unido (prefixo 44) para o
par 1
    %{
      gt_prefix: "44",                # Prefixo de Título Global
a ser correspondido
      peer_id: 1,                     # Par de destino
      priority: 1,                    # Prioridade (menor =
maior)
      description: "Números do Reino Unido"        # Descrição
para registro
    },

    # Roteie números dos EUA (prefixo 1) para o par 2
    %{
      gt_prefix: "1",
      peer_id: 2,
      priority: 1,
      description: "Números dos EUA"
    },

    # Rota mais específica: números móveis do Reino Unido
começando com 447
    %{
      gt_prefix: "447",               # A correspondência de
prefixo mais longa vence
      peer_id: 3,
      priority: 1,
      description: "Números móveis do Reino Unido"
    },

    # Roteamento específico de SSN (opcional)
    %{
      gt_prefix: "555",
      source_ssn: 8,                  # Corresponde apenas se SSN
de origem = 8 (SMSC)
      peer_id: 4,
```

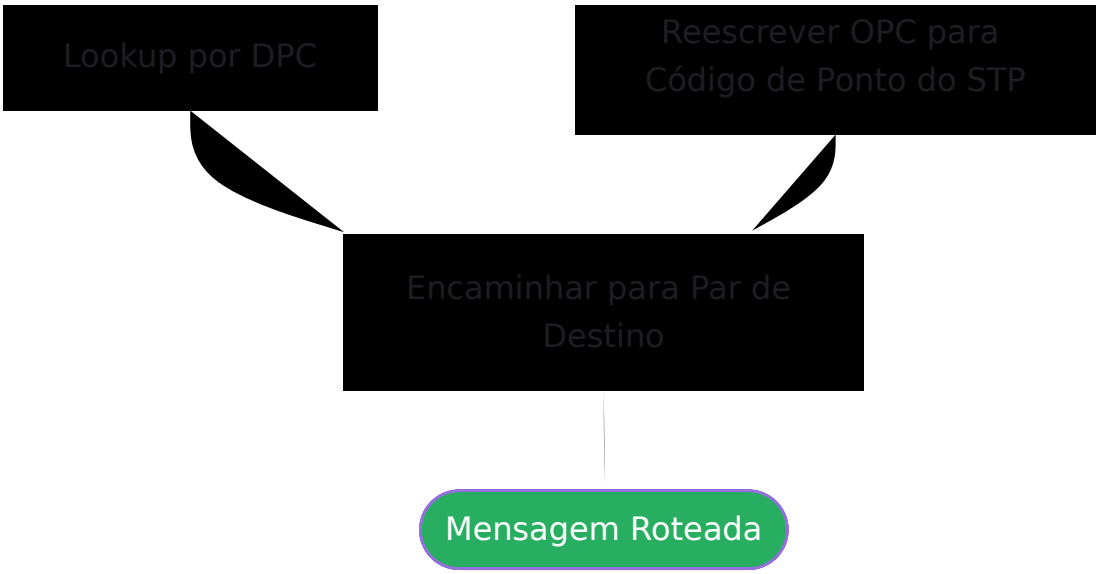
```
      dest_ssn: 6,                    # Reescreve SSN de destino
para 6 (HLR)
      priority: 1,
      description: "Tráfego SMS para prefixo 61"
    }
  ]
```

Lógica de Roteamento GT

O algoritmo de roteamento GT segue este processo de decisão:


Mensagem SCCP de Entrada
Extrair GT Chamado, SSN, TT, NPI, NAI
Roteamento GT Habilitado?
Sim
Encontrar Todas as Rotas Correspondentes prefixo GT + SSN + TT + NPI + NAI
Alguma Correspondência?
Sim


Não
Não
Ordenar por Especificidade:
1. Prefixo GT Mais Longo
2. SSN Específico > Coringa
3. TT Específico > Coringa
4. NPI Específico > Coringa
5. NAI Específico > Coringa
6. Menor Prioridade
Selecionar Rota Mais Específica
Rota Habilitada?
Não
Sim
Aplicar Reescritas:
- dest_ssn
- dest_tt
- dest_npi
- dest_nai
Usar Roteamento por Código de Ponto



Passos de Roteamento:

1. **Correspondência de Prefixo Mais Longo**: O STP encontra todas as rotas GT onde o prefixo corresponde ao início do Título Global
    - Exemplo: GT "447712345678" corresponde tanto a "44" quanto a "447", mas "447" vence (correspondência mais longa)
2. **Correspondência de SSN** (Opcional):
    - Se `source_ssn` estiver especificado, a rota corresponde apenas quando o SSN da Parte Chamadora SCCP é igual a esse valor
    - Se `source_ssn` for `nil`, a rota corresponde a qualquer SSN (coringa)
3. **Correspondência de TT/NPI/NAI** (Opcional):
    - Se `source_tt`, `source_npi` ou `source_nai` estiverem especificados, as rotas devem corresponder a esses indicadores
    - Valores `nil` atuam como coringas (correspondem a qualquer valor)
4. **Seleção Baseada em Especificidade**:
    - Rotas com critérios de correspondência mais específicos vencem sobre coringas
    - Ordem de prioridade: Comprimento do Prefixo GT → SSN → TT → NPI → NAI → Número de Prioridade
5. **Reescrita de Indicadores** (Opcional):

- Se `dest_ssn`, `dest_tt`, `dest_npi` ou `dest_nai` estiverem especificados, o STP reescreve esses indicadores
- Útil para normalização de protocolo e interconexão de rede

6. **Fallback para Código de Ponto**:
    - Se nenhuma rota GT corresponder, o STP recua para o roteamento por Código de Ponto usando o DPC

Roteamento GT Avançado: Tipo de Tradução, NPI e NAI

Além da correspondência de prefixo GT e SSN, o STP suporta roteamento e transformação com base nos indicadores de Título Global SCCP:

- **Tipo de Tradução (TT)**: Identifica o plano de numeração e o tipo de endereço
- **Indicador de Plano de Numeração (NPI)**: Define o plano de numeração (por exemplo, ISDN, Dados, Telex)
- **Indicador de Natureza de Endereço (NAI)**: Especifica o formato do endereço (por exemplo, Internacional, Nacional, Assinante)

Correspondência (Indicadores de Origem)

As rotas podem corresponder aos indicadores da mensagem de entrada:

- `source_tt`: Corresponder mensagens com Tipo de Tradução específico

- `source_npi`: Corresponder mensagens com Indicador de Plano de Numeração específico
- `source_nai`: Corresponder mensagens com Indicador de Natureza de Endereço específico
- Valor `nil` = coringa (corresponde a qualquer valor)

Transformação (Indicadores de Destino)

As rotas podem reescrever indicadores ao encaminhar:

- `dest_tt`: Transformar Tipo de Tradução para novo valor
- `dest_npi`: Transformar Indicador de Plano de Numeração para novo valor
- `dest_nai`: Transformar Indicador de Natureza de Endereço para novo valor
- Valor `nil` = preservar o valor original (sem transformação)

Seleção Baseada em Especificidade

Quando várias rotas correspondem, a rota mais específica é selecionada usando esta ordem de prioridade:

1. Correspondência de prefixo GT mais longa
2. SSN de origem específico sobre SSN coringa
3. TT de origem específico sobre TT coringa
4. NPI de origem específico sobre NPI coringa
5. NAI de origem específico sobre NAI coringa
6. Menor número de prioridade

Exemplos de Configuração

```
config :omniss7,
  enable_gt_routing: true,

  m3ua_gt_routes: [
    # Exemplo 1: Correspondência e transformação do Tipo de
Tradução
    %{
      gt_prefix: "44",
      peer_id: 1,
      source_tt: 0,      # Corresponder TT=0 (Desconhecido)
      dest_tt: 3,        # Transformar para TT=3 (Nacional)
      priority: 1,
      description: "Números do Reino Unido: transformação TT 0→3"
    },

    # Exemplo 2: Correspondência de NPI específico e transformação
de NAI
    %{
      gt_prefix: "1",
      peer_id: 2,
      source_npi: 1,     # Corresponder NPI=1 (ISDN/Telefônia)
      source_nai: 4,     # Corresponder NAI=4 (Internacional)
      dest_nai: 3,       # Transformar para NAI=3 (Nacional)
      priority: 1,
      description: "Números dos EUA: NAI Internacional→Nacional"
    },

    # Exemplo 3: Roteamento combinado de SSN e indicador
    %{
      gt_prefix: "33",
      source_ssn: 8,     # Corresponder tráfego SMS
      source_tt: 0,      # Corresponder TT=0
      dest_ssn: 6,       # Reescrever SSN para HLR
      dest_tt: 2,        # Transformar para TT=2
      dest_npi: 1,       # Definir NPI=1 (ISDN)
      dest_nai: 4,       # Definir NAI=4 (Internacional)
      peer_id: 3,
      priority: 1,
      description: "SMS francês: normalização completa"
    },

    # Exemplo 4: TT coringa, NPI específico
    %{
```

```
      gt_prefix: "49",
      source_tt: nil,    # Corresponder a qualquer TT (coringa)
      source_npi: 6,     # Corresponder NPI=6 (Dados)
      dest_npi: 1,       # Transformar para NPI=1 (ISDN)
      peer_id: 4,
      priority: 1,
      description: "Normalização da rede de dados alemã"
    }
  ]
```

Valores Comuns de TT/NPI/NAI

Tipo de Tradução (TT):

- `0` = Desconhecido
- `1` = Internacional
- `2` = Nacional
- `3` = Específico da Rede

Indicador de Plano de Numeração (NPI):

- `0` = Desconhecido
- `1` = ISDN/Telefônia (E.164)
- `3` = Dados (X.121)
- `4` = Telex (F.69)
- `6` = Móvel Terrestre (E.212)

Indicador de Natureza de Endereço (NAI):

- `0` = Desconhecido
- `1` = Número de Assinante
- `2` = Reservado para Uso Nacional
- `3` = Número Significativo Nacional
- `4` = Número Internacional

Exemplo de Decisão de Roteamento

Para uma mensagem de entrada com:

- GT: "447712345678"
- SSN: 8
- TT: 0
- NPI: 1
- NAI: 4

Com estas rotas configuradas:

```
# Rota A: TT coringa
%{gt_prefix: "447", peer_id: 1, priority: 1}

# Rota B: TT específico
%{gt_prefix: "447", source_tt: 0, peer_id: 2, priority: 1}

# Rota C: TT específico + NPI
%{gt_prefix: "447", source_tt: 0, source_npi: 1, peer_id: 3,
priority: 1}
```

Resultado: A Rota C é selecionada (mais específica: corresponde a GT + TT + NPI)

A mensagem é encaminhada com indicadores transformados de acordo com os valores de `dest_tt`, `dest_npi`, `dest_nai` da Rota C.

Exemplos de Roteamento GT

GT Chamado	SSN de Origem	TT	NPI	NAI	Rota Correspondente	Ra
447712345678	6	-	-	-	"447" → par 3	Corresp de prefi longa
441234567890	6	-	-	-	"44" → par 1	Corresp de prefi nenhum mais es
12125551234	6	-	-	-	"1" → par 2	Corresp de prefi número EUA
555881234567	8	-	-	-	"555" (SSN 8) → par 4	Corresp de GT + reescrev para 6
555881234567	6	-	-	-	"555" (SSN coringa) → par X	Corresp de GT, s reescrit
441234567890	6	0	1	4	"44" (TT=0) → par 1	Corresp de GT + transfor para 3
12125551234	8	0	1	4	"1" (TT=0, NPI=1, NAI=4)	Mais esp corresp GT+TT+

Casos de Uso Práticos para Roteamento TT/NPI/NAI

1. **Normalização de Interconexão de Rede**
    - Redes diferentes podem usar convenções de indicadores diferentes
    - Transformar indicadores no ponto de interconexão para garantir compatibilidade
    - Exemplo: A rede parceira usa TT=0 para internacional, sua rede usa TT=1
2. **Conversão de Protocolo**
    - Converter entre planos de numeração ao roteamento entre diferentes tipos de rede
    - Exemplo: Roteie de uma rede móvel (NPI=6) para PSTN (NPI=1)
3. **Padronização de Formato de Endereço**
    - Normalizar todo o tráfego de entrada para usar valores NAI consistentes
    - Exemplo: Converter todo o formato internacional (NAI=4) para formato nacional (NAI=3) para roteamento doméstico
4. **Roteamento Específico de Operadora**
    - Roteie com base no tipo de tradução para diferentes provedores de serviço
    - Exemplo: TT=0 roteia para a Operadora A, TT=2 roteia para a Operadora B
5. **Integração de Sistemas Legados**
    - Sistemas modernos podem usar valores de indicadores diferentes dos sistemas legados
    - Transformar no STP para manter compatibilidade retroativa

---

Manipulação de Trânsito SCCP

Além de selecionar um par de destino, o STP realiza as funções de trânsito da camada SCCP exigidas de um verdadeiro Ponto de Controle de Conexão de Sinalização. Isso se aplica às classes de mensagens sem conexão que o STP retransmite: **UDT** (0x09), **XUDT** (0x11) e **LUDT** (0x13).

Roteamento por SSN Após Tradução de Título Global

Por padrão, o STP **preserva o indicador de roteamento da Parte Chamadora como roteamento por GT** quando encaminha uma mensagem correspondida por uma rota GT. Isso é deliberado: selecionar um próximo salto por prefixo GT é *roteamento GT* (intermediário), não *tradução final de GT*. De acordo com ITU-T Q.714 §2.4, um nó intermediário deve deixar o indicador de roteamento como roteamento por GT para que o nó a jusante realize a tradução. Mudar para roteamento por SSN nesse caso rotula um resultado intermediário como final. A mensagem então carrega um GT com roteamento por SSN e sem código de ponto, que um nó a jusante (por exemplo, um MSS que esperava executar seu próprio GTT) rejeita com `providerAbort(abnormalDialogue)`.

Defina o indicador de roteamento para roteamento por SSN **somente** quando *este* STP for o tradutor final e a rota entregar a um ponto final apenas SSN que não pode realizar mais GTT. Isso é uma opção explícita por rota via a flag `route_on_ssn: true` na rota GT:

```
m3ua_gt_routes: [
  # Roteamento de Trânsito - preserva roteamento por GT (padrão);
a jusante realiza GTT
  %{gt_prefix: "44", peer_id: 2, priority: 1, description:
"Números do Reino Unido"},

  # Entrega final a um ponto final apenas SSN - muda para
roteamento por SSN
  %{
    gt_prefix: "1234",
    peer_id: 1,
    priority: 1,
    dest_ssn: 8,
    route_on_ssn: true,
    description: "Entrega final para MSC apenas SSN"
  }
]
```

A flag também pode ser definida por rota via a API de roteamento (`route_on_ssn` em `POST /api/routing/gt-routes`) e o padrão é `false`. Quando habilitada, a mudança se aplica apenas a uma rota atualmente endereçada por GT que carrega um SSN não zero (ITU-T Q.714 §2.4).

Aplicação do Contador de Saltos (XUDT / LUDT)

Mensagens XUDT e LUDT carregam um **contador de saltos SCCP** que protege contra loops de roteamento entre nós de retransmissão SCCP. Em cada salto de trânsito, o STP:

1. **Decrementa** o contador de saltos na mensagem encaminhada (Q.714 §2.4).
2. Se decrementar levar o contador a **zero**, a mensagem é **descartada** em vez de ser encaminhada (Q.714 §2.8.6).
3. Um descarte incrementa `m3ua_stp_routing_failures_total{reason="hcounter_violation"}`.
4. Se a mensagem original solicitou retorno em caso de erro, uma mensagem `XUDTS`/`LUDTS` de retorno é gerada de volta para o originador (veja abaixo).

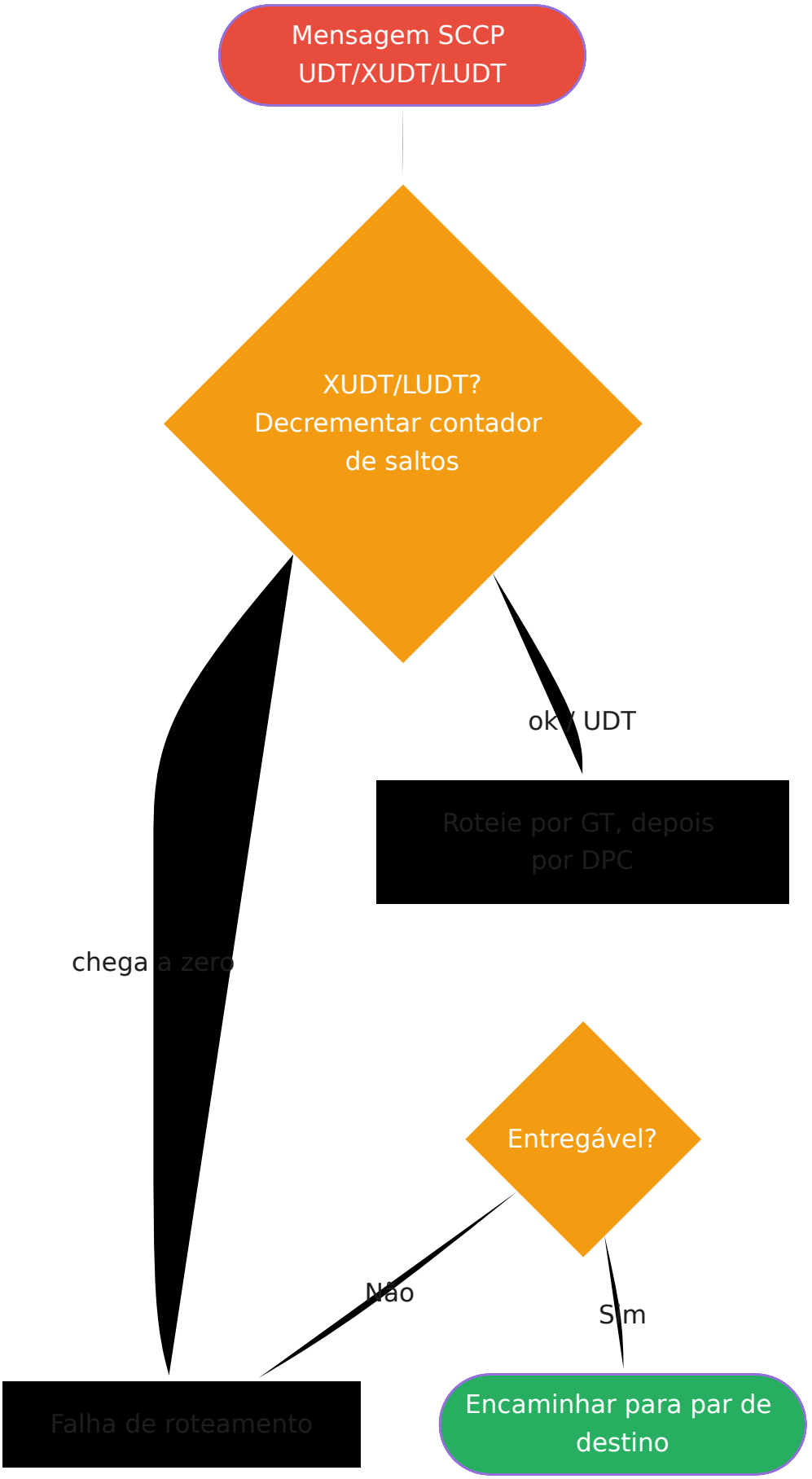
UDT não tem contador de saltos e é encaminhada inalterada nesta etapa.

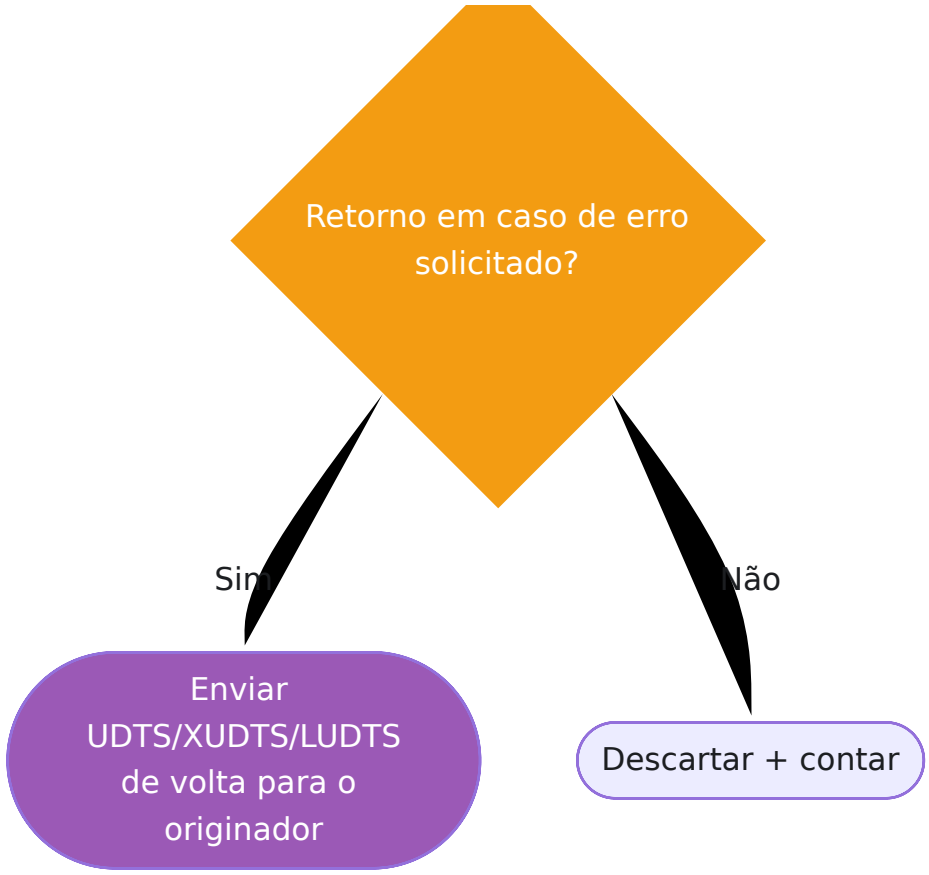
Retorno de Mensagem em Caso de Falha de Roteamento (UDTS / XUDTS / LUDTS)

Quando uma mensagem sem conexão não pode ser entregue (sem rota, destino proibido, contador de saltos esgotado, etc.) **e o originador definiu a opção de retorno em caso de erro**, o STP constrói uma mensagem de retorno SCCP e a envia de volta em direção ao originador (ITU-T Q.714 §4.2):

Mensagem Original	Retornada como
UDT (0x09)	UDTS (0x0A)
XUDT (0x11)	XUDTS (0x12)
LUDT (0x13)	LUDTS (0x14)

A causa de retorno carregada na mensagem reflete a razão da falha (por exemplo, uma falha de MTP para um destino proibido, ou uma violação de contador de saltos para um contador esgotado). Se o originador **não** solicitou retorno em caso de erro, a mensagem é simplesmente descartada e contada.


Mensagem SCCP
UDT/XUDT/LUDT
XUDT/LUDT?
Decrementar contador
de saltos
ok / UDT
Roteie por GT, depois
por DPC
chega a zero
Entregável?
Não
Sim
Falha de roteamento
Encaminhar para par de
destino



Estado de Destino (Gerenciamento de Rota de Sinalização)

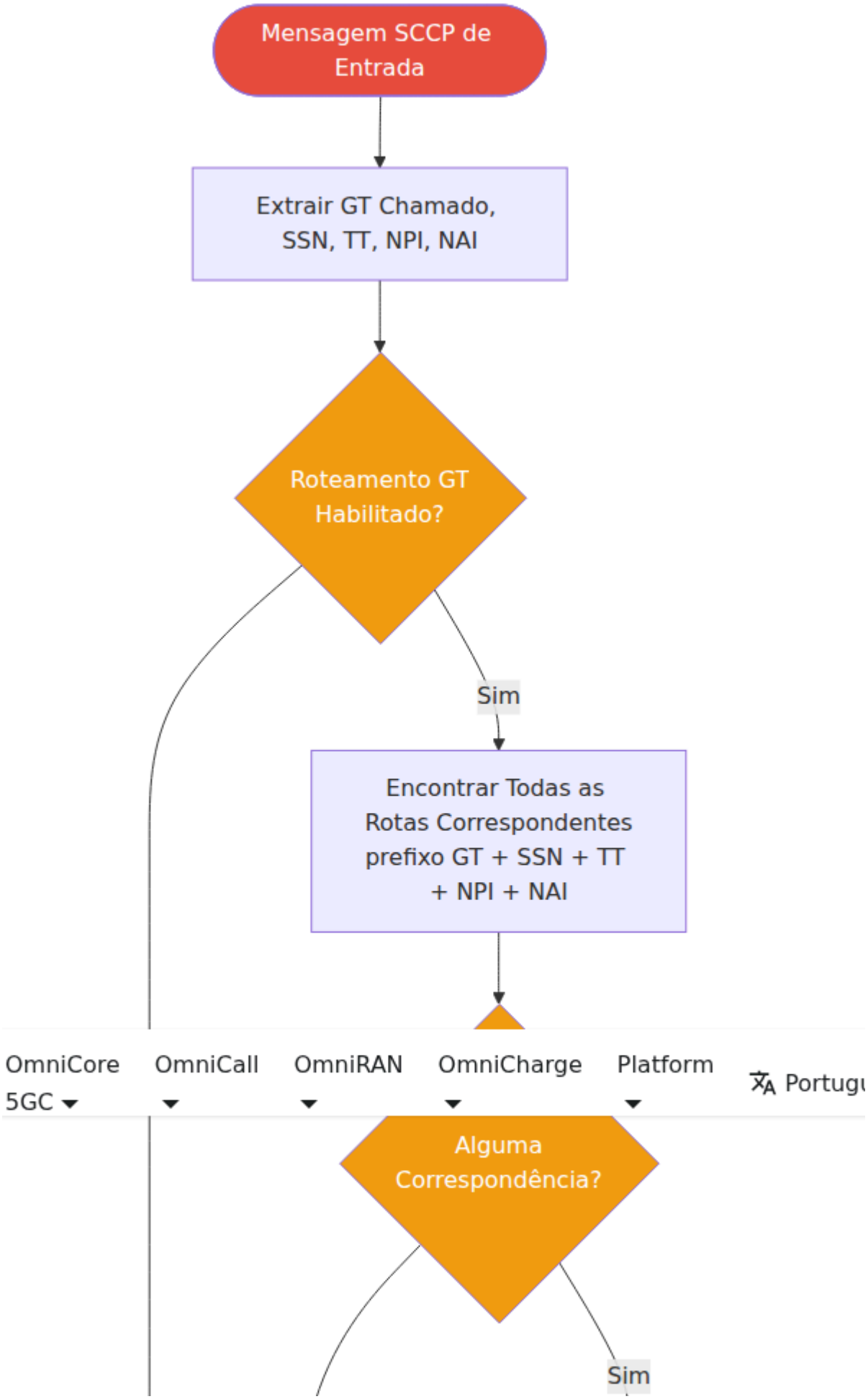
O STP rastreia a disponibilidade de cada código de ponto de destino usando mensagens de gerenciamento de rota de sinalização MTP3, conforme ITU-T Q.704 §13. O estado é mantido na tabela Mnesia `m3ua_dest_state`; a ausência de uma linha significa que o destino é **permitido** (o padrão).

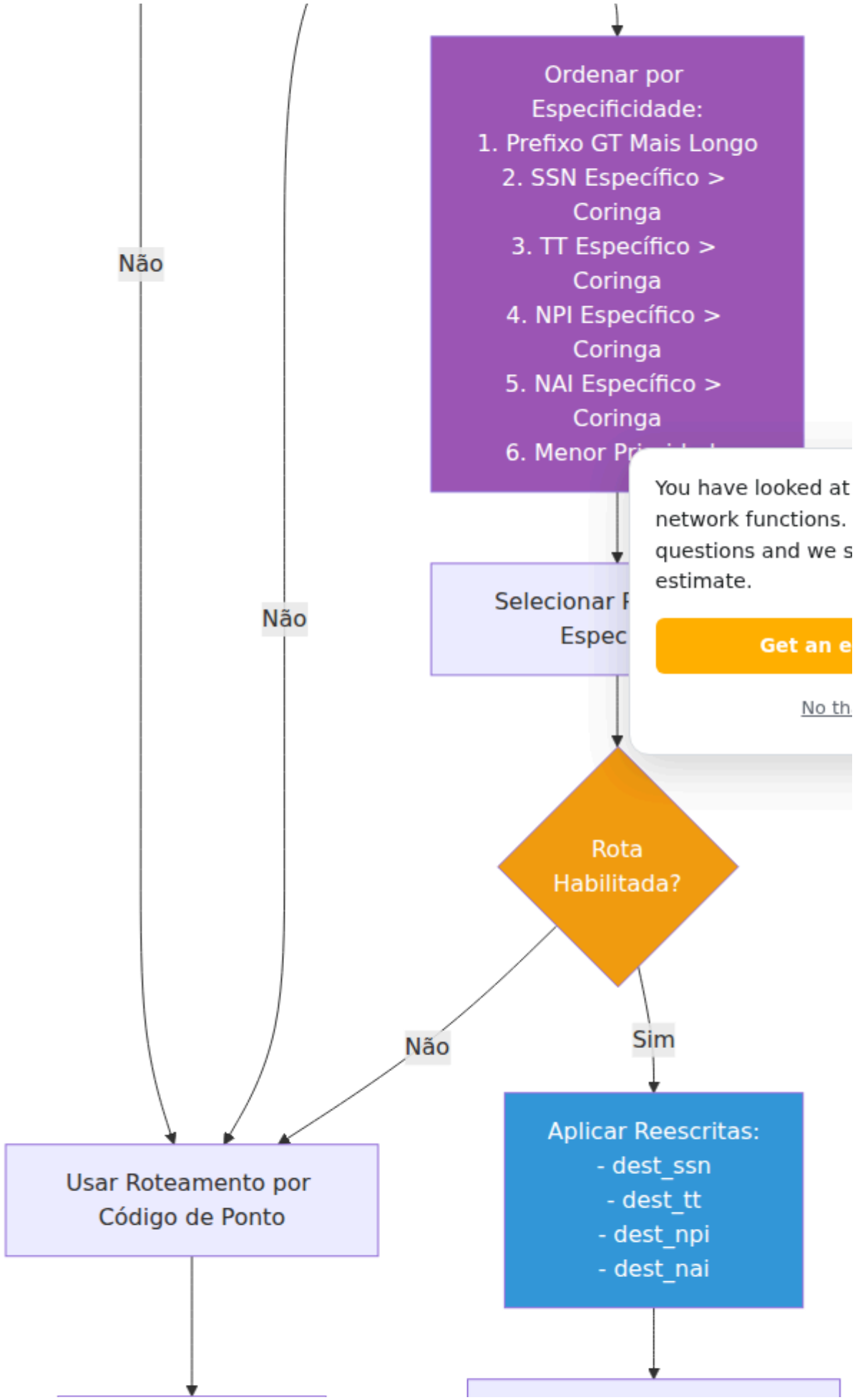
Manipulação de TFP / TFA / TFR

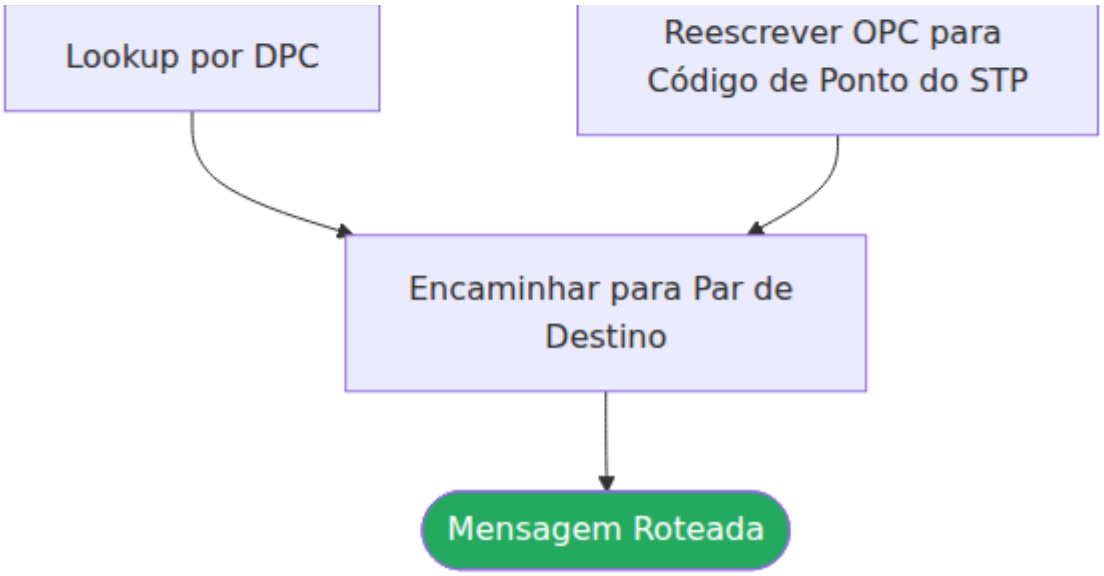
Mensagem	Significado	Efeito no estado do STP
TFP (Transferência Proibida)	Um destino não é mais acessível através desta rota	Destino marcado como `:prohibited`
TFA (Transferência Permitida)	Um destino é acessível novamente	Estado do destino limpo (de volta ao permitido)
TFR (Transferência Restrita)	Um destino deve ser usado apenas quando não houver alternativa	Destino marcado como `:restricted`

Efeito no Roteamento

- Durante uma pesquisa de Código de Ponto, o STP primeiro verifica o estado do destino. Um destino marcado como **transferência proibida** é ignorado. A pesquisa retorna `{:error, :prohibited}` e a mensagem não é roteada para lá (Q.704 §13.2.3). Isso incrementa `m3ua_stp_routing_failures_total{reason="prohibited"}`.
- **DAUD** (Auditoria de Destino) solicitações são respondidas a partir deste estado real em vez de com um "disponível" genérico: destinos acessíveis são relatados com **DAVA**, e destinos inacessíveis ou proibidos com **DUNA** (RFC 4666 §3.4.3).


Mensagem SCCP de Entrada
Extrair GT Chamado, SSN, TT, NPI, NAI
Roteamento GT Habilitado?
Sim
Encontrar Todas as Rotas Correspondentes prefixo GT + SSN + TT + NPI + NAI
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
Portugu
Alguma Correspondência?
Sim


Não
Não
Ordenar por Especificidade:
1. Prefixo GT Mais Longo
2. SSN Específico > Coringa
3. TT Específico > Coringa
4. NPI Específico > Coringa
5. NAI Específico > Coringa
6. Menor Pri
You have looked at
network functions.
questions and we s
estimate.
Get an e
No th
Selecionar F
Espec
Rota Habilitada?
Não
Sim
Usar Roteamento por Código de Ponto
Aplicar Reescritas:
- dest_ssn
- dest_tt
- dest_npi
- dest_nai



Filtragem de Gateway (Firewall SCCP)

A filtragem de gateway permite que o STP **permita ou negue tráfego de trânsito** antes de ser roteado. Sem ela, o STP encaminha cada mensagem incondicionalmente. Com ela, os operadores podem bloquear sinalizações indesejadas no gateway. Este é um controle de segurança central do SS7.

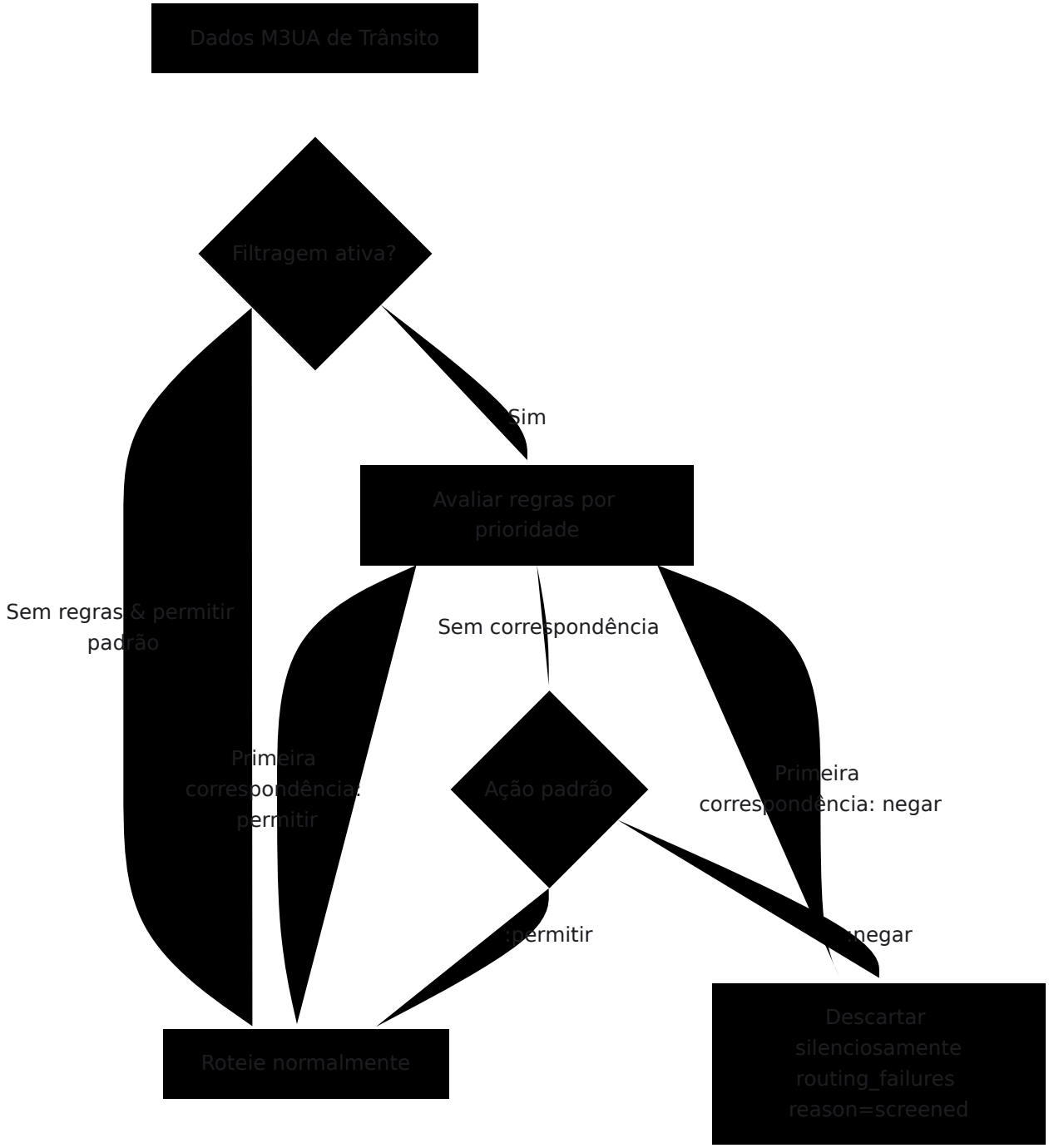
Cada mensagem de dados M3UA de trânsito é avaliada contra um conjunto ordenado de regras. Uma regra corresponde a qualquer combinação dos atributos abaixo (um atributo não definido é um coringa):

Atributo	Corresponde
Par de origem	O par de onde a mensagem chegou
OPC	Código de Ponto de Origem
DPC	Código de Ponto de Destino
Indicador de Serviço	SI MTP3
Prefixo GTA da Parte Chamadora	Título Global da Parte Chamadora SCCP (correspondência de prefixo)
Prefixo GTA da Parte Chamada	Título Global da Parte Chamada SCCP (correspondência de prefixo)
SSN	Número de Subsistema SCCP

A **primeira regra habilitada correspondente** (menor valor de `priority` primeiro) decide o resultado: `permitir` ou `negar`. Se nenhuma regra corresponder, a **ação padrão** configurável se aplica (`config :omniss7, :screening_default_action`, padrão `:allow`).

Uma mensagem negada é **descartada silenciosamente** e contada como `m3ua_stp_routing_failures_total{reason="screened"}`. Nenhum UDTS/retorno é gerado, porque um firewall não deve vazar o bloqueio de volta para o remetente.

Comportamento



A triagem é totalmente ignorada (custo zero no caminho quente) quando nenhuma regra está configurada **e** a ação padrão é `:allow`; ela se torna ativa assim que uma regra existe ou a ação padrão é definida como `:deny`.

Ação Padrão

Parâmetro	Tipo	Padrão	Descrição
`screening_default_action`	`:allow` | `:deny`	`:allow`	Ação aplicada quando nenhuma regra corresponde. Defina como `:deny` para uma postura de negação padrão (lista de permissão).

```
config :omniss7,
  # Bloquear tudo, exceto o que as regras permitem explicitamente.
  screening_default_action: :deny
```

As regras são armazenadas na tabela Mnesia `ss7_screen_rule` e gerenciadas programaticamente (cada regra possui uma prioridade, os atributos de correspondência acima, uma ação `allow`/`deny` e uma flag `enabled`). Consulte a Referência de Configuração para o parâmetro de ação padrão.

Recursos de Gerenciamento de Rota

Desabilitando Rotas

As rotas podem ser temporariamente desabilitadas sem serem excluídas. Isso é útil para testes, manutenção ou gerenciamento de tráfego.

Flag Habilitada

Tanto as rotas de Código de Ponto quanto as de Título Global suportam uma flag `enabled` opcional:

```
config :omniss7,
  m3ua_routes: [
    # Rota ativa
    %{
      dest_pc: 100,
      peer_id: 1,
      priority: 1,
      network_indicator: :international,
      enabled: true  # Rota está ativa (padrão se omitido)
    },

    # Rota desabilitada (não avaliada durante o roteamento)
    %{
      dest_pc: 200,
      peer_id: 2,
      priority: 1,
      network_indicator: :international,
      enabled: false  # Rota está desabilitada
    }
  ],

  m3ua_gt_routes: [
    # Rota GT desabilitada
    %{
      gt_prefix: "44",
      peer_id: 1,
      priority: 1,
      description: "Números do Reino Unido - temporariamente
desabilitados",
      enabled: false
    }
  ]
```

Comportamento Padrão:

- Se `enabled` não for especificado, as rotas têm como padrão `enabled: true`
- Rotas desabilitadas são completamente ignoradas durante a busca de rotas
- Use a interface da Web para alternar rotas on/off sem editar a configuração

Casos de Uso:

- Testando mudanças no fluxo de tráfego
- Janelas de manutenção temporárias
- Testes A/B de diferentes caminhos de roteamento
- Implantação gradual de novas rotas

Rotas DROP: Prevenindo Loops de Roteamento

Rotas DROP (com `peer_id: 0`) descartam silenciosamente o tráfego em vez de encaminhá-lo. Isso previne loops de roteamento e permite filtragem avançada de tráfego.

Configurando Rotas DROP

```
config :omniss7,
  m3ua_routes: [
    # Rota DROP para código de ponto específico
    %{
      dest_pc: 999,
      peer_id: 0,        # peer_id=0 significa DROP
      priority: 1,
      network_indicator: :international
    }
  ],

  m3ua_gt_routes: [
    # Rota DROP para prefixo GT
    %{
      gt_prefix: "999",
      peer_id: 0,        # peer_id=0 significa DROP
      priority: 99,
      description: "Bloquear intervalo de teste"
    }
  ]
```

Como Funcionam as Rotas DROP

Quando uma mensagem corresponde a uma rota DROP:

1. O mecanismo de roteamento identifica `peer_id: 0`

2. A mensagem é **silenciosamente descartada** (não é encaminhada)
3. Um **log INFO** é gerado: `"DROP route matched for DPC 999"` ou `"DROP route matched for GT 999"`
4. A busca de roteamento retorna `{:error, :dropped}`

Importante: O tráfego descartado é registrado no nível INFO para monitoramento e solução de problemas.

Caso de Uso Comum: Lista de Permissão de Prefixos

Um dos usos mais poderosos das rotas DROP é a **lista de permissão de prefixos** - permitindo apenas números específicos dentro de um grande intervalo, enquanto bloqueia todos os outros.

O Padrão:

1. Crie uma rota DROP para todo o prefixo com um **número de prioridade alto** (por exemplo, 99)
2. Crie rotas de permissão específicas para números individuais com **números de prioridade baixos** (por exemplo, 1)
3. Como os números de prioridade mais baixos são avaliados primeiro, as rotas permitidas correspondem antes da rota DROP
4. Qualquer número não explicitamente permitido é capturado pela rota DROP

Cenário de Exemplo:

Você tem um prefixo GT `1234` que representa um intervalo de 10.000 números (1234000000 - 1234999999), mas você só quer roteá-los para 3 números específicos: `1234567890`, `1234555000` e `1234111222`.

```
config :omniss7,
  m3ua_gt_routes: [
    # Rota DROP com número de prioridade ALTA (avaliada por
último)
    %{
      gt_prefix: "1234",
      peer_id: 0,              # DROP
      priority: 99,            # Número alto = baixa prioridade =
avaliada por último
      description: "Bloquear todos os 1234* exceto números na
lista de permissão"
    },

    # Rotas de permissão específicas com números de prioridade
BAIXA (avaliadas primeiro)
    %{
      gt_prefix: "1234567890",
      peer_id: 1,              # Rota para o peer 1
      priority: 1,             # Número baixo = alta prioridade =
avaliada primeiro
      description: "Número permitido 1"
    },
    %{
      gt_prefix: "1234555000",
      peer_id: 1,
      priority: 1,
      description: "Número permitido 2"
    },
    %{
      gt_prefix: "1234111222",
      peer_id: 1,
      priority: 1,
      description: "Número permitido 3"
    }
  ]
```

Comportamento de Roteamento:

GT de Entrada	Rotas Correspondentes	Rota Selecionada	Ação
1234567890	✅ "1234567890" (prioridade 1) ✅ "1234" DROP (prioridade 99)	"1234567890" (mais específica, maior prioridade)	**Roteada para o peer 1**
1234555000	✅ "1234555000" (prioridade 1) ✅ "1234" DROP (prioridade 99)	"1234555000" (mais específica, maior prioridade)	**Roteada para o peer 1**
1234111222	✅ "1234111222" (prioridade 1) ✅ "1234" DROP (prioridade 99)	"1234111222" (mais específica, maior prioridade)	**Roteada para o peer 1**
1234999999	✅ "1234" DROP (prioridade 99)	"1234" DROP (única correspondência)	**Descartada + registrada**
1234000000	✅ "1234" DROP (prioridade 99)	"1234" DROP (única correspondência)	**Descartada + registrada**

Resultado:

- ✅ Apenas 3 números específicos são roteados para o peer 1
- ✅ Todos os outros números `1234*` são silenciosamente descartados
- ✅ Todo o tráfego descartado é registrado para monitoramento

Logs Gerados:

```
[INFO] DROP route matched for GT 1234999999
[INFO] DROP route matched for GT 1234000000
```

Rotas DROP para Códigos de Ponto

O mesmo padrão de lista de permissão funciona para roteamento de Código de Ponto:

```
config :omniss7,
  m3ua_routes: [
    # DROP todo o intervalo /8 (64 códigos de ponto: 1000-1063)
    %{
      dest_pc: 1000,
      peer_id: 0,
      priority: 99,
      mask: 8,
      network_indicator: :international
    },

    # Permitir PCs específicos
    %{dest_pc: 1010, peer_id: 1, priority: 1, network_indicator:
:international},
    %{dest_pc: 1020, peer_id: 1, priority: 1, network_indicator:
:international},
    %{dest_pc: 1030, peer_id: 1, priority: 1, network_indicator:
:international}
  ]
```

Resultado: Apenas os PCs 1010, 1020 e 1030 são roteados. Todos os outros PCs no intervalo de 1000-1063 são descartados.

Monitorando Rotas DROP

Verifique os Logs:

```
# Monitore o tráfego descartado
tail -f logs/app.log | grep "DROP route matched"

# Saída esperada:
[INFO] DROP route matched for GT 1234999999
[INFO] DROP route matched for DPC 1050
```

Via Interface da Web:

- Navegue até a aba **Logs do Sistema**

- Filtre por nível `INFO`
- Pesquise por "DROP route matched"

Melhores Práticas:

1. ⚠ Monitore os logs regularmente para garantir que as rotas DROP não estejam bloqueando tráfego legítimo
2. □ Use campos `description` descritivos para documentar por que as rotas são descartadas
3. □ Use números de prioridade altos (90-99) para rotas DROP para garantir que sejam rotas de captura
4. □ Teste o comportamento da rota DROP antes de implantar em produção
5. □ Configure alertas para aumentos inesperados no tráfego descartado

---

Roteamento Avançado: Roteamento e Reescrita Baseados em SSN

Números de Subsystema (SSN)

Os Números de Subsystema identificam a camada de aplicação:

- **SSN 6**: HLR (Home Location Register)
- **SSN 7**: VLR (Visitor Location Register)
- **SSN 8**: MSC (Mobile Switching Center) / SMSC (SMS Center)
- **SSN 9**: GMLC (Gateway Mobile Location Center)

Exemplo de Roteamento Baseado em SSN

Roteie o tráfego SMS para diferentes HLRs com base no prefixo do número:

```
m3ua_gt_routes: [
  # Roteie SMS para números do Reino Unido para HLR do Reino
Unido, reescreva SSN de 8 (SMSC) para 6 (HLR)
  %{
    gt_prefix: "44",
    source_ssn: 8,                    # Correspondência com SSN de
entrada 8 (SMSC)
    peer_id: 1,
    dest_ssn: 6,                      # Reescreva para SSN 6 (HLR)
    priority: 1,
    description: "SMS do Reino Unido para HLR"
  },

  # Roteie tráfego de voz para números do Reino Unido (SSN 6) sem
reescrita
  %{
    gt_prefix: "44",
    source_ssn: 6,                    # Correspondência com SSN de
entrada 6 (HLR)
    peer_id: 1,
    dest_ssn: nil,                    # Sem reescrita de SSN
    priority: 1,
    description: "Tráfego de voz do Reino Unido"
  }
]
```

Testando a Configuração de Roteamento STP

Após configurar peers e rotas, verifique sua configuração:

1. Verifique o Status do Peer

Via Interface da Web:

- Navegue até http://localhost
- Verifique a página de Status M3UA

- Verifique se os peers mostram **Status: ATIVO**

Via Console IEx:

```
# Obtenha todos os status dos peers
SCTP.SocketHandler.get_peers_status()

# Saída esperada:
# [
#   %{peer_id: 1, name: "Partner_STP_West", status: :active,
point_code: 100, ...},
#   %{peer_id: 2, name: "Local_HLR", status: :active, point_code:
200, ...}
# ]
```

2. Teste o Roteamento de Código de Ponto

```
# Envie uma mensagem M3UA de teste para DPC 100
test_payload = <<1, 2, 3, 4>>  # Payload fictício
SCTP.SocketHandler.route_by_pc(100, test_payload, 0)

# Verifique os logs para a decisão de roteamento
# Log esperado: "Routing message: OPC=... -> DPC=100 via peer 1"
```

3. Teste o Roteamento de Título Global

```
# Procure a rota GT manualmente
M3UARouting.lookup_peer_by_gt("447712345678")

# Retorna {:ok, peer, rewrites}. `peer` é o registro m3ua_peer de
13 elementos;
# `rewrites` é um MAP de reescritas de indicador (nil = manter
original):
# {:ok, {:m3ua_peer, 3, "UK_Mobile_Peer", ...},
#       %{dest_ssn: nil, dest_tt: nil, dest_npi: nil, dest_nai:
nil}}

# Procure a rota GT com SSN (uma rota que reescreve SSN 8 → 6)
M3UARouting.lookup_peer_by_gt("555881234567", 8)

# A reescrita de SSN aparece no mapa de reescritas, NÃO como um
inteiro isolado:
# {:ok, {:m3ua_peer, 4, "SMS_HLR_Peer", ...},
#       %{dest_ssn: 6, dest_tt: nil, dest_npi: nil, dest_nai:
nil}}
```

4. Monitore Métricas de Roteamento

Acesse as métricas do Prometheus em `/metrics`

Métricas chave:

```
# Mensagens recebidas por peer
m3ua_stp_messages_received_total{peer_name="Partner_STP_West",point_c
1523

# Mensagens enviadas por peer
m3ua_stp_messages_sent_total{peer_name="Local_HLR",point_code="200"}

# Falhas de roteamento (por motivo)
m3ua_stp_routing_failures_total{reason="no_route"} 5
m3ua_stp_routing_failures_total{reason="no_gt_route"} 2
m3ua_stp_routing_failures_total{reason="no_active_peer"} 1
m3ua_stp_routing_failures_total{reason="prohibited"} 3
m3ua_stp_routing_failures_total{reason="hcounter_violation"} 4
m3ua_stp_routing_failures_total{reason="screened"} 7
```

Métricas e Monitoramento STP

Métricas Disponíveis

Métricas de Tráfego por Peer:

- `m3ua_stp_messages_received_total` - Total de mensagens recebidas de cada peer
  - Rótulos: `peer_name`, `point_code`
- `m3ua_stp_messages_sent_total` - Total de mensagens encaminhadas para cada peer
  - Rótulos: `peer_name`, `point_code`

Métricas de Falha de Roteamento:

- `m3ua_stp_routing_failures_total` - Contagem de falhas de roteamento por motivo
  - Rótulos: `reason`

`reason`	Significado
`no_route`	Nenhuma rota de Código de Ponto encontrada para o destino
`no_gt_route`	Nenhuma rota de Título Global encontrada, e o roteamento de PC também falhou
`no_active_peer`	Uma rota corresponde, mas seu peer é desconhecido/não ativo
`prohibited`	O destino é proibido de transferência (um TFP foi recebido, Q.704 §13)
`hcounter_violation`	O contador de salto SCCP atingiu zero em um XUDT/LUDT (Q.714 §2.8.6)

Interpretação das Métricas

- **Contagens de mensagens altas**: Indica fluxo de tráfego ativo
- **Falhas de roteamento**: Indica rotas ausentes, um destino inacessível ou uma mensagem em loop
  - `no_route`: Nenhuma rota de Código de Ponto encontrada para o destino
  - `no_gt_route`: Nenhuma rota de Título Global encontrada, e o roteamento de PC também falhou
  - `no_active_peer`: A rota existe, mas o peer de destino está fora/indisponível
  - `prohibited`: Destino marcado como proibido de transferência pela gestão de rotas de sinalização
  - `hcounter_violation`: Contador de salto esgotado; a mensagem foi descartada (e retornada se o retorno em caso de erro foi definido)

Solução de Problemas com Métricas

Cenário: Nenhum tráfego chegando ao destino

1. Verifique se as mensagens estão sendo recebidas:

```
m3ua_stp_messages_received_total{peer_name="Source_Peer"} > 0
```

2. Verifique se as mensagens estão sendo enviadas:

```
m3ua_stp_messages_sent_total{peer_name="Dest_Peer"} > 0
```

3. Verifique se há falhas de roteamento:

```
m3ua_stp_routing_failures_total{reason="no_route"} > 0
```

Solução: Se as falhas de roteamento forem altas, adicione rotas ausentes na configuração.

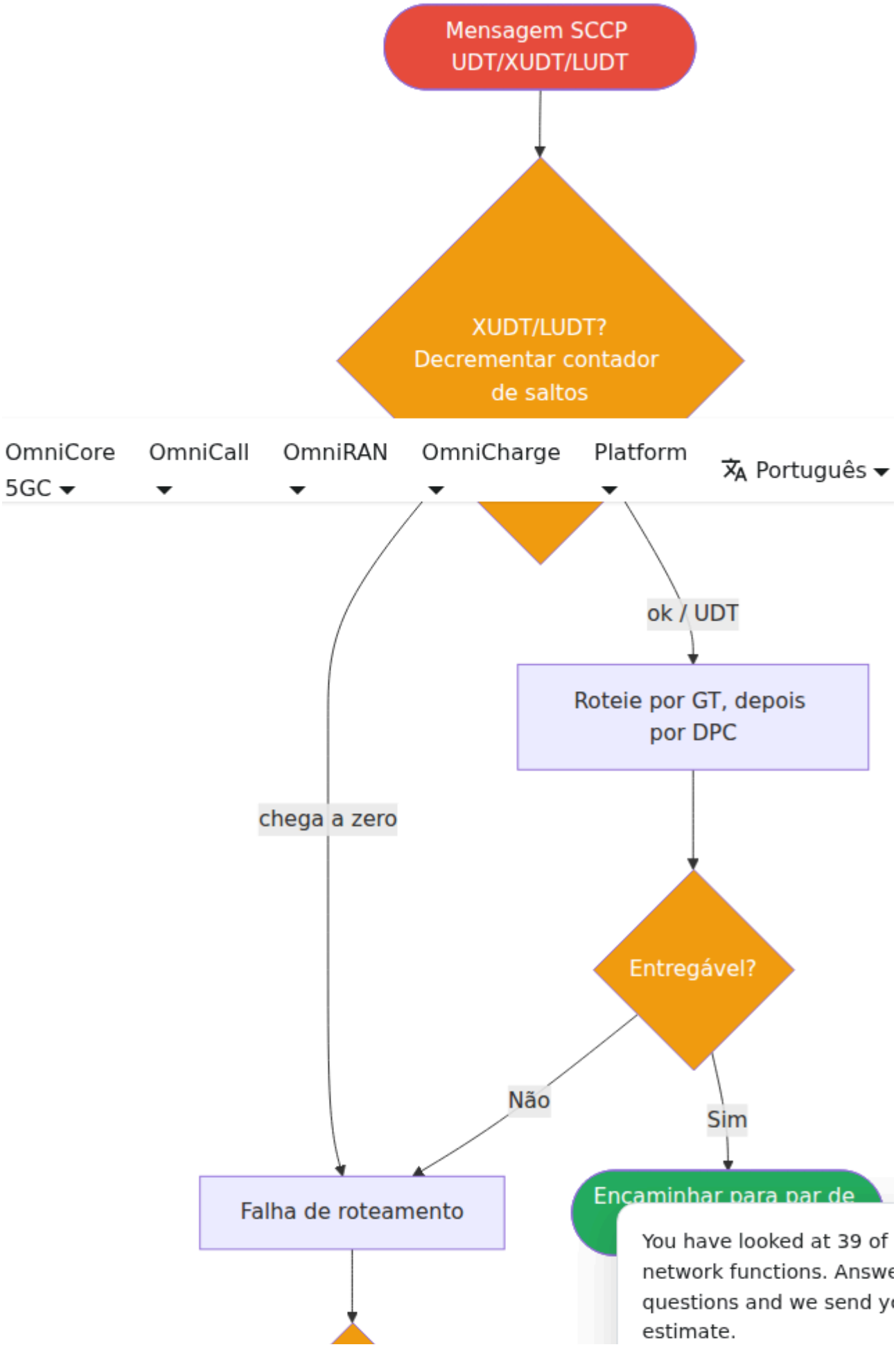
Monitoramento de Status de Peer M3UA

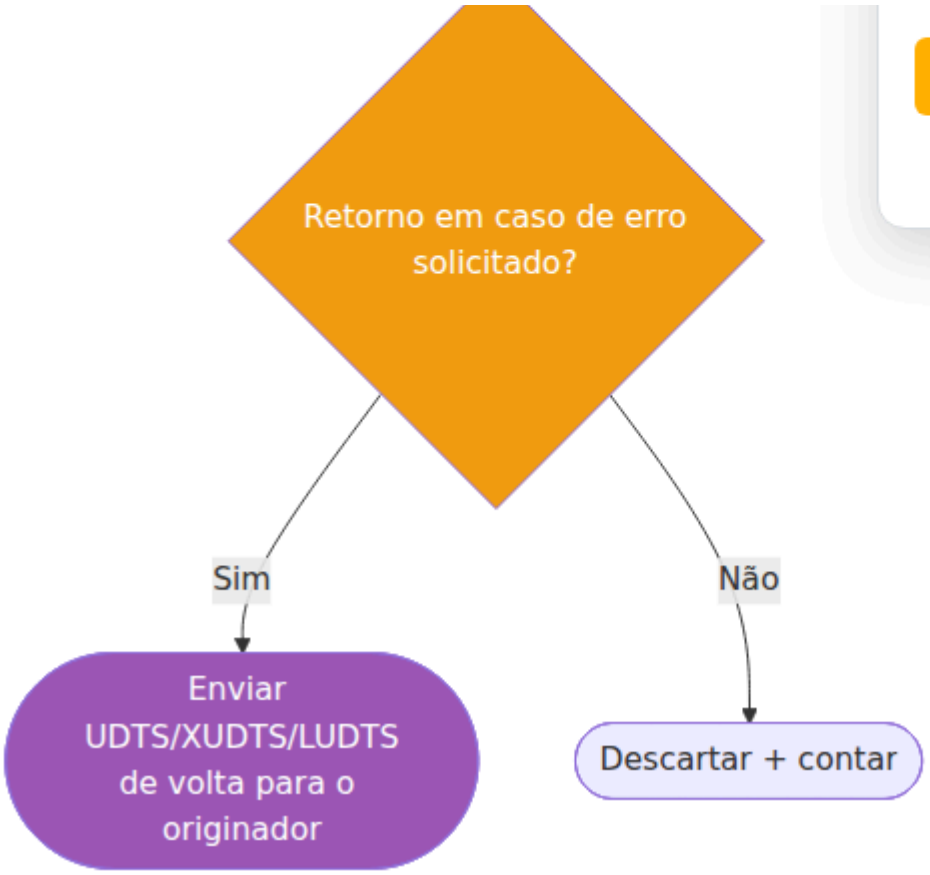
Entendendo M3UA

M3UA (MTP3 User Adaptation Layer) é um protocolo que permite que a sinalização SS7 seja transportada sobre redes IP usando SCTP.

Estados de Conexão M3UA

As conexões M3UA progridem através de vários estados:


Mensagem SCCP
UDT/XUDT/LUDT
XUDT/LUDT?
Decrementar contador
de saltos
OmniCore
5GC
OmniCall
OmniRAN
OmniCharge
Platform
Português
ok / UDT
Roteie por GT, depois
por DPC
chega a zero
Entregável?
Não
Sim
Falha de roteamento
Encaminhar para par de
You have looked at 39 of
network functions. Answe
questions and we send y
estimate.



Get an estima
No thanks

Descrições dos Estados:

- **DOWN** - Sem conexão SCTP
- **CONNECTING** - Conexão SCTP em andamento
- **ASPUP_SENT** - Aguardando reconhecimento de ASPUP
- **INACTIVE** - ASP está ativo, mas não está transportando tráfego ativamente (pode transitar para ATIVO via ASPAC)
- **ASPAC_SENT** - Aguardando reconhecimento de ASPAC
- **ACTIVE** - Pronto para tráfego, totalmente operacional
- **ASPDOWN_SENT** - Desligamento gracioso em andamento

Monitorando Peers M3UA via Interface da Web

A interface da Web fornece monitoramento em tempo real das conexões de peers M3UA.

Acessando a Página de Status M3UA:

1. Navegue até a página inicial da interface da Web
2. Clique em "Status M3UA" no menu de navegação

3. A página é atualizada automaticamente a cada segundo

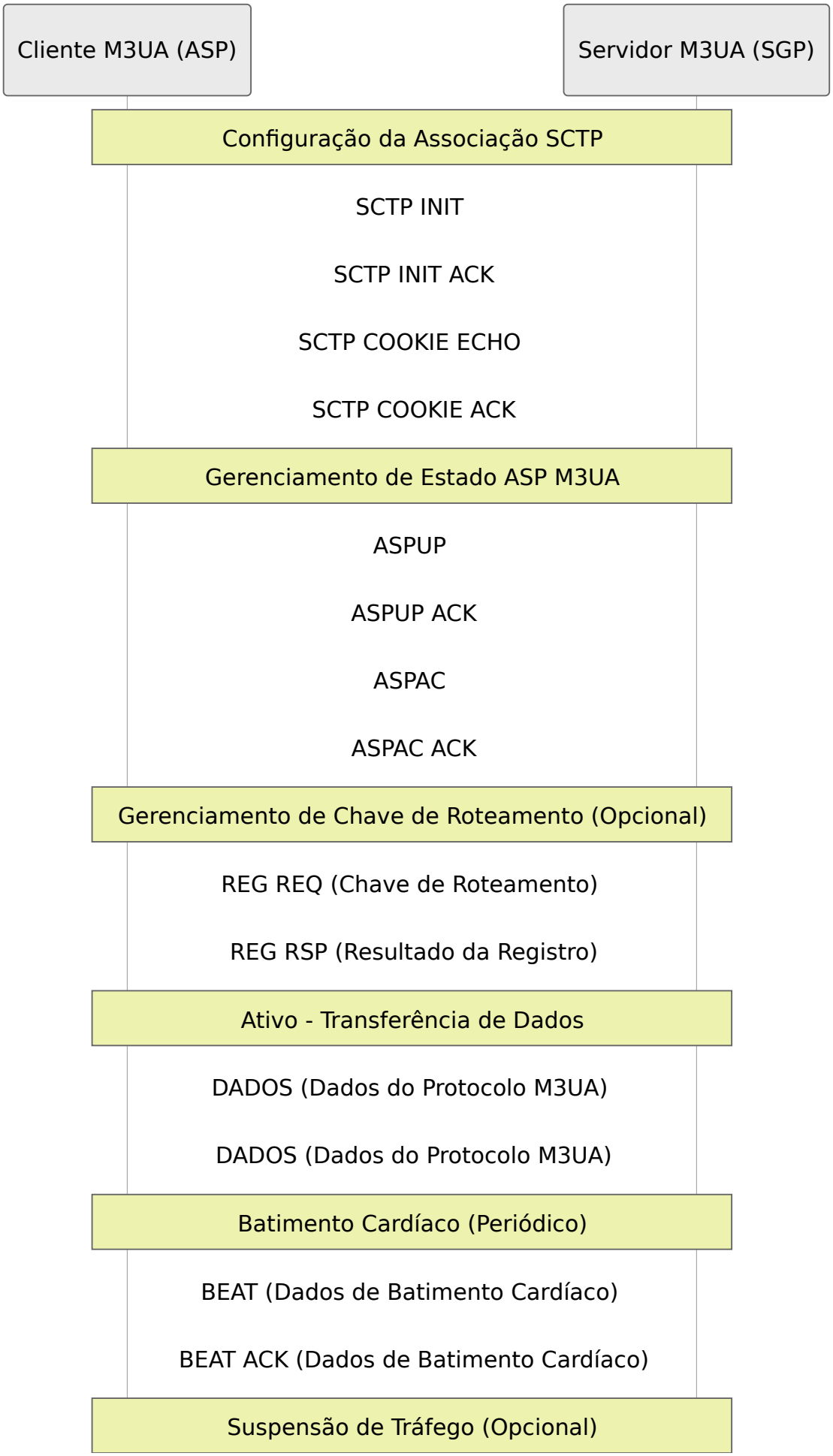
Tabela de Status M3UA:

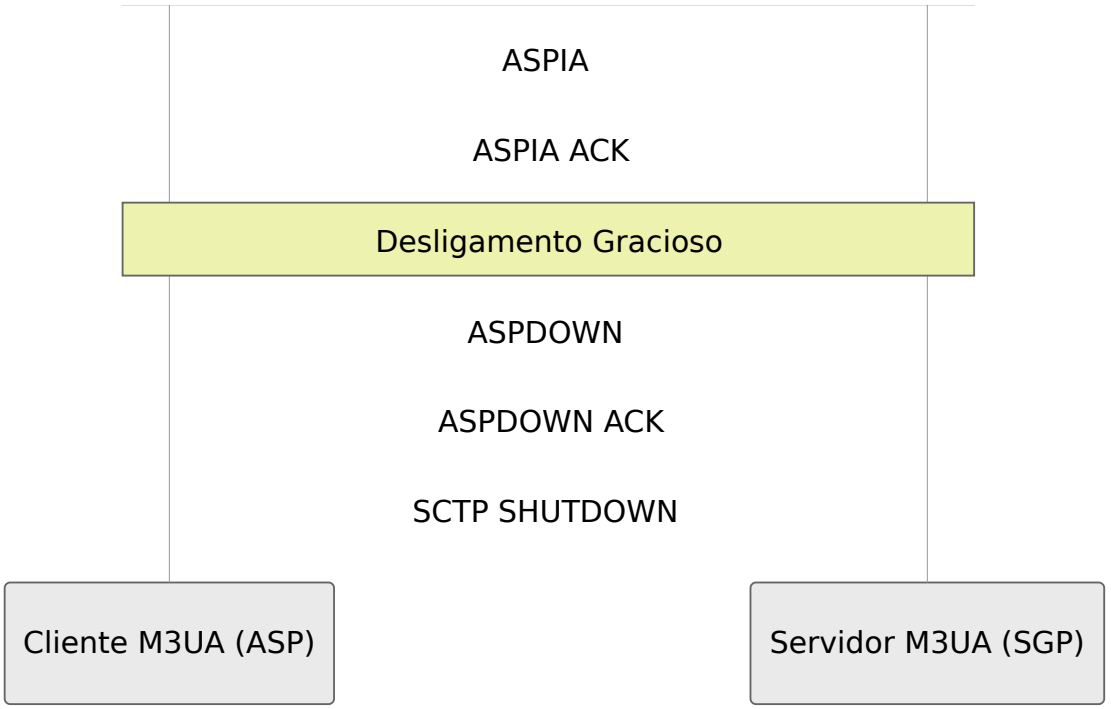
Coluna	Descrição
Nome	Nome da conexão (por exemplo, `testASP`)
PID	Identificador do processo
Status	ATIVO (verde) ou INATIVO (vermelho)
Estado ASP	Estado atual do M3UA (por exemplo, ATIVO, INATIVO)
Assoc/SCTP	Estado da associação SCTP
Local	IP:Porta local
Remoto	IP:Porta remota
RC	ID do Contexto de Roteamento

Indicadores de Status:

- **Verde (ATIVO)** - Conexão está ativa e saudável
- **Vermelho (INATIVO)** - Conexão está inativa ou indisponível
- **Estado ASP** - Mostra o estado atual da conexão M3UA
- **Assoc/SCTP** - Mostra o status da associação SCTP

Fluxo de Mensagens M3UA


Cliente M3UA (ASP)
Servidor M3UA (SGP)
Configuração da Associação SCTP
SCTP INIT
SCTP INIT ACK
SCTP COOKIE ECHO
SCTP COOKIE ACK
Gerenciamento de Estado ASP M3UA
ASPUP
ASPUP ACK
ASPAC
ASPAC ACK
Gerenciamento de Chave de Roteamento (Opcional)
REG REQ (Chave de Roteamento)
REG RSP (Resultado da Registro)
Ativo - Transferência de Dados
DADOS (Dados do Protocolo M3UA)
DADOS (Dados do Protocolo M3UA)
Batimento Cardíaco (Periódico)
BEAT (Dados de Batimento Cardíaco)
BEAT ACK (Dados de Batimento Cardíaco)
Suspensão de Tráfego (Opcional)



Suporte a Mensagens M3UA RFC 4666

OmniSS7 implementa o conjunto completo de tipos de mensagens M3UA definidos na RFC 4666:

Classe	Tipo	Mensagem	Direção	Comportamento
0 (MGMT)	0	ERR	Ambos	Registra código de erro e informações de diagnóstico
0 (MGMT)	1	NTFY	Ambos	Registra notificação
1 (XFER)	1	DADOS	Ambos	Roteia payload via PC ou GT
2 (SSNM)	1	DUNA	Receber	Registra destino indisponível
2 (SSNM)	2	DAVA	Receber	Registra destino disponível
2 (SSNM)	3	DAUD	Receber	Responde conforme o estado do destino: DAVA para PCs acessíveis, DUNA para PCs inacessíveis/proibidos
2 (SSNM)	4	SCON	Receber	Registra congestionamento
2 (SSNM)	5	DUPU	Receber	Registra parte do usuário indisponível
2 (SSNM)	6	DRST	Receber	Registra destino restrito
3 (ASPSM)	1	ASPUP	Ambos	Envia/recebe ACK de ASPUP
3 (ASPSM)	2	ASPDN	Ambos	Envia ACK de ASPDN, derruba rotas de peer

Classe	Tipo	Mensagem	Direção	Comportamento
3 (ASPSM)	3	BEAT	Ambos	Ecoa dados de batimento cardíaco no ACK de BEAT
3 (ASPSM)	4	ASPUP ACK	Receber	Transita peer para ATIVO
3 (ASPSM)	5	ASPDN ACK	Receber	Transita peer para INATIVO
3 (ASPSM)	6	BEAT ACK	Receber	Acknowledges heartbeat
4 (ASPTM)	1	ASPAC	Ambos	Envia/recebe ACK de ASPAC, ativa rotas
4 (ASPTM)	2	ASPIA	Ambos	Envia ACK de ASPIA, transita peer para INATIVO
4 (ASPTM)	3	ASPAC ACK	Receber	Transita peer para ATIVO
4 (ASPTM)	4	ASPIA ACK	Receber	Transita peer para INATIVO
9 (RKM)	1	REG REQ	Receber SGP	Analisa chave de roteamento, responde com REG RSP
9 (RKM)	2	REG RSP	Receber ASP	Registra resultado do registro
9 (RKM)	3	DEREG REQ	Receber SGP	Responde com DEREG RSP
9 (RKM)	4	DEREG RSP	Receber ASP	Registra resultado da desregistrar

Batimento Cardíaco (BEAT/BEAT ACK): Quando o STP recebe uma mensagem BEAT, ele ecoa os dados de batimento cardíaco de volta em um BEAT ACK. Isso permite que os peers detectem falhas de link sem depender apenas de timeouts no nível SCTP.

Gerenciamento de Chave de Roteamento (RKM): Quando um ASP envia um REG REQ, o SGP analisa a chave de roteamento (identificador RK local, contexto de roteamento, modo de tráfego, DPC) e responde com um REG RSP contendo o status do registro. Isso permite o provisionamento dinâmico de rotas sem exigir configuração estática.

ASP Down (ASPDN): Quando o SGP recebe um ASPDN, ele responde com ASPDN ACK, marca o peer como INATIVO e remove as rotas associadas da tabela de roteamento.

ASP Inativo (ASPIA): Quando o SGP recebe um ASPIA, ele responde com ASPIA ACK e transita o peer para INATIVO, pausando o tráfego sem derrubar a associação.

Manipulação do Contexto de Roteamento M3UA

O **Contexto de Roteamento (RC)** é um identificador de 32 bits no M3UA que identifica o contexto do Servidor de Aplicação (AS) em cada ponto final. Compreender como o RC é tratado em diferentes pontos do caminho de sinalização é importante para a configuração correta da rede.

O que é Contexto de Roteamento?

De acordo com RFC 4666 Seção 3.3.1, o Contexto de Roteamento:

- Identifica exclusivamente um Servidor de Aplicação (AS) em um ponto final
- É transportado em mensagens M3UA DATA (tag de parâmetro 6)
- Permite que um Gateway de Sinalização determine qual AS deve receber uma mensagem
- É significativo por salto, não de ponta a ponta

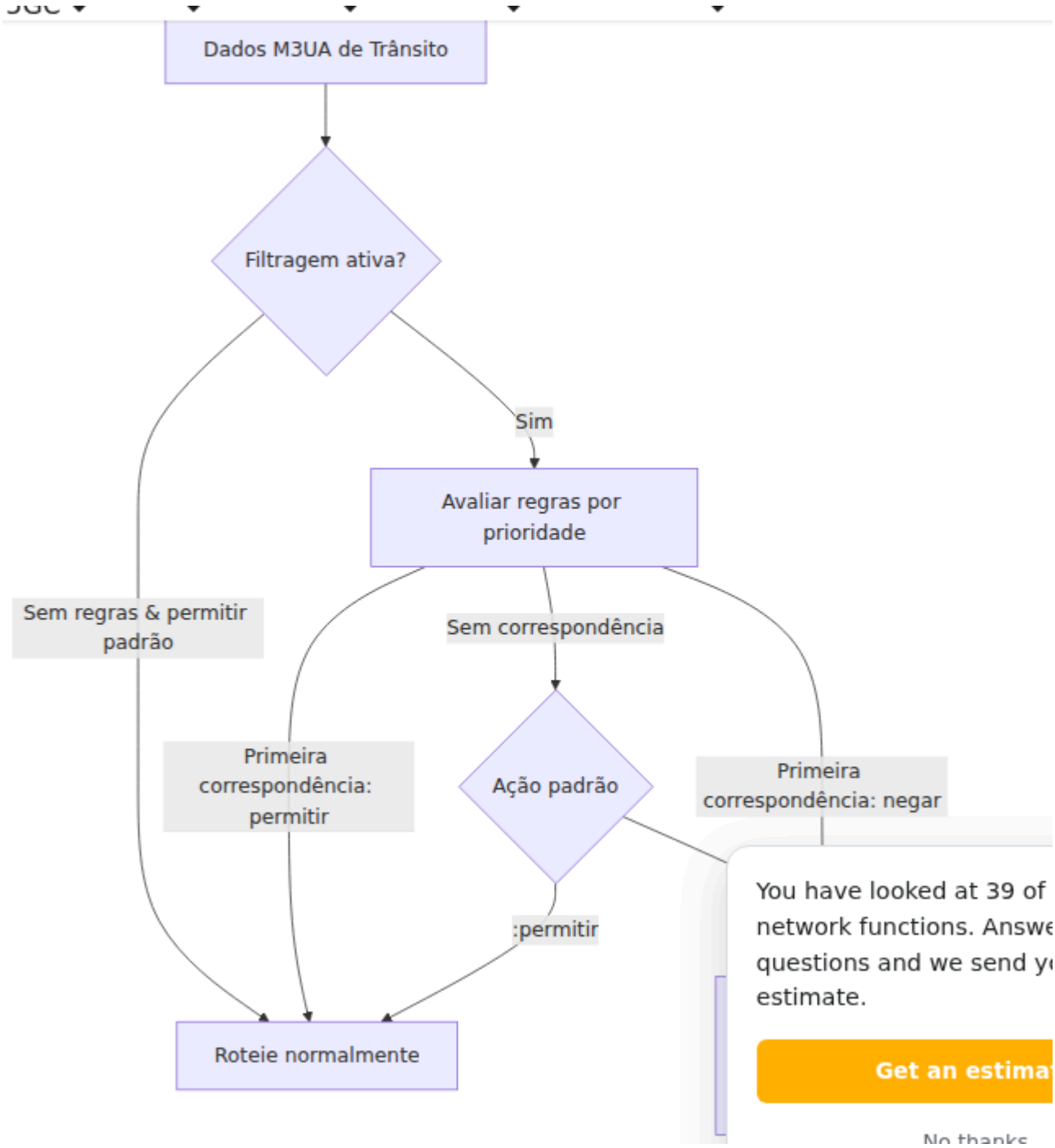
Contexto de Roteamento no Protocolo M3UA



O parâmetro de Contexto de Roteamento (tag 6) aparece nas mensagens M3UA DATA junto com o parâmetro de Dados do Protocolo (tag 528) que contém o payload real MTP3/SCCP.

Como o OmniSS7 Manipula o Contexto de Roteamento

O OmniSS7 manipula o Contexto de Roteamento de maneira diferente, dependendo do componente e da direção da mensagem:



Cliente MAP e Cliente CAP

Para clientes de camada de aplicação (MAP e CAP), o contexto de roteamento é tratado da seguinte forma:

Cenário	Tratamento de RC	Configuração
Resposta a solicitação recebida	Espelhar o RC recebido na resposta	Automático - nenhuma configuração necessária
Solicitação iniciada pelo cliente (SRI, PRN, IDP, etc.)	Usar RC configurado na configuração da aplicação	Definir `routing_context` em `map_client_m3ua` ou `cap_client_m3ua`
Sem RC na solicitação recebida	Omitir RC da resposta	Mantém compatibilidade retroativa
Sem RC configurado para solicitações do cliente	Omitir RC da mensagem de saída	Mantém compatibilidade retroativa

Exemplo de Configuração:

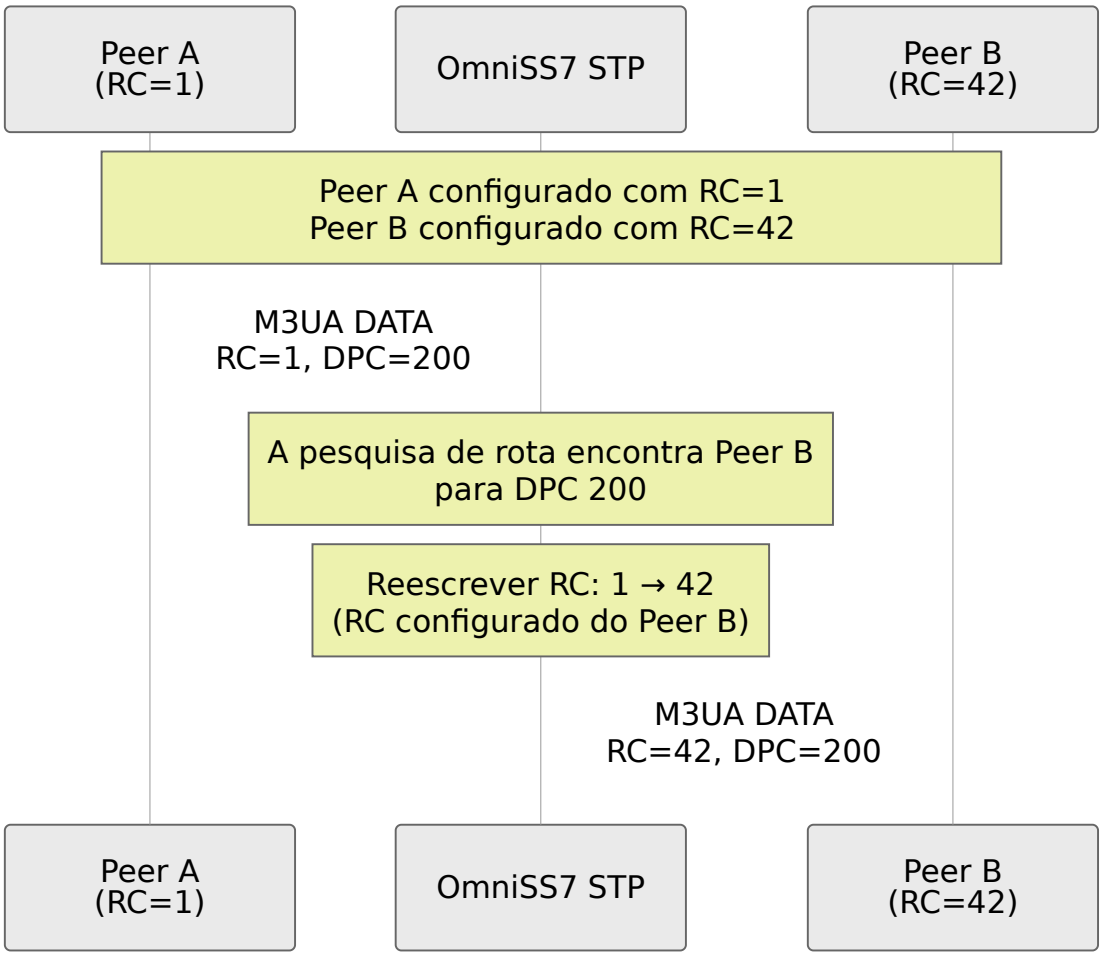
```
config :omniss7,
  # Configuração de RC do Cliente MAP
  map_client_m3ua: %{
    mode: "ASP",
    local_ip: {10, 0, 0, 1},
    local_port: 2905,
    remote_ip: {10, 0, 0, 10},
    remote_port: 2905,
    routing_context: 1    # Usado para solicitações MAP iniciadas
pelo cliente (SRI, PRN, etc.)
  },

  # Configuração de RC do Cliente CAP
  cap_client_m3ua: %{
    mode: "ASP",
    local_ip: {10, 0, 0, 1},
    local_port: 2906,
    remote_ip: {10, 0, 0, 20},
    remote_port: 2906,
    routing_context: 2    # Usado para solicitações CAP iniciadas
pelo cliente (InitialDP, etc.)
  }
```

Reescrita do Contexto de Roteamento STP

Quando o STP encaminha mensagens entre pares, ele **reescreve** o Contexto de Roteamento para o RC configurado do par de destino. Este é o comportamento correto segundo o RFC 4666 porque:

1. O RC identifica o contexto AS em cada ponto final
2. Diferentes pares podem estar associados a diferentes contextos AS
3. O STP traduz valores de RC ao roteá-los entre diferentes ASs

Fluxo de Reescrita de RC do STP:



Configuração:

O contexto de roteamento de cada par é configurado na definição do par:

```
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "Network_A_STP",
      role: :server,
      remote_ip: {10, 0, 0, 100},
      remote_port: 2905,
      routing_context: 1,        # RC esperado por/de este par
      point_code: 100,
      network_indicator: :international
    },
    %{
      peer_id: 2,
      name: "Network_B_HLR",
      role: :client,
      local_ip: {10, 0, 0, 1},
      local_port: 0,
      remote_ip: {10, 0, 0, 200},
      remote_port: 2905,
      routing_context: 42,       # RC esperado por este par
      point_code: 200,
      network_indicator: :international
    }
  ]
```

Quando uma mensagem chega do Peer 1 (RC=1) destinada ao Peer 2, o STP:

1. Roteia a mensagem com base no DPC (200 → Peer 2)
2. Reescreve o RC de 1 para 42 (RC configurado do Peer 2)
3. Encaminha a mensagem para o Peer 2

Resumo do Tratamento de RC

Componente	Solicitação Recebida	Resposta de Saída	Iniciada pelo Cliente
Cliente MAP	Extrair RC	Espelhar RC recebido	Usar RC configurado
Cliente CAP	Extrair RC	Espelhar RC recebido	Usar RC configurado
STP	Extrair RC	N/A	Reescrever para RC do par de destino

Quando o RC é Omitido

Se um par não tiver `routing_context` configurado (ou estiver definido como `nil`):

- **Respostas de Clientes MAP/CAP**: O parâmetro RC é omitido da resposta
- **Solicitações de Clientes MAP/CAP**: O parâmetro RC é omitido da solicitação
- **Encaminhamento STP**: O RC não é reescrito (RC original preservado ou omitido)

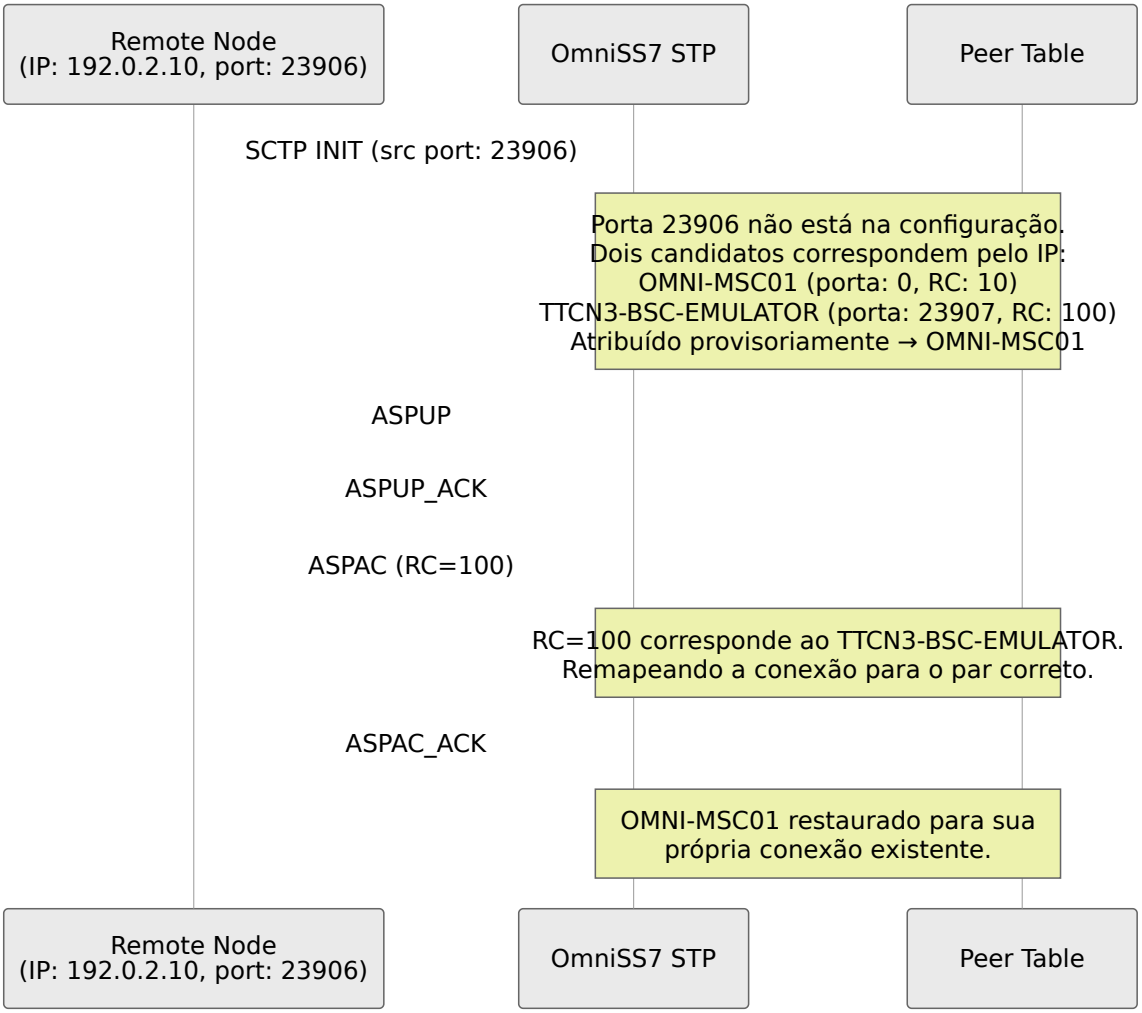
Isso mantém a compatibilidade retroativa com pares que não usam ou esperam contexto de roteamento.

Desambiguação de Múltiplos Pares com o Mesmo IP

Dois pares configurados podem compartilhar o mesmo endereço IP remoto: por exemplo, um MSC de produção e um emulador de teste rodando no mesmo host. Quando uma conexão SCTP de entrada chega, o OmniSS7 não pode determinar a identidade correta do par apenas a partir do IP de origem. Um par configurado com `remote_port: 0` (curinga) corresponderá a qualquer conexão daquele IP, o que pode fazer com que uma nova conexão do outro par seja atribuída provisoriamente à entrada errada.

O OmniSS7 resolve isso automaticamente usando o **contexto de roteamento ASPAC M3UA**. A conexão é aceita e atribuída provisoriamente ao par que melhor corresponde (a correspondência de porta específica é preferida em relação ao curinga). Quando o extremo remoto envia uma mensagem `ASPAC` carregando um contexto de roteamento, o sistema compara esse RC com o `routing_context` configurado de cada par. Se o RC corresponder a um par diferente, a conexão é silenciosamente remapeada para o correto.

Como Funciona



Saída de Log

Quando múltiplos pares correspondem a uma conexão de entrada, o OmniSS7 registra todos os candidatos para que os operadores possam verificar se a desambiguação está funcionando corretamente:

```
Conexão de entrada de 192.0.2.10:23906 corresponde aos pares:
["OMNI-MSC01", "TTCN3-BSC-EMULATOR"] - usando provisoriamente
"OMNI-MSC01" até que o RC ASPAC confirme a identidade
```

Após o contexto de roteamento ASPAC ser recebido:

```
RC ASPAC=100 corresponde ao par 9 (TTCN3-BSC-EMULATOR), mas a
conexão foi atribuída ao par 8 (OMNI-MSC01) - remapeando
```

Requisitos de Configuração

Para que a desambiguação de pares com o mesmo IP funcione corretamente, **cada par deve ter um `routing_context` único configurado**. Se dois pares no mesmo IP compartilharem o mesmo RC (ou ambos tiverem `nil`), o remapeamento não poderá determinar qual par é o correto.

```
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 8,
      name: "OMNI-MSC01",
      role: :server,
      remote_ip: {10, 5, 198, 200},
      remote_port: 0,              # Curva - aceita qualquer porta
de origem
      routing_context: 10,         # Deve ser único entre pares
que compartilham este IP
      point_code: 500,
      network_indicator: :national
    },
    %{
      peer_id: 9,
      name: "TTCN3-BSC-EMULATOR",
      role: :server,
      remote_ip: {10, 5, 198, 200},
      remote_port: 23907,          # Porta de origem esperada
(pode diferir para portas efêmeras)
      routing_context: 100,        # RC diferente - usado para
identificar este par após ASPAC
      point_code: 193,
      network_indicator: :national
    }
  ]
```

> **Nota sobre portas de origem efêmeras**: A correspondência de portas é exata. Se um par estiver configurado com `remote_port: 23907`, mas conectar a partir da porta 23906, esse par será excluído da correspondência inicial e o par curinga vencerá em vez disso. Isso é esperado. O contexto de roteamento ASPAC é o sinal de identidade autoritativo. A `remote_port` configurada em um par é usada apenas para a heurística de atribuição da conexão inicial, não para a identidade final. Dê a cada par um `routing_context` distinto e o remapeamento será resolvido corretamente, independentemente de qual porta de origem o remoto usar.

Solução de Problemas em Conexões M3UA

Problema: Conexão Não Estabelece

Sintomas:

- Status mostra DOWN
- Nenhuma associação SCTP

Verificações:

1. Verifique a conectividade da rede: `ping remote_ip`
2. Verifique se o firewall permite SCTP (protocolo 132)
3. Verifique se o STP/SGP remoto está ouvindo na porta correta
4. Verifique `remote_ip` e `remote_port` na configuração
5. Revise os logs da aplicação em busca de erros SCTP

Problema: Conexão Estabelecida, mas ASP Não Está Ativo

Sintomas:

- Associação SCTP existe
- Estado do ASP preso em INACTIVE ou ASPUP_SENT

Verificações:

1. Verifique se o contexto de roteamento corresponde à configuração remota
2. Verifique se o STP remoto aceita seu código de ponto
3. Revise os logs para rejeições de ASPUP/ASPAC
4. Verifique se não há requisitos de autenticação/segurança

Problema: Dados Não Fluindo

Sintomas:

- Estado do ASP mostra ACTIVE
- Nenhuma mensagem sendo roteada

Verificações:

1. Verifique o contexto de roteamento nas mensagens
2. Verifique o endereçamento SCCP (formato GT, valores SSN)
3. Verifique se as tabelas de roteamento estão configuradas corretamente
4. Revise a página `/events` em busca de erros SCCP
5. Verifique o roteamento do código de ponto no nível do STP

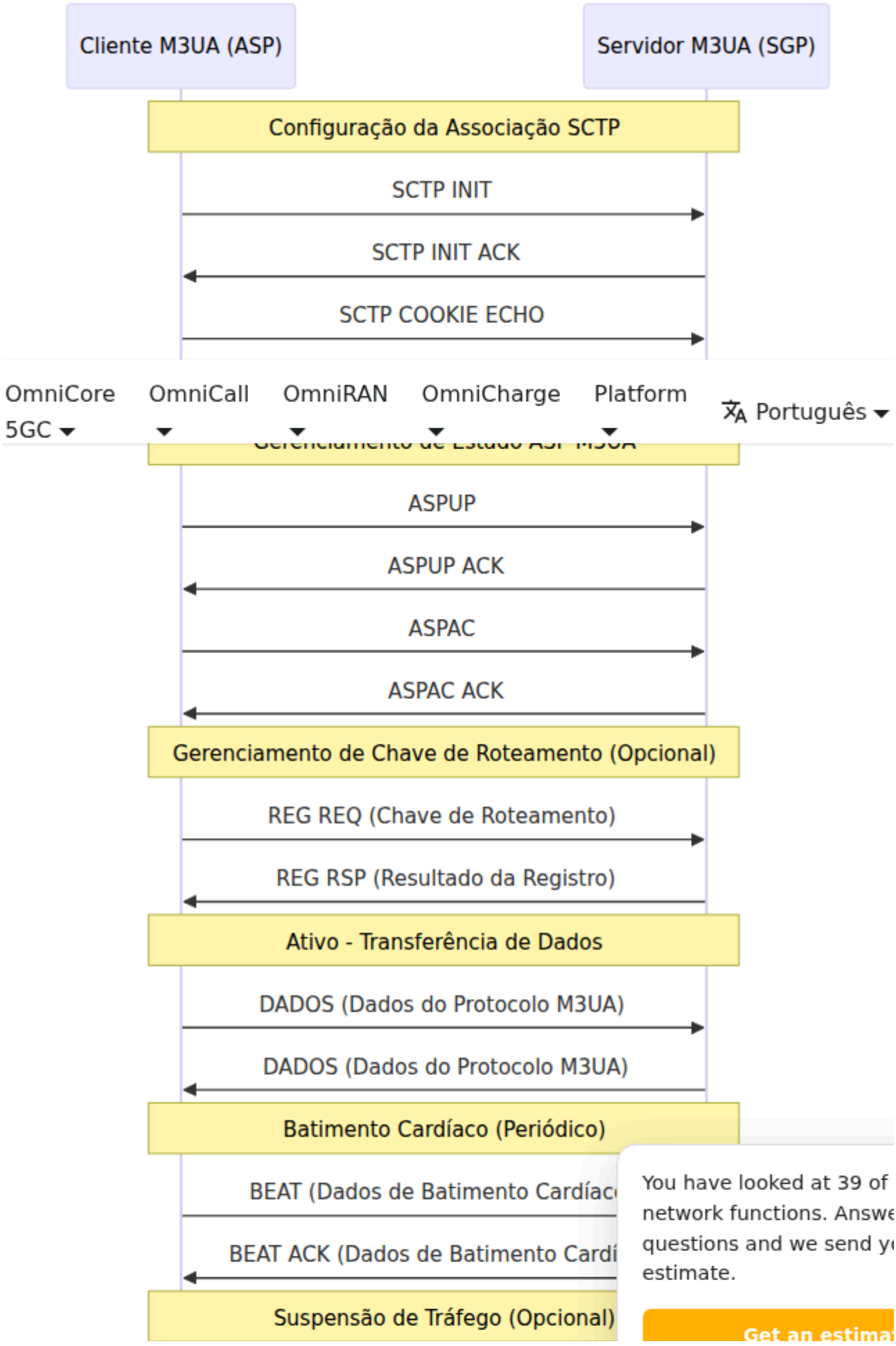
Monitoramento de Status de Pares M2PA

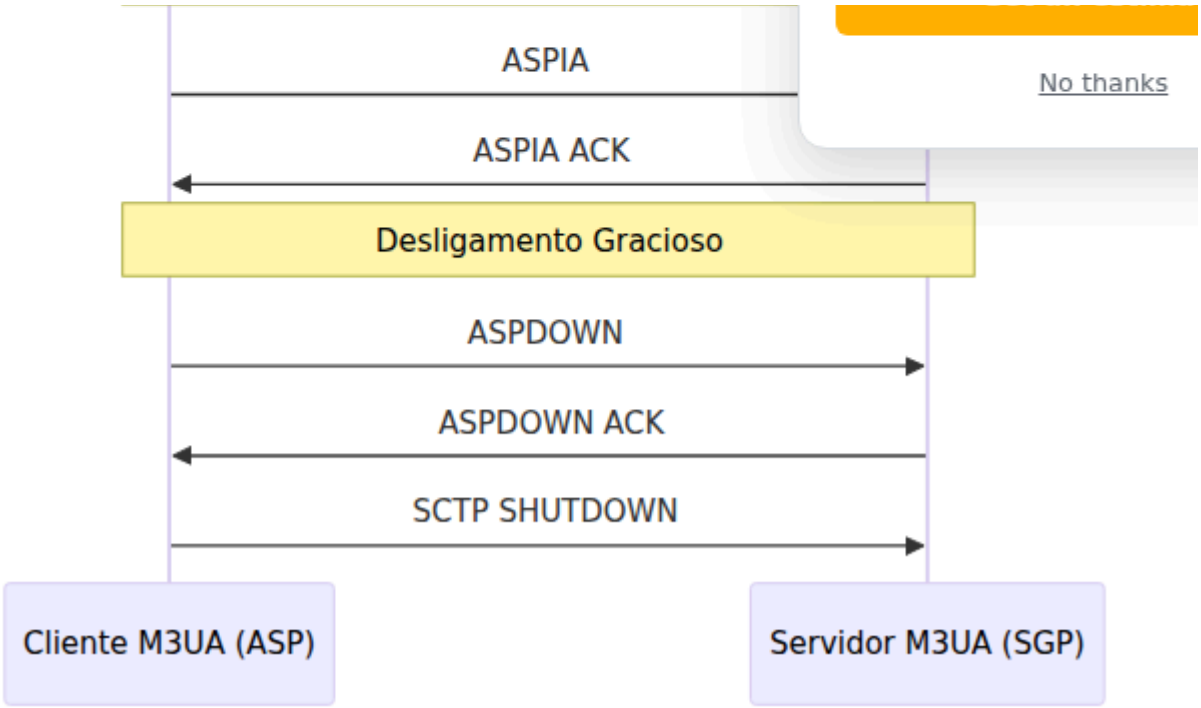
Compreendendo M2PA

M2PA (MTP2 User Peer-to-Peer Adaptation Layer) é um protocolo definido no RFC 4165 que fornece transporte de mensagens MTP3 ponto a ponto sobre SCTP. Ao contrário do M3UA, que usa uma arquitetura ASP/SGP, o M2PA fornece links ponto a ponto semelhantes aos links tradicionais TDM SS7.

Estados do Link M2PA

Os links M2PA progridem por vários estados durante o estabelecimento:


Cliente M3UA (ASP)
Servidor M3UA (SGP)
Configuração da Associação SCTP
SCTP INIT
SCTP INIT ACK
SCTP COOKIE ECHO
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
Português
ASPUP
ASPUP ACK
ASPAC
ASPAC ACK
Gerenciamento de Chave de Roteamento (Opcional)
REG REQ (Chave de Roteamento)
REG RSP (Resultado da Registro)
Ativo - Transferência de Dados
DADOS (Dados do Protocolo M3UA)
DADOS (Dados do Protocolo M3UA)
Batimento Cardíaco (Periódico)
BEAT (Dados de Batimento Cardíac
BEAT ACK (Dados de Batimento Cardi
Suspensão de Tráfego (Opcional)
You have looked at 39 of
network functions. Answe
questions and we send y
estimate.
Get an estimat



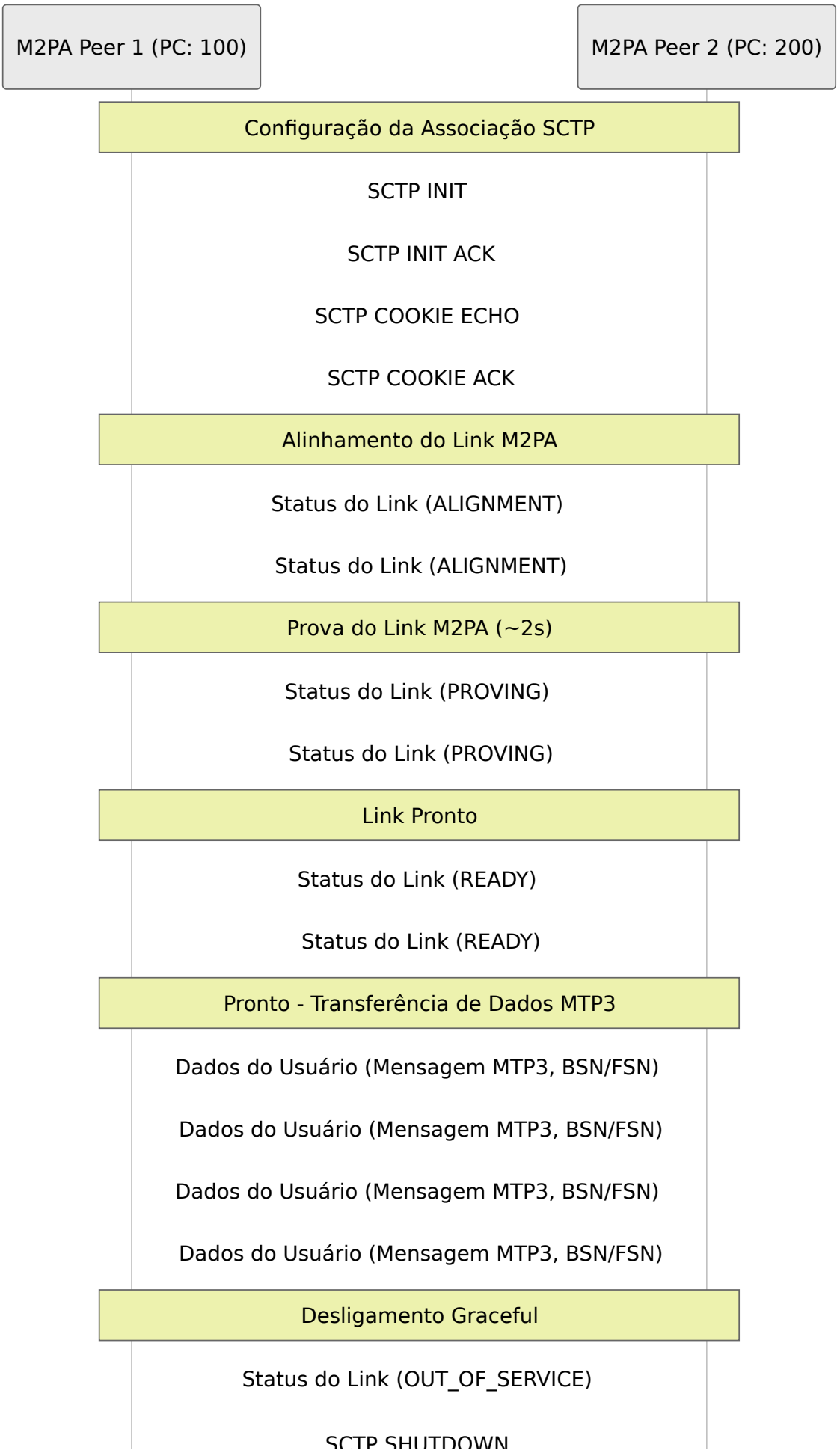
Descrições dos Estados:

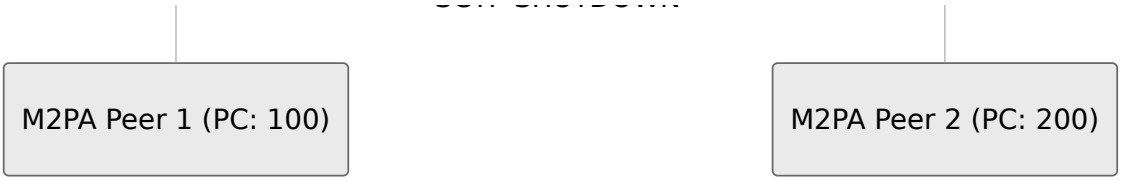
- **DOWN** - Sem conexão SCTP, link inativo
- **CONNECTING** - Associação SCTP em progresso
- **ALIGNMENT** - Mensagens de Status do Link trocadas (~1 segundo)
- **PROVING** - Período de prova do link, testando a integridade do link (~2 segundos)
- **READY** - Link operacional, pronto para transferência de dados do usuário MTP3
- **ALIGNMENT (reentrada)** - Mudança de status do link requer re-alinhamento

Progressão do Estado do Link:

1. **Conexão SCTP**: Estabelece a associação SCTP (DOWN → CONNECTING)
2. **Alinhamento**: Troca mensagens de Status do Link para sincronizar (CONNECTING → ALIGNMENT)
3. **Prova**: Testa a confiabilidade do link e a sincronização do número de sequência (ALIGNMENT → PROVING)
4. **Pronto**: O link se torna operacional para transferência de dados (PROVING → READY)

Fluxo de Mensagens M2PA


M2PA Peer 1 (PC: 100)
M2PA Peer 2 (PC: 200)
Configuração da Associação SCTP
SCTP INIT
SCTP INIT ACK
SCTP COOKIE ECHO
SCTP COOKIE ACK
Alinhamento do Link M2PA
Status do Link (ALIGNMENT)
Status do Link (ALIGNMENT)
Prova do Link M2PA (~2s)
Status do Link (PROVING)
Status do Link (PROVING)
Link Pronto
Status do Link (READY)
Status do Link (READY)
Pronto - Transferência de Dados MTP3
Dados do Usuário (Mensagem MTP3, BSN/FSN)
Dados do Usuário (Mensagem MTP3, BSN/FSN)
Dados do Usuário (Mensagem MTP3, BSN/FSN)
Dados do Usuário (Mensagem MTP3, BSN/FSN)
Desligamento Graceful
Status do Link (OUT_OF_SERVICE)
SCTP SHUTDOWN



Monitoramento de Pares M2PA via UI Web

A UI Web fornece monitoramento em tempo real das conexões de pares M2PA.

Acessando a Página de Gerenciamento de Roteamento:

1. Navegue até a página inicial da UI Web
2. Clique em "Gerenciamento de Roteamento" no menu de navegação
3. Veja a tabela "Pares M3UA/M2PA"

Tabela de Pares M2PA:

Coluna	Descrição
ID do Par	Identificador único do par
Nome	Nome do par (por exemplo, `M2PA_Link_STP_A`)
Protocolo	Mostra "M2PA" em verde
Código de Ponto	Código de ponto local
PC Adjacente	Código de ponto do par adjacente
Local	IP:Porta local (tipicamente porta 3565)
Remoto	IP:Porta remota
Status	Estado do link (por exemplo, READY, ALIGNMENT, DOWN)

Indicadores de Status:

- **READY (Verde)** - O link está operacional e passando tráfego
- **ALIGNMENT (Amarelo)** - O link está alinhando, ainda não pronto
- **PROVING (Amarelo)** - O link está em estado de prova
- **DOWN (Vermelho)** - O link está inativo ou indisponível

Solução de Problemas em Conexões M2PA

Problema: Link Preso em ALIGNMENT

Sintomas:

- O estado do link mostra ALIGNMENT por um período prolongado
- Sem progressão para PROVING ou READY

Verificações:

1. Verifique se ambos os lados estão configurados com os códigos de ponto corretos
2. Verifique se o firewall SCTP permite o protocolo 132
3. Verifique se `point_code` e `adjacent_point_code` estão configurados corretamente
4. Revise os logs da aplicação em busca de erros de mensagem de Status do Link
5. Certifique-se de que o par remoto também está no estado ALIGNMENT

Problema: Link Preso em PROVING

Sintomas:

- O link atinge PROVING, mas não transita para READY
- O período de prova excede 2-3 segundos

Verificações:

1. Verifique a estabilidade da rede (sem perda de pacotes)
2. Verifique se há erros de associação SCTP
3. Revise os logs em busca de incompatibilidades de números de sequência
4. Certifique-se de que o par remoto também está no estado PROVING

5. Verifique se o multihoming SCTP não está causando problemas de roteamento

Problema: Flutuação do Link (DOWN ↔ READY)

Sintomas:

- O link alterna repetidamente entre READY e DOWN
- Re-alinhamentos frequentes

Verificações:

1. Verifique a estabilidade da conectividade da rede
2. Verifique as configurações de heartbeat SCTP
3. Revise as configurações de tempo limite de sessão do firewall
4. Verifique se há problemas de MTU/fracionamento
5. Verifique se não há endereços IP duplicados

Problema: Dados Não Fluindo

Sintomas:

- O estado do link mostra READY
- Nenhuma mensagem MTP3 sendo transferida

Verificações:

1. Verifique se as tabelas de roteamento incluem rotas para este par
2. Verifique se o roteamento do código de ponto MTP3 está configurado
3. Revise os valores DPC nas mensagens correspondem às rotas esperadas
4. Verifique a página `/events` em busca de erros de roteamento
5. Verifique se os números de sequência (BSN/FSN) estão incrementando

---

Documentação Relacionada

- ← Voltar à Documentação Principal

- Guia de Recursos Comuns - UI Web, API, Monitoramento
- Guia do Cliente MAP - Enviando solicitações MAP
- Guia do Centro de SMS - Entrega de SMS
- Referência Técnica - Especificações de protocolo

OmniSS7 por Omnitouch Network Services

Guia do Gateway USSD

← Voltar para a Documentação Principal

Este guia cobre o **Gateway USSD** do OmniSS7, que conecta diálogos USSD SS7/MAP a callbacks HTTP/JSON, permitindo que desenvolvedores de terceiros construam aplicações USSD com um simples endpoint HTTP.

Índice

1. Visão Geral
2. Arquitetura
3. Habilitando o Gateway USSD
4. Configuração
5. Protocolo de Callback HTTP
6. USSD Originado pela Rede (Push API)
7. Ciclo de Vida da Sessão
8. Tratamento de Erros
9. Métricas e Monitoramento
10. Servidor de Callback de Exemplo
11. Solução de Problemas

Visão Geral

O Gateway USSD lida com duas direções de tráfego USSD:

- **Originado pelo Móvel (Entrada)**: Um assinante disca um código curto (por exemplo, `*100#`). O gateway recebe o `processUnstructuredSS-Request` MAP (opcode 59). Ele encaminha a solicitação para seu callback HTTP. Em seguida, retransmite sua resposta de volta via SS7.
- **Originado pela Rede (Saída)**: Sua aplicação envia uma mensagem USSD para um assinante via a API REST. O gateway envia um `unstructuredSS-`

`Request` MAP (opcode 60) via SS7. Ele então roteia a resposta do assinante para seu callback.

Ambas as direções suportam **diálogos de múltiplas interações**: menus interativos onde o assinante responde e recebe prompts de acompanhamento.

Características Principais

Propriedade	Valor
Transporte	HTTP POST síncrono por interação
Codificação	Alfabeto padrão GSM de 7 bits (DCS 0x0F) conforme 3GPP TS 23.038
Máximo de caracteres	182 caracteres (configurável)
Rastreamento de sessão	UUID gerado pelo gateway por diálogo
Autenticação	Nenhuma (confia na rede SS7)
Roteamento	Correspondência de prefixo de código curto para URLs de callback

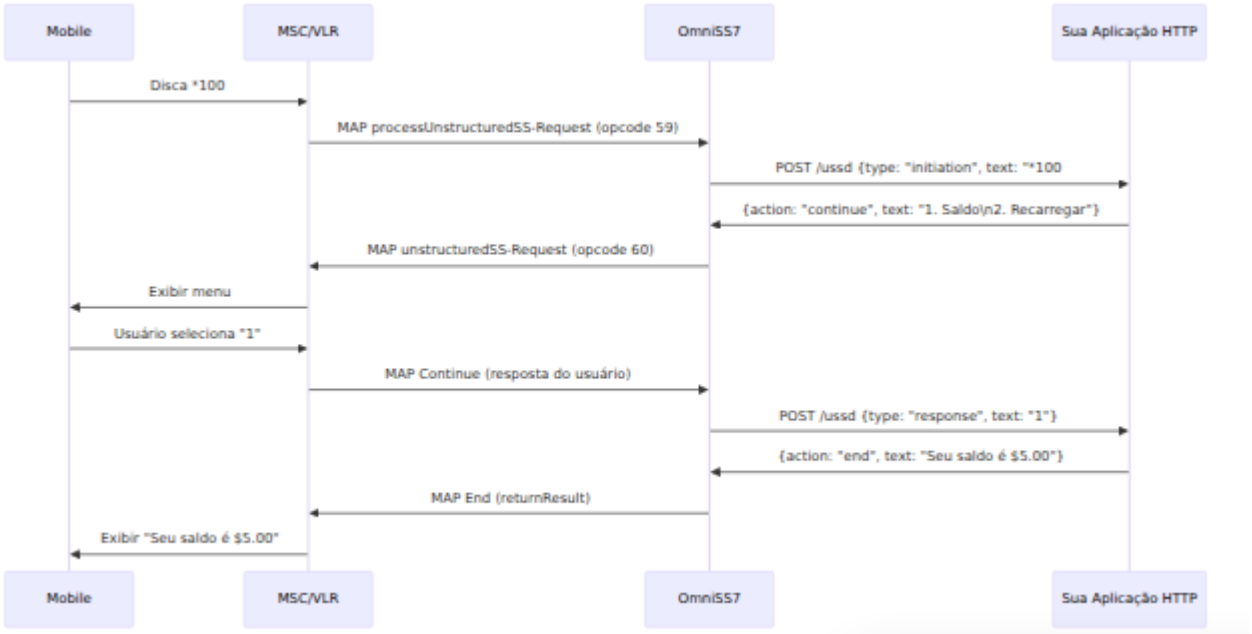
Referências 3GPP

Especificação	Relevância
3GPP TS 23.090	USSD Fase 2: arquitetura e procedimentos
3GPP TS 24.090	USSD Fase 3: detalhes do protocolo
3GPP TS 29.002	Protocolo MAP: USSD-Arg, USSD-Res, opcodes 59/60/61
3GPP TS 23.038	Alfabeto padrão GSM de 7 bits e esquema de codificação de dados

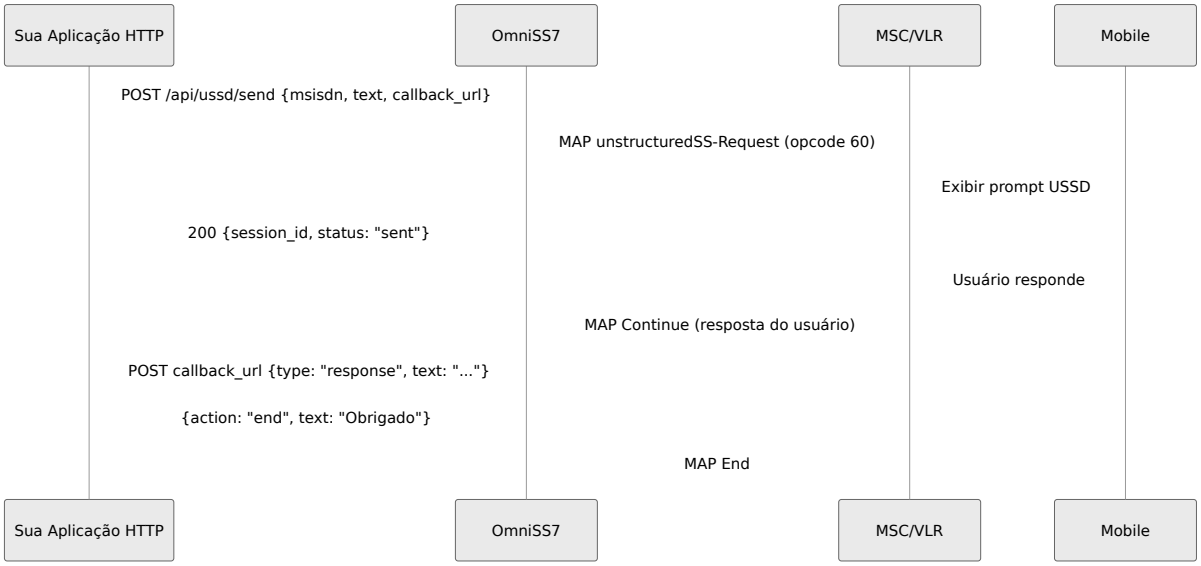
---

Arquitetura

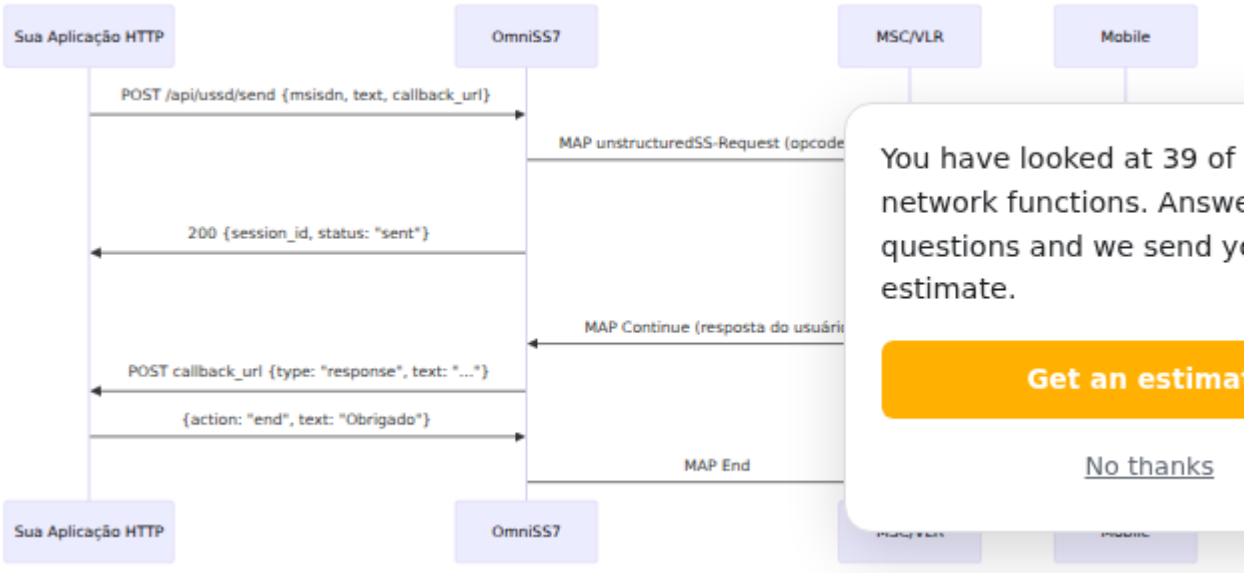
Fluxo Originado pelo Móvel (Entrada)



Fluxo Originado pela Rede (Push Saída)



Visão Geral dos Componentes



Habilitando o Gateway USSD

O Gateway USSD requer que o **modo Cliente MAP** esteja habilitado, além de sua própria flag de recurso.

```
config :omniss7,
  map_client_enabled: true,
  ussd_gateway_enabled: true
```

O gateway também requer uma conexão M3UA funcional (veja o Guia do Cliente MAP para configuração do M3UA).

Configuração

Parâmetros do Gateway USSD

```
config :omniss7,
  ussd_gateway_enabled: true,
  ussd_gateway: %{
    # Roteamento de código curto - correspondência de prefixo mais
longa
    routes: [
      %{pattern: "*100", url: "http://balance-app:9000/ussd"},
      %{pattern: "*200", url: "http://topup-app:9000/ussd"},
      %{pattern: "*",    url: "http://default-app:9000/ussd"}
    ],

    # Timeouts de sessão
    session_timeout_ms: 180_000,    # Duração total da sessão (3
minutos)
    turn_timeout_ms: 30_000,        # Máximo de espera pela
resposta do assinante por interação (30 segundos)

    # Configurações de callback HTTP
    http_timeout_ms: 5_000,         # Timeout para HTTP POST para
seu aplicativo (5 segundos)

    # Limites de texto
    max_text_length: 182            # Máximo de 7 bits GSM (trunca
com aviso se excedido)
  }
```

Referência de Parâmetros

Parâmetro	Tipo	Obrigatório	Padrão
`ussd_gateway_enabled`	Booleano	Sim	`false`
`ussd_gateway.routes`	Lista de Mapas	Sim	`[]`
`ussd_gateway.session_timeout_ms`	Inteiro	Não	`180_000`
`ussd_gateway.turn_timeout_ms`	Inteiro	Não	`30_000`

Parâmetro	Tipo	Obrigatório	Padrão
`ussd_gateway.http_timeout_ms`	Inteiro	Não	`5_000`
`ussd_gateway.max_text_length`	Inteiro	Não	`182`

Parâmetros de Roteamento

Cada entrada na lista `routes` é um mapa:

Parâmetro	Tipo	Obrigatório	Descrição
`pattern`	String	Sim	Prefixo de código curto a ser correspondido. Use `"*"` como um fallback geral. Prefixos mais longos têm prioridade.
`url`	String	Sim	URL do endpoint HTTP para receber POSTs de callback para códigos curtos correspondentes.

Correspondência de Roteamento

As rotas são correspondidas por **prefixo mais longo primeiro**. Para a string de discagem `*100#`:

1. `"*100"` corresponde (comprimento 4): selecionado
2. `"*10"` corresponde (comprimento 3): ignorado, mais curto
3. `"*"` corresponde (comprimento 1): fallback

Se nenhuma rota corresponder, o gateway retorna um erro MAP para o móvel e registra um aviso.

Protocolo de Callback HTTP

Sua aplicação recebe requisições HTTP POST do gateway e responde com instruções JSON.

Requisição do Gateway para Sua Aplicação

Content-Type: `application/json`

Primeira interação (iniciação da sessão):

```
{
  "session_id": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
  "msisdn": "+254712345678",
  "type": "initiation",
  "text": "*100#",
  "turn": 1
}
```

Interações subsequentes (assinante respondeu):

```
{
  "session_id": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
  "msisdn": "+254712345678",
  "type": "response",
  "text": "1",
  "turn": 2
}
```

Campos da Requisição

Campo	Tipo	Descrição
`session_id`	String	UUID gerado pelo gateway. Único por diálogo USSD. Use isso para correlacionar interações.
`msisdn`	String	MSISDN do assinante (se disponível na mensagem MAP). Pode estar vazio para algumas redes.
`type`	String	`"initiation"` para a primeira interação, `"response"` para respostas subsequentes do assinante.
`text`	String	A string de discagem (por exemplo, `*100#`) na iniciação, ou a entrada do assinante (por exemplo, `1`) na resposta.
`turn`	Inteiro	Contador de interações começando em 1. Incrementa com cada interação do assinante.

Resposta da Sua Aplicação

Sua aplicação deve responder com JSON contendo uma `action` e `text`:

Continuar (mostrar menu, aguardar entrada do assinante):

```
{
  "action": "continue",
  "text": "1. Saldo\n2. Recarregar\n3. Transferir"
}
```

Encerrar (mostrar mensagem final, fechar sessão):

```
{
  "action": "end",
  "text": "Seu saldo é $5.00"
}
```

Campos da Resposta

Campo	Tipo	Obrigatório	Descrição
`action`	String	Sim	`"continue"` para manter a sessão aberta e aguardar a entrada do assinante, ou `"end"` para exibir uma mensagem final e fechar a sessão.
`text`	String	Sim	Texto a ser exibido no dispositivo do assinante. O comprimento máximo é regido por `max_text_length` (padrão 182). Use `\n` para quebras de linha.

USSD Originado pela Rede (Push API)

Envie uma mensagem USSD para um assinante a partir de sua aplicação.

Endpoint

`POST /api/ussd/send`

Requisição

```
{
  "msisdn": "+254712345678",
  "text": "Você tem uma fatura pendente. Responda 1 para pagar.",
  "callback_url": "http://billing-app:9000/ussd"
}
```

Campos da Requisição

Campo	Tipo	Obrigatório	Descrição
`msisdn`	String	Sim	MSISDN do assinante de destino no formato internacional.
`text`	String	Sim	Texto inicial USSD a ser exibido. Codificado como GSM de 7 bits.
`callback_url`	String	Sim	URL para receber a resposta do assinante via o protocolo de callback padrão.

Resposta

Sucesso (200 OK):

```
{
  "session_id": "xyz-789-abc-123",
  "status": "sent"
}
```

Respostas de erro:

Status HTTP	Corpo	Causa
400	`{"error": "invalid request", "required": ["msisdn", "text", "callback_url"]}`	Falta um ou mais campos obrigatórios, ou o corpo não é um JSON válido
500	`{"error": "{:gsm7_encode_failed, ...}"}`	O texto contém caracteres não no alfabeto GSM de 7 bits. A falha de codificação é apresentada através do ramo de erro genérico, então o status é **500**, não 400.
500	`{"error": "..."}`	Qualquer outra falha de envio (por exemplo, conectividade M3UA). O campo `error` carrega a razão interna inspecionada.
503	`{"error": "USSD gateway not enabled"}`	`ussd_gateway_enabled` é `false`

> **Nota sobre erros de validação vs. envio:** Problemas de validação de campo (chaves faltando, JSON malformado) retornam **400** **`invalid request`**. Uma vez que a validação passe, qualquer falha do caminho de envio (incluindo falhas de codificação GSM de 7 bits) é relatada como **500** com a razão inspecionada no campo `error`.

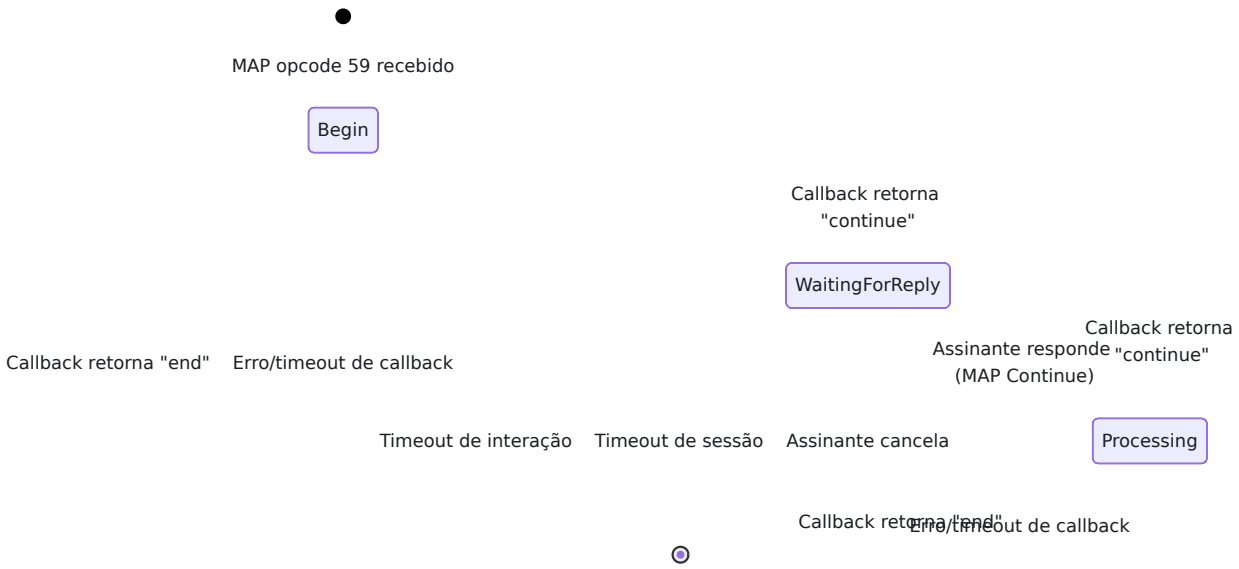
Exemplo cURL

```
curl -X POST http://localhost:9090/api/ussd/send \
  -H "Content-Type: application/json" \
  -d '{
    "msisdn": "+254712345678",
    "text": "Você tem uma fatura pendente. Responda 1 para
pagar.",
    "callback_url": "http://billing-app:9000/ussd"
  }'
```

Ciclo de Vida da Sessão

Cada diálogo USSD é rastreado como uma **sessão** com um `session_id` único.

Estados da Sessão



Sessão de Uma Interação

Se seu callback retornar `"end"` na primeira interação, nenhuma sessão persistente é criada. O gateway envia um MAP End com o resultado e retorna imediatamente.

Sessão de Múltiplas Interações

Se seu callback retornar `"continue"`, o gateway:

1. Cria um **Session GenServer** registrado em `UssdGateway.Registry`
2. Envia um MAP Continue com opcode 60 (unstructuredSS-Request) para o móvel
3. Aguarda a resposta do assinante (até `turn_timeout_ms`)
4. Encaminha a resposta para seu callback
5. Repete até que seu callback retorne `"end"` ou ocorra um timeout

Comportamento de Timeout

Timeout	Padrão	Efeito
Timeout de interação	30 segundos	Se o assinante não responder dentro desta janela, a sessão é encerrada com um erro MAP.
Timeout de sessão	3 minutos	Duração total da sessão. Encerra a sessão independentemente da atividade.
Timeout de callback HTTP	5 segundos	Se sua aplicação não responder a tempo, o gateway envia um erro MAP para o móvel e encerra a sessão.

Tratamento de Erros

O gateway lida com falhas de forma elegante e sempre tenta enviar uma resposta de erro MAP para o móvel, para que o assinante veja uma mensagem significativa em vez de um timeout de rede.

Cenário	Ação do Gateway	Código de Erro MAP
Timeout de callback HTTP ou 5xx	Encerrar sessão, enviar MAP End com erro	34 (systemFailure)
JSON inválido do callback	Encerrar sessão, enviar MAP End com erro	34 (systemFailure)
Texto USSD excede `max_text_length`	Truncar texto, registrar aviso, continuar normalmente	N/A (truncado, não é um erro)
Timeout do assinante (sem resposta)	Encerrar sessão, enviar MAP End com erro	34 (systemFailure)
Nenhuma rota corresponde ao código curto	Enviar MAP End com erro, registrar aviso	34 (systemFailure)
Falha do Session GenServer	Sessão morre, assinante vê timeout de rede	N/A (saída do processo)
Gateway USSD não habilitado	Retornar Facility Not Supported	21 (facilityNotSupported)

Métricas e Monitoramento

O Gateway USSD expõe métricas Prometheus no endpoint padrão `/metrics` (porta 9090).

Métricas USSD

Métrica: `ussd_requests_total` **Tipo**: Contador **Descrição**: Total de requisições USSD processadas **Rótulos**:

- `direction`: `"inbound"` (originado pelo móvel) ou `"outbound"` (push originado pela rede)

Métrica: `ussd_active_sessions` **Tipo**: Gauge **Descrição**: Declarado como um gauge para o número de sessões USSD ativas.

> **Caveat:** Este gauge é **declarado mas nunca atualizado** na versão atual: ele sempre lê `0`. Não confie nele para contagens de sessões ativas. Acompanhe a atividade da sessão via `ussd_requests_total` em vez disso.

Métrica: `map_request_duration_milliseconds` **Tipo**: Histograma **Descrição**: Duração das operações de envio USSD em milissegundos **Rótulos**:

- `operation`: `"ussd_send"` para requisições de push de saída

Exemplos de Consultas Prometheus

```
# Taxa de requisições USSD por direção (entrada vs saída)
rate(ussd_requests_total[5m])

# Latência USSD de saída (p95)
histogram_quantile(0.95,
rate(map_request_duration_milliseconds_bucket{operation="ussd_send"}
[5m]))
```

Servidor de Callback de Exemplo

Python (Flask)

```
from flask import Flask, request, jsonify

app = Flask(__name__)
sessions = {}

@app.route('/ussd', methods=['POST'])
def ussd():
    data = request.json
    session_id = data['session_id']
    text = data['text']
    turn = data['turn']

    if data['type'] == 'initiation':
        sessions[session_id] = {'state': 'main_menu'}
        return jsonify({
            'action': 'continue',
            'text': 'Bem-vindo!\n1. Ver saldo\n2. Comprar
crédito\n3. Transferir'
        })

    state = sessions.get(session_id, {}).get('state')

    if state == 'main_menu':
        if text == '1':
            del sessions[session_id]
            return jsonify({
                'action': 'end',
                'text': 'Seu saldo é $5.00'
            })
        elif text == '2':
            sessions[session_id]['state'] = 'buy_airtime'
            return jsonify({
                'action': 'continue',
                'text': 'Digite o valor:'
            })
        else:
            del sessions[session_id]
            return jsonify({
```

```
                'action': 'end',
                'text': 'Opção inválida. Adeus.'
            })

    elif state == 'buy_airtime':
        del sessions[session_id]
        return jsonify({
            'action': 'end',
            'text': f'Você comprou ${text} de crédito. Obrigado!'
        })

    return jsonify({'action': 'end', 'text': 'Sessão expirada.'})

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=9000)
```

Node.js (Express)

```
const express = require('express');
const app = express();
app.use(express.json());

const sessions = new Map();

app.post('/ussd', (req, res) => {
  const { session_id, text, type } = req.body;

  if (type === 'initiation') {
    sessions.set(session_id, { state: 'main_menu' });
    return res.json({
      action: 'continue',
      text: 'Bem-vindo!\n1. Ver saldo\n2. Comprar crédito'
    });
  }

  const session = sessions.get(session_id);
  if (!session) {
    return res.json({ action: 'end', text: 'Sessão expirada.' });
  }

  if (session.state === 'main_menu' && text === '1') {
    sessions.delete(session_id);
    return res.json({ action: 'end', text: 'Seu saldo é $5.00' });
  }

  sessions.delete(session_id);
  return res.json({ action: 'end', text: 'Adeus.' });
});

app.listen(9000, () => console.log('Callback USSD na porta
9000'));
```

Solução de Problemas

Discar USSD Retorna "Serviço N��o Disponível"

Sintomas: O assinante disca um código curto e imediatamente recebe um erro de rede.

Possíveis causas:

- `ussd_gateway_enabled` é `false`
- Nenhuma rota corresponde ao código curto discado
- Conexão M3UA está inativa

Resolução:

1. Verifique `ussd_gateway_enabled: true` na configuração
2. Verifique se um padrão de rota corresponde ao código curto (lembre-se de incluir `*` como fallback)
3. Verifique o status do par M3UA na interface Web (página de Pares)

Callback Não Recebendo Requisições

Sintomas: Os logs do gateway mostram o início do USSD, mas sua aplicação nunca recebe o POST HTTP.

Possíveis causas:

- URL de callback é inatingível a partir do host OmniSS7
- Firewall bloqueando HTTP de saída do OmniSS7
- Aplicação de callback não está em execução

Resolução:

1. Teste a conectividade: `curl -v http://your-app:9000/ussd` a partir do host OmniSS7
2. Verifique as regras do firewall para HTTP de saída

3. Verifique se sua aplicação de callback está ouvindo na porta configurada

Sessões Expirando Prematuramente

Sintomas: Sessões de múltiplas interações terminam com "systemFailure" antes que o assinante possa responder.

Possíveis causas:

- `turn_timeout_ms` é muito curto para sua base de assinantes
- `http_timeout_ms` é muito curto para o tempo de processamento da sua aplicação
- Latência de rede entre OmniSS7 e seu servidor de callback

Resolução:

1. Aumente `turn_timeout_ms` (o padrão de 30 segundos deve ser suficiente para a maioria dos casos)
2. Aumente `http_timeout_ms` se sua aplicação precisar de mais tempo de processamento
3. Implemente o servidor de callback próximo ao OmniSS7 para reduzir a latência

Erros de Codificação GSM de 7 bits

Sintomas: Erros `gsm7_encode_failed` nos logs, ou uma resposta **500** de `/api/ussd/send` cujo campo `error` contém `{:gsm7_encode_failed, ...}`.

Possíveis causas:

- O texto contém caracteres fora do alfabeto padrão de 7 bits GSM (por exemplo, emoji, caracteres CJK)

Resolução:

- Restringir o texto USSD ao conjunto básico de caracteres GSM: letras ASCII, dígitos, pontuação comum e alguns caracteres gregos/nórdicos
- Veja 3GPP TS 23.038 Seção 6.2.1 para a tabela completa de caracteres

Documentação Relacionada

- **Guia da API**: Referência completa da API REST (todos os endpoints, incluindo `/api/ussd/send`)
- **Guia do Cliente MAP**: Configuração da conexão M3UA necessária para USSD
- **Referência de Configuração**: Todos os parâmetros de configuração
- **Guia de Recursos Comuns**: UI da Web, monitoramento e configuração do Prometheus

Guia da Interface Web

← Voltar para a Documentação Principal

Este guia fornece documentação abrangente para o uso da **Interface Web** do OmniSS7 (interface Phoenix LiveView).

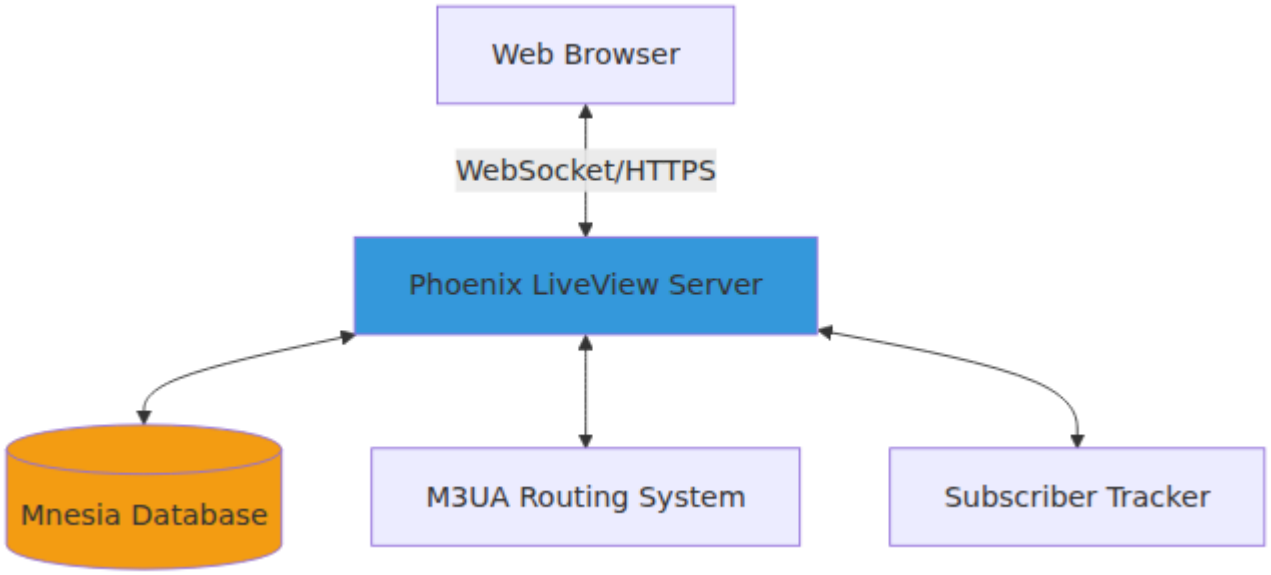
Índice

1. Visão Geral
2. Acessando a Interface Web
3. Página de Gerenciamento de Roteamento
4. Página de Assinantes Ativos
5. Operações Comuns
6. Comportamento de Atualização Automática

Visão Geral

A Interface Web do OmniSS7 é uma aplicação **Phoenix LiveView** que fornece capacidades de monitoramento e gerenciamento em tempo real. As páginas disponíveis dependem de qual modo operacional está ativo (STP, HLR ou SMSc).

Arquitetura da Interface Web



Configuração do Servidor

Parâmetro	Valor
Protocolo	HTTPS (`force_ssl` habilitado)
Porta	`8087` (configurado em `config :control_panel, ControlPanelWeb.Endpoint` → `https: [port: 8087]`)
IP Padrão	`0.0.0.0` (escuta em todas as interfaces)
Certificado	`priv/cert/omnitouch.crt`
Chave privada	`priv/cert/omnitouch.pem`
Adaptador	Bandit (Phoenix)

URL de Acesso: `https://[server-ip]:8087`

Acessando a Interface Web

Pré-requisitos

1. **Certificados SSL**: Certifique-se de que certificados SSL válidos estão presentes em `priv/cert/`:
   - `omnitouch.crt` - Arquivo de certificado
   - `omnitouch.pem` - Arquivo de chave privada
2. **Aplicação em Execução**: Inicie a aplicação com `iex -S mix`
3. **Firewall**: Certifique-se de que a porta `8087` está aberta para tráfego HTTPS

Páginas Disponíveis por Modo

O rótulo da página mostrado aqui é exatamente o rótulo registrado em `config/config.exs` (`use_builtin_pages` / `use_additional_pages`). As colunas de modo indicam o modo operacional (STP/HLR/SMSc) em que uma página é tipicamente relevante; a configuração em tempo de execução filtra a lista de páginas por implantação.

Página (rota)	Rótulo	Modo STP	Modo HLR	Modo SMSc	Descri
`/application`	**Recursos**	✅	✅	✅	Recursos sistema e monitorar
`/configuration`	**Configuração**	✅	✅	✅	Visualizad configura
`/license`	**Licença**	✅	✅	✅	Status da licença
`/events`	**Eventos SS7**	✅	✅	✅	Registro d eventos e captura d mensage SCCP
`/logs`	**Logs do Sistema**	✅	✅	✅	Visualizad logs da aplicação vivo
`/client`	**Cliente SS7**	✅	✅	✅	Teste de operação manual M
`/m3ua`	**Pares**	✅	✅	✅	Status da conexão d pares / AS M3UA
`/sctp`	**Conexões SCTP**	✅	✅	✅	Status da associaçã SCTP ao v

Página (rota)	Rótulo	Modo STP	Modo HLR	Modo SMSc	Descri
`/routing`	**Roteamento**	□	□	□	Gerenciar da tabela roteamen M3UA
`/routing_test`	**Teste de Roteamento**	□	□	□	Teste e validação rotas
`/hlr_links`	**Links HLR**	□	□	□	Status da HLR e gerencian de assina
`/subscribers`	**Assinantes Ativos**	□	□	□	Rastream em tempo da localiz de assina (HLR)
`/smsc_links`	**Links SMSc**	□	□	□	Status da SMSc e gerencian de filas
`/smsc_subscribers`	**Assinantes SMSc**	□	□	□	Rastream em tempo de assina (SMSc)
`/camel_sessions`	**Sessões CAMEL**	□	□	□	Monitoran de diálog CAMEL/CA vivo

Página (rota)	Rótulo	Modo STP	Modo HLR	Modo SMSc	Descri
`/camel_request`	**Construtor de Solicitações CAMEL**	□	□	□	Construto interativo solicitaçõ CAP
`/msisdn_imsi_test`	**Teste MSISDN/IMSI**	□	□	□	Pesquisa / teste de conversão MSISDN ⇔

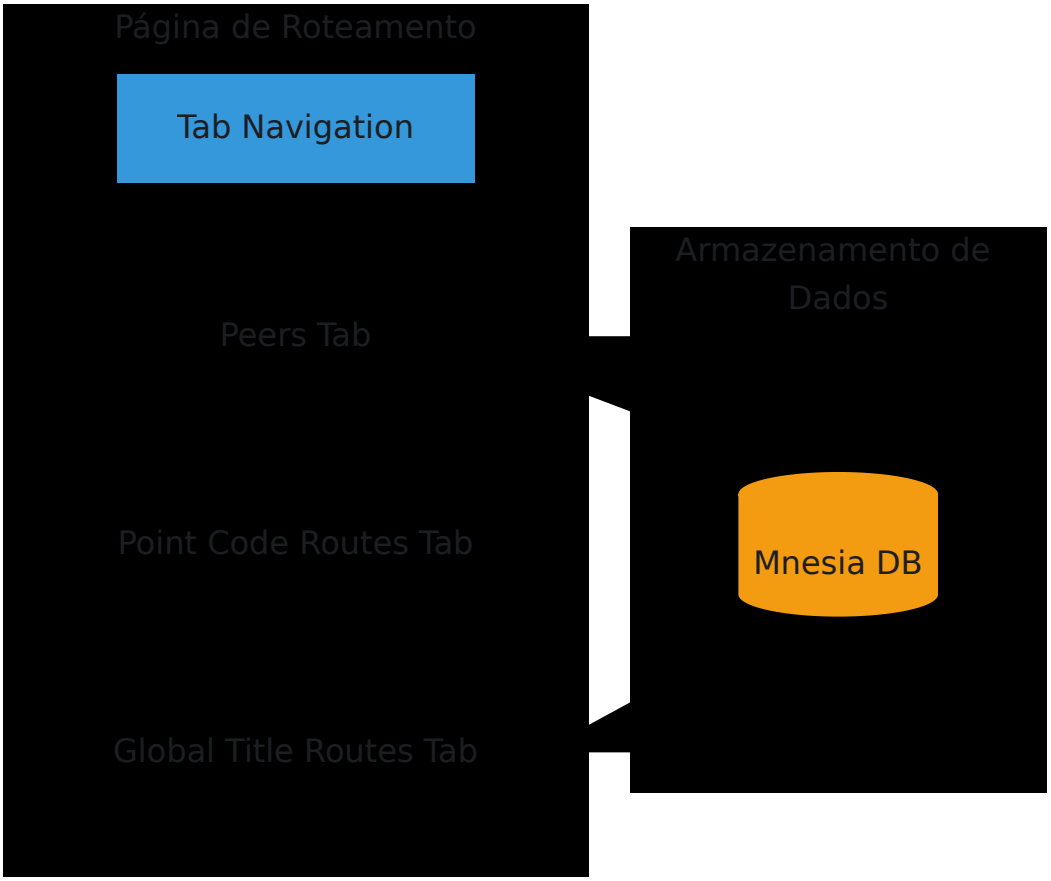
> A página M3UA é rotulada como **"Pares"** na navegação (rota `/m3ua`).

Gerenciamento de Roteamento

Página: `/routing`
Modos: STP, SMSc
Atualização Automática: A cada 5 segundos

A página de Gerenciamento de Roteamento fornece uma interface em abas para gerenciar tabelas de roteamento M3UA.

Layout da Página



Aba de Pares

Gerencie conexões de pares M3UA (outros STPs, HLRs, MSCs, SMSCs).

Colunas da Tabela de Pares

Coluna	Descrição	Exemplo
ID	Identificador único do par	`1`
Nome	Nome do par legível por humanos	`"STP_West"`
Função	Função da conexão	`client`, `server`, `stp`
Point Code	Código de ponto SS7 do par	`100`
Remoto	IP:Porta remota	`10.0.0.10:2905`
Status	Status da conexão	`active`, `aspup`, `down`
Ações	Botões Editar/Excluir	-

Adicionando um Par

1. **Clique** na aba Pares
2. **Preencha** os campos do formulário:
   - **ID do Par**: Gerado automaticamente se deixado em branco
   - **Nome do Par**: Nome descritivo (obrigatório)
   - **Função**: Selecione `client`, `server` ou `stp`
   - **Point Code**: Código de ponto SS7 (obrigatório)
   - **IP Local**: Endereço IP do seu sistema
   - **Porta Local**: 0 para atribuição dinâmica de porta
   - **IP Remoto**: Endereço IP do par
   - **Porta Remota**: Porta do par (tipicamente 2905)
   - **Contexto de Roteamento**: ID do contexto de roteamento M3UA
   - **Indicador de Rede**: `international` ou `national`
3. **Clique** em "Adicionar Par"

Persistência: O par é imediatamente salvo no Mnesia e sobrevive a reinicializações.

Editando um Par

1. **Clique** no botão "Editar" na linha do par
2. **Modifique** os campos do formulário conforme necessário
3. **Clique** em "Atualizar Par"

Nota: Se você alterar o ID do Par, o par antigo será excluído e um novo será criado.

Excluindo um Par

1. **Clique** no botão "Excluir" na linha do par
2. **Confirme** a exclusão (todas as rotas que usam este par também serão removidas)

Indicadores de Status do Par

Status	Cor	Descrição
`active`	□ Verde	O par está conectado e roteando mensagens
`aspup`	□ Amarelo	ASP está ativo, mas ainda não está em operação
`down`	□ Vermelho	O par está desconectado

Aba de Rotas de Código de Ponto

Configure regras de roteamento com base nos Códigos de Ponto de Destino.

Colunas da Tabela de Rotas

Coluna	Descrição	Exemplo
PC de Destino	Código de ponto alvo (formato zone.area.id)	`1.2.3` (100)
Máscara	Máscara de sub-rede para correspondência de PC	`/14` (exato), `/8` (intervalo)
ID do Par	Par alvo para esta rota	`1`
Nome do Par	Nome do par alvo	`"STP_West"`
Prioridade	Prioridade da rota (1 = mais alta)	`1`
Rede	Indicador de rede	`international`
Ações	Botões Editar/Excluir	-

Adicionando uma Rota de Código de Ponto

1. **Clique** na aba "Rotas de Código de Ponto"
2. **Preencha** os campos do formulário:
   - **Código de Ponto de Destino**: Insira como `zone.area.id` (por exemplo, `1.2.3`) ou inteiro (0-16383)
   - **Máscara**: Selecione a máscara `/14` para correspondência exata, valores mais baixos para intervalos
   - **ID do Par**: Selecione o par alvo no dropdown
   - **Prioridade**: Insira a prioridade (1 = mais alta, padrão)
   - **Indicador de Rede**: Selecione `international` ou `national`
3. **Clique** em "Adicionar Rota"

Formato do Código de Ponto: Você pode inserir códigos de ponto em dois formatos:

- **Formato 3-8-3**: `zone.area.id` (por exemplo, `1.2.3`)
- **Formato Inteiro**: `0-16383` (por exemplo, `1100`)

O sistema converte automaticamente entre os formatos.

Compreendendo Máscaras

Os códigos de ponto são valores de 14 bits (0-16383). A máscara especifica quantos bits mais significativos devem corresponder:

Máscara	PCs Correspondidos	Caso de Uso
`/14`	1 (correspondência exata)	Roteie para um destino específico
`/13`	2 PCs	Pequeno intervalo
`/8`	64 PCs	Intervalo médio
`/0`	Todos os 16.384 PCs	**Rota padrão/catch-all**

Exemplos:

- `PC 1000 /14` → Corresponde apenas ao PC 1000
- `PC 1000 /8` → Corresponde ao PC 1000-1063 (64 PCs consecutivos)
- `PC 0 /0` → Corresponde a todos os códigos de ponto (rota padrão)

Cartão de Referência de Máscara de Código de Ponto

A página inclui uma referência interativa mostrando todos os valores de máscara e seus intervalos.

Aba de Rotas de Título Global

Configure regras de roteamento com base nos endereços de Título Global SCCP.

Requisito: O roteamento de Título Global deve estar habilitado na configuração:

```
config :omniss7,
  enable_gt_routing: true
```

Colunas da Tabela de Rotas

Coluna	Descrição	Exemplo
Prefixo GT	Prefixo GT da parte chamada (vazio = fallback)	`"1234"`, `""`
SSN de Origem	Correspondência no SSN da parte chamada (opcional)	`6` (HLR), `any`
ID do Par	Par alvo	`1`
Par	Nome do par	`"HLR_West (1)"`
SSN de Destino	Reescrever SSN ao encaminhar (opcional)	`6`, `preserve`
Prioridade	Prioridade da rota	`1`
Descrição	Descrição da rota	`"Números dos EUA"`
Ações	Botões Editar/Excluir	-

Adicionando uma Rota de Título Global

1. **Clique** na aba "Rotas de Título Global"
2. **Preencha** os campos do formulário:
   - **Prefixo GT**: Deixe vazio para rota de fallback, ou insira dígitos (por exemplo, `"1234"`)
   - **SSN de Origem**: Opcional - filtrar pelo SSN da parte chamada
   - **ID do Par**: Selecione o par alvo

    - **SSN de Destino**: Opcional - reescrever SSN ao encaminhar
    - **Prioridade**: Prioridade da rota (1 = mais alta)
    - **Descrição**: Descrição legível por humanos
3. **Clique** em "Adicionar Rota"

Rotas de Fallback: Se o Prefixo GT estiver vazio, a rota atua como um catch-all para GTs que não correspondem a nenhuma outra rota.

Valores Comuns de SSN

A página inclui um cartão de referência com valores comuns de SSN:

SSN	Elemento de Rede
6	HLR (Home Location Register)
7	VLR (Visitor Location Register)
8	MSC (Mobile Switching Center)
9	EIR (Equipment Identity Register)
10	AUC (Authentication Center)
142	RANAP
145	gsmSCF (Service Control Function)
146	SGSN

Reescrita de SSN

- **SSN de Origem**: Correspondência no SSN da Parte Chamadora em mensagens de entrada
- **SSN de Destino**: Se definido, reescreve o SSN da Parte Chamadora ao encaminhar
    - Vazio = preservar SSN original

- Valor = substituir por este SSN

Caso de Uso: Roteie mensagens com SSN=6 (HLR) para um par e reescreva para SSN=7 (VLR) no lado de saída.

Persistência da Tabela de Roteamento

Todas as rotas são armazenadas no Mnesia e sobrevivem a reinicializações da aplicação.

Como as Rotas Persistem

1. **Mudanças na Interface Web**: Todas as operações de adicionar/editar/excluir são imediatamente salvas no Mnesia
2. **Reinicialização da Aplicação**: As rotas são carregadas do Mnesia na inicialização
3. **Mesclagem de runtime.exs**: Rotas estáticas de `config/runtime.exs` são mescladas com rotas do Mnesia (sem duplicatas)

Prioridade da Rota

Quando várias rotas correspondem a um destino:

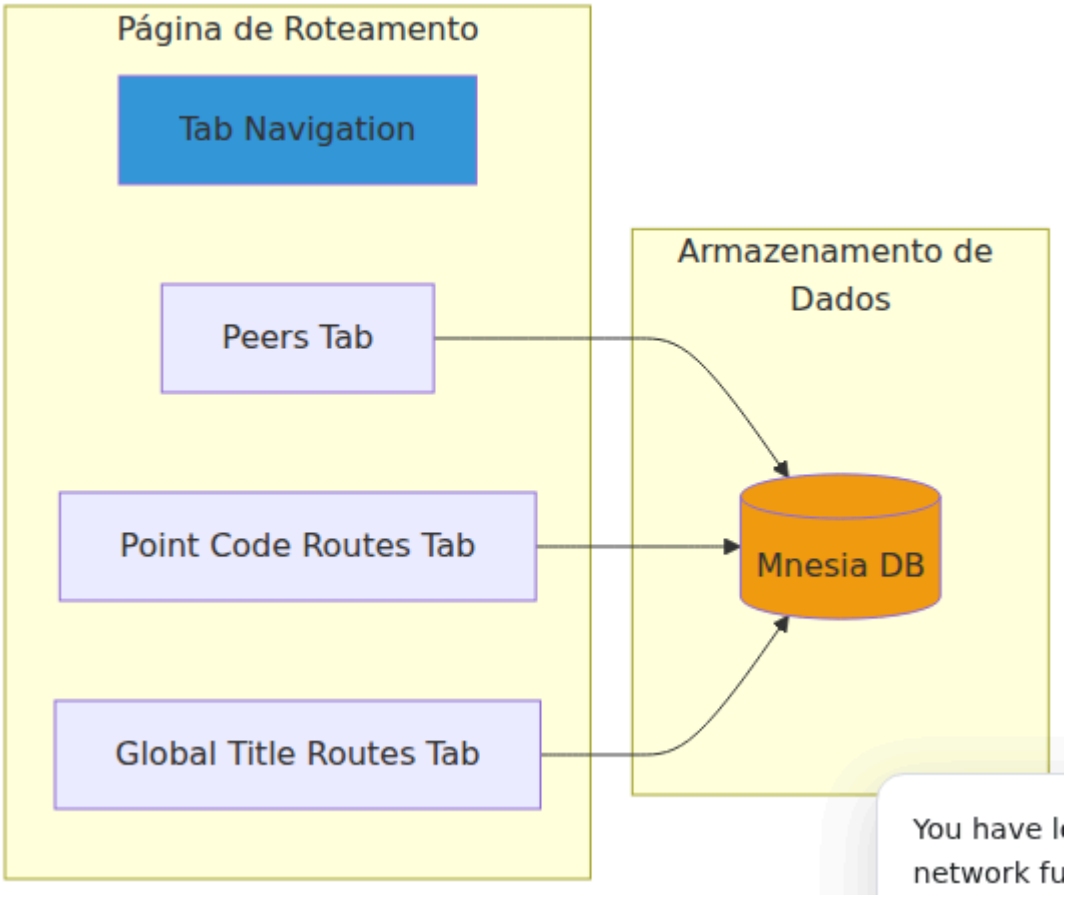
1. **Mais Específico Primeiro**: Valores de máscara mais altos (mais específicos) têm precedência
2. **Campo de Prioridade**: Números de prioridade mais baixos roteiam primeiro (1 = maior prioridade)
3. **Status do Par**: Apenas rotas para pares `ativos` são usadas

Assinantes Ativos

Página: `/subscribers`
Modo: Apenas HLR
Atualização Automática: A cada 2 segundos

Exibe rastreamento em tempo real de assinantes que enviaram solicitações de UpdateLocation.

Recursos da Página



Colunas da Tabela de Assinantes

Coluna	Descrição	Exemplo
IMSI	IMSI do assinante	`"50557123456789"`
Número VLR	Endereço GT do VLR atual	`"555123155"`
Número MSC	Endereço GT do MSC atual	`"555123155"`
Atualizado Em	Último timestamp de UpdateLocation	`"2025-10-25 14:23:45 UTC"`
Duração	Tempo desde o registro	`"2h 15m 34s"`

Resumo de Estatísticas

Quando assinantes estão presentes, um cartão de resumo exibe:

- **Total Ativo**: Total de assinantes registrados
- **VLRs Únicos**: Número de endereços VLR distintos
- **MSCs Únicos**: Número de endereços MSC distintos

Limpando Assinantes

Botão Limpar Todos: Remove todos os assinantes ativos do rastreador.

Confirmação: Requer confirmação antes de limpar (não pode ser desfeito).

Caso de Uso: Limpar registros de assinantes obsoletos após manutenção ou teste de rede.

Atualização Automática

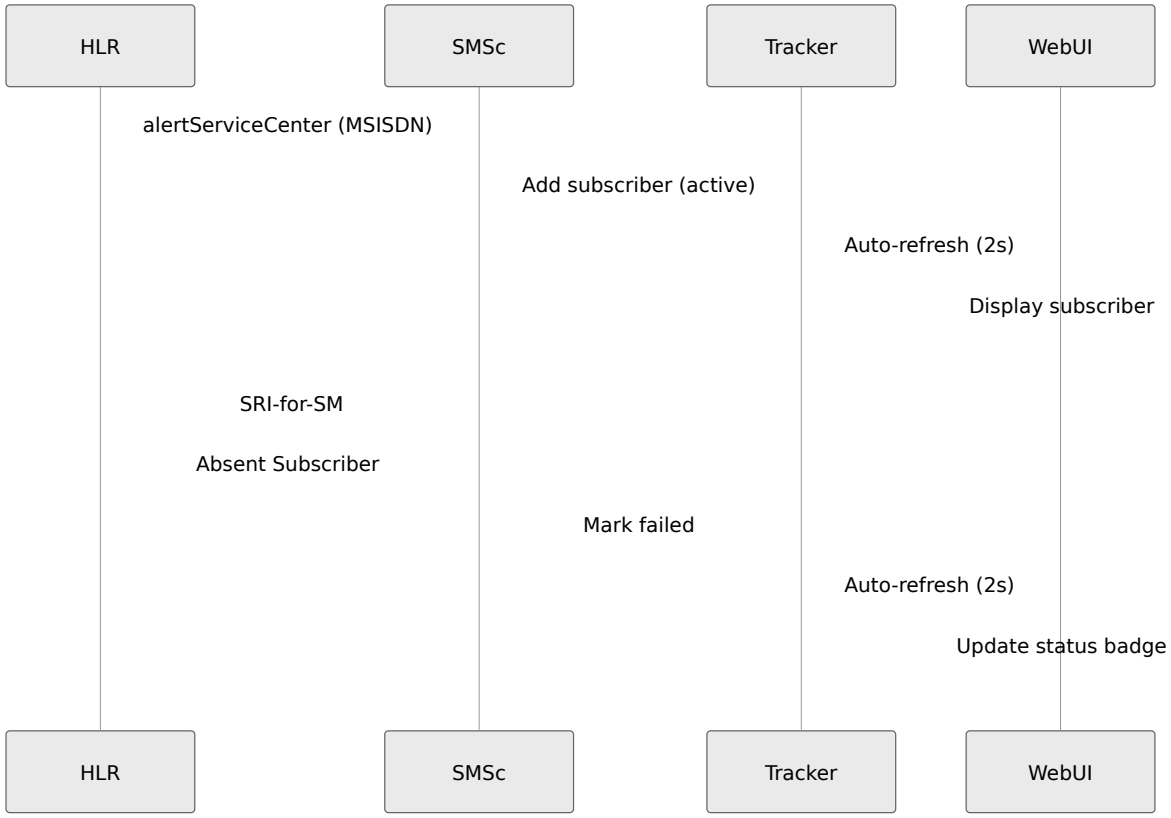
A página atualiza automaticamente a cada **2 segundos** para mostrar atualizações em tempo real dos assinantes.

Assinantes SMSc

Página: `/smsc_subscribers`
Modo: Apenas SMSc
Atualização Automática: A cada 2 segundos

Exibe rastreamento em tempo real de assinantes com base nas mensagens alertServiceCenter recebidas dos HLRs, status de entrega de mensagens e rastreamento de falhas.

Recursos da Página



Colunas da Tabela de Assinantes

Coluna	Descrição	Exemplo
MSISDN	Número de telefone do assinante	`"15551234567"`
IMSI	IMSI do assinante	`"001010123456789"`
HLR GT	HLR GT que enviou alertServiceCenter	`"15551111111"`
Msgs Enviadas	Contagem de mensagens MT-FSM enviadas	`5`
Msgs Recebidas	Contagem de mensagens MO-FSM recebidas	`2`
Status	`Ativo` ou `Falhou` (codificado por cor)	`● Ativo`
Última Atualização	Timestamp da última atualização	`"2025-10-30 14:23:45 UTC"`
Duração	Tempo desde a última atualização	`"15m 34s"`

Indicadores de Status

- ● **Ativo** (Verde): Assinante é acessível, último alertServiceCenter recebido com sucesso
- ○ **Falhou** (Vermelho): Última tentativa de entrega falhou (erro SRI-for-SM ou assinante ausente)

Resumo de Estatísticas

Quando assinantes estão presentes, um cartão de resumo exibe:

- **Total Rastreado**: Total de assinantes rastreados
- **Ativos**: Número de assinantes com status ativo
- **Falhados**: Número de assinantes com status falhado
- **HLRs Únicos**: Número de HLRs distintos enviando alertas

Gerenciando Assinantes

Botão Remover: Remove assinante individual do rastreamento.

Botão Limpar Todos: Remove todos os assinantes rastreados.

Confirmação: Limpar Todos requer confirmação antes de limpar (não pode ser desfeito).

Caso de Uso:

- Remover entradas obsoletas após problemas de rede
- Limpar dados de teste após desenvolvimento
- Monitorar quais HLRs estão enviando alertas

Contadores de Mensagens

O rastreador incrementa automaticamente os contadores:

- **Mensagens Enviadas**: Incrementado quando SRI-for-SM é bem-sucedido e MT-FSM é enviado
- **Mensagens Recebidas**: Incrementado quando MO-FSM é recebido do assinante

Atualização Automática

A página atualiza automaticamente a cada **2 segundos** para mostrar atualizações em tempo real dos assinantes e status.

Operações Comuns

Pesquisa e Filtragem

Atualmente, a Interface Web não inclui funcionalidade de pesquisa/filtragem embutida. Para encontrar rotas específicas:

1. Use a função de busca do seu navegador (Ctrl+F / Cmd+F)
2. Pesquise por nomes de pares, códigos de ponto ou prefixos GT

Operações em Lote

Para realizar alterações em lote nas rotas:

1. **Opção 1**: Use a API REST para acesso programático
2. **Opção 2**: Edite `config/runtime.exs` e reinicie a aplicação
3. **Opção 3**: Use a Interface Web para alterações individuais nas rotas

Exportar/Importar

Nota: A Interface Web atualmente não suporta exportação ou importação de tabelas de roteamento. As rotas são:

- Armazenadas em arquivos de banco de dados Mnesia
- Configuradas em `config/runtime.exs`

Para fazer backup das rotas:

1. **Mnesia**: Faça backup do diretório `Mnesia.{node_name}/`
2. **Config**: Controle de versão de `config/runtime.exs`

Comportamento de Atualização Automática

Diferentes páginas têm diferentes intervalos de atualização:

Página	Intervalo de Atualização	Razão
Gerenciamento de Roteamento	5 segundos	Mudanças de rota são infrequentes
Assinantes Ativos	2 segundos	O estado do assinante muda frequentemente
Status M3UA	Varia por página	Monitoramento do estado da conexão

Conexão WebSocket: Todas as páginas usam conexões WebSocket do Phoenix LiveView para atualizações em tempo real.

Interrupção de Rede: Se a conexão WebSocket for perdida, a página tentará reconectar automaticamente.

Solução de Problemas

Página Não Carregando

1. **Verifique o Certificado HTTPS**: Certifique-se de que `priv/cert/omnitouch.crt` e `priv/cert/omnitouch.pem` estão presentes
2. **Verifique a Porta 8087**: Verifique se as regras do firewall permitem tráfego HTTPS na porta `8087`
3. **Aplicação em Execução**: Confirme se a aplicação está em execução com `iex -S mix`

4. **Console do Navegador**: Verifique se há erros de certificado SSL (avisos de certificado autoassinado)

Rotas Não Persistindo

1. **Verifique o Armazenamento Mnesia**: Verifique `mnesia_storage_type: :disc_copies` na configuração
2. **Diretório Mnesia**: Certifique-se de que o diretório Mnesia é gravável
3. **Verifique os Logs**: Procure por erros do Mnesia nos logs da aplicação

Atualização Automática Não Funcionando

1. **Conexão WebSocket**: Verifique o console do navegador para erros do WebSocket
2. **Rede**: Verifique a conexão de rede estável
3. **Recarregar a Página**: Tente atualizar a página (F5)

Documentação Relacionada

- **Guia STP** - Configuração detalhada de roteamento
- **Guia HLR** - Gerenciamento de assinantes
- **Guia API** - API REST para acesso programático
- **Referência de Configuração** - Todos os parâmetros de configuração

---

Resumo

A Interface Web do OmniSS7 fornece gerenciamento intuitivo e em tempo real de tabelas de roteamento e rastreamento de assinantes:

 Atualizações em Tempo Real - Atualização automática mantém os dados atuais
 Armazenamento Persistente - Mnesia garante que as rotas sobrevivam a reinicializações

 Interface Baseada em Função - Páginas se adaptam ao modo operacional (STP/HLR/SMSc)
 Gerenciamento Interativo - Adicione, edite, exclua rotas sem reinicialização
 Monitoramento de Status - Status de conexão e par ao vivo

Para operações avançadas ou automação, consulte o Guia API.

Exemplos Avançados de Roteamento de Título Global

← Voltar para a Documentação Principal

Este documento fornece exemplos práticos de uso do Tipo de Tradução (TT), Indicador do Plano de Numeração (NPI) e Indicador da Natureza do Endereço (NAI) para cenários avançados de roteamento de GT.

Índice

1. Conceitos Básicos
2. Exemplos de Tipo de Tradução (TT)
3. Exemplos de Indicador do Plano de Numeração (NPI)
4. Exemplos de Natureza do Endereço (NAI)
5. Exemplos Combinados
6. Casos de Uso do Mundo Real

Conceitos Básicos

Referência de Valores de Indicador

Tipo de Tradução (TT):

- `0` = Desconhecido
- `1` = Internacional
- `2` = Nacional
- `3` = Específico da Rede

Indicador do Plano de Numeração (NPI):

- `0` = Desconhecido
- `1` = ISDN/Telefonia (E.164)
- `3` = Dados (X.121)
- `4` = Telex (F.69)
- `6` = Móvel Terrestre (E.212)

Indicador da Natureza do Endereço (NAI):

- `0` = Desconhecido
- `1` = Número do Assinante
- `2` = Reservado para Uso Nacional
- `3` = Número Nacional Significativo
- `4` = Número Internacional

Correspondência vs Transformação

- **Indicadores de origem** (source_tt, source_npi, source_nai): Usados para **corresponder** mensagens recebidas
- **Indicadores de destino** (dest_tt, dest_npi, dest_nai): Usados para **transformar** valores ao encaminhar
- Deixe os campos vazios (`nil`) para correspondência de curinga ou para preservar valores originais

---

Exemplos de Tipo de Tradução (TT)

Exemplo 1: Conversão de TT Desconhecido para Internacional

Cenário: A rede parceira envia todos os GTs com TT=0 (Desconhecido), mas seu HLR a montante espera TT=1 (Internacional).

```
%{
  gt_prefix: "44",
  peer_id: 1,
  priority: 1,
  description: "Reino Unido: Converter TT Desconhecido para
Internacional",
  source_tt: 0,      # Corresponder TT=0 recebido
  dest_tt: 1         # Transformar para TT=1
}
```

Fluxo de tráfego:

- Recebido: GT=441234567890, TT=0
- Roteamento: Corresponde gt_prefix="44", source_tt=0
- Saída: GT=441234567890, TT=1 (transformado)

Exemplo 2: Rotas Separadas por Tipo de Tradução

Cenário: Roteie diferentes valores de TT para diferentes redes.

```
# Roteie TT=0 para a Rede A
%{
  gt_prefix: "33",
  peer_id: 1,
  priority: 1,
  description: "Números franceses com TT=0 para a Rede A",
  source_tt: 0
},

# Roteie TT=2 para a Rede B
%{
  gt_prefix: "33",
  peer_id: 2,
  priority: 1,
  description: "Números franceses com TT=2 para a Rede B",
  source_tt: 2
}
```

Fluxo de tráfego:

- GT="33123456789", TT=0 → Roteia para o par 1
- GT="33123456789", TT=2 → Roteia para o par 2

Exemplos de Indicador do Plano de Numeração (NPI)

Exemplo 1: Conversão de NPI Móvel para ISDN

Cenário: A rede móvel usa NPI=6 (E.212), PSTN espera NPI=1 (E.164).

```
%{
  gt_prefix: "49",
  peer_id: 1,
  priority: 1,
  description: "Conversão móvel→ISDN alemã",
  source_npi: 6,     # Corresponder NPI=6 (Móvel/E.212)
  dest_npi: 1        # Transformar para NPI=1 (ISDN/E.164)
}
```

Caso de uso: Interconectar redes móveis e fixas com diferentes planos de numeração.

Exemplo 2: Roteamento Baseado no Plano de Numeração

Cenário: Separe o tráfego da rede de dados do tráfego de voz.

```
# Roteie o tráfego da rede de dados (NPI=3)
%{
  gt_prefix: "1",
  peer_id: 1,
  priority: 1,
  description: "Tráfego da rede de dados",
  source_npi: 3      # Corresponder NPI=3 (Dados/X.121)
},

# Roteie o tráfego da rede de voz (NPI=1)
%{
  gt_prefix: "1",
  peer_id: 2,
  priority: 1,
  description: "Tráfego da rede de voz",
  source_npi: 1      # Corresponder NPI=1 (ISDN)
}
```

Exemplos de Natureza do Endereço (NAI)

Exemplo 1: Conversão de NAI Internacional para Nacional

Cenário: Formato internacional recebido (NAI=4), converter para formato nacional (NAI=3) para roteamento doméstico.

```
%{
  gt_prefix: "1",
  peer_id: 1,
  priority: 1,
  description: "EUA: Formato Internacional→Nacional",
  source_nai: 4,     # Corresponder NAI=4 (Internacional)
  dest_nai: 3        # Transformar para NAI=3 (Nacional)
}
```

Fluxo de tráfego:

- Recebido: GT=12125551234, NAI=4 (formato internacional)
- Saída: GT=12125551234, NAI=3 (formato nacional)

Exemplo 2: Roteamento de NAI de Assinante vs Rede

Cenário: Roteie com base em se é um número de assinante ou número de rede.

```
# Roteie números de assinantes para HLR
%{
  gt_prefix: "44",
  peer_id: 1,
  priority: 1,
  description: "Números de assinantes do Reino Unido para HLR",
  source_nai: 1      # Corresponder NAI=1 (Número do Assinante)
},

# Roteie números de rede para MSC
%{
  gt_prefix: "44",
  peer_id: 2,
  priority: 1,
  description: "Endereços de rede do Reino Unido para MSC",
  source_nai: 3      # Corresponder NAI=3 (Número Nacional
Significativo)
}
```

Exemplos Combinados

Exemplo 1: Normalização de Tráfego SMS

Cenário: Normalizar todo o tráfego SMS de várias fontes antes de roteá-lo para HLR.

```
%{
  gt_prefix: "86",
  source_ssn: 8,           # Corresponder SSN=8 (SMSC)
  peer_id: 1,
  dest_ssn: 6,             # Reescrever para SSN=6 (HLR)
  priority: 1,
  description: "Normalização de SMS chineses",
  source_tt: 0,            # Corresponder TT=0 (Desconhecido)
  dest_tt: 2,              # Transformar para TT=2 (Nacional)
  source_npi: 6,           # Corresponder NPI=6 (Móvel)
  dest_npi: 1,             # Transformar para NPI=1 (ISDN)
  source_nai: 4,           # Corresponder NAI=4 (Internacional)
  dest_nai: 3              # Transformar para NAI=3 (Nacional)
}
```

O que isso faz:

- Corresponde ao tráfego SMS (SSN=8) para números chineses (prefixo GT "86")
- Normaliza todos os indicadores para valores padrão
- Reescreve o SSN de destino de SMSC (8) para HLR (6)
- Perfeito para cenários de interconexão com uso inconsistente de indicadores

Exemplo 2: Roteamento Multi-Operadora com Conversão de Formato

Cenário: Roteie para diferentes operadoras com base no TT, convertendo formatos conforme necessário.

```
# Operadora A: Espera TT Nacional e NPI ISDN
%{
  gt_prefix: "44",
  peer_id: 1,
  priority: 1,
  description: "Operadora A do Reino Unido: Normalizar para
Nacional/ISDN",
  source_tt: 0,            # Corresponder qualquer TT=0
  dest_tt: 2,              # Converter para TT=2 (Nacional)
  source_npi: 6,           # Corresponder NPI Móvel
  dest_npi: 1              # Converter para NPI ISDN
},

# Operadora B: Espera TT Internacional, preservar NPI
%{
  gt_prefix: "44",
  peer_id: 2,
  priority: 2,             # Prioridade mais baixa (número maior)
  description: "Operadora B do Reino Unido: Converter para TT
Internacional",
  source_tt: 2,            # Corresponder TT Nacional
  dest_tt: 1               # Converter para TT Internacional
  # NPI deixado vazio = preservar original
}
```

Casos de Uso do Mundo Real

Caso de Uso 1: Integração de Sistema Legado

Problema: Integrar rede moderna com sistema legado que usa convenções de indicadores diferentes.

Solução: Transformar todos os indicadores para o formato legado.

```
%{
  gt_prefix: "555",
  peer_id: 1,
  priority: 1,
  description: "Normalização de HLR legado",
  # Corresponder a qualquer formato moderno
  source_tt: 1,            # Moderno: Internacional
  dest_tt: 0,              # Legado: Desconhecido
  source_npi: 1,           # Moderno: ISDN
  dest_npi: 0,             # Legado: Desconhecido
  source_nai: 4,           # Moderno: Internacional
  dest_nai: 0              # Legado: Desconhecido
}
```

Caso de Uso 2: Padronização de Interconexão Internacional

Problema: Múltiplos parceiros internacionais enviam tráfego com indicadores inconsistentes.

Solução: Normalizar todo o tráfego recebido para um formato consistente.

```
# O parceiro A envia TT=0, converter para TT padrão=1
%{
  gt_prefix: "1",
  peer_id: 1,
  priority: 1,
  description: "Normalização do parceiro A dos EUA",
  source_tt: 0,
  dest_tt: 1,
  dest_npi: 1,             # Garantir NPI ISDN
  dest_nai: 4              # Garantir NAI Internacional
},

# O parceiro B envia TT=3 (Específico da Rede), converter para
TT=1
%{
  gt_prefix: "44",
  peer_id: 2,
  priority: 1,
  description: "Normalização do parceiro B do Reino Unido",
  source_tt: 3,
  dest_tt: 1,
  dest_npi: 1,
  dest_nai: 4
}
```

Caso de Uso 3: Gerenciamento de Tráfego de Roaming

Problema: A rede doméstica e os parceiros de roaming usam formatos de endereço diferentes.

Solução: Converter o tráfego de roaming para o formato da rede doméstica.

```
# Roaming de entrada: Converter para formato da rede doméstica
%{
  gt_prefix: "12025",      # Área de Washington DC
  peer_id: 1,
  priority: 1,
  description: "Normalização de roaming de entrada",
  source_nai: 4,           # Internacional do parceiro de roaming
  dest_nai: 3,             # Nacional para a rede doméstica
  source_npi: 6,           # Rede móvel
  dest_npi: 1              # ISDN para HLR doméstico
}
```

Caso de Uso 4: Migração de Rede

Problema: Migrar gradualmente de um antigo esquema de endereçamento para um novo esquema.

Solução: Roteie com base nos indicadores de origem, transformando para o destino.

```
# Tráfego de formato antigo: Transformar para novo formato
%{
  gt_prefix: "999",
  peer_id: 1,
  priority: 1,
  description: "Migração de formato legado para novo",
  source_tt: 0,            # Antigo: Desconhecido
  dest_tt: 2,              # Novo: Nacional
  source_npi: 0,           # Antigo: Desconhecido
  dest_npi: 1,             # Novo: ISDN
  source_nai: 0,           # Antigo: Desconhecido
  dest_nai: 3              # Novo: Nacional
},

# Tráfego de novo formato: Passar sem alterações
%{
  gt_prefix: "999",
  peer_id: 1,
  priority: 2,
  description: "Passagem de novo formato",
  source_tt: 2,            # Já no novo formato
  # Sem campos dest_ = preservar todos os valores
}
```

Testando Suas Rotas

Via Interface Web

1. Navegue até **Roteamento → Rotas de Título Global**
2. Clique em **"Mostrar Roteamento Avançado (TT/NPI/NAI)"**
3. Preencha seus indicadores de origem e destino
4. Clique em **Adicionar Rota**
5. Vá para a página de **Teste de Roteamento** para testar a rota

Via API

O endpoint de rota GT está montado em `/routing/gt-routes` no listener da API (porta **8445**, TLS habilitado). O corpo JSON usa o campo `prefix` (não `gt_prefix`), e `prefix` e `peer_id` são obrigatórios. Os correspondentes e reescritas de TT/NPI/NAI são opcionais.

```
curl -k -X POST https://localhost:8445/routing/gt-routes \
  -H "Content-Type: application/json" \
  -d '{
    "prefix": "44",
    "peer_id": 1,
    "priority": 1,
    "description": "Conversão TT do Reino Unido",
    "source_tt": 0,
    "dest_tt": 1
  }'
```

Campo	Tipo	Obrigatório	Descrição
`prefix`	String	Sim	Prefixo GT a ser correspondido
`peer_id`	Inteiro	Sim	ID do par alvo (`0` = DESCARTAR)
`priority`	Inteiro	Não	Prioridade da rota (padrão `1`, menor vence)
`description`	String	Não	Descrição legível por humanos
`source_ssn` / `dest_ssn`	Inteiro	Não	Correspondência / reescrita do SSN da Parte Chamadora
`source_tt` / `dest_tt`	Inteiro	Não	Correspondência / reescrita do Tipo de Tradução
`source_npi` / `dest_npi`	Inteiro	Não	Correspondência / reescrita do Indicador do Plano de Numeração
`source_nai` / `dest_nai`	Inteiro	Não	Correspondência / reescrita do Indicador da Natureza do Endereço

Para excluir uma rota, emita um `DELETE` para `/routing/gt-routes/{prefix}:{peer_id}` (por exemplo, `/routing/gt-routes/44:1`).

Testando a Correspondência de Rota

Use a página de Teste de Roteamento para simular tráfego recebido com indicadores específicos:

1. Insira o prefixo GT (por exemplo, "441234567890")
2. Defina os valores de TT, NPI, NAI
3. Clique em "Testar Rota"

4. Veja qual rota corresponde e quais transformações se aplicam

---

Solução de Problemas

Rotas Não Correspondendo

Verifique a ordem de especificidade:

1. O prefixo GT mais longo vence
2. Em seguida, source_ssn específico sobre curinga
3. Em seguida, source_tt específico sobre curinga
4. Em seguida, source_npi específico sobre curinga
5. Em seguida, source_nai específico sobre curinga
6. Finalmente, o número de prioridade mais baixo vence

Transformações Não Aplicadas

Verifique se os campos dest_ estão definidos:

- Campos dest_tt/dest_npi/dest_nai vazios significam "preservar valor original"
- Deve-se definir explicitamente dest_* para aplicar a transformação

Problemas de Rota Duplicada

A tabela :bag permite duplicatas:

- A tabela de rotas GT é do tipo `:bag` (permite múltiplos registros com a mesma chave)
- A chave é `{gt_prefix, source_ssn}`
- Use TT/NPI/NAI para diferenciar rotas com o mesmo prefixo+SSN
- Exclua rotas duplicadas indesejadas

Melhores Práticas

1. **Documente suas conversões:** Use descrições de rota descritivas
2. **Teste minuciosamente:** Use a página de Teste de Roteamento antes da produção
3. **Monitore logs:** Verifique os logs INFO para transformações aplicadas
4. **Comece simples:** Adicione rotas de prefixo básicas primeiro, depois adicione regras de indicadores
5. **Use curingas com sabedoria:** Campos source_* vazios correspondem a qualquer valor
6. **Preserve quando possível:** Transforme apenas o que precisa ser alterado
7. **Considere a especificidade:** Rotas mais específicas substituem as menos específicas

---

Para mais informações, veja:

- ← Voltar para a Documentação Principal
- Guia STP - Documentação completa de roteamento
- Referência de Configuração - Todas as opções de configuração
- Guia da API - Documentação da API REST

Roteamento de Título Global M3UA STP

← Voltar para a Documentação Principal

Visão Geral

O M3UA STP suporta **roteamento de Título Global (GT)** além do roteamento por Código de Ponto (PC). O roteamento GT roteia mensagens SS7 com base no Título Global da Parte Chamadora na camada SCCP, com fallback automático para o roteamento por PC.

Como Funciona

Prioridade de Roteamento

1. **Roteamento GT (se habilitado)**: Extrair GT Chamado da SCCP → Consulta na tabela de roteamento GT
2. **Roteamento PC (fallback)**: Extrair DPC do MTP3 → Consulta na tabela de roteamento PC

Correspondência de Prefixo GT

- Usa o algoritmo de **correspondência de prefixo mais longo**
- Exemplo: GT "447123456789"
  - Corresponde a "447" (comprimento 3) em vez de "44" (comprimento 2)
  - Roteia para o par configurado para o prefixo "447"

Configuração

Habilitar Roteamento GT

Em `config/runtime.exs`:

```
config :omniss7,
  enable_gt_routing: true,  # Defina como false para usar apenas
roteamento PC
```

Definir Rotas GT

```
config :omniss7,
  m3ua_gt_routes: [
    # Roteia números dos EUA (começando com 1234) para o par 1
    %{gt_prefix: "1234", peer_id: 1, priority: 1, description:
"Números dos EUA"},

    # Roteia números do Reino Unido (começando com 44) para o par
2
    %{gt_prefix: "44", peer_id: 2, priority: 1, description:
"Números do Reino Unido"},

    # Rota mais específica para móvel do Reino Unido (447xxx)
    %{gt_prefix: "447", peer_id: 2, priority: 1, description:
"Móvel do Reino Unido"},

    # Roteia números australianos (começando com 61) para o par 3
    %{gt_prefix: "61", peer_id: 3, priority: 1, description:
"Números australianos"}
  ]
```

> Uma configuração STP completa e pronta para edição é enviada em `config/stp_example_runtime.exs`. Consulte o Guia STP para saber como habilitá-lo.

Exemplos de Uso

Exemplo 1: Roteamento GT Básico

Mensagem: SCCP UDT com GT Chamado = "447700123456"

Decisão de Roteamento:

- GT extraído: "447700123456"
- Melhor correspondência: "447" (móvel do Reino Unido) → par 2
- Mensagem roteada para o par 2

Exemplo 2: GT Não Encontrado, Fallback para PC

Mensagem: SCCP UDT com GT Chamado = "331234567890", DPC = 100

Decisão de Roteamento:

- GT extraído: "331234567890"
- Nenhuma rota GT encontrada para "33"
- Fallback para roteamento PC
- DPC 100 → par 1 (da tabela de roteamento PC)
- Mensagem roteada para o par 1

Exemplo 3: Nenhum GT Presente

Mensagem: M3UA DATA sem SCCP, apenas MTP3 com DPC = 200

Decisão de Roteamento:

- Sem camada SCCP, sem GT
- Usar apenas roteamento PC
- DPC 200 → par 2
- Mensagem roteada para o par 2

Gerenciamento em Tempo de Execução

Essas funções são expostas por `M3UARouting` e podem ser chamadas de um console `iex -S mix`.

Consultar Rotas GT

```
# Obter todas as rotas GT
{:ok, routes} = M3UARouting.get_all_gt_routes()

# Cada registro é uma tupla de 14 elementos. Os campos são:
# {:m3ua_gt_route, gt_prefix, source_ssn, peer_id, dest_ssn,
priority,
#  description, enabled, source_tt, dest_tt, source_npi, dest_npi,
#  source_nai, dest_nai}
#
# Exemplo de saída (source_ssn e os campos de reescrita/indicador
padrão para nil):
[
  {:m3ua_gt_route, "1234", nil, 1, nil, 1, "Números dos EUA",
true, nil, nil, nil, nil, nil, nil},
  {:m3ua_gt_route, "44", nil, 2, nil, 1, "Números do Reino Unido",
true, nil, nil, nil, nil, nil, nil},
  {:m3ua_gt_route, "447", nil, 2, nil, 1, "Móvel do Reino Unido",
true, nil, nil, nil, nil, nil, nil},
  {:m3ua_gt_route, "61", nil, 3, nil, 1, "Números australianos",
true, nil, nil, nil, nil, nil, nil}
]
```

Adicionar Rota GT Dinamicamente

`add_gt_route/3` aceita o prefixo e o ID do par como parâmetros; tudo o mais (`priority`, `description`, `source_ssn`, `dest_ssn`, os correspondentes TT/NPI/NAI e reescritas, e `enabled`) é uma opção de palavra-chave.

```
# Assinatura: add_gt_route(gt_prefix, peer_id, opts \ [])

# Adicionar uma rota para números alemães
M3UARouting.add_gt_route("49", 4, priority: 1, description:
"Números da Alemanha")

# Adicionar uma rota de SMS que corresponde ao SSN 8 (SMSC) e
reescreve o SSN para 6 (HLR)
M3UARouting.add_gt_route("55", 4,
  source_ssn: 8,
  dest_ssn: 6,
  priority: 1,
  description: "Tráfego de SMS para HLR"
)
```

Remover Rota GT

```
# Remover rota (gt_prefix, peer_id, source_ssn opcional)
M3UARouting.remove_gt_route("49", 4)
```

Testar Consulta GT

```
# Testar qual par lidaria com este GT
# Assinatura: lookup_peer_by_gt(global_title, source_ssn \ nil,
opts \ [])
{:ok, peer, rewrites} =
M3UARouting.lookup_peer_by_gt("447700123456")

# `peer` é o registro m3ua_peer de 13 elementos:
# {:m3ua_peer, peer_id, peer_name, protocol, role, local_ip,
local_port,
#  remote_ip, remote_port, routing_context, point_code,
network_indicator, status}
#
# `rewrites` é um MAP das reescritas de indicador a serem
aplicadas ao encaminhar.
# valores nil significam "manter o original":
#   %{dest_ssn: nil, dest_tt: nil, dest_npi: nil, dest_nai: nil}
```

Quando uma rota especifica reescritas, elas aparecem no mapa. Por exemplo, uma rota "55" com `dest_ssn: 6`:

```
{:ok, peer, rewrites} =
M3UARouting.lookup_peer_by_gt("5512345678", 8)
# rewrites => %{dest_ssn: 6, dest_tt: nil, dest_npi: nil,
dest_nai: nil}
```

Uma rota com `peer_id: 0` (DROP) retorna `{:error, :dropped}`; um GT não correspondido retorna `{:error, :no_gt_route}`.

Registro

O STP registra decisões de roteamento GT:

```
[info] Roteando M3UA DATA: OPC=1000, DPC=200, GT
Chamado="447700123456" (do par 1)
[info] Roteando por GT: 447700123456
[info] Mensagem roteada do par 1 para o par 2
```

Se o roteamento GT falhar:

```
[info] Roteando M3UA DATA: OPC=1000, DPC=200, GT
Chamado="331234567" (do par 1)
[info] Nenhuma rota GT encontrada para 331234567
[info] Mensagem roteada do par 1 para o par 2 (via roteamento PC)
```

Suporte SCCP

Mensagens SCCP Suportadas

Mensagem	Código	Suporte
UDT (Unitdata)	0x09	Completo: extração de GT, reescrita de indicador, roteamento
UDTS (Unitdata Service)	0x0A	Completo: gerado como uma mensagem de retorno em caso de falha de roteamento
XUDT (Extended Unitdata)	0x11	Completo: extração de GT, reescrita de indicador, decremento de contador de saltos, mensagem de retorno
XUDTS (Extended Unitdata Service)	0x12	Completo: gerado como uma mensagem de retorno em caso de falha de roteamento
LUDT (Long Unitdata)	0x13	Completo: extração de GT, reescrita de indicador, decremento de contador de saltos, mensagem de retorno
LUDTS (Long Unitdata Service)	0x14	Completo: gerado como uma mensagem de retorno em caso de falha de roteamento

XUDT e LUDT carregam um contador de saltos SCCP. O STP o decrementa em cada salto de trânsito e, se chegar a zero, descarta a mensagem e a retorna ao originador (quando a devolução em caso de erro é solicitada). Consulte o Guia STP para detalhes.

Formatos GT Suportados

- **GTI 0-4**: Todos os tipos de indicadores GT
- **Codificação BCD**: Codificação padrão de dígitos de telefonia
- **Natureza do Endereço**: Todos os tipos (Internacional, Nacional, etc.)

Desempenho

- **Consulta GT**: O(n) onde n = número de rotas GT
- **Correspondência de prefixo**: Otimizada com ordenação de correspondência mais longa primeiro
- **Recomendação**: Mantenha a contagem de rotas GT < 1000 para desempenho ideal

Solução de Problemas

GT Não Está Sendo Extraído

Problema: Os logs mostram `Called GT=nil`

Soluções:

1. Verifique se o tipo de mensagem SCCP é UDT/XUDT/LUDT
2. Verifique se o Endereço da Parte Chamadora contém um Título Global
3. Habilite o registro de depuração para ver os detalhes da análise SCCP

Par Errado Selecionado

Problema: Mensagem roteada para o par errado

Soluções:

1. Verifique a configuração do prefixo GT (a correspondência mais longa vence)
2. Verifique se o `peer_id` corresponde aos pares configurados

3. Verifique os valores de prioridade (menor = maior prioridade)

Fallback para PC Não Funcionando

Problema: Mensagem descartada quando GT não encontrado

Soluções:

1. Verifique se `enable_gt_routing` está `true`
2. Verifique se a tabela de roteamento PC tem uma rota para o DPC
3. Certifique-se de que o par para o PC está ativo

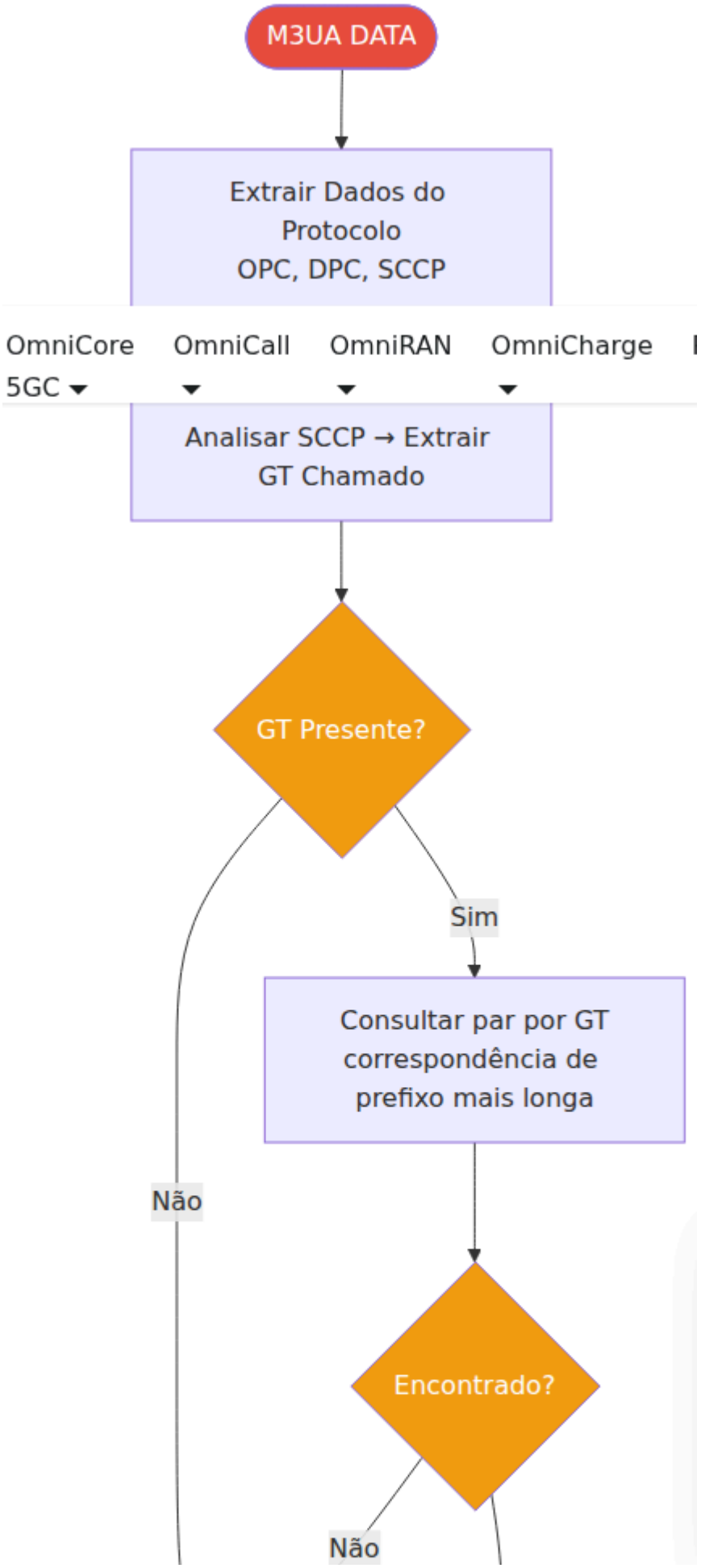
Melhores Práticas

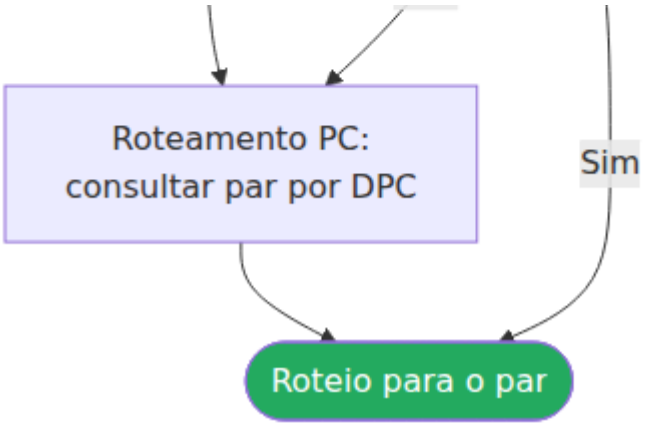
1. **Use Prefixos Específicos**: Rotas mais específicas substituem as menos específicas
   - Bom: "447", "448", "449" para diferentes operadoras do Reino Unido
   - Ruim: Um único "44" para todo o Reino Unido
2. **Documente as Rotas**: Use o campo de descrição

   ```
   M3UARouting.add_gt_route("1234", 1, priority: 1,
     description: "ACME Corp - Costa Leste dos EUA")
   ```

3. **Teste as Rotas**: Use `lookup_peer_by_gt/1` para verificar
4. **Monitore**: Verifique os logs para falhas de roteamento
5. **Fallback**: Sempre configure rotas PC como backup

Arquitetura


M3UA DATA
Extrair Dados do Protocolo OPC, DPC, SCCP
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Analisar SCCP → Extrair GT Chamado
GT Presente?
Sim
Não
Consultar par por GT correspondência de prefixo mais longa
Encontrado?
Não



OmniSS7 por Omnitouch Network Services