OmniLCS □作指南

OmniLCS □一个集成的 LTE/GSM 位置和蜂窝广播平台。它结合了 **E-SMLC**（演进服务移动位置中心）用于用户设备定位，以及 **CBC**（蜂窝广播中心）用于公共警报和商业广播服务，所有这些功能都在一个 Elixir/OTP 应用程序中实现。

架构概述

功能概述

E-SMLC: 位置服务

- 定位方法：蜂窝 ID、增强蜂窝 ID (E-CID)、GNSS/A-GPS、OTDOA
- **LCS-AP over SCTP (SLs 接口)**：根据 3GPP TS 29.171 使用原生 LCS-AP 与 MME 通信，端□ 9082, PPID 29
- **LPPA/LPP** 中继：通过 LCS-AP 连接导向信息向 eNB 和 UE 发送定位协议消息
- 蜂窝数据库：存储蜂窝 ID 和 OTDOA 定位的蜂窝位置。支持从华为 U2020 XLSX (GSM/UMTS/LTE/NR)、特定厂商的 CSV、JSON 和 InfluxDB 同步导��。持久化到 Mnesia，以便在重启后生存
- **OTDOA** 多边测量：使用迭代最小二乘法从 RSTD 测量计算 UE 位置

- 位置日志记录：将每个位置修正持久化到 CSV 和内存历史中。配置时可选择 InfluxDB 日志记录
- 用户跟踪：根据 IMSI 进行定期位置轮询，具有可配置的间隔和方法。结果存储在 Mnesia 中，并支持 KML/CSV 导出
- 虚拟驱动测试：多 IMSI 活动结合 GNSS 定位和 E-CID 信号测量 (RSRP/RSRQ)。活动级管理，实时监控，InfluxDB 日志记录，以及带有信号质量数据的 KML/CSV 导出

GMLC: 网关移动位置中心

- **Le** 接口：根据 3GPP TS 29.172 为外部 LCS 客户端（PSAP、车队管理、合法拦截）提供的 Diameter 接口
- 客户端授权：每个客户端的访问控制，具有类型匹配和可配置的速率限制
- 定期位置：延迟会话在可配置的间隔内执行位置修正，并进行 InfluxDB 日志记录和 Diameter LRR 交付
- 触发位置（地理围栏）：针对圆形或多边形区域的进入/离开/在内触发的区域事件订阅
- 位置报告交付：对于每个延迟修正，向发起客户端发送 Diameter 位置报告请求 (LRR) 消息

CBC: 蜂窝广播

- **2G CBSP**: 在端口 48049 上的 TCP 服务器，接受来自 BSC 的连��
- **3G SABP**: 在端口 3452 上的 TCP（Iu-BC 接口，3GPP TS 25.419，传输根据 TS 25.414 §7.1.3.3）。CBC 连接到配置的 RNC 对等体；同时监听来自 RNC 的入站连接（故障/重启指示）
- **4G SBC-AP**: 在端口 29168 上连接到 MME 对等体的 SCTP 客户端
- 多语言广播：同时以多种语言发送相同的警报，每种语言都有其自己的 CBS 数据编码方案和消息代码，符合 3GPP TS 23.038
- 广播更新：通过增加更新编号来更新活动广播。在新的写入替换之前，自动发送停止警告以停止旧的序列
- 消息格式：GSM 7 位和 UCS-2 编码，CBS 页面组装，序列号构造
- 警告类型：支持 ETWS（地震、海啸、测试）以及紧急警报和弹出激活
- 广播状态管理：活动广播跟踪，持久化跨重启，响应超时监控
- **Mnesia** 持久性：蜂窝数据库和 CAP 警报状态持久化到 Mnesia disc_copies 表中，能够在应用重启后生存而无需重新导入
- **PWS** 程序：处理来自 MMEs 的 PWS-Restart-Indication 和 PWS-Failure-Indication
- **CAP** 警报摄取：通过 HTTP POST 或 Atom 订阅轮询接受外部警报机构发送的通用警报协议 (CAP) v1.2 XML，将多边形警告区域解析为 TACs/LACs，并触发蜂窝广播，需可选的运营商批准

控制接口

- **REST API**: 在端口 8443 上的 HTTPS，具有位置请求、蜂窝管理和系统状态的端点
- **LiveView** 控制面板：在端口 443 上的 HTTPS，具有实时仪表板、位置测试、蜂窝数据库管理、Diameter 对等体监控和 CBC 广播组合

文档结构

文档	描述
配置参考	所有配置参数及其表格和示例
蜂窝广播操作	CBSP (2G)、SABP (3G) 和 SBC-AP (4G) 操作、消息格式、故障排除
CAP 警报操作	CAP v1.2 摄取、多边形解析、批准工作流、订阅轮询
位置服务	E-SMLC 定位方法、LCS-AP 接口、蜂窝数据库、OTDOA
用户跟踪与驱动测试	定期跟踪、虚拟驱动测试活动、RSRP/RSRQ 测量、KML/CSV 导出
GMLC 与 Le 接口	外部 LCS 客户端访问、定期和地理围栏位置、InfluxDB 日志记录
REST API 参考	所有 API 端点及请求/响应示例
Web 接口指南	LiveView 控制面板页面和功能

接口摘要

接口	协议	传输	端口	方向	**3GPP** 参考
SLs (E-SMLC ↔ MME)	LCS-AP	SCTP	9082	OmniLCS 连接到 MMEs	TS 29.171
SLg/Le (GMLC ↔ LCS 客户端/DRA)	Diameter	SCTP	3868	从 LCS 客户端入站，向 DRA 出站	TS 29.172
CBSP	CBSP	TCP	48049	BSC 连接到 OmniLCS	TS 48.049
SABP (Iu-BC)	SABP	TCP	3452	向 RNC 出站 / 从 RNC 入站	TS 25.414 §7.1.3.3
SBC-AP	SBC-AP	SCTP	29168	OmniLCS 连接到 MMEs	TS 29.168
REST API	HTTPS	TCP	8443	客户端连接到 OmniLCS	--
控制面板	HTTPS	TCP	443	浏览器连接到 OmniLCS	--
蜂窝同步	HTTP	TCP	8086	OmniLCS 查询 InfluxDB	--

监督树

OmniLCS 在一对一监督器下启动以下进程：

1. **OmniLcs.Persistence**：Mnesia 初始化和蜂窝数据库及 CAP 警报的磁盘持久性

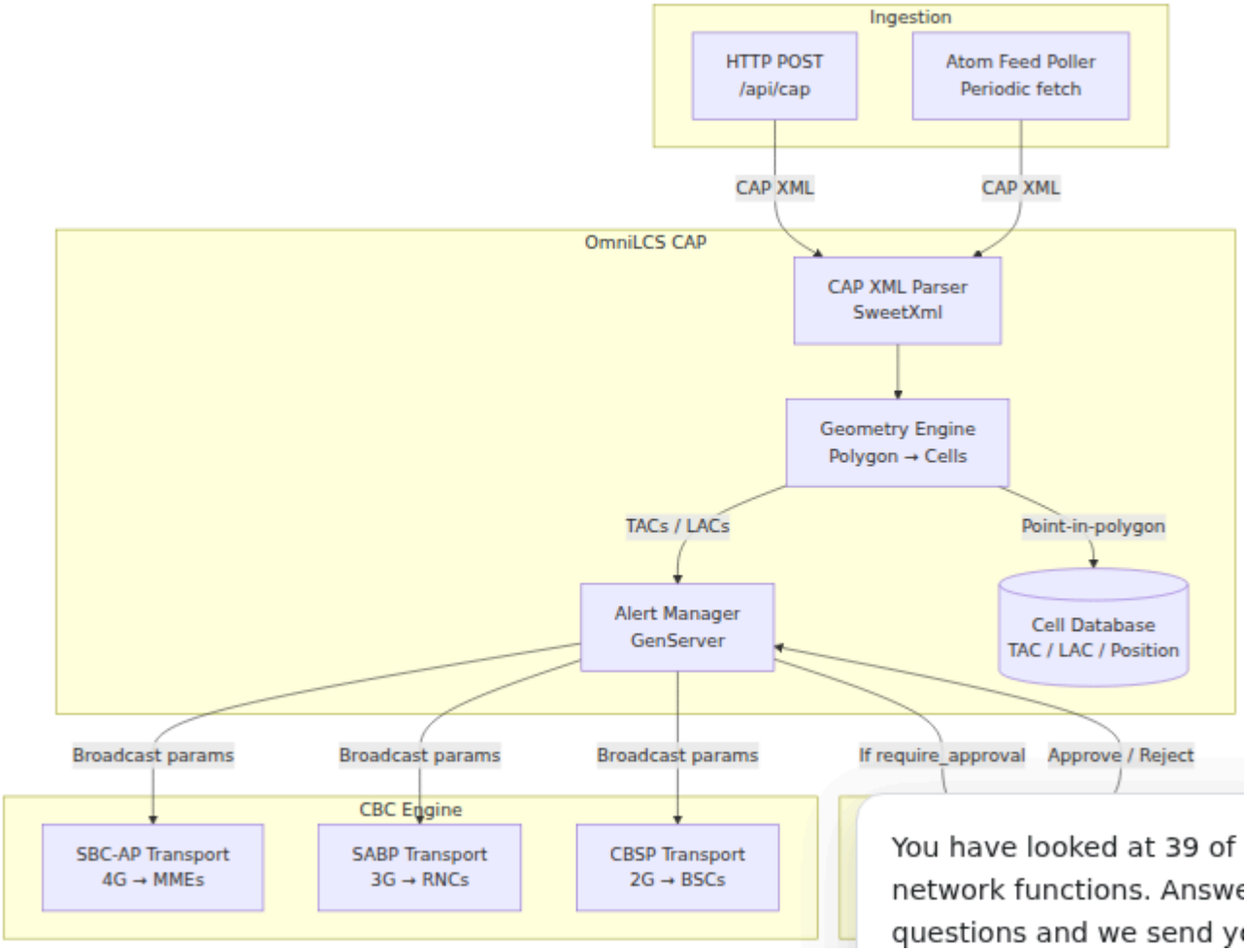
2. **OmniLcs.Context**: ETS 表初始化（cell_database、location_session、pending_transactions）；在启动时从 Mnesia 加载持久化的蜂窝
3. **OmniLcs.InfluxDb**: InfluxDB 连接池
4. **OmniLcs.Esmlc.LocationLogger**: 位置修正日志记录到 CSV 和 ETS
5. **Task.Supervisor**: 异步任务执行
6. **OmniLcs.Sls.SctpTransport**: SLs LCS-AP SCTP 客户端连接到 MMEs
7. **DiameterEx.Supervisor**: Diameter 服务和对等体管理（SLg 和其他 Diameter 接口）
8. **OmniLcs.Esmlc.CellSync**: 定期 InfluxDB 蜂窝同步
9. **OmniLcs.Cbc.CbspConnectionSupervisor**: 动态监督器用于 2G CBSP TCP 连接
10. **OmniLcs.Cbc.CbspTransport**: CBSP TCP 监听器（端口 48049）
11. **OmniLcs.Cbc.SabpConnectionSupervisor**: 动态监督器用于 3G SABP TCP 连接
12. **OmniLcs.Cbc.SabpTransport**: SABP TCP 监听器（端口 3452）
13. **OmniLcs.Cbc.Engine**: CBC 广播状态管理（2G、3G 和 4G）
14. **OmniLcs.Cbc.SctpTransport**: SBC-AP SCTP 客户端连接到 MMEs
15. **OmniLcs.Cap.AlertManager**: CAP 警报生命周期管理（解析、解析、批准、广播）；持久化到 Mnesia
16. **OmniLcs.Cap.FeedPoller**: 定期 CAP Atom 订阅轮询
17. **OmniLcs.Gmlc.ClientRegistry**: 授权的外部 LCS 客户端管理
18. **OmniLcs.Gmlc.SessionSupervisor**: 动态监督器用于定期和触发的位置会话
19. **OmniLcs.Tracking.SessionSupervisor**: 动态监督器用于用户跟踪监视会话
20. **OmniLcs.DriveTest.CampaignSupervisor**: 动态监督器用于虚拟驱动测试活动

CAP 警报□作指南

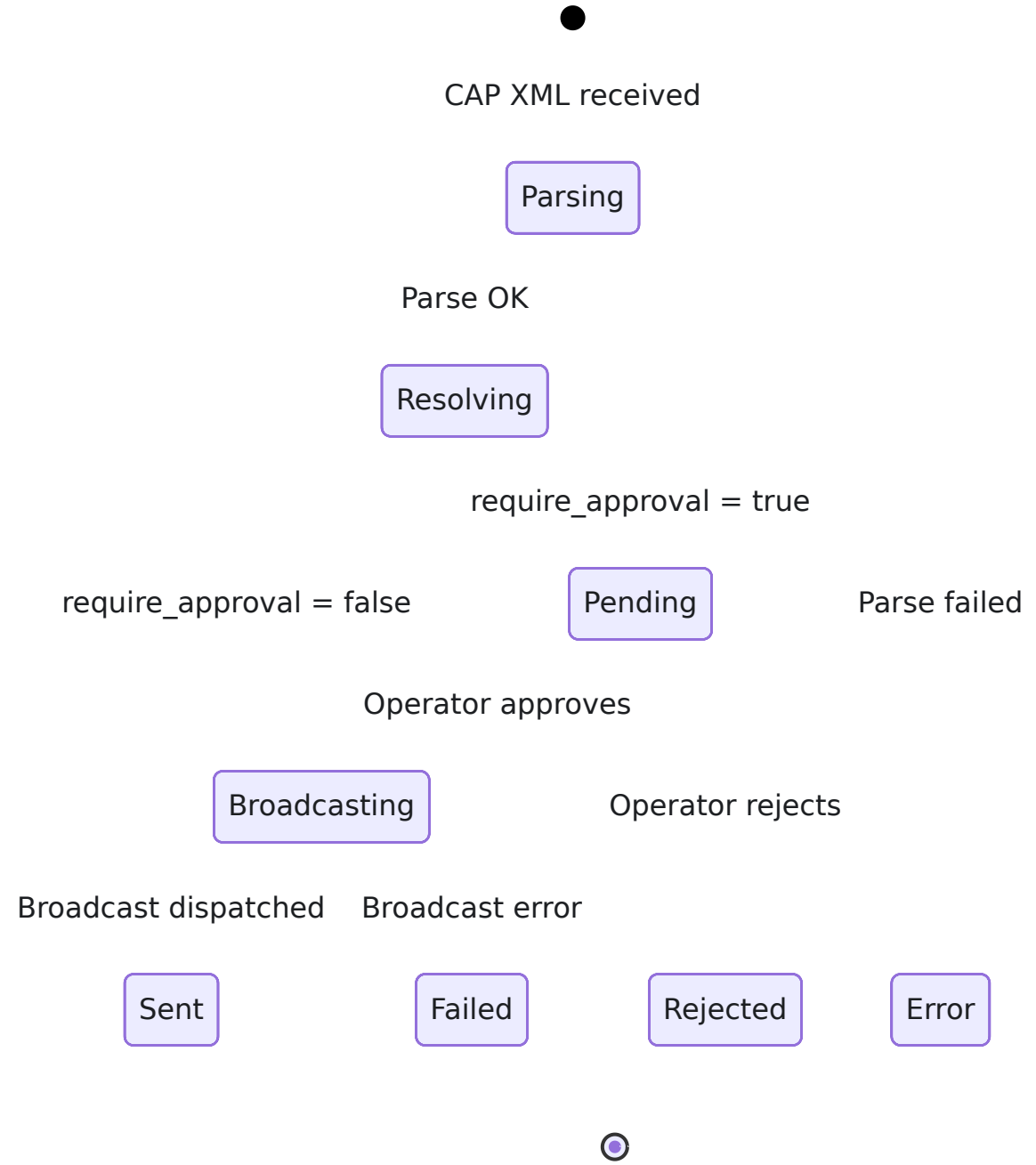
OmniLCS 接受来自外部警报机构（天气服务、应急管理机构等）的 通用警报协议（**CAP**）**v1.2** XML 消息，解析地理警告区域到网络的蜂窝基础设施，并通过现有的 CBC 引擎在 4G（SBC-AP）、3G（SABP）和 2G（CBSP）接口上触发蜂窝广播。

支持两种摄取方法：**HTTP POST**（推送）用于直接集成和 **Atom** 订阅轮询（拉取）用于监控标准 CAP 源。可配置的 手动审批步骤 允许操作员在广播前预览警报。

□□



You have looked at 39 of
network functions. Answe
questions and we send y



警报生命周期

当接收到 CAP 警报（通过 HTTP POST 或源轮询）时：

1. **CAP XML** 解析器 提取警报信封、信息块、警告区域（多边形）和 CB 参数
2. 几何引擎 使用射线投射（点在多边形内）将每个多边形与蜂窝数据库匹配，将警告区域解析为 TAC（4G）、SAI（3G）和 LAC（2G）列表
3. 如果 `require_approval` 为 `true`，则警报被排队为 待处理，并在操作员 LiveView UI 中显示以供预览和批准
4. 如果 `require_approval` 为 `false`，则警报自动批准并立即发送到 CBC 引擎
5. 在广播时，CBC 引擎向所有连接的 MMEs、RNCs 和 BSCs 发送写替换警告请求（4G SBC-AP）、写替换（3G SABP）和/或 写替换（2G CBSP）

□□

CAP □□

```
config :omnilcs, :cap,
  # Require operator approval before broadcasting
  require_approval: true,

  # PLMN identity for broadcast messages
  plmn: %{mcc: "001", mnc: "01"},

  # Use cell coverage radius for polygon matching (vs center-point
only)
  coverage_aware: false,

  # Atom feed sources to poll (empty = no polling)
  feeds: []
```

CAP 参数

参数	类型	必需	默认	描述
`require_approval`	boolean	否	`true`	当为 `true` 时，警报在广播前会排队等待操作员批准。当为 `false` 时，警报自动批准并立即广播。
`plmn`	map	否	`%{mcc: "001", mnc: "01"}`	广播消息中使用的 PLMN 身份（MCC/MNC）。必须与服务网络匹配。
`plmn.mcc`	string	□	`"001"`	移动国家代码（3 位数字）。
`plmn.mnc`	string	□	`"01"`	移动网络代码（2-3 位数字）。
`coverage_aware`	boolean	否	`false`	当为 `true` 时，多边形匹配使用每个蜂窝的覆盖半径来确定交集，而不仅仅□中心点。对于位于多边形边缘的大型蜂窝很有用。
`feeds`	list	否	`[]`	要轮询的 CAP Atom 源配置列表。请参见 Feed Configuration。

源配置

`feeds` 列表中的每个条目定义一个要监控的 CAP Atom 源。

```
config :omnilcs, :cap,
  feeds: [
    %{url: "https://alerts.weather.gov/cap/us.php?x=1",
poll_interval_seconds: 60},
    %{url: "https://feeds.meteoalarm.org/api/v1/warnings/atom",
poll_interval_seconds: 120}
  ]
```

参数	类型	必需	默认	描述
`url`	string	□	--	CAP Atom 源的 URL。必须返回嵌入 CAP 警报条目的 Atom XML（RFC 4287）。
`poll_interval_seconds`	integer	否	`60`	源轮询之间的秒数。较低的值提供更快的警报摄取，但增加网络流量。

源轮询器跟踪已见警报标识符以避免重新处理。源中的每个新 `<entry>` 都会提取其嵌入的 `<alert>` XML 并提交给警报管理器。

配置示例：完整 CAP 设置

```
config :omnilcs, :cap,
  require_approval: true,
  plmn: %{mcc: "001", mnc: "01"},
  coverage_aware: true,
  feeds: [
    %{url: "https://alerts.weather.gov/cap/us.php?x=1",
poll_interval_seconds: 60}
  ]
```

工作原理：来自 NWS 天气源的 CAP 警报每 60 秒检查一次。新警报被解析，其多边形警告区域与蜂窝数据库进行覆盖感知交集匹配。匹配的警报在 CAP 警报 LiveView 页面中显示以供操作员审核。操作员可以预览警报详细信息（消息文本、严重性、匹配的蜂窝、TACs/LACs），然后批准广播或拒绝丢弃。

蜂窝数据库：TAC、LAC 和 RAT 字段

CAP 多边形解析依赖于数据库中具有 **TAC**（4G）、**LAC**（2G 和 3G）、**SAC**（3G）和 **RAT** 字段的蜂窝。这些字段决定每个蜂窝使用的广播接口。

CAP 的蜂窝字段

字段	类型	描述
`tac`	integer	跟踪区域代码。用于 4G SBC-AP 广播。具有 TAC 的蜂窝包含在发送给 MMEs 的 TAI 列表中。
`lac`	integer	位置区域代码。用于 2G CBSP 和 3G SABP 广播。具有 LAC 的蜂窝包含在发送给 BSCs 的蜂窝列表和发送给 RNCs 的 SAI 列表中。
`rat`	string	无线接入技术：`"4g"`、`"3g"` 或 `"2g"`。信息性；`tac` 或 `lac` 的存在决定广播目标。

这些字段可以通过以下方式设置：

- **REST API** 在创建或更新蜂窝时（`POST /api/cells`，`PUT /api/cells/:id`）
- **JSON** 导入（`priv/cells.json`），每个蜂窝包含 `"tac"`、`"lac"` 和 `"rat"` 字段
- 手动输入 在蜂窝数据库 LiveView 页面中

从 InfluxDB 同步蜂窝时，现有的 TAC/LAC/RAT 值将被保留（InfluxDB 不提供这些字段，因此不会用 `nil` 覆盖）。

蜂窝 JSON 示例

```
[
  {
    "cell_id": "eNB-001-cell-01",
    "latitude": 40.7128,
    "longitude": -74.0060,
    "pci": 100,
    "earfcn": 1300,
    "radius": 500,
    "tac": 100,
    "lac": null,
    "rat": "4g"
  },
  {
    "cell_id": "BTS-001-cell-01",
    "latitude": 40.7130,
    "longitude": -74.0065,
    "pci": null,
    "earfcn": null,
    "radius": 2000,
    "tac": null,
    "lac": 5001,
    "rat": "2g"
  }
]
```

多边形解析

当 CAP 警报包含多边形警告区域时，几何引擎确定哪些蜂窝位于每个多边形内。

多边形匹配的工作原理

标准模式（`coverage_aware: false`）：如果蜂窝的中心点（纬度/经度）位于多边形内，则该蜂窝匹配，使用射线投射算法。

覆盖感知模式（`coverage_aware: true`）：如果蜂窝的覆盖区域（建模为一个带有蜂窝 `radius` 字段的圆）与多边形相交，则该蜂窝匹配。这可以捕捉到中心点刚好在多边形外但覆盖范围延伸到多边形内的蜂窝。

CAP 多边形格式

CAP v1.2 将多边形定义为以空格分隔的 `lat,lon` 对。第一个和最后一个点应相同（闭合多边形）：

```
38.47,-120.14 38.34,-119.95 38.52,-119.74 38.62,-119.89
38.47,-120.14
```

CAP XML 格式

解析器根据 OASIS 标准处理 CAP v1.2。提取的关键元素：

警报信封

元素	描述
`<identifier>`	唯一的警报标识符
`<sender>`	警报发起者
`<sent>`	警报发出时间戳
`<status>`	`Actual`、`Exercise`、`System`、`Test`、`Draft`
`<msgType>`	`Alert`、`Update`、`Cancel`、`Ack`、`Error`
`<scope>`	`Public`、`Restricted`、`Private`

信息块

每个 `<info>` 块包含特定语言/受众的警报内容：

元素	描述
`<category>`	警报类别（地理、气象、安全、安保等）
`<event>`	事件类型描述（例如，“龙卷风警告”）
`<urgency>`	立即、预期、未来、过去、未知
`<severity>`	极端、严重、中等、轻微、未知
`<certainty>`	观察到、可能、可能、不太可能、未知
`<headline>`	短警报标题
`<description>`	完整的警报描述文本（用作广播消息）
`<instruction>`	推荐的行动

CB 参数

蜂窝广播参数从每个 `<info>` 块中的 `<parameter>` 元素提取：

参数名称	描述
`CBMessageIdentifier`	用于广播的 16 位 CB 消息 ID
`CBRepetitionInterval`	广播重复之间的秒数
`CBNumberOfBroadcasts`	广播的总次数

如果这些参数在 CAP XML 中不存在，则使用默认值（消息 ID `0x1112`，重复周期 30 秒，10 次广播）。

区域块

每个 `<info>` 中的 `<area>` 可以包含：

元素	描述
`<areaDesc>`	可读的区域描述
`<polygon>`	定义警告区域的以空格分隔的 `lat,lon` 坐标对
`<circle>`	中心点和半径（供将来使用）

REST API

POST /api/cap

提交 CAP XML 警报以进行处理。

请求体

```
{
  "xml": "<alert xmlns=\"urn:oasis:names:tc:emergency:cap:1.2\">...</alert>"
}
```

参数	类型	必需	描述
`xml`	string	□	完整的 CAP v1.2 XML 警报文档

响应 **(201)** -- 警报成功提交

```
{
  "status": "ok",
  "data": {
    "id": "a1b2c3d4-e5f6-...",
    "status": "pending",
    "source": "http_post",
    "received_at": "2025-01-15T10:30:00Z",
    "matched_cells": 42,
    "tacs": [100, 101, 102],
    "lacs": [5001, 5002],
    "mcc": "001",
    "mnc": "01",
    "broadcast_params": {
      "message_id": 4370,
      "repetition_period": 30,
      "num_broadcasts": 10,
      "message_text": "Tornado Warning for Springfield County...",
      "event": "Tornado Warning",
      "severity": "Extreme",
      "urgency": "Immediate"
    }
  }
}
```

如果 `require_approval` 为 `true`，则 `status` 字段将为 `"pending"`，如果自动批准，则为 `"sent"`。

错误响应

状态	原因
400	`"xml field is required"`
422	解析错误详情

GET /api/cap

列出所有州的所有警报。

响应 **(200)**

```
{
  "status": "ok",
  "data": {
    "pending": [...],
    "active": [...],
    "history": [...]
  }
}
```

每个数组包含与 POST 响应相同格式的警报对象。

GET /api/cap/:id

按 ID 获取单个警报。

路径参数

参数	类型	描述
`id`	string	警报 UUID

响应 **(200)**

返回警报对象。

错误响应

状态	原因
404	`"Alert not found: <id>"`

PUT /api/cap/:id

批准或拒绝待处理的警报。

请求体

```
{
  "action": "approve",
  "operator": "operator1"
}
```

参数	类型	必需	描述
`action`	string	□	`"approve"` 或 `"reject"`
`operator`	string	否	审计跟踪的操作员名称。默认为 `"unknown"`。

响应 **(200)**

返回更新后的警报对象，带有新状态（`"sent"`、`"broadcasting"` 或 `"rejected"`）。

错误响应

状态	原因
400	`"action must be 'approve' or 'reject'"`
404	`"Alert not found: <id>"`

Web 界面：CAP 警报页面

路径：`/cap` 刷新：每 3 秒 + 实时 PubSub 更新

CAP 警报页面提供操作员审批工作流和广播监控界面。

统计卡片

���部有四个摘要卡片：

卡片	描述
待批准	等待操作员操作的警报数量（当 > 0 时为琥珀色）
活动广播	当前正在广播的警报数量
总处理	历史中的总警报（已发送 + 已拒绝 + 失败）
审批模式	当前模式：“手动”或“自动”

待处理警报面板

仅在 `require_approval` 为 `true` 时可见。显示等待操作员批准的警报。

列	描述
时间	警报接收时间（HH:MM:SS）
事件	警报事件类型（例如，“龙卷风警告”）
严重性	警报严重性级别
蜂窝	通过多边形解析匹配的蜂窝数量
状态	待处理徽章（琥珀色）
操作	预览、批准和拒绝按钮

预览 展开行以显示：

详细信息	描述
描述	完整的警报描述文本（广播消息内容）
来源	警报接收方式（`http_post` 或 `feed_poll`）
TACs	匹配的 4G 广播跟踪区域代码
LACs	匹配的 2G 和 3G 广播位置区域代码
消息 ID	CB 消息标识符
PLMN	用于广播的 MCC/MNC

批准 触发通过 CBC 引擎立即广播到所有连接的 MMEs（4G）、RNCs（3G）和 BSCs（2G）。

拒绝 将警报移至历史记录，状态为 `:rejected`。

活动广播面板

显示当前正在广播的警报：

列	描述
事件	警报事件类型
消息 ID	CB 消息标识符
TACs	目标跟踪区域代码
开始时间	广播开始时间
状态	广播中（蓝色）或已发送（绿色）徽章

警报历史面板

显示已发送、已拒绝和已失败的警报（最后 200 条）：

列	描述
时间	发送或接收时间戳
事件	警报事件类型
严重性	警报严重性级别
蜂窝	匹配的蜂窝数量
TACs/LACs	目标 TACs 和 LACs
状态	已发送（绿色）、已拒绝（红色）或失败（红色）徽章

实时更新

该页面订阅 `cap:alerts` PubSub 主题。当新警报到达或警报状态更改时，页面会自动刷新。当收到新的待处理警报时，会出现吐司通知（“收到新的 CAP 警报：等待批准”）。

警报状态持久性

警报状态持久化到 `priv/cap_alerts.json` 中，格式为 JSON，遵循与 CBC 引擎的 `priv/active_broadcasts.json` 相同的模式。在启动时，警报管理器重新加载此文件以恢复待处理和活动警报。

历史记录限制为 **200** 条 以防止无限增长。

Atom 源轮询

源轮询器支持国家天气服务��警报机构使用的标准 CAP Atom 源。源格式遵循 RFC 4287（Atom 订阅）与 CAP 警报条目。

预期的源格式

```
<feed xmlns="http://www.w3.org/2005/Atom">
  <entry>
    <id>urn:oid:2.49.0.1.840.0.abc123</id>
    <title>Tornado Warning</title>
    <content type="text/xml">
      <alert xmlns="urn:oasis:names:tc:emergency:cap:1.2">
        <!-- 整的 CAP 警报 XML -->
      </alert>
    </content>
  </entry>
</feed>
```

去重

轮询器跟踪其已见的每个条目的 `<id>`。具有先前见过的 ID 的条目将被跳过。去重集保存在内存中，并在应用程序重启时重置。

错误处理

源获取失败（网络错误、HTTP 错误、XML 解析错误）将作为警告记录，并不会影响后续的轮询周期。轮询器将继续按照其配置的间隔进行，无论单个获取结果如何。

监督

CAP 子系统向 OmniLCS 监督器添加两个进程：

进程	描述
`OmniLcs.Cap.AlertManager`	警报生命周期 GenServer。管理待处理/活动/历史状态、多边形解析和广播调度。
`OmniLcs.Cap.FeedPoller`	Atom 源轮询 GenServer。为每个配置的源 URL 安排定期获取。

两者都会自动启动，并与现有的 CBC 和 E-SMLC 进程一起使用 `:one_for_one` 策略进行监督。

故障排除

没有蜂窝与多边形匹配

症状：警报被接受，但 `matched_cells` 为 0，TAC/LAC 列表为空。

可能原因：

- 蜂窝数据库中没有地理位置位于警告多边形内的蜂窝
- 蜂窝存在于多边形区域内，但缺少 `tac` 和 `lac` 字段
- CAP XML 中的多边形坐标顺序或格式错误

解决方案：

1. 通过 `GET /api/cells` 验证警告区域中的蜂窝□否设置了 `tac` 和/或 `lac` 值
2. 检查蜂窝位置（纬度/经度）□否正确
3. 如果蜂窝位于多边形边缘附近，请尝试启用 `coverage_aware: true`
4. 验证 CAP 多边形坐标与地图的匹配

警报卡在待处理状态

症状：警报显示为待处理，但操作员无法批准。

可能原因：

- `require_approval` 为 `true`，且没有操作员批准警报
- LiveView 页面未连接（检查浏览器 WebSocket 连接）

解决方案：

1. 导航到控制面板中的 `/cap` 并单击批准
2. 或者，使用 REST API：`PUT /api/cap/<id>`，内容为 `{"action": "approve", "operator": "operator1"}`

源轮询器未获取警报

症状：配置的源 URL 有新警报，但它们未出现在 OmniLCS 中。

可能原因：

- 源 URL 不正确或无法访问
- 源格式不□嵌入 CAP 的标准 Atom
- TLS ��书问题
- 警报条目不包含带有嵌入 `<alert>` XML 的 `<content>`

解决方案：

1. 检查日志中□否有 `CAP FeedPoller: Failed to fetch` 消息
2. 验证源 URL □否返回有效的 Atom XML
3. 确认源包含 `<entry>` 元素，并且 `<content>` 中包含 `<alert>` XML
4. 对于 TLS 问题，检查源 URL □否可以从 OmniLCS 服务器访问

广播已发送但 UE 未接收

症状：警报状态显示为“已发送”，但没有 UE 接收到广播消息。

可能原因：

- TAC/LAC 值与实际网络配置不匹配
- MME 或 BSC 拒绝了广播（检查 CBC 4G / CBC 页面以获取响应详情）
- 消息编码问题

解决方案：

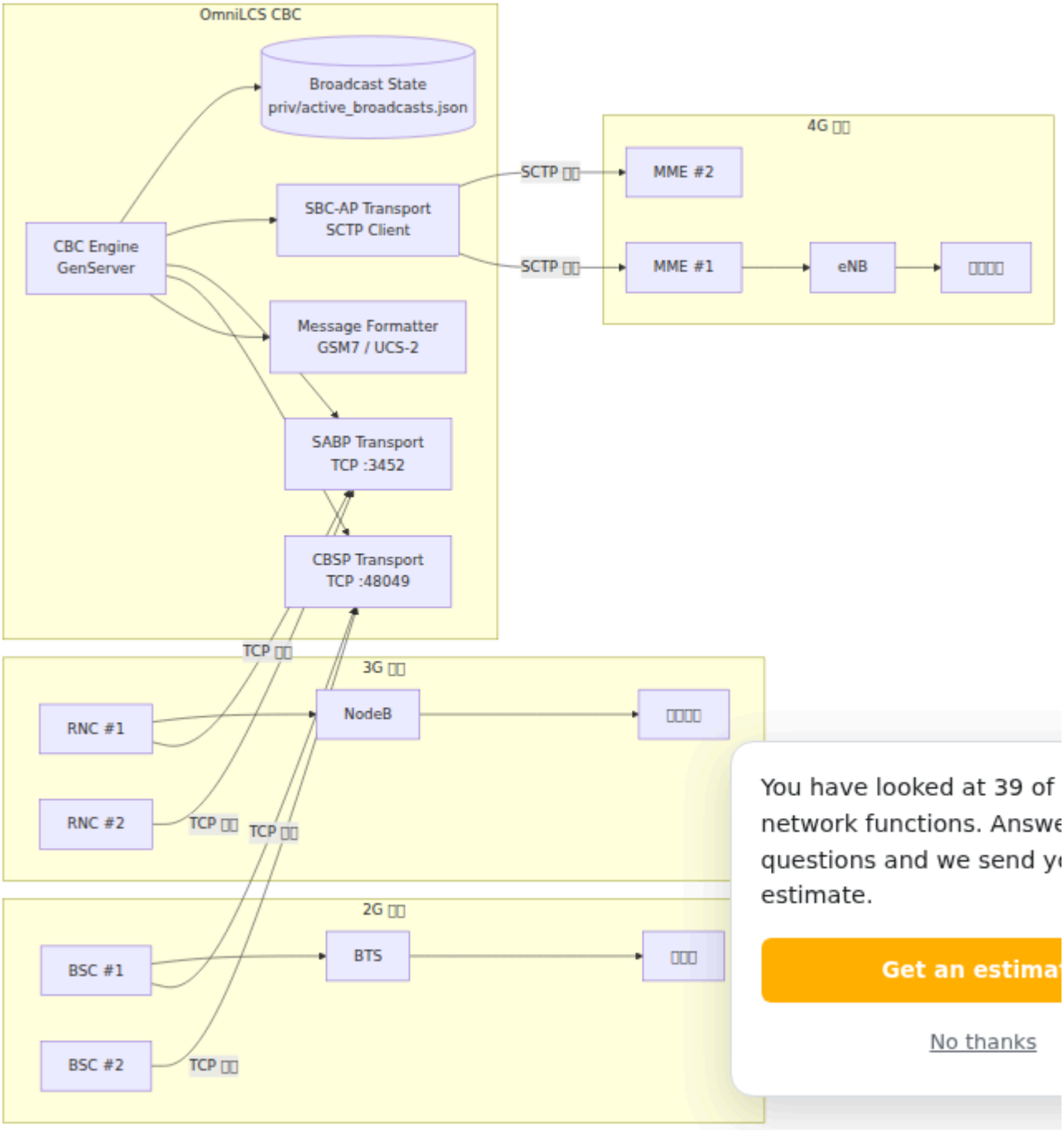
1. 验证 TAC 值□否与 MME 的跟踪区域配置匹配
2. 检查 CBC 4G 页面上的写替换警告响应状态
3. 检查 CBC 页面上的写替换完成/失败消息
4. 在警报历史中查看广播结果（展开警报行）

3GPP 和标准参考

规范	标题
OASIS CAP v1.2	通用警报协议版本 1.2
RFC 4287	Atom 订阅格式
TS 29.168	蜂窝广播中心与演进分组核心（SBC-AP）的接口
TS 48.049	蜂窝广播中心协议（CBSP）
TS 23.041	蜂窝广播服务（CBS）的技术实现
TS 23.038	字母表和特定语言的信息

短信广播□作指南

OmniLCS 实现了一个短信广播中心 (CBC)，支持通过 CBSP 的 2G 网络、通过 SABP 的 3G UTRAN 网络和通过 SBC-AP 的 4G LTE 网络。CBC 可以同时在所有连接的无线接入网络元素之间发送、更新和停止广播消息。



架构

You have looked at 39 of
network functions. Answe
questions and we send y
estimate.
Get an estima
No thanks

2G CBSP □作

连接模型

CBSP 根据 3GPP TS 48.049 使用 TCP 传输。CBC 监听 TCP 端口 **48049**（IANA 注册），BSC 建立入站连接。

- 每个接受的连接在 DynamicSupervisor 下生成一个连接处理程序
- 连接在 `:cbsp_connections` ETS 表中进行跟踪
- 连接状态通过 PubSub 广播到 LiveView UI

保持活动

CBC 实现了主动保持活动:

参数	默认值	描述
保持活动间隔	30 秒	CBC 向每个对等体发送 KEEP-ALIVE 的频率
保持活动超时	10 秒	在标记为不健康之前等待 KEEP-ALIVE COMPLETE 的时间

CBC 同时发送主动的 KEEP-ALIVE 消息，并对 BSC 发起的 KEEP-ALIVE 消息以 KEEP-ALIVE COMPLETE 作出响应。

CBSP 消息类型

消息	方向	代码	描述
WRITE-REPLACE	CBC -> BSC	0x01	发送或���新广播消息
WRITE-REPLACE COMPLETE	BSC -> CBC	0x02	BSC 接受广播
WRITE-REPLACE FAILURE	BSC -> CBC	0x03	BSC 拒绝广播
KILL	CBC -> BSC	0x04	停止广播
KILL COMPLETE	BSC -> CBC	0x05	广播已停止
KILL FAILURE	BSC -> CBC	0x06	停止广播失败
LOAD-QUERY	CBC -> BSC	0x07	查询无线资源加载
LOAD-QUERY COMPLETE	BSC -> CBC	0x08	返回加载信息
LOAD-QUERY FAILURE	BSC -> CBC	0x09	加载查询失败
STATUS-QUERY	CBC -> BSC	0x0A	查询广播交付状态
STATUS-QUERY COMPLETE	BSC -> CBC	0x0B	返回状态信息
STATUS-QUERY FAILURE	BSC -> CBC	0x0C	状态查询失败
RESET	CBC -> BSC	0x10	重置 BSC 上的广播状态
RESET COMPLETE	BSC -> CBC	0x11	重置已确认
RESET FAILURE	BSC -> CBC	0x12	重置失败
RESTART	BSC -> CBC	0x13	BSC 重启（信息性）
FAILURE	BSC -> CBC	0x14	BSC 故障指示

消息	方向	代码	描述
ERROR INDICATION	任意	0x15	协议错误
KEEP-ALIVE	任意	0x16	连接监督
KEEP-ALIVE COMPLETE	任意	0x17	保持活动响应

CBSP 消息封装

每个 CBSP 消息在传输中为：

```
+----------+----------+------+
| □度      | 类型     | IEs  |
| (3 字节) | (1 字节) | ...  |
+----------+----------+------+
```

3 字节的长��字段涵盖类型字节和所有 IEs（不包括 3 个长度字节本身）。

小区列表格式

小区列表 IE 确定广播目标的小区。支持的格式：

区分符	值	格式	描述
CGI	0x00	MCC+MNC+LAC+CI	完整小区全球标识
LAC+CI	0x01	PLMN+LAC+CI	位置区码 + 小区标识
CI	0x02	仅 CI	仅小区标识
LAI	0x04	MCC+MNC+LAC	位置区标识
LAC	0x05	仅 LAC	位置区码
所有在 BSC	0x06	（无小区）	BSC 管理的所有小区

通道指示符

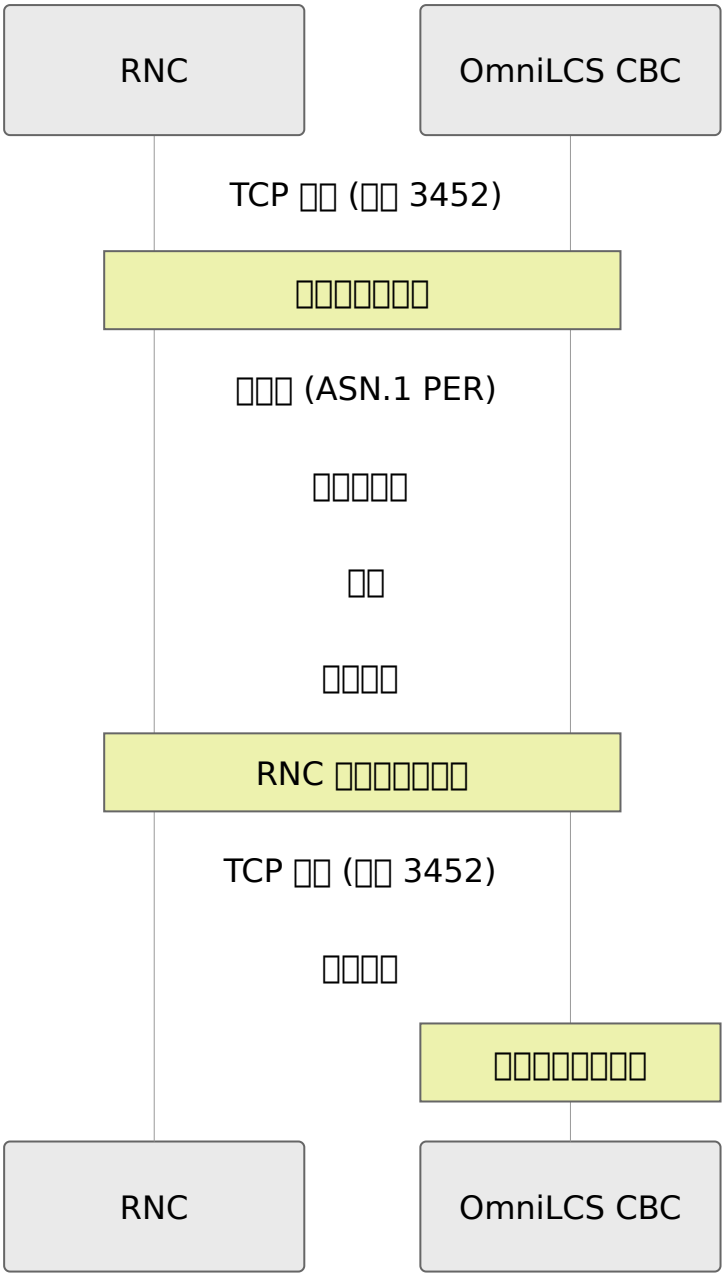
值	名称	描述
0	基本 CBCH	主要小区广播信道
1	扩展 CBCH	扩展小区广播信道（更高容量）

3G SABP □作

服务区广播协议 (SABP) 根据 3GPP TS 25.419 提供 CBC 与 3G UTRAN 网络中的 RNC 之间的 Iu-BC 接口。SABP 使用 ASN.1 PER 对 TCP 进行对齐编码。

连接模型

SABP 根据 TS 25.414 §7.1.3.3 使用 TCP 传输。SABP 的 IANA 注册端口为 **3452**。根据 TS 25.419 §5，CBC 为正常操作（写替换、杀死等）发起连接，RNC 仅在故障/重启指示时发起连接。双方在建立新连接时使用目标端口 3452。



RNC
OmniLCS CBC
TCP 连接 (端口 3452)
出站连接已建立
写替换 (ASN.1 PER)
写替换完成
杀死
杀死完成
RNC 也可以连接入站
TCP 连接 (端口 3452)
重启指示
重新发送活动广播
RNC
OmniLCS CBC

- 每个接受的 TCP 连接在 SABP 连接 DynamicSupervisor 下生成一个连接处理程序
- 连接在 `:sabp_connections` ETS 表中进行跟踪
- 连接状态更改通过 PubSub 广播到 LiveView UI
- 启用了 TCP 保持活动套接字选项，因为 SABP 没有协议级别的保持活动机制

SABP 消息封装

SABP 消息通过 TCP 封装，前面带有 4 字节的大端长度前缀：

```
+--------------------+----------------------------+
| □度 (4 字节)       | ASN.1 PER 编码的 PDU       |
| 大端 uint32        | ...                        |
+--------------------+----------------------------+
```

长度字段指定后续 ASN.1 PER 编码有效负载的字节数。多个消���可能在单个 TCP 段中到达；连接处理程序会根据需要缓冲和重新组装。

SABP 程序

程序	代码	类别	方向	描述
写替换	0	类别 1	CBC -> RNC	发送或更新广播消息
杀死	1	类别 1	CBC -> RNC	停止活动广播
加载状态查询	2	类别 1	CBC -> RNC	查询无线资源加载
消息状态查询	3	类别 1	CBC -> RNC	查询广播交付状态
重置	4	类别 1	CBC -> RNC	重置 RNC 上的广播状态
重启指示	5	类别 2	RNC -> CBC	RNC 重启，重新发送活动广播
故障指示	6	类别 2	RNC -> CBC	RNC 故障指示
错误指示	7	类别 2	任意	报告协议错误

SABP 信息元素

SABP 协议使用 ASN.1 定义的信息元素 (IEs)。在写替换中使用的关键 IEs:

IE	ID	重要性	类型	描述
广播消息内容	0	拒绝	二进制	编码的 CBS 消息内容
类别	1	忽略	枚举	消息优先级（高、正常、背景、默认）
原因	2	忽略	整数	故障/错误指示的原因值
数据编码方案	4	忽略	8 位	消息内容的编码方式（例如，0x0F 表示 GSM 7 位）
消息标识符	6	拒绝	16 位	CB 消息标识符
新序列号	7	拒绝	16 位	新/更新广播的序列号
请求的广播数量	9	拒绝	整数	总广播数量 (0..65535)
旧序列号	10	忽略	16 位	被替换消息的序列号（可选）
重复周期	13	拒绝	整数	重复之间的秒数 (1..4096)
服务区列表	15	拒绝	列表	目标服务区（SAI 列表）

服务区标识符 **(SAI)**

SABP 将广播目标定向到特定的服务区，服务区通过 SAI 识别。每个 SAI 由以下部分组成：

字段	大小	描述
PLMN 标识	3 字节	根据 3GPP TS 24.008 编码的 BCD 编码 MCC+MNC
LAC	2 字节	位置区码
SAC	2 字节	服务区码

广播类别

可选的类别 IE 设置广播消息的优先级：

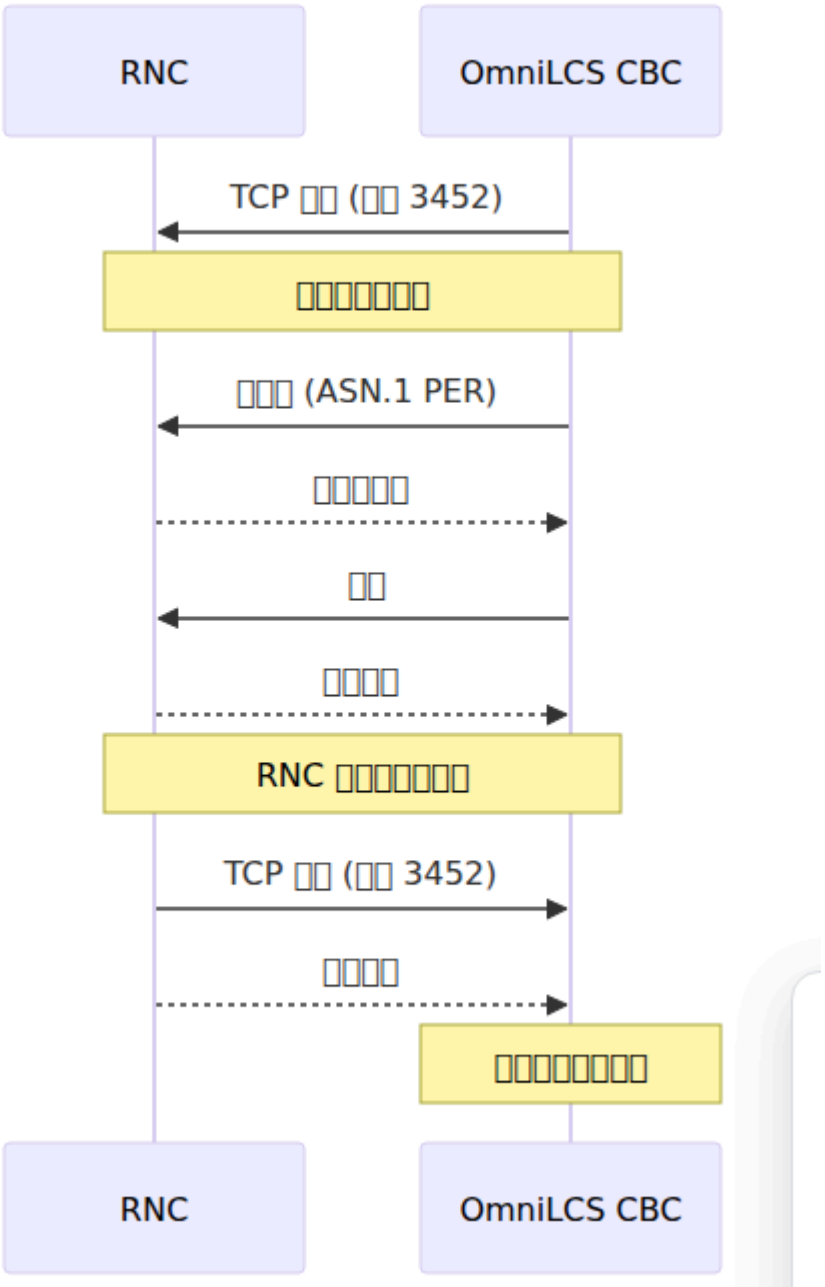
类别	描述
高优先级	最高优先级，抢占其他广播
正常优先级	标准优先级
背景优先级	低优先级，不抢占
默认优先级	未指定类别时的默认值

重启指示和故障指示处理

当 CBC 接收到来自 RNC 的 重启指示 时，RNC 已重启并丢失其广播状态。CBC 应重新发送所有活动广播给该 RNC。

当 CBC 接收到 故障指示 时，RNC 报告某些服务区失去了广播能力。这将被记录并作为遥测事件发出。

4G SBC-AP 操作



连接模型

根据 3GPP TS 29.168 第 4.3 节，**CBC** 向每个 **MME** 发起 **SCTP** 关联。MME 监听端口 **29168**（IANA 注册）。SCTP 有效负载协议标识符（PPID）为 **24**。

该传输实现了自动重连和指数退避：

参数	值	描述
初始重连延迟	5 秒	连接失败后的第一次重试延迟
最大重连延迟	60 秒	最大退避上限
SCTP 心跳间隔	10 秒	对等地址心跳
SCTP 路径最大重传次数	7	在声明路径失败之前
SCTP RTO 最大	30 秒	最大重传超时
SCTP RTO 最小	1 秒	最小重传超时

SBC-AP 程序

程序	代码	类别	方向	描述
写替换警告	0	类别 1	CBC -> MME	发送或更新警告消息
停止警告	1	类别 1	CBC -> MME	停止��动警告
错误指示	2	类别 2	任意	报告协议错误
PWS 重启指示	3	类别 2	MME -> CBC	MME 重启，重新发送活动警告
PWS 故障指示	4	类别 2	MME -> CBC	eNB 故障，失去广播能力

写替换警告请求 **IEs**

IE	ID	重要性	类型	描述
消息标识符	5	拒绝	16 位	CB 消息标识符（例如，0x1112 表示 CMAS）
序列号	11	拒绝	16 位	带有地理范围、消息代码、更新编号的序列号
TAIs 列表	14	拒绝	列表	广播的跟踪区域标识
警告区域列表	15	忽略	选择	目标区域（TAI 列表、小区列表或紧急区域）
重复周期	10	拒绝	整数	重复之间的秒数 (0..4096)
请求的广播数量	7	拒绝	整数	总广播数量 (0..65535)
警告类型	18	忽略	2 字节	ETWS 警告类型 + 激活标志
数据编码方案	3	忽略	8 位	消息��容的编码方式
警告消息内容	16	忽略	二进制	编码的 CBS 页面
OMC ID	19	忽略	二进制	操作中心标识
并发警告消息指示符	20	拒绝	布尔值	允许并发警告

停止警告请求 IEs

IE	ID	重要性	描述
消息标识符	5	拒绝	要停止的 CB 消息标识符
序列号	11	拒绝	要停止的广播的序列号
TAIs 列表	14	拒绝	（可选）限制停止到特定 TAIs
警告区域列表	15	忽略	（可选）限制停止到特定区域

PWS 重启指示处理

当 CBC 接收到来自 MME 的 PWS 重启指示时，它会重新发送所有活动广播（状态为 `:sent` 或 `:acknowledged`）给该 MME。这确保了 MME 重启后的广播连续性。

PWS 故障指示处理

当 CBC 接收到 PWS 故障指示时，它会将受影响的广播标记为 `:pws_failure` 响应，以便报告 MME 关联。这将被记录为警告。

消息格式

GSM 7 位编码

CBS 消息的默认编码。每个 CBS 页面最多可容纳 **93** 个七位字符，打包为 **82** 字节。

GSM 7 位默认字母表映射标准拉丁字符以及希腊字母和特殊符号。基本字母表中不存在的字符使用扩展表（转义序列 0x1B + 扩展代码），消耗两个七位字符。

警告消息内容的二进制结构：

```
+----------+-----------------------------------------------+
| 页面数量 | 页面 1（82 字节打包 + 1 字节□度）              |
| （1 字节）| 页面 2 ...                                    |
+----------+-----------------------------------------------+
```

UCS-2 编码

用于非拉丁文字（CJK、阿拉伯语、泰语等）。每个 CBS 页面最多可容纳 **40** 个 **UCS-2** 字符（80 字节的 UTF-16BE 数据），填充为 82 字节。

数据编码方案 **(DCS)**

编码	**DCS** 值	描述
GSM 7 位	0x0F	GSM 7 位默认字母表，语言未指定
UCS-2	0x48	UCS-2 (UTF-16BE)，语言未指定

DCS 可以在每个广播中显式设置，或根据编码参数自动派生。

序列号结构

根据 3GPP TS 23.041 第 9.4.1.2.1 节，16 位序列号具有内部结构：

```
+----+------------+--------+
| GS | 消息代码    | 更新    |
| 2b | 10 位      | 4 位    |
+----+------------+--------+
```

字段	位数	描述
地理范围 (GS)	15-14	0 = 小区范围内立即，1 = PLMN 范围，2 = LA/TA 范围，3 = 小区范围
消息代码	13-4	在其范围内识别广播 (0..1023)
更新编号	3-0	消息更新时递增 (0..15)

序列号由这些组件构成。

警告类型 (ETWS)

警告类型 IE 根据 3GPP TS 23.041 第 9.3.24 节为 2 字节：

警告类型	八位组 1 值	描述
地震	0x00	地震警告
海啸	0x01	海啸警告
地震 + 海啸	0x02	组合地震和海啸
测试	0x03	测试警告
其他	0x04	其他紧急情况

八位组 2 激活标志（默认 0xC0）：

位	值	描述
位 8	0x80	紧急用户警报（激活警报音/振动）
位 7	0x40	弹出（自动在屏幕上显示消息）

默认情况下这两个位都被设置（0xC0），以确保最大可见性。

广播状态管理

状态生命周期

持久性

2G、3G 和 4G 的活动广播被持久化到 `priv/active_broadcasts.json` 作为 JSON。在启动时，Engine 重新加载该文件以恢复广播状态。这允许 CBC 在应用程序重启后重新发送活动警告。持久性文件存储三个单独的广播映射：`active_broadcasts_4g`、`active_broadcasts_3g` 和 `active_broadcasts_2g3g`。

响应跟踪

对于 4G 广播，每个 `send_broadcast_4g` 启动一个 30 秒的响应超时计时器。来自 MMEs 的响应通过消息标识符和序列号进行关联。当所有预期的 MME 对等体都已响应时，计时器被取消。超时的对等体被记录为 `:timeout` 原因。



活动广播视图

活动广播页面提供了 2G、3G 和 4G 网络所有进行中的广播的统一视图。

广播历史

Engine 维护了最后 100 个 4G 广播的滚动历史，以供 LiveView UI 和 API 查询。

故障排除

没有连接的 MMEs (SBC-AP)

1. 验证 `mme_peers` 配置包含正确的 IP 地址和端口
2. 检查 `local_ip` 是否可以从 MME 网络访问
3. 在日志中查找 SCTP 连接错误：`SBC-AP: 无法连接到 MME`
4. 验证 SCTP 是否未被防火墙阻止（协议 132）
5. 确认 MME 监听端口 29168

没有连接的 BSCs (CBSP)

1. 验证 `listen_ip` 和 `listen_port` 配置在 `:cbsp` 下

2. 检查 TCP 端口 48049 □否未被防火墙阻止
3. 查找接受错误：`CBSP 接受失败`
4. 确认 BSC 配置了 CBC 的 IP 地址和端口

没有连接的 RNCs (SABP)

1. 验证 `listen_ip` 和 `listen_port` 配置在 `:sabp` 下
2. 检查 TCP 端口 3452 □否未被防火墙阻止
3. 查找接受错误：`SABP 接受失败`
4. 确认 RNC 配置了 CBC 的 IP 地址和端口
5. 检查日志中□否有 `SABP 连接被对等体关闭` 或 `SABP TCP 错误` 消息

广播未交付

1. 检查活动广播表以获取响应状态
2. 在 CBSP/SABP 消息日志中查找 WRITE-REPLACE FAILURE 或 KILL FAILURE 消息
3. 验证小区列表格式□否与 BSC/RNC/MME 预期相符
4. 对于 4G，检查 TAC 值□否对��标 MME 有效
5. 对于 3G，检查服务区标识符 (SAIs) □否对目标 RNC 有效
6. 检查广播历史以查找超时条目

编码问题

1. 对于非拉丁文本，确保选择了 `:ucs2` 编码
2. 验证 DCS □否与编码匹配（0x0F 表示 GSM 7 位，0x48 表示 UCS-2）
3. GSM 7 位只能表示 GSM 默认字母表；不支持的字符将被替换为 `?`

3GPP 参考

规范	标题
TS 25.419	UTRAN Iu-BC 接口: 服务区广播协议 (SABP)
TS 29.168	短信广播中心与演进分组核心的接口 (SBC-AP)
TS 48.049	短信广播中心协议 (CBSP)
TS 23.041	短信广播服务 (CBS) 的技术实现
TS 23.038	字母表和特定语言信息 (DCS, GSM 7 位)
TS 24.008	移动无线接口层 3 规范 (PLMN 标识编码)

OmniLCS 配置参考

所有配置都在 `config/config.exs` 中定义。本文档涵盖每个配置部分和参数。

控制面板（LiveView UI）

控制面板提供基于网页的管理界面，通过 HTTPS 在 443 端口提供服务。

```
config :control_panel,
  parent_application: :omnilcs,
  parent_application_version: "1.0.0",
  parent_application_readable_name: "OmniLCS",
  home_page: ControlPanelWeb.ApplicationLive,
  use_builtin_pages: [...],
  use_additional_pages: [...],
  page_order: ["/dashboard", "/location", "/cells", "/diameter",
               "/cbc", "/cbc4g", "/application", "/configuration",
"/log"],
  licensee_name: "Omnitouch"
```

参数	类型	描述
`parent_application`	atom	OTP 应用程序名称
`parent_application_version`	string	在 UI 中显示的版本
`parent_application_readable_name`	string	人类可读的产品名称
`home_page`	module	默认登录页面模块
`use_builtin_pages`	list	内置页面 (应用程序，配置，日志)
`use_additional_pages`	list	自定义 OmniLCS 页面 (仪表板，位置，单元，直径，CBC 2G，CBC 3G，CBC 4G，CAP 警报)
`page_order`	list	导航栏中的标签顺序
`licensee_name`	string	在页脚中显示的被许可方名称

控制面板端点

```
config :control_panel, ControlPanelWeb.Endpoint,
  server: true,
  https: [
    port: 443,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ],
  secret_key_base: "...",
  check_origin: false,
  pubsub_server: ControlPanel.PubSub,
  live_view: [signing_salt: "LcsLvSlt"]
```

参数	类型	默认值	
`server`	boolean	`true`	启用 HTT
`https.port`	integer	`443`	HTTPS
`https.keyfile`	string	`"priv/cert/omnitouch.pem"`	TLS 私
`https.certfile`	string	`"priv/cert/omnitouch.crt"`	TLS 证
`secret_key_base`	string	--	Phoen (使用 `mi` `phx.ge` 生成)
`check_origin`	boolean	`false`	否验证 WebSo
`pubsub_server`	atom	`ControlPanel.PubSub`	用于 Live PubSu
`live_view.signing_salt`	string	`"LcsLvSlt"`	LiveVi

REST API

REST API 通过 api_ex 框架在 HTTPS 上的 8443 端口提供服务。

```
config :api_ex,
  api: %{
    port: 8443,
    listen_ip: "0.0.0.0",
    product_name: "OmniLCS",
    title: "API - OmniLCS",
    hostname: "localhost",
    enable_tls: true,
    tls_cert_path: "priv/cert/omnitouch.crt",
    tls_key_path: "priv/cert/omnitouch.pem",
    routes: [...]
  }
```

参数	类型	默认值	描述
`port`	integer	`8443`	API 监听端口
`listen_ip`	string	`"0.0.0.0"`	API 绑定地址
`product_name`	string	`"OmniLCS"`	OpenAPI 规范中的产品名称
`title`	string	`"API - OmniLCS"`	Swagger UI 中的 API 标题
`hostname`	string	`"localhost"`	OpenAPI 规范 URL 的主机名
`enable_tls`	boolean	`true`	启用 HTTPS
`tls_cert_path`	string	`"priv/cert/omnitouch.crt"`	TLS 证书路径
`tls_key_path`	string	`"priv/cert/omnitouch.pem"`	TLS 私钥路径
`routes`	list	--	路由定义列表（路径，模块，操作）

注册的 **API** 路由

路径	控制器	□作
`/status`	`OmniLcs.Api.StatusController`	`index`
`/location`	`OmniLcs.Api.LocationController`	`index`, `create`, `show`
`/cells`	`OmniLcs.Api.CellController`	`index`, `create`, `show`, `update`, `delete`
`/cap`	`OmniLcs.Api.CapController`	`index`, `create`, `show`, `update`

E-SMLC 设置

一般的 E-SMLC 配置。

```
config :omnilcs,
  esmlc_name: "OmniLCS",
  cell_database_path: "priv/cells.json"
```

参数	类型	默认值	描述
`esmlc_name`	string	`"OmniLCS"`	E-SMLC 实例名称（用于状态 API）
`cell_database_path`	string	`"priv/cells.json"`	用于导入的单元数据库 JSON 文件路径

CBSP (2G 单元广播)

CBSP TCP 监听器的配置。BSC 通过此端口连接到 CBC。

```
config :omnilcs, :cbsp,
  listen_ip: "0.0.0.0",
  listen_port: 48049
```

参数	类型	默认值	描述
`listen_ip`	string	`"0.0.0.0"`	绑定 CBSP TCP 监听器的 IP 地址
`listen_port`	integer	`48049`	CBSP 连接的 TCP 端口 (IANA 注册的 CBSP 端口)

SABP (3G 单元广播)

SABP TCP 监听器的配置。RNC 通过 Iu-BC 接口连接到 CBC。

```
config :omnilcs, :sabp,
  listen_ip: "0.0.0.0",
  listen_port: 3452
```

参数	类型	默认值	描述
`listen_ip`	string	`"0.0.0.0"`	绑定 SABP TCP 监听的 IP 地址
`listen_port`	integer	`3452`	RNC 连接的 SABP TCP 端口 (IANA 注册的，根据 TS 25.414 §7.1.3.3)

SBC-AP (4G 单元广播)

SBC-AP SCTP 连接的配置。根据 3GPP TS 29.168，CBC 向每个 MME 对等体发起 SCTP 关联。

```
config :omnilcs, :sbcap,
  local_ip: "10.5.198.200",
  mme_peers: [
    %{host: "mme01", ip: "10.179.2.100", port: 29168},
    %{host: "mme02", ip: "10.179.2.101", port: 29168}
  ]
```

参数	类型	默认值	描述
`local_ip`	string	`"0.0.0.0"`	绑定 SCTP 套接字的本地 IP 地址
`mme_peers`	list	`[]`	MME 对等体配置列表

MME 对等体配置

`mme_peers` 中的每个条目□一个包含以下字段的映射：

字段	类型	必需	默认值	描述
`host`	string	否	`ip` 的值	人类可读的 MME 主机名（用于日志和 UI）
`ip`	string	□	--	MME IP 地址
`port`	integer	否	`29168`	MME SBC-AP SCTP 端口（IANA 注册）

SBC-AP 传输使用 SCTP 负载协议标识符（PPID）24。连接失败时，它会以指数退避的方式重试，初始延迟为 5 秒，最长为 60 秒。

InfluxDB

单元位置数据定期从 InfluxDB 实例同步。

```
config :omnilcs, OmniLcs.InfluxDb,
  database: "nokia-monitor",
  host: "172.19.3.68",
  port: 8086,
  auth: [method: :basic, username: "monitor", password: "..."],
  http_opts: [recv_timeout: 30_000],
  pool: [max_overflow: 10, size: 5]
```

参数	类型	默认值	描述
`database`	string	--	InfluxDB 数据库名称
`host`	string	--	InfluxDB 服务器主机名或 IP
`port`	integer	`8086`	InfluxDB HTTP API 端口
`auth.method`	atom	`:basic`	认证方法
`auth.username`	string	--	InfluxDB 用户名
`auth.password`	string	--	InfluxDB 密码
`http_opts.recv_timeout`	integer	`30000`	HTTP 接收超时（毫秒）
`pool.size`	integer	`5`	连接池大小
`pool.max_overflow`	integer	`10`	最大池溢出连接数

单元同步每 5 分钟自动运行一次，启动后初始延迟为 10 秒。也可以通过 REST API 或控制面板手动触发。

SLs 接口 (LCS-AP 通过 SCTP)

SLs 接口使用 LCS-AP 通过 SCTP 将 E-SMLC 连接到 MME，符合 3GPP TS 29.171。OmniLCS 在 9082 端口向每个配置的 MME 对等体发起 SCTP 关联，使用 PPID 29。

```
config :omnilcs, :sls,
  local_ip: "10.5.198.200",
  mme_peers: [
    %{host: "mme01", ip: "10.179.1.15", port: 9082}
  ]
```

参数	类型	默认值	描述
`local_ip`	string	`"0.0.0.0"`	绑定 SCTP 套接字的本地 IP 地址
`mme_peers`	list	`[]`	MME 对等体配置列表

MME 对等体配置（SLs）

`mme_peers` 中的每个条目□一个包含以下字段的映射：

字段	类型	必需	默认值	描述
`host`	string	否	`ip` 的值	人类可读的 MME 主机名（用于日志和 UI）
`ip`	string	□	--	MME IP 地址
`port`	integer	否	`9082`	MME LCS-AP SCTP 端口（IANA 注册的 LCS-AP）

SLs 传输使用 SCTP 负载协议标识符（PPID）29。连接失败时，它会以指数退避的方式重试，初始延迟为 5 秒，最长为 60 秒。

SCTP 调优参数：

参数	值	描述
心跳间隔	10 秒	对等地址心跳
路径最大重传次数	7	在声明路径失败之前
RTO 最大值	30 秒	最大重传超时
RTO 最小值	1 秒	最小重传超时
SACK 延迟	200 毫秒	选择性确认延迟

Diameter (SLg 和其他接口)

Diameter 配置控制 OmniLCS (作为 GMLC) 和 MME 之间的 SLg 接口，通过 DRA 路由。这与使用原生 LCS-AP 通过 SCTP 的 SLs 接口分开的。

```
config :diameter_ex,
  diameter: %{
    service_name: :omnitouch_esmlc,
    listen_ip: "10.5.198.200",
    listen_port: 3868,
    host: "amanaki",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    product_name: "OmniLCS",
    request_timeout: 5000,
    peer_selection_algorithm: :random,
    allow_undefined_peers_to_connect: true,
    log_unauthorized_peer_connection_attempts: true,
    control_module: OmniLcs.Control.Diameter,
    vendor_id: 10415,
    supported_vendor_ids: [5535, 10415],
    applications: [...],
    peers: [...]
  }
```

服务参数

参数	类型	默认值
`service_name`	atom	`:omnitouch_esml`
`listen_ip`	string	--
`listen_port`	integer	`3868`
`host`	string	--
`realm`	string	--
`product_name`	string	`"OmniLCS"`
`request_timeout`	integer	`5000`
`peer_selection_algorithm`	atom	`:random`

□□	□□	□□□
`allow_undefined_peers_to_connect`	boolean	`true`
`log_unauthorized_peer_connection_attempts`	boolean	`true`
`control_module`	module	`OmniLcs.Control`
`processor_module`	module	`DiameterEx.Proc`
`vendor_id`	integer	`10415`
`supported_vendor_ids`	list	`[5535, 10415]`

Diameter □□□□

```
applications: [
  %{
    application_name: :slg,
    application_dictionary: :diameter_gen_3gpp_slg,
    vendor_specific_application_ids: [
      %{vendor_id: 10415, auth_application_id: 16_777_264,
acct_application_id: nil}
    ]
  }
]
```

字段	类型	描述
`application_name`	atom	应用程序标识符 (`:slg` 用于 GMLC 到 MME 接口)
`application_dictionary`	atom	Erlang Diameter 字典模块
`vendor_specific_application_ids`	list	Vendor-Specific-Application-Id AVP 值

SLg 应用程序使用应用程序 ID **16777264** 和 3GPP Vendor-Id **10415**。请注意，E-SMLC 到 MME 接口 (SLs) 使用原生 LCS-AP 通过 SCTP，而不是 Diameter。

Diameter 对等体

```
peers: [
  %{
    host: "dra01.epc.mnc001.mcc001.3gppnetwork.org",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    ip: "10.179.2.233",
    port: 3868,
    tls: false,
    transport: :diameter_sctp,
    initiate_connection: true
  }
]
```

字段	类型	默认值	描述
`host`	string	--	对等 Diameter 主机身份 (FQDN)
`realm`	string	--	对等 Diameter 领域
`ip`	string	--	对等 IP 地址
`port`	integer	`3868`	对等 Diameter 端口
`tls`	boolean	`false`	启用对等连接的 TLS
`transport`	atom	`:diameter_sctp`	传输协议 (`:diameter_sctp` 或 `:diameter_tcp`)
`initiate_connection`	boolean	`true`	OmniLCS □否发起与此对等体的连接

GMLC / Le 接口

网关移动位置中心和 Le 接口到外部 LCS 客户端的配置。有关请求流程、延迟会话和 InfluxDB 日志的完整详细信息，请参见 GMLC & Le 接口操作指南。

```
config :omnilcs, :gmlc,
  enabled: true,
  allow_unknown_clients: false,
  authorized_clients: [
    %{
      name: "psap-01",
      type: :emergency_services,
      allowed_methods: [:cell, :ecid, :gnss, :otdoa],
      rate_limit: 100,
      description: "□□ PSAP"
    }
  ],
  allow_deferred: true,
  max_periodic_sessions: 100,
  max_triggered_sessions: 50,
  default_periodic_poll_interval_ms: 60_000,
  default_triggered_poll_interval_ms: 30_000,
  influx_logging: true
```

参数	类型	默认值	描述
`enabled`	boolean	`false`	启用 GMLC Le 接口处理程序
`allow_unknown_clients`	boolean	`false`	接受来自未列出客户端的请求
`authorized_clients`	list	`[]`	授权的 LCS 客户端定义（请参见 GMLC 指南）
`allow_deferred`	boolean	`true`	接受定期和触发的位置请求
`max_periodic_sessions`	integer	`100`	最大并发定期会话
`max_triggered_sessions`	integer	`50`	最大并发触发会话
`default_periodic_poll_interval_ms`	integer	`60000`	默认定期修复间隔（毫秒）
`default_triggered_poll_interval_ms`	integer	`30000`	默认地理围栏轮询间隔（毫秒）
`influx_logging`	boolean	`true`	将所有 GMLC 位置修复写入 InfluxDB

2G/3G 位置（MAP / Lg）

通过 **Lg** 接口使用 MAP 访问 GERAN (2G) 或 UTRAN (3G) 上的订阅者位置，使用 `omniss7` SS7/SIGTRAN 堆栈。默认禁用；请参见 通过 MAP/Lg 的 2G/3G 位置。

```
# OmniLCS MAP/Lg 客户端 (OmniLcs.Control.MapClient)
config :omnilcs, :map,
  enabled: true,
  gmlc_number: "61400000000",   # 此 GMLC 的 ISDN 号码 (E.164)
  hlr_gt: "61400000001",        # 用于 SendRoutingInfoForLCS 的 HLR 全
球标题
  lcs_client_type: :emergencyServices

# 将 omniss7 SS7 堆栈作为库运行 (没有自己的网页 UI / SCTP / 许可证客户端)
config :omniss7, headless: true

# MAP 客户端的 SS7 传输 - M3UA 与 STP 的关联。
# 仅在启用 :omnilcs, :map 时需要。
config :omniss7, :map_client_enabled, true

config :omniss7, :map_client_m3ua, %{
  local_ip: {10, 0, 0, 100},
  local_port: 2905,
  remote_ip: {10, 0, 0, 1},     # STP / SGP
  remote_port: 2905,
  routing_context: 1
}
```

参数	类型	默认值	描述
`:omnilcs, :map enabled`	boolean	`false`	启用 2G/3G MAP/Lg 位置路径。当 `false` 时，`rat: "2g"`/`"3g"` 请求返回 `501`。
`gmlc_number`	string	`nil`	此 GMLC 的 ISDN 号码，用作 MAP 发起者。
`hlr_gt`	string	`nil`	用于 SendRoutingInfoForLC 的 HLR 全球标题（服务 MSC/SGSN 发现）。如果请求提 `node_gt`，则不需要。
`lcs_client_type`	atom	`:emergencyServices`	请求的 MAP LCS-ClientType。
`:omniss7 headless`	boolean	`true`（在 `config.exs` 中设置）	将 omniss7 作为库运行：仅编解器 + MAP 事务注册。
`:omniss7 map_client_m3ua`	map	--	SCTP/M3UA 与 STP/SGP 的关联。实时操作所需。

> 2G/3G 的精度为小区 ID / 定时提前类。服务 MSC/SGSN 返回服务小区，而不精确修复。

当 `:omnilcs, :map enabled` 为 `true` 且 `:map_client_m3ua` 被设置时，OmniLCS 会启动与 STP 的 M3UA 关联（由 `OmniLcs.Control.MapStack` 监督）。没有单独的传输步骤。`omnilcs_map_request_total` Prometheus 指标计算每个 MAP 事务（标签 `type`，`result`）。如果 `enabled` 为 `true` 但缺少 `:map_client_m3ua`，OmniLCS 会跳过传输并发出警告。其余的 OmniLCS 继续运行。

CAP 警报

CAP（通用警报协议）警报摄取和广播的配置。有关警报生命周期、多边形解析和操作员批准工作流的完整详细信息，请参见 CAP 警报操作指南。

```
config :omnilcs, :cap,
  require_approval: true,
  plmn: %{mcc: "001", mnc: "01"},
  coverage_aware: false,
  feeds: []
```

参数	类型	默认值	描述
`require_approval`	boolean	`true`	当 `true` 时，警报排队等待操作员批准。当 `false` 时，警报自动广播。
`plmn`	map	`%{mcc: "001", mnc: "01"}`	广播消息的 PLMN 身份 (MCC/MNC)
`coverage_aware`	boolean	`false`	使用小区覆盖半径进行多边形交集（与仅中心点相比）
`feeds`	list	`[]`	CAP Atom feed URL 进行轮询

Feed 配置

```
feeds: [
  %{url: "https://alerts.weather.gov/cap/us.php?x=1",
poll_interval_seconds: 60}
]
```

字段	类型	必需	默认值	描述
`url`	string	□	--	CAP Atom feed URL
`poll_interval_seconds`	integer	否	`60`	轮询之间的秒数

日志记录

```
config :logger,
  backends: [:console, ControlPanel.Logger]
```

参数	类型	描述
`backends`	list	日志记录后端。`:console` 写入 stdout; `ControlPanel.Logger` 将日志馈送到网页 UI 中的日志页面。

□□□□□□

```
import Config

config :control_panel,
  parent_application: :omnilcs,
  parent_application_version: "1.0.0",
  parent_application_readable_name: "OmniLCS",
  home_page: ControlPanelWeb.ApplicationLive,
  use_builtin_pages: [
    {ControlPanelWeb.ApplicationLive, "/application", "□□"},
    {ControlPanelWeb.ConfigurationLive, "/configuration", "□□"},
    {ControlPanelWeb.LogLive, "/log", "□□"}
  ],
  use_additional_pages: [
    {OmniLcs.Web.DashboardLive, "/dashboard", "□□□"},
    {OmniLcs.Web.LocationLive, "/location", "□□"},
    {OmniLcs.Web.CellDatabaseLive, "/cells", "□□"},
    {OmniLcs.Web.DiameterLive, "/diameter", "□□"},
    {OmniLcs.Web.CbcLive, "/cbc", "CBC 2G"},
    {OmniLcs.Web.Cbc3gLive, "/cbc3g", "CBC 3G"},
    {OmniLcs.Web.Cbc4gLive, "/cbc4g", "CBC 4G"},
    {OmniLcs.Web.CapAlertsLive, "/cap", "CAP □□"}
  ],
  page_order: [
    "/dashboard", "/location", "/cells", "/diameter",
    "/cbc", "/cbc3g", "/cbc4g", "/cap", "/application",
"/configuration", "/log"
  ],
  licensee_name: "Omnitouch"

# REST API
config :api_ex,
  api: %{
    port: 8443,
    listen_ip: "0.0.0.0",
    product_name: "OmniLCS",
    title: "API - OmniLCS",
    hostname: "localhost",
    enable_tls: true,
    tls_cert_path: "priv/cert/omnitouch.crt",
    tls_key_path: "priv/cert/omnitouch.pem",
```

```
    routes: [
      %{path: "/status", module: OmniLcs.Api.StatusController,
actions: [:index]},
      %{path: "/location", module: OmniLcs.Api.LocationController,
actions: [:index, :create, :show]},
      %{path: "/cells", module: OmniLcs.Api.CellController,
actions: [:index, :create, :show, :update, :delete]},
      %{path: "/cap", module: OmniLcs.Api.CapController, actions:
[:index, :create, :show, :update]}
    ]
  }

# 控制面板 HTTPS 端点
config :control_panel, ControlPanelWeb.Endpoint,
  server: true,
  url: [host: "0.0.0.0", path: "/"],
  https: [port: 443, keyfile: "priv/cert/omnitouch.pem", certfile:
"priv/cert/omnitouch.crt"],
  adapter: Bandit.PhoenixAdapter,
  secret_key_base: "REPLACE_WITH_64_BYTE_RANDOM_SECRET",
  check_origin: false,
  pubsub_server: ControlPanel.PubSub,
  live_view: [signing_salt: "LcsLvSlt"]

# 日志记录
config :logger,
  backends: [:console, ControlPanel.Logger]

# E-SMLC 一般设置
config :omnilcs,
  esmlc_name: "OmniLCS",
  cell_database_path: "priv/cells.json"

# CBSP (2G 单元广播) - BSC 连接到此端口
config :omnilcs, :cbsp,
  listen_ip: "0.0.0.0",
  listen_port: 48049

# SABP (3G 单元广播) - RNC 连接到此端口
config :omnilcs, :sabp,
  listen_ip: "0.0.0.0",
  listen_port: 3452

# SLs (LCS-AP 通过 SCTP) - E-SMLC 连接到这些 MMEs 进行定位
```

```
config :omnilcs, :sls,
  local_ip: "10.5.198.200",
  mme_peers: [
    %{host: "mme01", ip: "10.179.1.15", port: 9082}
  ]

# SBC-AP (4G 单元广播) - OmniLCS 连接到这些 MMEs
config :omnilcs, :sbcap,
  local_ip: "10.5.198.200",
  mme_peers: [
    %{host: "mme01", ip: "10.179.2.100", port: 29168},
    %{host: "mme02", ip: "10.179.2.101", port: 29168}
  ]

# CAP 警报摄取和广播
config :omnilcs, :cap,
  require_approval: true,
  plmn: %{mcc: "001", mnc: "01"},
  coverage_aware: false,
  feeds: []

# InfluxDB 用于单元位置同步
config :omnilcs, OmniLcs.InfluxDb,
  database: "nokia-monitor",
  host: "172.19.3.68",
  port: 8086,
  auth: [method: :basic, username: "monitor", password:
"REPLACE_WITH_PASSWORD"],
  http_opts: [recv_timeout: 30_000],
  pool: [max_overflow: 10, size: 5]

# Diameter (SLg 接口到 DRA/MME - GMLC 角色，而不是 E-SMLC SLs)
config :diameter_ex,
  diameter: %{
    service_name: :omnitouch_esmlc,
    listen_ip: "10.5.198.200",
    listen_port: 3868,
    host: "amanaki",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    product_name: "OmniLCS",
    request_timeout: 5000,
    peer_selection_algorithm: :random,
    allow_undefined_peers_to_connect: true,
    log_unauthorized_peer_connection_attempts: true,
```

```
    control_module: OmniLcs.Control.Diameter,
    processor_module: DiameterEx.Processor,
    auth_application_ids: [],
    acct_application_ids: [],
    vendor_id: 10415,
    supported_vendor_ids: [5535, 10415],
    applications: [
      %{
        application_name: :slg,
        application_dictionary: :diameter_gen_3gpp_slg,
        vendor_specific_application_ids: [
          %{vendor_id: 10415, auth_application_id: 16_777_264,
acct_application_id: nil}
        ]
      }
    ],
    peers: [
      %{
        host: "dra01.epc.mnc001.mcc001.3gppnetwork.org",
        realm: "epc.mnc001.mcc001.3gppnetwork.org",
        ip: "10.179.2.233",
        port: 3868,
        tls: false,
        transport: :diameter_sctp,
        initiate_connection: true
      }
    ]
  }
```

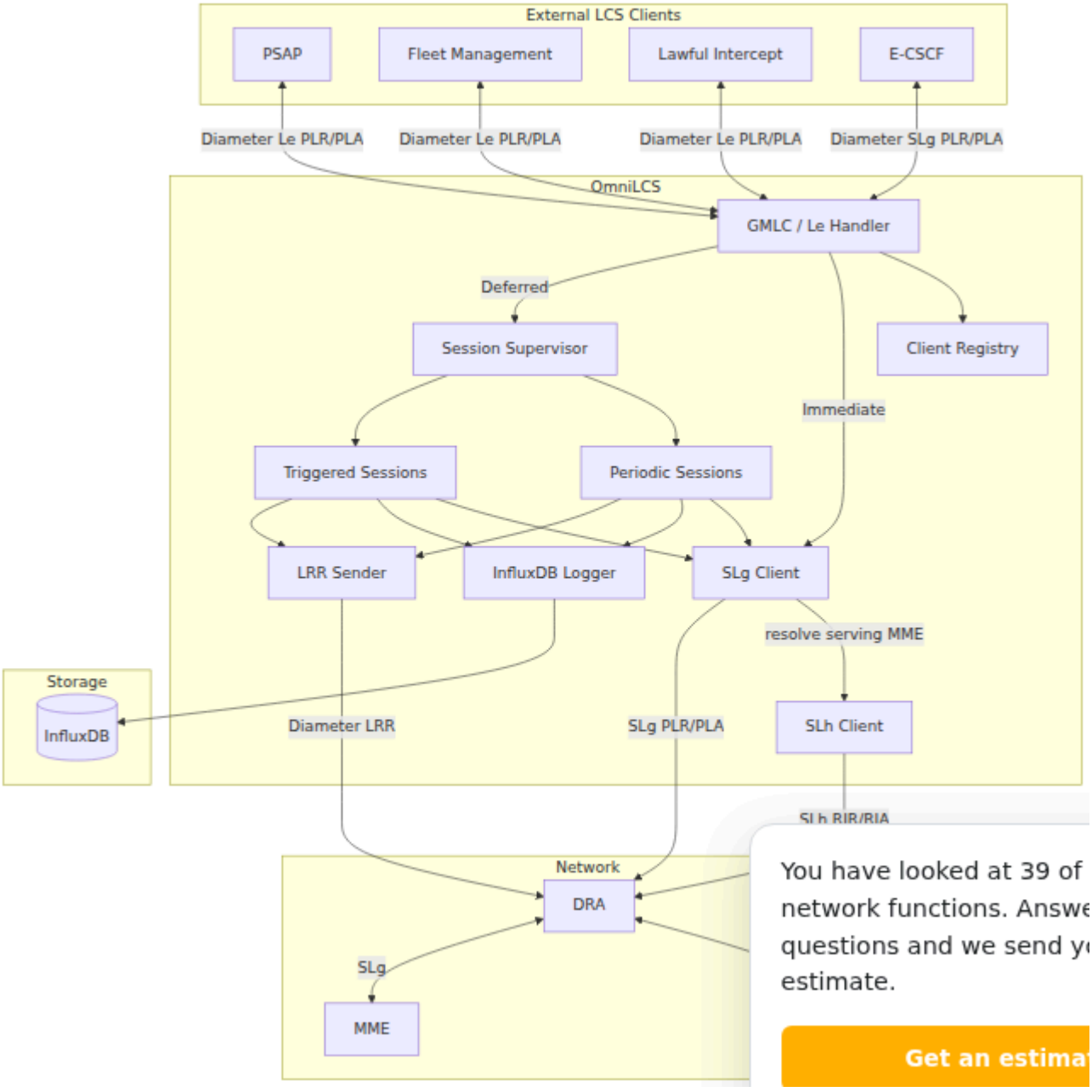
GMLC & Le 接口□作指南

OmniLCS 包含一个网关移动位置中心 (GMLC) 功能，该功能向外部 LCS 客户端暴露 **Le** 接口。这允许公共安全应答点 (PSAP)、车队管理系统、合法拦截平台和其他外部方通过 3GPP TS 29.172 中定义的基于 Diameter 的 Le 参考点请求订阅者位置。

GMLC 支持 即时 和 延迟 位置请求。延迟请求包括 周期性位置（在固定间隔报告位置）和 触发位置（当订阅者进入、离开或停留在地理区域内时报告位置）。

为了满足请求，GMLC 遵循标准的 3GPP 程序：它通过 **SLh** 接口发现订阅者的服务 MME（向 HSS 发送 `LCS-Routing-Info-Request`，根据 TS 29.173），然后通过 **SLg** 接口向该 MME 发送 **Provide-Location-Request (PLR)**。MME 执行定位（直接或通过 SLs/LCS-AP 委托给 E-SMLC）并在 Provide-Location-Answer (PLA) 中返回 `Location-Estimate`。在此 **GMLC** 角色 中 ���OmniLCS 不直接定位 UE。它查询服务 MME 并转发结果。

> 这□ OmniLCS 提供的几种位置选项之一。对于主动的高精度定位，它使用其 **E-SMLC** 角色 通过 SLs，而不□直接驱动 LPP/LPPa 到 UE 和 eNB（请参见 两个 LCS 角色，一个位置请求）。对于 **2G/3G** 的订阅者，GMLC 通过 **MAP/Lg** 到达服务 MSC/SGSN，而不□通过 Diameter SLg（请参见 2G/3G 通过 MAP/Lg 定位）。相同的 `POST /api/location` 调用可以到达每条路径。这里描述的 GMLC/SLg 路径□为 4G `cell`/最后已知请求选择的。

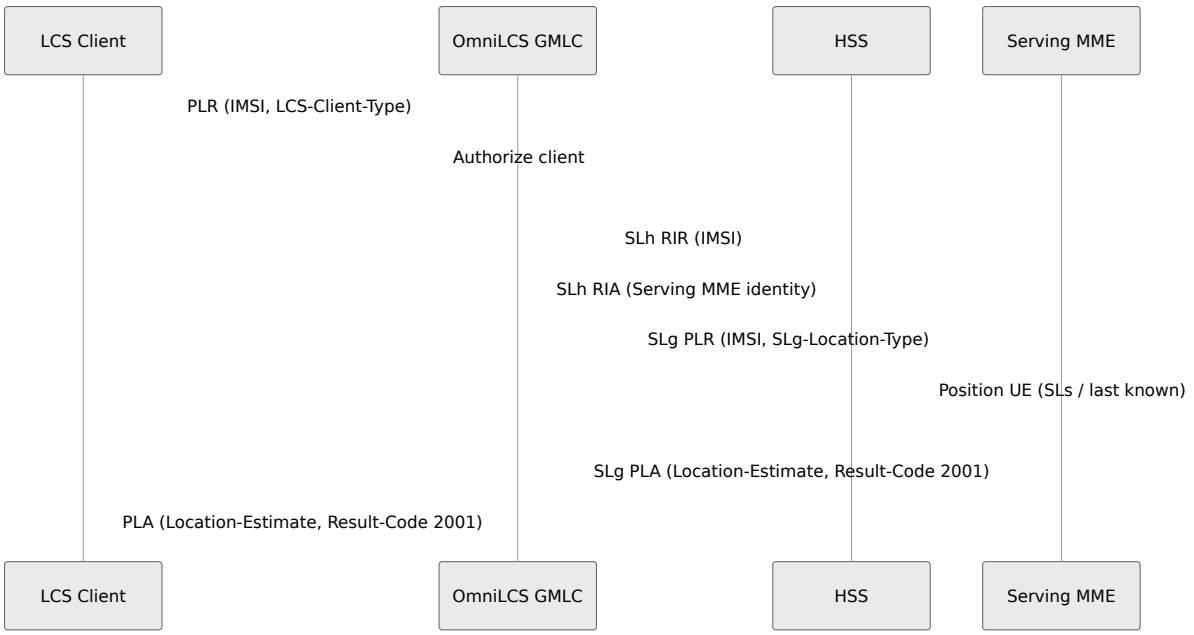


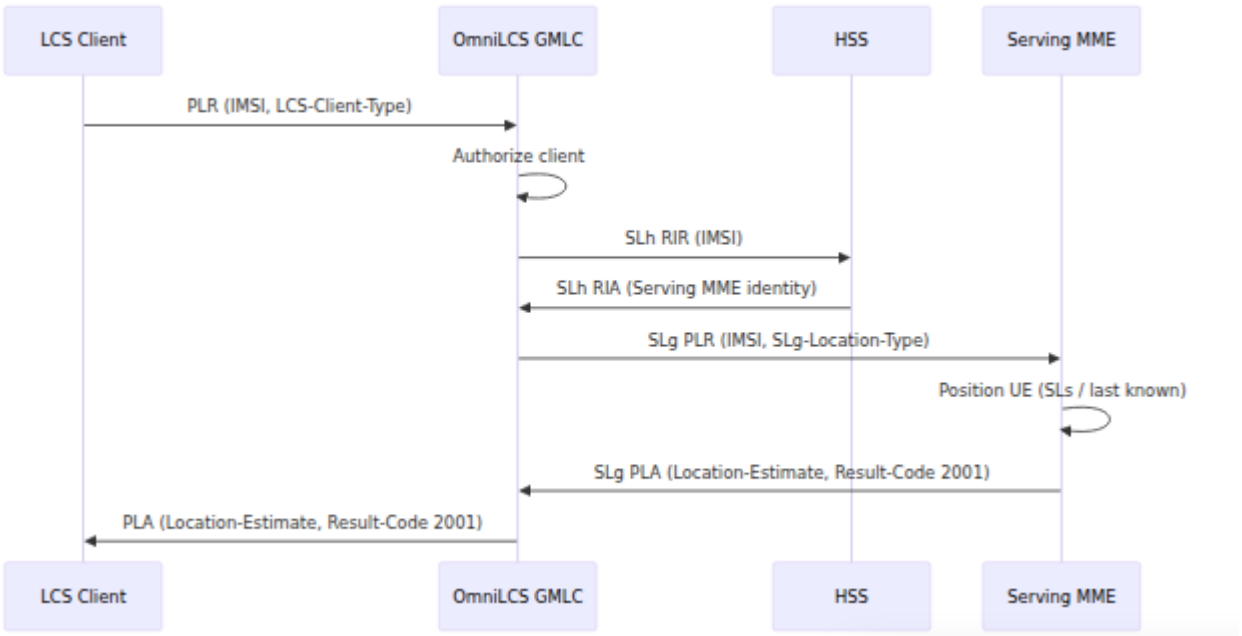
架构

请求流程

即时位置请求

外部 LCS 客户端通过 Diameter 发送 Provide-Location-Request (PLR) ◦OmniLCS 授权客户端，发现服务 MME，通过 SLg 向该 MME 发送 PLR，并在 Provide-Location-Answer (PLA) 中返回 MME 的结果。



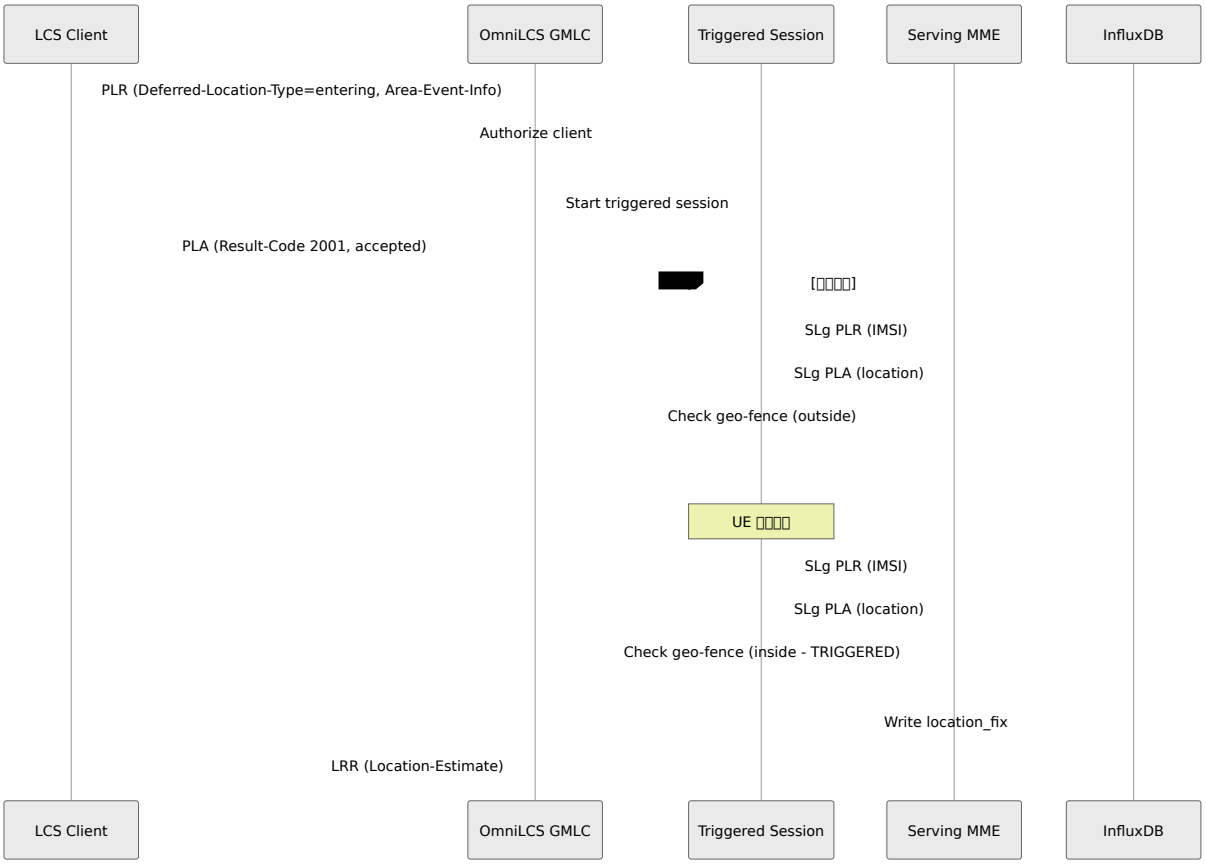


周期性位置请求

当 PLR 包含带有周期位设置的 `Deferred-Location-Type` 和分组 AVP `Periodic-LDR-Information` 时，OmniLCS 创建一个周期会话，在指定的间隔内运行位置修正，并通过位置报告请求 (LRR) 交付每个结果。

触发（地理围栏）位置请求

当 PLR 包含区域事件信息时，OmniLCS 创建一个触发会话，轮询 UE 位置，并在触发条件满足时触发。

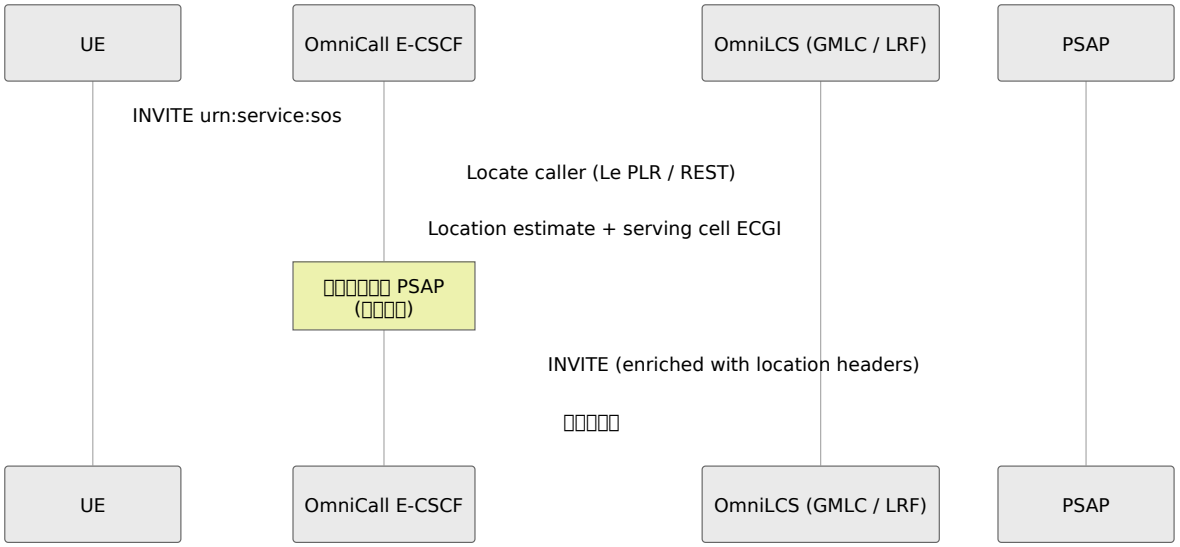


使用 **OmniCall (IMS)** 的紧急呼叫路由

GMLC 的主要消费者 **OmniCall**，IMS 核心。当订阅者拨打紧急号码时，OmniCall 的 **E-CSCF**（紧急呼叫会话控制功能）必须将呼叫路由到服务呼叫者物理位置的 **PSAP**。路由到错误的 PSAP 会浪费宝贵的时间。OmniCall 从 OmniLCS 获取该位置，后者充当 3GPP TS 23.167 中的 **LRF / RDF**（位置检索功能 / 路由确定功能）。

流程：

1. UE 发起 IMS 紧急呼叫（`urn:service:sos`），到达 OmniCall 的 E-CSCF。
2. OmniCall 通过 **Le** 接口（Diameter PLR，使用 `:emergency_services` 客户端类型）或 REST API 请求 OmniLCS 获取呼叫者的位置。OmniLCS 根据适合的角色进行响应：服务小区的 GMLC/SLg，或处于活动状态的 GNSS/OTDOA 修正的 E-SMLC/SLs（请参见 两个 LCS 角色，一个位置请求）。
3. OmniLCS 返回位置（坐标 + 服务小区 ECGI）。
4. OmniCall 将该位置映射到服务 PSAP（路由确定），并将 INVITE 转发到那里。
5. OmniCall 丰富 **SIP** 信令，使 PSAP 在接到呼叫时接收到位置。



SIP 位□□□头丰富

使用 OmniLCS 返回的位置，OmniCall 将标准 IMS 紧急位置元素添加到外发 INVITE 中，以便 PSAP（和任何中介）在呼叫到达时接收到呼叫者的位置，而不□必须单独查询：

SIP 元素	标准	承载
`Geolocation` header + PIDF-LO body	RFC 6442 / RFC 4119	通过值或引用传递的地理（纬度/经度 + 不确定性）和/或市民位置
`P-Access-Network-Info`	RFC 7315	服务接入网络和小区标识 (ECGI)
`Geolocation-Routing`	RFC 6442	下游代理□否可以根据传递的位置进行路由

因此，PSAP 呼叫接听员在呼叫到达时看到呼叫者的位置，该位置源自 OmniLCS 在其他地方使用的相同定位：服务小区以快速修正，或在 UE 和无线电支持时的活动 GNSS/OTDOA 位置。

配置

GMLC 配置

```
config :omnilcs, :gmlc,
  # 启用/禁用 GMLC Le 接口
  enabled: true,

  # 允许来自未授权列表中的客户端的请求
  allow_unknown_clients: false,

  # 授权的外部 LCS 客户端
  authorized_clients: [
    %{
      name: "psap-01",
      type: :emergency_services,
      allowed_methods: [:cell, :ecid, :gnss, :otdoa],
      rate_limit: 100,
      description: "主要 PSAP"
    },
    %{
      name: "fleet-mgmt",
      type: :value_added_services,
      allowed_methods: [:cell, :ecid],
      rate_limit: 50,
      description: "车队管理系统"
    }
  ],

  # 允许延迟（周期性/触发）位置请求
  allow_deferred: true,

  # 最大并发周期会话
  max_periodic_sessions: 100,

  # 最大并发触发（地理围栏）会话
  max_triggered_sessions: 50,

  # 默认周期会话的间隔（毫秒）
  default_periodic_poll_interval_ms: 60_000,

  # 默认触发会话的轮询间隔（毫秒）
```

```
  default_triggered_poll_interval_ms: 30_000,

  # 将所有位置修正记录到 InfluxDB
  influx_logging: true
```

GMLC 参数

参数	类型	必需	默认	描述
`enabled`	布尔值	否	`false`	启用 GMLC Le 接口处理[当禁用时，所有 PLR 都作为标 SLg 请求处理。
`allow_unknown_clients`	布尔值	否	`false`	接受未列在 `authorized_clier` 中的客户端的请求。集成测试期间 为 `true` 以允许模式。
`authorized_clients`	列表	否	`[]`	授权的外部 LCS 客户端配置[请参见下面的 客户端参数。
`allow_deferred`	布尔值	否	`true`	接受延迟（周期性和触发）位置请 当 `false` 时，仅处理即时[
`max_periodic_sessions`	整数	否	`100`	最大并发周期位置会话数。超过时 Diameter 3004 (TOO_BUSY)。
`max_triggered_sessions`	整数	否	`50`	最大并发触发位置会话数。
`default_periodic_poll_interval_ms`	整数	否	`60000`	当 PLR 中未指定时，周期位置 之间的默认间隔。
`default_triggered_poll_interval_ms`	整数	否	`30000`	当在 Area-Event-Info AVP 中未指定时，触发会话的[轮询间隔。
`influx_logging`	布尔	否	`true`	将所有来自 GMLC 会话的位 正写入 InfluxDB。

□□	□□	□□	□□	□□
	□			

□□□□□

□□ `authorized_clients` □□□□□

□□	□□	□□	□□	
`name`	□□□	□	--	□□□□□□□ PLR □□ LCS-EPS-Client-Name
`type`	□□	□	`:any`	□□□ LCS-Client-Type□ `:emergency_services`□`:value_added` □ `:any`□
`allowed_methods`	□□	□	`[:cell, :ecid, :gnss, :otdoa]`	□□□□□□□□□□□□□□□
`rate_limit`	□□	□	`0`	□□□□□□□□□□ `0` □□□□□□□
`description`	□□□	□	`""`	□□□□□□□□□□□

LCS □□□□□

□□ 3GPP TS 29.172 □ 7.4.4 □□

值	类型	描述
0	紧急服务	PSAP, E-CSCF 紧急呼叫路由
1	增值服务	车队管理、资产跟踪、基于位置的服务
2	PLMN 运营商服务	运营商内部服务（O&M、网络优化）
3	合法拦截服务	执法位置请求

延迟位置类型

周期性位置

周期性位置会话在固定间隔内执行可配置数量的位置修正。每个修正：

1. 通过向订阅者的服务 MME（通过 SLh 解析）发送 SLg PLR 来执行
2. 作为带有会话元数据的 `location_fix` 测量写入 InfluxDB
3. 通过标准位置记录器（CSV + ETS）记录
4. 通过 Diameter LRR 将其交付给发起的 LCS 客户端

当报告计数耗尽时会话终止。

使用的 **Diameter AVP**：

AVP	代码	描述
`Deferred-Location-Type`	1480	位 4 设置表示周期性 LDR
`Periodic-LDR-Information`	2025	包含报告参数的分组 AVP
`Reporting-Amount`	2026	要执行的位置修正数量
`Reporting-Interval`	2027	修正之间的间隔（以秒为单位）

触发（地理围栏）位置

触发位置会话轮询 UE 位置并将其与一个或多个地理区域进行评估。当触发条件满足时，报告位置。

事件类型：

事件	**Deferred-Location-Type** 位	触发条件
进入	位 1	UE 从区域外部过渡到内部
离开	位 2	UE 从区域内部过渡到外部
在内部	位 3	UE 在每次轮询时都在区域内

支持的区域定义：

- 圆形：中心点（纬度、经度）和半径（米）
- 多边形：定义封闭多边形的顶点列表

地理围栏评估器重用 CAP 警报多边形解析器中的射线投射算法进行多边形包含检查，并使用哈弗辛距离进行圆形区域检查。

InfluxDB 位置记录

所有 GMLC 会话修正都使用 `location_fix` 测量写入 InfluxDB：

标签：

标签	描述
`imsi`	订阅者 IMSI
`method`	使用的位置方法（cell、ecid、gnss、otdoa）
`source`	来自引擎结果的位置来源
`session_type`	`periodic`、`triggered` 或 `immediate`
`client_name`	请求 LCS 客户端的名称

字段：

字段	类型	描述
`latitude`	浮点数	WGS84 纬度（以度为单位）
`longitude`	浮点数	WGS84 经度（以度为单位）
`altitude`	浮点数	海拔（米）（可用时）
`uncertainty`	浮点数	位置不确定性（米）
`confidence`	整数	置信水平（0-100）
`duration_ms`	整数	执行修正所需的时间

示例 **InfluxQL** 查询：

```
-- 获取订阅者的最新修正
SELECT * FROM location_fix WHERE imsi = '001010000000001' ORDER BY
time DESC LIMIT 10

-- 获取过去一小时的周期会话轨迹
SELECT latitude, longitude FROM location_fix
WHERE session_type = 'periodic' AND imsi = '001010000000001' AND
time > now() - 1h

-- 按方法获取修正成功率
SELECT COUNT(*) FROM location_fix WHERE time > now() - 24h GROUP
BY method

-- 地理围栏触发事件
SELECT * FROM location_fix WHERE session_type = 'triggered' AND
time > now() - 24h
```

REST API

延迟位置 API 可在 `https://<host>:8443/api/deferred_location` 上使用。

列出活动会话

```
GET /api/deferred_location
```

响应：

```
{
  "status": "ok",
  "count": 2,
  "data": [
    {
      "session_id": "a1b2c3d4-...",
      "type": "periodic",
      "imsi": "001010000000001",
      "method": "cell",
      "client_name": "rest-api",
      "status": "active",
      "interval_ms": 60000,
      "remaining_reports": 7,
      "total_reports": 10,
      "started_at": "2026-04-09T10:00:00Z",
      "last_fix_at": "2026-04-09T10:03:00Z"
    }
  ]
}
```


□□□□□□

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

{
  "type": "periodic",
  "imsi": "001010000000001",
  "method": "cell",
  "interval_seconds": 60,
  "count": 10
}
```

创建触发会话

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

{
  "type": "triggered",
  "imsi": "001010000000001",
  "method": "cell",
  "event_type": "entering",
  "poll_interval_seconds": 30,
  "max_reports": 0,
  "areas": [
    {
      "type": "circle",
      "center": {"lat": -33.8688, "lon": 151.2093},
      "radius_meters": 500
    }
  ]
}
```

取消会话

```
DELETE /api/deferred_location/:session_id
```

指标

Le 接口请求

指标: `omnilcs_gmlc_le_request_total` 类型: 计数器 描述: 来自外部 LCS 客户端的总 Le 接口请求 标签:

- `client_type`: LCS 客户端类型 (emergency_services, value_added_services, 等)
- `result`: 请求结果: `received`, `success`, `error`, `unauthorized`

��标: `omnilcs_gmlc_le_request_duration` 类型: 直方图 描述: Le 请求处理时间（以毫秒为单位） 标签:

- `client_type`: LCS 客户端类型

MAP / Lg 请求（2G/3G）

指标：`omnilcs_map_request_total` 类型：计数器 描述：2G/3G 订阅者的总 MAP/Lg 位置请求 (SendRoutingInfoForLCS, ProvideSubscriberLocation) 标签：

- `type`：MAP 操作（`MAP-PSL`）
- `result`：`sent`，`success`

位置报告请求

指标：`omnilcs_gmlc_lrr_total` 类型：计数器 描述：发送到外部客户端的总位置报告请求 标签：

- `session_type`：`periodic` 或 `triggered`
- `result`：`sent` 或 `error`

会话跟踪

指标：`omnilcs_gmlc_session_periodic_active` 类型：测量 描述：活动周期位置会话的数量

指标：`omnilcs_gmlc_session_triggered_active` 类型：测量 描述：活动触发/地理围栏位置会话的数量

指标：`omnilcs_gmlc_session_total` 类型：计数器 描述：创建的总延迟会话 标签：

- `type`：`periodic` 或 `triggered`

地理围栏事件

指标：`omnilcs_gmlc_geofence_trigger_total` 类型：计数器 描述：触发的总地理围栏事件 标签：

- `event_type`：`entering`，`leaving`，或 `being_inside`

InfluxDB ���入

指标：`omnilcs_gmlc_influx_write_total` 类型：计数器 描述：来自 GMLC 会话的总 InfluxDB 位置修正写入 标签：

- `result`：`success` 或 `error`

示例 **Prometheus** 查询：

```
# Le 请求速率
rate(omnilcs_gmlc_le_request_total[5m])

# 按类型的延迟会话创建速率
rate(omnilcs_gmlc_session_total[5m])

# 地理围栏触发速率
rate(omnilcs_gmlc_geofence_trigger_total[5m])

# LRR 交付错误比例
sum(rate(omnilcs_gmlc_lrr_total{result="error"}[5m]))
  / sum(rate(omnilcs_gmlc_lrr_total[5m]))
```

故障排除

Le 请求返回 5012 (UNABLE_TO_COMPLY)

症状：外部 LCS 客户端对所有 PLR 接收到结果代码 5012。

可能原因：

- GMLC 在配置中未启用（`enabled: false`）
- 客户端不在 `authorized_clients` 中，且 `allow_unknown_clients` 为 `false`
- SLh 解析失败或没有到服务 MME 的 SLg 路由（检查 DRA/HSS/MME 连接性）

解决方案：

1. 验证 `config :omnilcs, :gmlc, enabled: true` 已设置
2. 检查客户端名称□否与 PLR 中的 `LCS-EPS-Client-Name` AVP 匹配
3. 验证 SLh/SLg Diameter 对等方（DRA、HSS、MME）在仪表板上□否连接

周期会话未启动

症状：带有周期延迟类型的 PLR 返回 3004 (TOO_BUSY)。

可能原因：

- 达到 `max_periodic_sessions` 限制
- 会话监督器未运行

解决方案：

1. 检查 GMLC 控制面板页面上的活动会话计数
2. 如有需要，增加 `max_periodic_sessions`
3. 验证 `OmniLcs.Gmlc.SessionSupervisor` 是否在应用程序页面中运行

位置修正未出现在 InfluxDB 中

症状：周期/触发会话正在运行，但 InfluxDB 中没有数据。

可能原因：

- `influx_logging` 设置为 `false`
- InfluxDB 连接中断
- 数据库不存在

解决方案：

1. 验证 GMLC 配置中的 `influx_logging: true`
2. 检查 InfluxDB 连接性（单元同步状态是一个很好的代理）
3. 直接查询 InfluxDB: `SHOW MEASUREMENTS` 应该包括 `location_fix`

地理围栏未触发

症状：触发会话处于活动状态但从未触发。

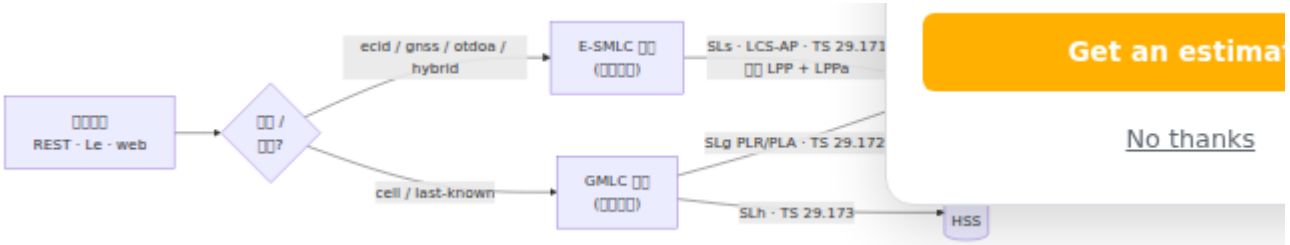
可能原因：

- 区域定义对于定位方法的精度过小
- 错误的事件类型（例如，当订阅者已经在外部时使用 `:leaving`）
- 定位方法返回 `nil` 坐标

解决方案：

1. 检查定位方法的精度。小区 ID 具有公里级的不确定性，这可能对小地理围栏来说过于粗糙。

2. 对于较小的区域，使用更精确的方法（`:ecid` 或 `:gnss`）
3. 验证区域坐标□否正确（纬度/经度顺序）



E-SMLC 位置服务指南

OmniLCS 实现了一个演进的服务移动位置中心 (E-SMLC)，为 LTE 网络提供 UE 定位。E-SMLC 使用多种定位方法确定 UE 位置，并通过 SLs 接口使用原生 LCS-AP 通过 SCTP 与 MME 通信，符合 3GPP TS 29.171。

两个 LCS 角色，一个位置请求

OmniLCS 同时作为 **E-SMLC** 和 **GMLC** 共同存在。这□ 3GPP LTE LCS 架构中定义的两个位置网络功能 (TS 23.271)。位置客户端从不在它们之间做选择。它通过 REST API、位置网页或 Le 接口请求 UE 的位置，OmniLCS 根据请求的 `method`/`accuracy` 选择角色和接口。这两个选项始终可通过一个“获取位置”调用使用。

- 主动/高精度方法 (E-CID, GNSS, OTDOA, 混合) 使用本指南中记录的 **E-SMLC** 角色。OmniLCS 驱动 **LPP** (到 UE, TS 36.355) 和 **LPPa** (到 eNB, TS 36.455) 通过 **SLs / LCS-AP** 接口 (TS 29.171)，由服务 MME 中继，以获得精确定位。
- 小区 **ID /** 最后已知请求使用 **GMLC** 角色 (参见 GMLC & Le 接口指南)。OmniLCS 通过 **SLh** (TS 29.173) 解析服务 MME，并通过 **SLg** (TS 29.172) 发送 Provide-Location-Request；MME 返回 UE 上下文中的服务小区。

在这两个角色中，服务 **MME** 是锚点：它□唯一与 UE (NAS) 和 eNB (S1AP) 之间有信令路径的节点，因此它要么□ SLg 响应者，要么□ LPP/LPPa 的 SLs 中继。GMLC 仅通过 SLh/SLg Diameter 链接，无法直接到达无线电。这正□主动定位□ E-SMLC 的工作的原因。

请参见 REST API 参考 以获取统一的请求/响应架构和完整的决策表。REST API 根据请求选择路径；Le Diameter 接口和延迟（周期性/触发）会话当前始终使用 GMLC/SLg 路���。

2G/3G 通过 MAP/Lg 定位

对于 **GERAN (2G)** 或 **UTRAN (3G)** 上的用户，无法使用 Diameter SLg/SLs。GMLC 通过 **Lg** 接口使用 **MAP** (TS 29.002) 通过 SS7/SIGTRAN 堆栈（SCTP → M3UA → SCCP → TCAP）访问传统核心。OmniLCS 通过 `OmniLcs.Control.MapClient` 模块提供此功能，支持 `omniss7` SS7 堆栈，作为同一 `/api/location` 调用后面的第三个选项（通过 `rat: "2g"` / `"3g"` 选择）。

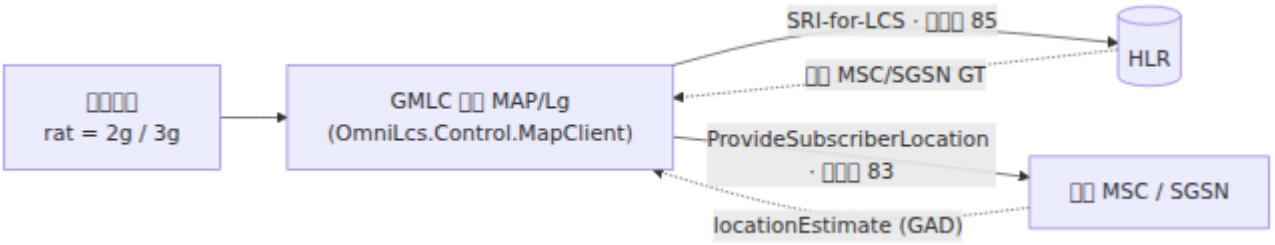
该流程□一个两步 MAP 事务:

1. **SendRoutingInfoForLCS** (操作码 85) 到 **HLR** → 返回用户的服务 **MSC/SGSN** 全球标题。(如果提供了 `node_gt` 则跳过。)
2. **ProvideSubscriberLocation** (操作码 83) 到该 **MSC/SGSN** → 返回位置估计。

`locationEstimate` 的形状与 4G 的 **TS 23.032 GAD** 相同,因此使用共享的 `decode_gad/1` 解码,并在标准位置映射中返回 `source: "map"`。

在 2G/3G 上的精度为小**□□-ID /** 时间提前类别:网络返回服务小区,而不□精确定位。此路径默认禁用 (`config :omnilcs, :map, enabled: false`)。要启用它,请提供 GMLC 号码、HLR 全球标题和通过 `omniss7` 的 `:map_client_m3ua` 传输配置的 SS7 对等体 (STP/MSC/SGSN)。请参见 GMLC 指南 和 配置。

□作。`omniss7` 以无头运行(库模式:仅 MAP 编解码器和事务注册)。当 `:omnilcs, :map` 被启用时,OmniLCS 启动到 STP 本身的 **M3UA** 关联(由 `OmniLcs.Control.MapStack` 监督)。当禁用时,它不启动任何内容。每个 MAP 事务都会增加 `omnilcs_map_request_total` Prometheus 计数器(标签 `type` = `MAP-PSL`, `result` = `sent`/`success`)。在路径禁用时,带有 `rat: "2g"`/`"3g"` 的请求返回 HTTP `501`。未解析的服务节点返回 `422`。



架构

定位方法

OmniLCS 支持全系列的 LTE 定位技术。请求指定一个 `method`（或目标 `accuracy`，自动选择一个）；OmniLCS 在适当的接口上运行该技术并返回估计。每种方法映射到一个标准的 3GPP 定位程序：

方法	技术	定位协议	角色 / 接口	典型精度	需求
小区 **ID (CID)**	服务小区位置	小区数据库查找，或 SLg PLR	E-SMLC（本地）或 GMLC (SLg)	小区半径，100 m – 5 km	无
增强小区 **ID (E-CID)**	单小区测距：时间提前 + RSRP/RSRQ	LPPa (eNB) → LPP (UE) 回退	E-SMLC / SLs	50 m – 500 m	eNB □ UE
OTDOA	从多个小区的 PRS 时间差进行下行多边定位（“三��测量”）	LPPa (PRS 配置) + LPP (RSTD)	E-SMLC / SLs	10 m – 100 m	UE + eNB PRS
A-GNSS	在 UE 中辅助卫星 (GPS/Galileo/…) 定位	LPP (UE)	E-SMLC / SLs	5 m – 50 m	UE GNSS
混合	在一个请求中结合 A-GNSS + E-CID，最佳选择	LPP (UE)	E-SMLC / SLs	最佳可用	UE

> 这里“三角测量”的含义。 LTE UE 的真正三角测量/多边定位 **OTDOA**：UE 测量来自多个小区的下行定位参考信号 (PRS) 的时间差，OmniLCS 解决由此产生的双曲线以确定位置（参见 OTDOA 计算）。E-CID 更粗略的单小区表亲。它从一个小区的时间提前范围，而不多小区几何。上行多边定位（**U-TDOA**，网络测量）未实现：它需要在仅 EPC 部署中不存在的定位测量单元 (LMUs)。

根据精度选择方法

如果请求提供 `accuracy`（以米为单位）而不明确的 `method`，OmniLCS 选择能够合理满足目标的最轻方法：

请求精度	选择的方法
≤ 50 m	A-GNSS
≤ 200 m	OTDOA
≤ 500 m	E-CID
> 500 m	小区 ID

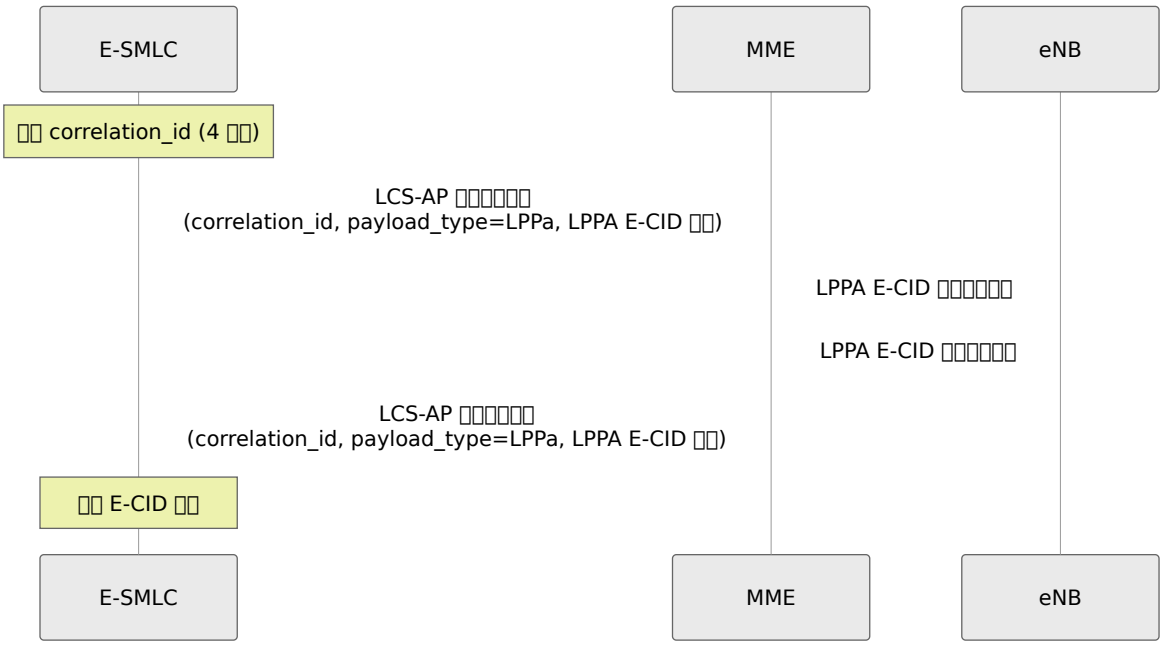
小区 ID

最简单的定位方法。返回服务小区在小区数据库中的地理位置。

- 精度：小区半径（通常根据小区类型为 100m - 5km）
- 延迟：毫秒（本地数据库查找）
- **UE** 支持要求：无
- **eNB** 支持要求：无

引擎首先尝试在本地小区数据库中查找服务小区。如果未找到，它通过 MME 使用 LCS-AP 连接导向信息向 eNB 发送 LPPA E-CID 测量启动请求（仅带小区 ID 数量）以获取小区身份。

同一服务小区也可以通过 GMLC 角色在不接触无线电的情况下获得。这使用 SLg Provide-Location-Request 向服务 MME 请求，后者返回 UE 上下文中的 ECGI。这□用于 `cell`/最后已知 REST 请求和所有 Le 接口请求的路径（参见 两个 LCS 角色，一个位置请求）。



增强小区 ID (E-CID)

使用 LPPa 从 eNB 获取无线电测量，包括时间提前和信号强度。如果 eNB 拒绝 LPPa E-CID 程序（例如 `requested-item-not-supported`），OmniLCS 会回退到直接从 **UE** 请求 E-CID 测量。

- 精度：50m - 500m
- ��迟：1-5 秒
- **UE** 支持要求：LPPa 路径不需要；LPP 回退需要
- **eNB** 支持要求：LPPa 路径需要；LPP 回退不需要

E-SMLC
MME
eNB
生成 correlation_id (4 字节)
LCS-AP 连接导向信息
(correlation_id, payload_type=LPPa, LPPA E-CID 请求)
LPPA E-CID 测量启动请求
LPPA E-CID 测量启动响应
LCS-AP 连接导向信息
(correlation_id, payload_type=LPPa, LPPA E-CID 响应)
处理 E-CID 测量
E-SMLC
MME
eNB

请求的测量数量：

数量	描述
小区 ID	服务小区身份
时间提前类型 2	UE 到 eNB 的传播延迟（距离估计）
RSRP	参考信号接收功率
RSRQ	参考信号接收质量

GNSS / A-GPS

请求 UE 通过 LPP 协议报告其 GPS 坐标。

- 精度: 5m - 50m
- 延迟: 5-30 秒
- **UE** 支持要求: □ (GNSS 功能)
- **eNB** 支持要求: 无

E-SMLC 通过 MME 通过 LCS-AP 向 UE 发送 LPP 请求位置信息消息以请求 GNSS。UE 执行 GNSS 定位并通过 LPP 报告其坐标，经过 MME 在连接导向信息消息中中继。

当配置 GNSS 辅助时（参见 GNSS 辅助数据），E-SMLC 首先发送 LPP 提供辅助数据消息，携带 GPS 广播导航模型（卫星历）、参考时间和参考位置。这使得 UE 能够在不首先解调卫星的 30 秒导航帧的情况下计算定位，从而缩短冷启动时间。如果没有辅助，UE 将从其存储的数据中执行无辅助定位，这在冷启动时较慢。

GNSS 辅助数据（卫星历）

辅助数据由 `OmniLcs.Esmlc.GnssAssistance` 构建，并根据 TS 36.355 §6.5.1 编码（GPS 模型-2：NAV 时钟 + NAV 开普勒）。默认情况下□关闭的，并根据部署启用：

```
config :omnilcs, :gnss_assistance,
  enabled: true,
  rinex_nav_file: "/var/lib/omnilcs/brdc.rnx",
  reference_location: %{latitude: 36.85, longitude: -76.28,
altitude: 5},
  gps_utc_offset_seconds: 18,
  max_age_hours: 4
```

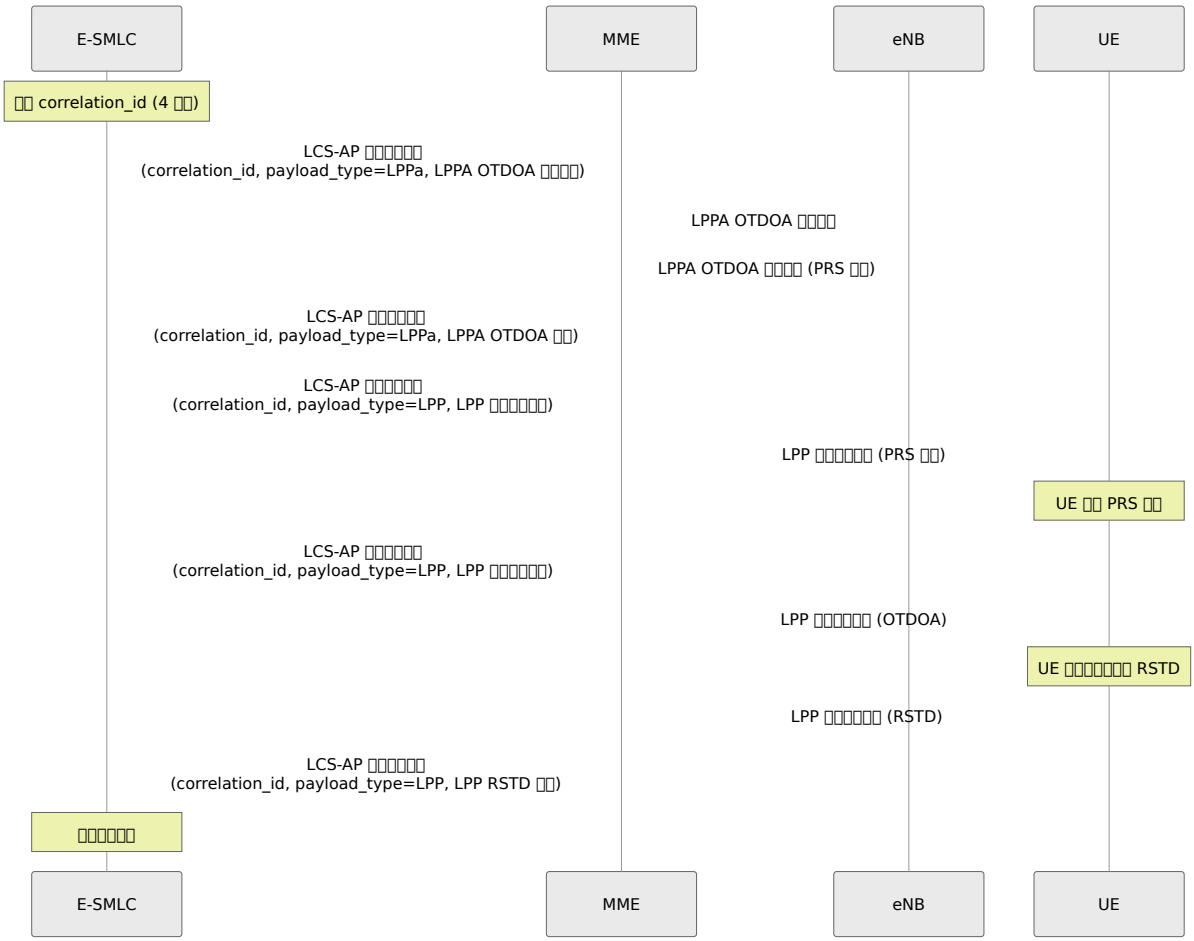
- **`rinex_nav_file`** — **RINEX 3.x GPS** 导航文件 的路径。操作员提供并在外部刷新（例如，每小时/每日的 cron 获取来自 IGS/NASA CDDIS 产品或本地 GNSS 接收器的广播卫星历）。GPS 卫星历有效期大约为两小时，因此过期的文件几乎没有用。
- **`reference_location`** — 近似站点位置（度/米），用于限制 UE 的搜索。设置为 `nil` 以省略。
- **`gps_utc_offset_seconds`** — 当前 GPS−UTC 偏移（闰秒，自 2017 年以来为 18）；用于标记参考时间。
- **`max_age_hours`** — 超出此窗口的时间戳卫星历将被拒绝。

当功能 启用但源缺失、不可读、错误的 **RINEX** 版本或未提供新鲜健康的 **GPS** 卫星历时，定位尝试将失败并返回错误，而不□静默回退到无辅助请求。今天仅编码 GPS；尚未构建 Galileo/GLONASS/BeiDou 辅助。

OTDOA

观察到的到达时间差。使用来自多个小区的定位参考信号 (PRS) 测量进行多边定位。

- 精度: 10m - 100m
- 延迟: 5-15 秒
- **UE ���**持要求: □ (OTDOA 功能)
- **eNB** 支持要求: □ (配置 PRS, LPPA)



E-SMLC
MME
eNB
UE
生成 correlation_id (4 字节)
LCS-AP 连接导向信息
(correlation_id, payload_type=LPPa, LPPA OTDOA 信息请求)
LPPA OTDOA 信息请求
LPPA OTDOA 信息响应 (PRS 配置)
LCS-AP 连接导向信息
(correlation_id, payload_type=LPPa, LPPA OTDOA 响应)
LCS-AP 连接导向信息
(correlation_id, payload_type=LPP, LPP 提供辅助数据)
LPP 提供辅助数据 (PRS 配置)
UE 处理 PRS 配置
LCS-AP 连接导向信息
(correlation_id, payload_type=LPP, LPP 请求位置信息)
LPP 请求位置信息 (OTDOA)
UE 从多个小区测量 RSTD
LPP 提供位置信息 (RSTD)
LCS-AP 连接导向信息
(correlation_id, payload_type=LPP, LPP RSTD 测量)
多边定位计算
E-SMLC
MME
eNB
UE

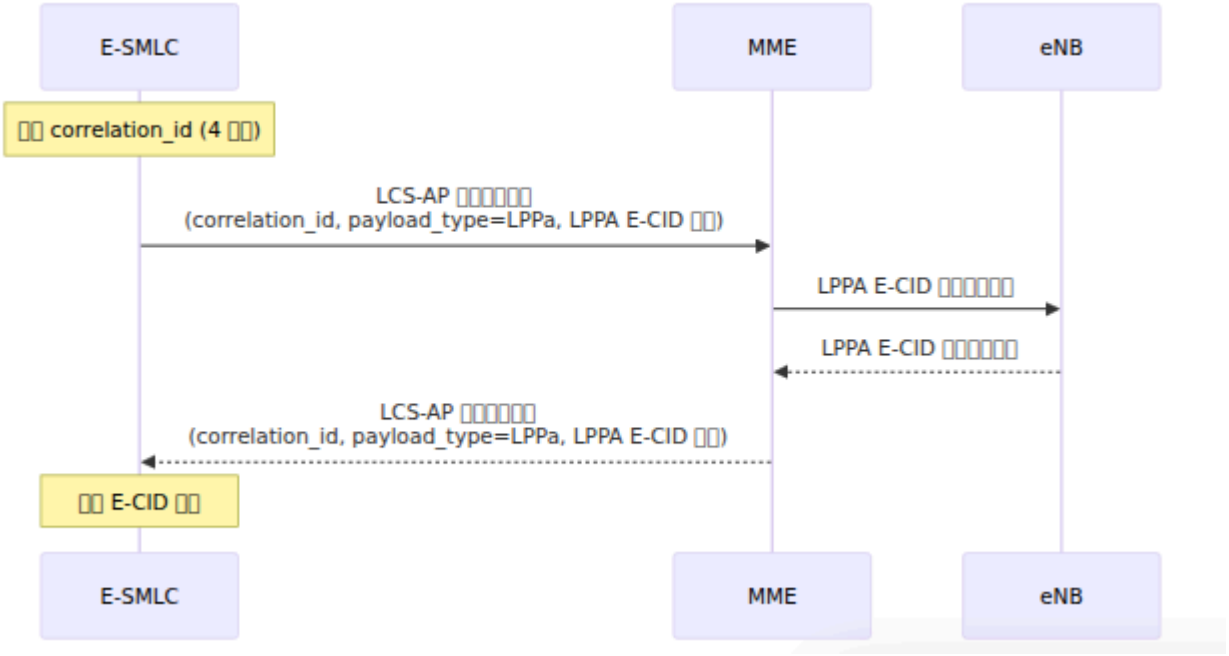
OTDOA 步骤:

1. 通过 LPPA 请求 eNB 的 OTDOA 小区信息（通过 LCS-AP 连接导向信息中继）
2. 接收 PRS 配置、小区 ID、参考和邻居小区的 EARFCN
3. ��过 LPP 向 UE 发送 PRS 辅助数据（通过 LCS-AP 连接导向信息中继）
4. 通过 LPP 请求 UE 的 RSTD 测量
5. 接收 RSTD 测量
6. 运行多边定位以计算位置

混合

在单个 LPP 事务中请求 A-GNSS 和 E-CID，并返回最佳可用定位，在 UE 无法完成 GNSS 定位时回退到较粗糙的 E-CID 测量。

- 精度：A-GNSS (5-50m) 和 E-CID (50-500m) 中的最佳
- 延迟：5-30 秒（受 GNSS 定位限制）
- **UE** 支持要求：□（GNSS + 测量报告）
- **eNB** 支持要求：无



OmniLCS 发送一个合并的 LPP 请求位置信息以请求 GNSS + E-CID，并合并 UE 返回的所有内容，优先考虑 GNSS 位置。这最小化了信令，相比单独运行这两种方法。

SLs 接口: LCS-AP 通过 SCTP

E-SMLC ��用 LCS 应用协议 (LCS-AP) 通过 SCTP 与 MME 通信，符合 3GPP TS 29.171。OmniLCS 向每个配置的 MME 对等体发起 SCTP 关联。

传输细节

参数	值
协议	LCS-AP 通过 SCTP
SCTP PPID	29
默认端口	9082 (IANA 注册)
连接方向	E-SMLC 向 MME 发起
编码	ASN.1 PER 对齐

LCS-AP 程序

程序	代码	描述
位置请求 / 位置响应	0	E-SMLC 请求 MME 定位 UE
连接导向信息	1	中继 LPPA/LPP PDU（与 UE 关联，使用 correlation_id）
无连接信息	2	中继非 UE 关联的 LPPA PDU
位置中止	3	中止进行中的位置程序
重置	4	重置所有位置程序

关联 ID

每个与 UE 关联的 LCS-AP 事务使用关联 **ID**。这□一个 4 字节的二进制值，将位置请求、所有连接导向信息交换和位置响应联系在一起，形成一个单一的定位会话。E-SMLC 在每个会话开始时生成一个随机的关联 ID，使用 `:crypto.strong_rand_bytes(4)`。

关联 ID 作为 IE (ID 2) 包含在:

- 位置请求 (E-SMLC -> MME)
- 位置响应 (MME -> E-SMLC)
- 连接导向信息（双向）
- 位置中止 (E-SMLC -> MME)

APDU 隧道

LPP 和 LPPA PDU 被包含在 LCS-AP 连接导向信息 消息中。有效负载类型 IE (ID 15) 标识内容:

有效负载类型	值	描述
`:lPP`	0	LPP PDU (UE 定位协议, TS 36.355)
`:lPPa`	1	LPPa PDU (eNB 定位协议, TS 36.455)

APDU IE (ID 1) 携带二进制 LPP 或 LPPa PDU。MME 在 E-SMLC 和目标 eNB 或 UE 之间透明地中继这些。

入站消息 (MME -> E-SMLC)

位置响应 (程序代码 **0,** 成功结果 / 失败结果)

MME 发送位置响应作为位置请求的结果。

IE	ID	严重性	描述
关联-ID	2	拒绝	来自原始位置请求的关联 ID
位置估计	12	拒绝	GAD 编码位置（成功时）
定位数据	16	拒绝	定位方法信息（成功时）
速度估计	21	拒绝	UE 的速度（可选，成功时）
精度满足指示	0	拒绝	□否满足请求的精度（成功时）
LCS 原因	11	忽略	失败原因（失败时）

面向连接的信息（过程代码 **1**，**initiatingMessage**）

MME 将 LPPA/LPP PDU 从 eNB 或 UE 转发回 E-SMLC。

IE	ID	关键性	描述
Correlation-ID	2	拒绝	会话关联
Payload-Type	15	拒绝	`:lPP` 或 `:lPPa`
APDU	1	拒绝	二进制 LPP 或 LPPa PDU

重置请求（过程代码 **4**，**initiatingMessage**）

MME 请求 E-SMLC 重置所有活动过程。E-SMLC 以重置确认响应。

IE	ID	关键性	描述
LCS-Cause	11	忽略	重置原因

出站消息（E-SMLC -> MME）

位置请求（过程代码 **0**，**initiatingMessage**）

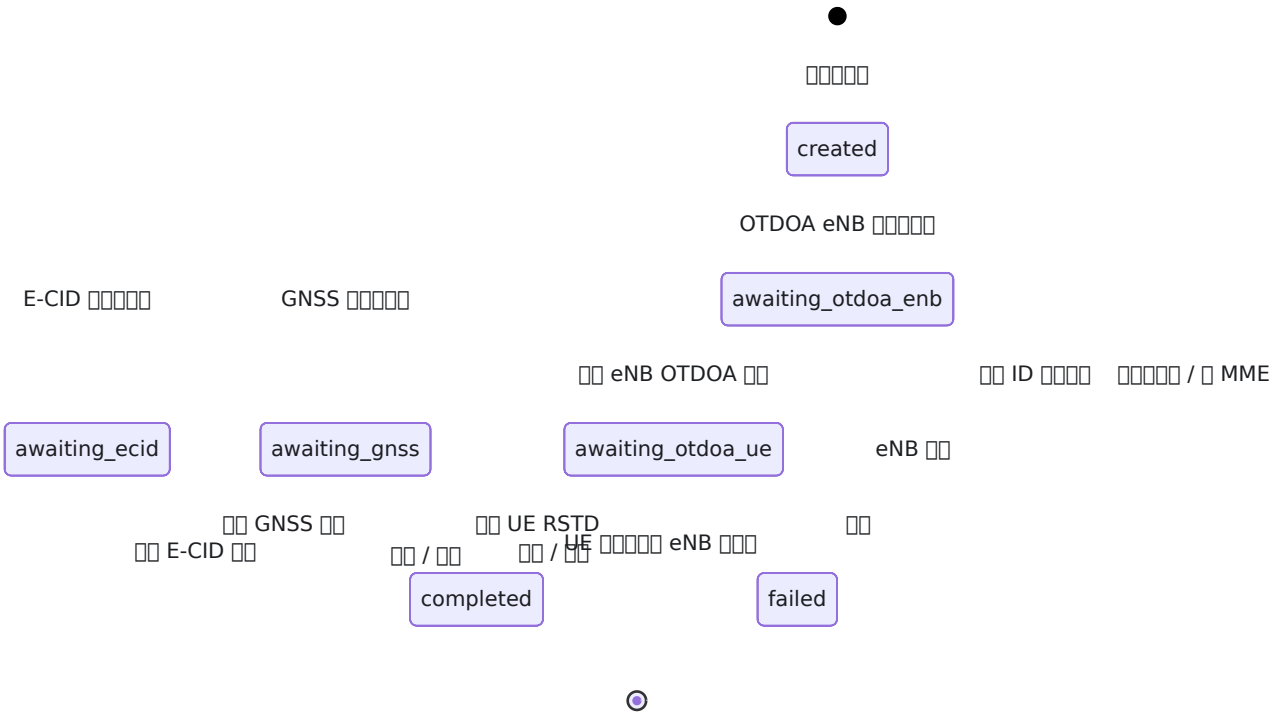
E-SMLC 请求 MME 定位 UE。

IE	ID	关键性	描述
Correlation-ID	2	拒绝	4 字节会话关联 ID
Location-Type	13	拒绝	`geographic-Information`、`assistance-Information` 或 `last-known-location`
E-UTRAN-Cell-Id	4	忽略	UE 服务小区的 E-CGI
LCS-Client-Type	8	拒绝	LCS 客户端类型（可选）
LCS-Priority	9	拒绝	位置请求优先级（可选）
LCS-QoS	10	拒绝	服务质量要求（可选）
IMSI	7	忽略	UE IMSI（可选）
IMEI	6	忽略	UE IMEI（可选）
Include-Velocity	5	拒绝	请求速度估计（可选）

面向连接的信息（过程代码 **1**，**initiatingMessage**）

E-SMLC 通过 MME 向 eNB 或 UE 发送 LPPA/LPP PDU。

IE	ID	关键性	描述
Correlation-ID	2	拒绝	会话关联
Payload-Type	15	拒绝	`:lPP` 或 `:lPPa`
APDU	1	拒绝	二进制 LPP 或 LPPa PDU



无连接信息（过程代码 2，initiatingMessage）

E-SMLC 通过 MME 发送非 UE 关联的 LPPA PDU（例如，eNB 位置请求）。

IE	ID	关键性	描述
Source-Identity	19	拒绝	E-SMLC 网络元素标识
Destination-Id	3	拒绝	目标 eNB 网络元素标识
APDU	1	拒绝	二进制 LPPa PDU

位置会话生命周期

会话已创建
created
OTDOA eNB 请求已发送
E-CID 请求已发送
GNSS 请求已发送
awaiting_otdoa_enb
收到 eNB OTDOA 数据
小区 ID 查找成功
无小区信息 / 无 MME
awaiting_ecid
awaiting_gnss
awaiting_otdoa_ue
eNB 超时
收到 GNSS 修正
收到 E-CID 测量
超时 / 错误
收到 UE RSTD
超时 / □□
UE 超时（返回 eNB 数据）
错误
completed
failed

每个会话跟踪：

字段	描述
`session_id`	唯一标识符（例如，`esmlc-1234567890-1`）
`imsi`	UE 标识符
`mme_host`	发起请求的 MME
`method`	使用的定位方法
`state`	当前会话状态
`created_at`	会话创建时间戳
`updated_at`	最后状态更改时间戳
`completed_at`	完成时间戳
`lppa_transactions`	交换的 LPPA PDU 列表
`lpp_transactions`	交换的 LPP PDU 列表
`measurements`	累积的测量数据
`result`	最终位置结果

超过 1 小时的会话会自动清理。

待处理事务关联

E-SMLC 使用 ETS 表（`:pending_transactions`）将出站 LCS-AP 消息与其响应关联。当定位会话发送面向连接的信息消息时：

1. 引擎生成一个 4 字节的关联 ID
2. ��� `:pending_transactions` 表中注册 `{correlation_id, {caller_pid, ref}}`

3. 通过 SCTP 传输发送 LCS-AP 消息
4. 在 `receive` 块中等待响应
5. 当 SLs 处理程序收到匹配的响应时，它通过关联 ID 查找待处理事务，并将结果发送给等待的进程

小区数据库管理

小区数据库存储每个小区站点的地理位置和无线参数。它用于小区 ID 定位和 OTDOA 计算。

小区记录字段

字段	类型	必需	描述
`cell_id`	任意	□	唯一小区标识符
`latitude`	浮点数	□	小区纬度（十进制度）
`longitude`	浮点数	□	小区经度（十进制度）
`pci`	整数	否	物理小区标识（0-503）
`earfcn`	整数	否	E-UTRA 绝对无线频率信道号
`radius`	整数	否	小区覆盖半径（米）（默认：1000）
`azimuth`	浮点数	否	天线方位角（度）
`height`	浮点数	否	天线高度（米）
`prs_config`	映射	否	OTDOA 的 PRS 配置

PRS 配置字段

字段	类型	描述
`bandwidth`	整数	PRS 带宽（资源块）（6、15、25、50、75、100）
`config_index`	整数	PRS 配置索引（0-4095）
`num_dl_frames`	整数	连续 DL 子帧的数量
`cp_length`	原子	循环前缀长度（`:normal` 或 `:extended`）
`num_antenna_ports`	整数	天线端口数量（1、2 或 4）

InfluxDB 小区同步

小区位置定期从 InfluxDB 同步：

参数	值	描述
同步间隔	5 分钟	自动同步周期
初始延迟	10 秒	启动后第一次同步前的延迟
同步超时	60 秒	同步操作的最大时间

同步也可以通过以下方式手动触发：

- REST API: `POST /api/cells/sync`
- LiveView UI: 小区页面上的“从 InfluxDB 同步”按钮

JSON 导入

小区可以从 JSON 文件导入：

```
[
  {
    "cell_id": "001-01-0001-01",
    "latitude": 40.7128,
    "longitude": -74.0060,
    "pci": 100,
    "earfcn": 1300,
    "radius": 500,
    "prs_config": {
      "bandwidth": 50,
      "config_index": 0,
      "num_dl_frames": 1
    }
  }
]
```

附近小区搜索

小区数据库支持使用哈弗辛公式进行地理邻近查询，以计算大圆距离。通过 REST API 查询：`GET /api/cells/nearby?lat=X&lon=Y&radius=R`，其中半径以公里为单位。

OTDOA 计算

OTDOA 计算器将 RSTD（参考信号时间差）测量转换为 UE 位置，使用多边定位法。

算法

1. **RSTD** 到��离差：`dd = RSTD * Ts * c`，其中 Ts = 1/(15000 * 2048) 秒，c = 光速
2. 坐标投影：小区的纬度/经度坐标投影到基于米的局部坐标系统
3. 迭代最小二乘法：使用加权最小二乘法和雅可比优化求解双曲定位方程
4. 收敛：迭代直到位置变化小于 1 米或最大 50 次迭代
5. 不确定性估计：根据测量几何和距离差计算

要求

- 至少 2 个邻近小区（加上参考小区 = 总共 3 个）以进行位置估计

- 推荐 3 个或更多邻近小区以获得明确的 2D 修正
- 所有小区必须在小区数据库中具有已知位置
- 小区位置通过 PCI、ECGI 或 cell_id 解析

位置的 REST API 参考

请参阅 REST API 参考 以获取完整的端点文档。关键位置端点：

端点	方法	描述
`/api/location`	POST	请求新的 UE 位置
`/api/location`	GET	列出最近的位置修正
`/api/location/:imsi`	GET	某个 IMSI 的最后已知位置
`/api/location/:imsi/history`	GET	某个 IMSI 的位置历史
`/api/location/:imsi/history/csv`	GET	位置历史的 CSV 导出

故障排除

无 SLs 对等体连接

1. 验证 `:sls` 配置：`local_ip` 必须可以从 MME 网络访问
2. 检查每个 `mme_peers` 条目□否具有正确的 IP 地址和端口（默认 9082）
3. 在日志中查找 SCTP 连接错误：`SLs: 无法连接到 MME`
4. 验证 SCTP 未被防火墙阻止（IP 协议 132）
5. 确认 MME 正在监听端口 9082 的 LCS-AP

位置请求返回 "no_mme_host"

E-SMLC 无法确定将 LPPA/LPP 消息发送到哪个 MME。确保：

1. 至少建立一个 SLs SCTP 关联
2. 对于 REST API 请求，提供 `mme_host` 参数
3. 检查仪表板页面上的 SLs 连接状态

MME 没有 LCS-AP 响应

1. 验证 SCTP 关联在连接状态视图中处于 `:established` 状态
2. 检查日志中的 SCTP 心跳失败
3. 确认 MME 支持 LCS-AP (TS 29.171)
4. 检查请求和响应之间的关联 ID 是否正确匹配

小区 ID 定位返回无坐标

小区数据库中没有匹配的小区。操作：

1. 触发 InfluxDB 同步：`POST /api/cells/sync`
2. 通过 REST API 或 LiveView UI 手动添加小区
3. 从 JSON 文件导入小区

OTDOA 超时

eNB 在超时期间未响应 OTDOA 信息。可能原因：

1. eNB 不支持 LPPA OTDOA 信息过程
2. eNB 上未配置 PRS
3. MME 和 eNB 之间的网络路径问题

GNSS 超时

UE 在超时内未报告 GNSS 坐标。可能原因：

1. UE 不支持 GNSS 定位
2. UE 在室内（无卫星可见）
3. 未提供 GNSS 辅助数据，因冷启动需要更长时间 — 启用 GNSS 辅助数据

3GPP 参考

规范	标题
TS 29.171	MME 与 E-SMLC 之间的 LCS 应用协议 (LCS-AP) (SLs 接口)
TS 29.172	GMLC 与 MME 之间的 EPC LCS 协议 (SLg Diameter 接口)
TS 36.455	eNB 与 E-SMLC 之间的 LTE 定位协议 A (LPPa)
TS 36.355	UE 与 E-SMLC 之间的 LTE 定位协议 (LPP)
TS 23.032	通用地理区域描述 (GAD 编码)

OmniLCS REST API 参考

OmniLCS REST API 通过 HTTPS 在端口 **8443** 提供服务。基本 URL 为 `https://<host>:8443/api`。

API 文档（Swagger UI）可在 `https://<host>:8443/api/docs` 获取。OpenAPI 架构位于 `https://<host>:8443/api/schema`。

身份验证

API 当前不强制执行身份验证。访问应在网络级别进行限制。

响应格式

所有响应使用 JSON。成功的响应包含 `"status": "ok"` 字段。错误响应包含 `"status": "error"` 和 `"reason"` 字段。

系统状态

GET /api/status

返回系统健康和操作状态。

响应 **(200)**

```
{
  "status": "ok",
  "version": "1.0.0",
  "name": "OmniLCS",
  "diameter_peers": [
    {
      "host": "dra01.epc.mnc001.mcc001.3gppnetwork.org",
      "realm": "epc.mnc001.mcc001.3gppnetwork.org",
      "state": "Connected",
      "transport": "sctp"
    }
  ],
  "active_sessions": 2,
  "completed_sessions": 47,
  "cells_loaded": 128,
  "cell_sync": {
    "last_sync": "2025-01-15T10:30:00Z",
    "last_result": "ok (128 cells)",
    "sync_count": 42
  },
  "uptime_seconds": 86400
}
```

字段	类型	描述
`version`	string	应用程序版本
`name`	string	实例名称
`diameter_peers`	array	连接的 Diameter 对等体及其状态
`active_sessions`	integer	正在进行的位置会话数量
`completed_sessions`	integer	已完成的位置会话数量
`cells_loaded`	integer	数据库中的小区数量
`cell_sync`	object	InfluxDB 同步状态
`uptime_seconds`	integer	进程运行时间（秒）

位置服务

POST /api/location

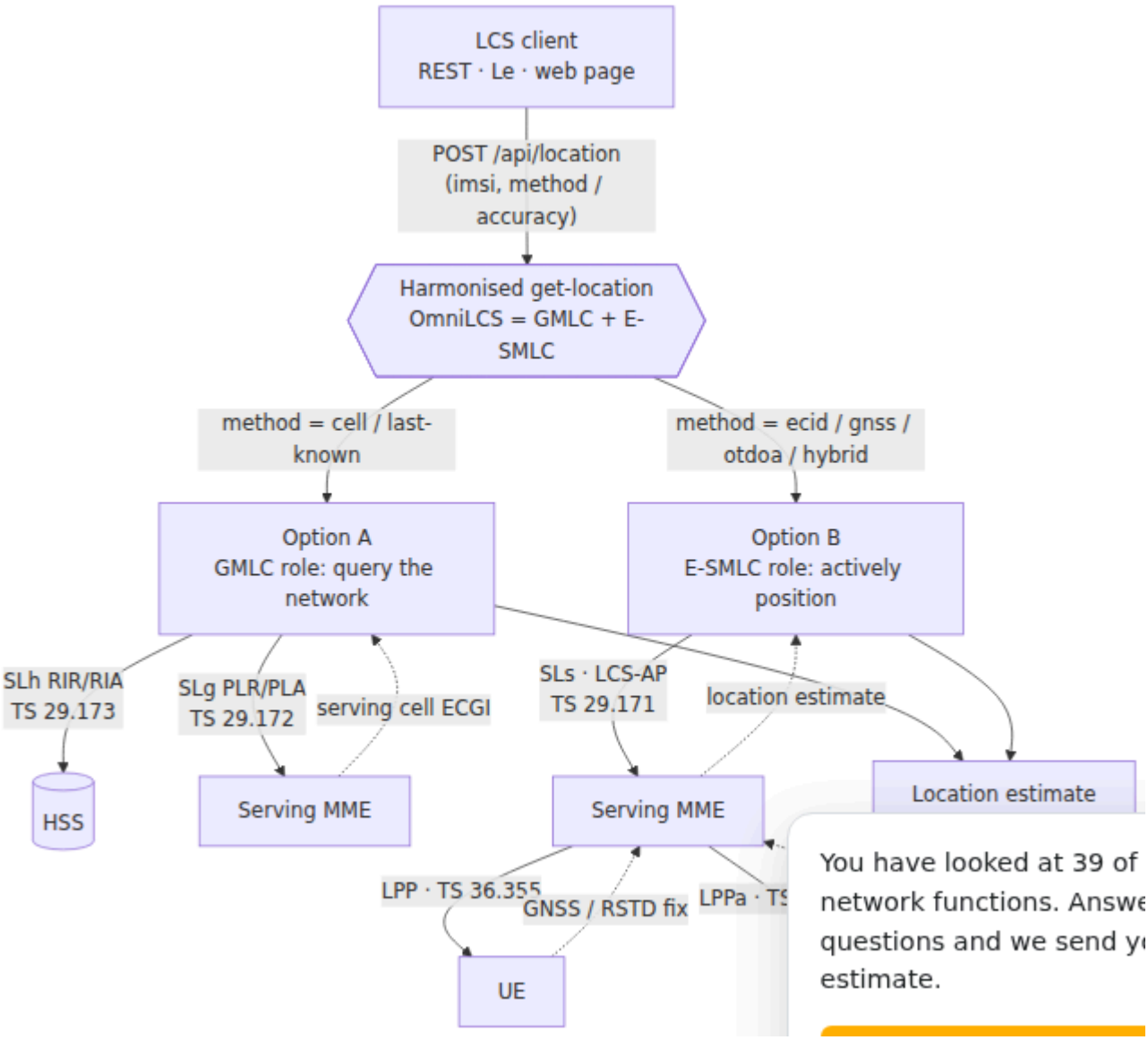
一个单一的、统一的“获取位置”调用。调用者提供 IMSI，并可选择性地提供所需的 `method`/`accuracy`。OmniLCS 然后决定 *□□□□ 3GPP LCS □□□□□*。它在 `source` 字段中报告哪个接口响应。

每个选项在这里可用

OmniLCS 被共同部署为 **GMLC** 和 **E-SMLC**（TS 23.271 的 LTE LCS 网络功能），同时也支持 **MAP/Lg** 用于传统的 2G/3G。因此，这一个端点可以访问每种标准方式来定位订阅者：

选项	**3GPP** 角色	接口	定位方式	最[illegible]适用
A: 查询网络 **(4G)**	GMLC	SLh (TS 29.173) + SLg (TS 29.172)	无主动定位。服务 MME 从其上下文返回 UE 的服务小区（ECGI）。	快速、始终可用的粗略位置；空闲的 UE（最后已知）
B: 主动定位 **(4G)**	E-SMLC	SLs / LCS-AP (TS 29.171)，通过 LPP (TS 36.355) 隧道到 UE，LPPa (TS 36.455) 到 eNB	OmniLCS 对 UE 和无线电运行 E-CID / GNSS / OTDOA	当 UE/无线电支持时，提供精确定位
C: 查询网络 **(2G/3G)**	GMLC	MAP / Lg 通过 SS7 (TS 29.002)：SRI-for-LCS 到 HLR，然后 ProvideSubscriberLocation 到服务 MSC/SGSN	服务 MSC/SGSN 返回服务小区（小区 ID / TA 类）	处于 GERAN/UTRAN 的订阅者

选择：对于 4G，请求的 `method`/`accuracy` 选择 **A**（`cell`/最后已知）或 **B**（`ecid`/`gnss`/`otdoa`/`hybrid`）。将 `rat` 设置为 `2g`/`3g` 始终选择 **C**，无论方法如何。在 A/B 中，服务 MME 始终在路径中（SLg 响应者或 SLs 中继）；在 C 中，服务 MSC/SGSN 在路径中。请参见 2G/3G 通过 MAP/Lg 定位。



请参见 定位方法 了解每种方法（小区 ID、E-CID、OTDOA、A-GNSS、混合）如何获得的，它们的准确性，以及准确性→方法的自动选择。

请求体

```
{
  "imsi": "001010000000001",
  "method": "cell",
  "location_type": 1,
  "timeout": 30000,
  "mme_host": "mme01.epc.mnc001.mcc001.3gppnetwork.org"
}
```

参数	类型	必需	默认	描述
`imsi`	string	□	--	要定位的 UE 的 IMSI
`method`	string	否	`cell`	定位方法: `cell`、`ecid`、`gnss`、`otdoa`、`hybrid`。主动方法（`ecid`/`gnss`/`otdoa`/`hybrid`）通过 SLs 驱动 E-SMLC; `cell` 使用 SLg PLR。
`accuracy`	integer	否	--	请求的水平精度（米）。如果设置，将自动选择方法（更严格 → GNSS/OTDOA/E-CID）；对于小区/最后已知路径，它也会作为 LCS-QoS 传递给 MME。
`location_type`	integer	否	`1`	SLg-Location-Type（小区/最后已知路径）：`0` 当前，`1` 当前或最后已知，`2` 初始。默认 `1` 返回最后已知的服务小区，而不□失败当前定位请求。
`mme_host`	string	否	--	目标主机（服务 MME）。如果省略，将通过 SLh `LCS-Routing-Info-Request` 解析到 HSS。
`rat`	string	否	`4g`	无线接入技术。`2g`/`3g` 通过 MAP/Lg（选项 C）路由到服务 MSC/SGSN; `4g` 使用 SLg/SLs（选项 A/B），由 `method` 选择。
`node_gt`	string	否	--	仅限 2G/3G: 服务 MSC/SGSN 全球标题。如果省略，将通过 MAP SendRoutingInfoForLCS 解析到 HLR。
`timeout`	integer	否	`30000`	超时时间（毫秒）

> 请求的 `method`（或 `accuracy`）选择 *OmniLCS □□□□ UE*: 主动方法运行 OmniLCS E-SMLC（LPP/LPPa 通过 SLs），而 `cell`/最后已知查询服务 MME 通过 SLg。在 SLg 路径中，MME 最终决定它可以返回什么（通常□服务小区 ECGI）。

响应 **(200)**

```
{
  "status": "ok",
  "imsi": "001010000000001",
  "method": "cell",
  "source": "slg",
  "latitude": 40.7128,
  "longitude": -74.0060,
  "altitude": null,
  "uncertainty": 10.5,
  "confidence": null,
  "cell_id": 43559,
  "plmn": {"mcc": "001", "mnc": "01"},
  "ecgi": "00F1100000AA27",
  "serving_node": {"name": "mme01...", "realm": "epc..."},
  "duration_ms": 5230,
  "timestamp": "2025-01-15T10:30:00Z"
}
```


响应 合并了定位器返回的每个字段 与请求信封，因此确切的字段集因路径而异。`latitude`/`longitude` 从 GAD `Location-Estimate` 解码；`source` 对于来自服务 MME 通过 SLg 的结果为 `"slg"`，或对于主动 E-SMLC 定位方法（`gnss`、`ecid` 等）。小区/SLg 定位还包括解码的 ECGI（`cell_id` + `plmn`）以及原始 `ecgi`。二进制 AVP（原始 ECGI、GAD 估计、服务节点）为十六进制编码。

字段	类型	描述
`imsi`	string	UE 标识符
`method`	string	请求的方法
`source`	string	实际使用的定位源（`slg`、`gnss`、`ecid` 等）
`latitude`	float/null	纬度（十进制度）
`longitude`	float/null	经度（十进制度）
`altitude`	float/null	海拔（米，若可用）
`uncertainty`	float/null	水平位置不确定性（米）
`uncertainty_altitude`	float/null	垂直不确定性（米，若存在）
`confidence`	integer/null	置信百分比
`shape`	string/null	估计的 GAD 形状（例如 `ellipsoid_point_with_uncertai`
`cell_id`	integer/null	解码的 E-UTRAN 小区 ID（小区/SLg 路径）
`plmn`	object/null	解码的服务 PLMN `{mcc, mnc}`（小区/SLg
`ecgi`	string/null	原始 7 字节 ECGI，十六进制（小区/SLg 路径）
`serving_node`	object/null	服务 MME 身份（名称/领域）
`age_of_location_estimate`	integer/null	估计的年龄（分钟，若存在）
`accuracy_fulfilment`	string/null	请求的准确性□否满足
`ecid_measurements`	object/null	E-CID 无线电测量（`ecid` 方法）

字段	类型	描述
`duration_ms`	integer	耗时（毫秒）
`timestamp`	string	ISO 8601 时间戳

错误响应

状态	原因	描述
400	`"imsi is required"`	缺少 IMSI 参数
404	`"User not found"`	IMSI 未知
404	`"User not connected"`	UE 未连接到网络
422	`"No MME host available for this subscriber"`	没有 MME 路由请求
504	`"Positioning timed out"`	定位方法未及时完成
500	(变化)	内部错误

示例

□□ A□□□ GMLC □□□SLh + SLg□□□□□□□□ 请求 `cell`（或任何最后已知请求）解析服务 MME 并返回其服务小区：

```
curl -sk -X POST https://localhost:8443/api/location \
  -H "Content-Type: application/json" \
  -d '{"imsi": "001010000000001", "method": "cell"}'
```

```
{
  "status": "ok",
  "imsi": "001010000000001",
  "method": "cell",
  "source": "slg",
  "latitude": -37.8183,
  "longitude": 144.9671,
  "altitude": null,
  "uncertainty": 1273.5,
  "confidence": null,
  "cell_id": 43559,
  "plmn": {"mcc": "001", "mnc": "01"},
  "ecgi": "00F1100000AA27",
  "serving_node": {"name":
"mme01.epc.mnc001.mcc001.3gppnetwork.org", "realm":
"epc.mnc001.mcc001.3gppnetwork.org"},
  "duration_ms": 412,
  "timestamp": "2026-06-22T01:14:55.218000Z"
}
```


□□ B□□□ E-SMLC □□□SLs / LPP□□□□□□□□ 请求 `gnss`（或紧密的 `accuracy` 值）驱动主动修正。`source` 字段反映了响应的方法：

```
curl -sk -X POST https://localhost:8443/api/location \
  -H "Content-Type: application/json" \
  -d '{"imsi": "001010000000001", "method": "gnss", "timeout":
30000}'
```

```
{
  "status": "ok",
  "imsi": "001010000000001",
  "method": "gnss",
  "source": "gnss",
  "latitude": -37.8200,
  "longitude": 144.9834,
  "altitude": 558,
  "uncertainty": 9.5,
  "uncertainty_altitude": 1.1,
  "confidence": 90,
  "shape":
"ellipsoid_point_with_altitude_and_uncertainty_ellipsoid",
  "duration_ms": 10077,
  "timestamp": "2026-06-22T01:18:02.640000Z"
}
```


同样，`-d '{"imsi": "001010000000001", "accuracy": 20}'` 自动选择 A-GNSS（≤ 50 m → A-GNSS）并返回相同的响应格式。

GET /api/location

列出最近完成的位置修正。

查询参数

参数	类型	默认	描述
`limit`	integer	`50`	最大结果数量

响应 **(200)**

```
{
  "status": "ok",
  "data": [
    {
      "imsi": "001010000000001",
      "method": "gnss",
      "state": "completed",
      "latitude": 40.7128,
      "longitude": -74.0060,
      "uncertainty": 10.5,
      "source": "gnss",
      "created_at": "2025-01-15T10:29:55Z",
      "completed_at": "2025-01-15T10:30:00Z"
    }
  ],
  "count": 1
}
```

GET /api/location/:imsi

通过 IMSI 获取 UE 的最后已知位置。

路径参数

参数	类型	描述
`imsi`	string	UE 的 IMSI

响应 **(200)**

```
{
  "status": "ok",
  "imsi": "001010000000001",
  "latitude": 40.7128,
  "longitude": -74.0060,
  "altitude": null,
  "uncertainty": 10.5,
  "confidence": null,
  "source": "gnss",
  "timestamp": "2025-01-15T10:30:00Z"
}
```

错误响应

状态	原因
404	`"No location found for IMSI"`
404	`"No completed location for IMSI"`

GET /api/location/:imsi/history

获取 IMSI 的位置修正历史。

路径参数

参数	类型	描述
`imsi`	string	UE 的 IMSI

查询参数

参数	类型	默认	描述
`from`	string	--	时间范围的开始（ISO 8601 或日期）
`to`	string	--	时间范围的结束（ISO 8601 或日期）
`limit`	integer	`100`	最大结果数量

响应 **(200)**

```
{
  "status": "ok",
  "data": [
    {
      "timestamp": "2025-01-15T10:30:00Z",
      "imsi": "001010000000001",
      "method": "gnss",
      "latitude": 40.7128,
      "longitude": -74.0060,
      "altitude": null,
      "uncertainty": 10.5,
      "confidence": null,
      "source": "gnss",
      "duration_ms": 5230
    }
  ],
  "count": 1
}
```

GET /api/location/:imsi/history/csv

将 IMSI 的位置历史导出为 CSV 文件。

路径参数

参数	类型	描述
`imsi`	string	UE 的 IMSI

查询参数

与 `/api/location/:imsi/history` 相同。

响应 **(200)**

返回带有标题的 CSV 文件下载：

```
timestamp,imsi,method,latitude,longitude,altitude,uncertainty,confide
```

内容类型：`text/csv` 内容处置：`attachment; filename="location_history_<imsi>_<date>.csv"`

小区数据库

GET /api/cells

列出数据库中的所有小区。

响应 **(200)**

```
{
  "status": "ok",
  "data": [
    {
      "cell_id": "001-01-0001-01",
      "latitude": 40.7128,
      "longitude": -74.0060,
      "pci": 100,
      "earfcn": 1300,
      "radius": 500,
      "azimuth": null,
      "height": null,
      "prs_config": null,
      "updated_at": "2025-01-15T10:00:00Z"
    }
  ],
  "count": 1
}
```

GET /api/cells/:id

通过 cell_id 获取单个小区。

路径参数

参数	类型	描述
id	string	小区标识符

响应 **(200)**

```
{
  "status": "ok",
  "data": {
    "cell_id": "001-01-0001-01",
    "latitude": 40.7128,
    "longitude": -74.0060,
    "pci": 100,
    "earfcn": 1300,
    "radius": 500,
    "azimuth": null,
    "height": null,
    "prs_config": {
      "bandwidth": 50,
      "config_index": 0,
      "num_dl_frames": 1,
      "cp_length": null,
      "num_antenna_ports": null
    },
    "updated_at": "2025-01-15T10:00:00Z"
  }
}
```


错误响应

状态	原因
404	`"Cell not found: <id>"`

POST /api/cells

创建一个新小区。

请求体

```
{
  "cell_id": "001-01-0001-01",
  "latitude": 40.7128,
  "longitude": -74.0060,
  "pci": 100,
  "earfcn": 1300,
  "radius": 500,
  "azimuth": 120.0,
  "height": 30.0,
  "prs_config": {
    "bandwidth": 50,
    "config_index": 0,
    "num_dl_frames": 1,
    "cp_length": "normal",
    "num_antenna_ports": 2
  }
}
```

参数	类型	必需	默认	描述
`cell_id`	string	□	--	唯一小区标识符
`latitude`	float	□	--	小区纬度（-90 到 90）
`longitude`	float	□	--	小区经度（-180 到 180）
`pci`	integer	否	--	物理小区标识（0-503）
`earfcn`	integer	否	--	E-UTRA 绝对无线电频率信道号
`radius`	integer	否	`1000`	覆盖半径（米）
`azimuth`	float	否	--	天线方位角（度）
`height`	float	否	--	天线高度（米）
`prs_config`	object	否	--	OTDOA 的 PRS 配置
`tac`	integer	否	--	跟踪区域代码（用于 4G CAP 警报广播目标）
`lac`	integer	否	--	位置区域代码（用于 2G 和 3G CAP 警报广播目标）
`rat`	string	否	--	无线接入技术：`"4g"`、`"3g"` 或 `"2g"`

响应 **(201)**

以与 `GET /api/cells/:id` 相同的格式返回创建的小区。

错误响应

状态	原因
400	`"cell_id is required"`
400	`"latitude and longitude are required"`

PUT /api/cells/:id

更新现有小区。仅更新提供的字段。

路径参数

参数	类型	描述
`id`	string	小区标识符

请求体

任何要更新的小区字段（与 POST 相同，但所有字段都可选的）。

响应 (200)

返回更新后的小区。

错误响应

状态	原因
404	`"Cell not found: <id>"`

DELETE /api/cells/:id

从数据库中删除小区。

路径参数

参数	类型	描述
`id`	string	小区标识符

响应 (204)

成功时无内容。

错误响应

状态	原因
404	`"Cell not found: <id>"`

GET /api/cells/nearby

查找地理位置附近的小区。

查询参数

参数	类型	必需	默认	描述
`lat`	float	□	--	搜索中心的纬度
`lon`	float	□	--	搜索中心的经度
`radius`	float	否	`10`	搜索半径（公里）

响应 **(200)**

```
{
  "status": "ok",
  "data": [
    {
      "cell_id": "001-01-0001-01",
      "latitude": 40.7128,
      "longitude": -74.0060,
      "pci": 100,
      "earfcn": 1300,
      "distance_km": 0.523
    }
  ],
  "count": 1
}
```

结果按距离排序（最近的在前）。每个条目包括 `distance_km` 字段。

错误响应

状态	原因
400	`"lat and lon query parameters are required"`

POST /api/cells/sync

触发立即的 InfluxDB 小区同步。

请求体

不需要。

响应 **(200)**

```
{
  "status": "ok",
  "cells_synced": 128
}
```

错误响应

状态	原因
500	`"Sync failed: <reason>"`
503	`"Cell sync service unavailable"`

延迟定位 (GMLC)

管理周期性和地理围栏触发的位置会话。有关会话类型和 Diameter 集成的完整详细信息，请参见 GMLC & Le 接口指南。

GET /api/deferred_location

列出所有活动的延迟定位会话。

响应 **(200):**

```
{
  "status": "ok",
  "count": 1,
  "data": [
    {
      "session_id": "a1b2c3d4-e5f6-...",
      "type": "periodic",
      "imsi": "001010000000001",
      "method": "cell",
      "client_name": "rest-api",
      "status": "active",
      "interval_ms": 60000,
      "remaining_reports": 7,
      "total_reports": 10,
      "started_at": "2026-04-09T10:00:00Z",
      "last_fix_at": "2026-04-09T10:03:00Z"
    }
  ]
}
```

POST /api/deferred_location

创建一个新的延迟定位会话。

周期性会话请求体：

```
{
  "type": "periodic",
  "imsi": "001010000000001",
  "method": "cell",
  "interval_seconds": 60,
  "count": 10
}
```

字段	类型	必需	描述
`type`	string	□	`"periodic"`
`imsi`	string	□	订阅者 IMSI
`method`	string	否	定位方法：`cell`、`ecid`、`gnss`、`otdoa`。默认：`cell`
`interval_seconds`	integer	□	修正之间的秒数
`count`	integer	□	要执行的修正总数

触发会话请求体：

```
{
  "type": "triggered",
  "imsi": "001010000000001",
  "method": "cell",
  "event_type": "entering",
  "poll_interval_seconds": 30,
  "max_reports": 0,
  "areas": [
    {
      "type": "circle",
      "center": {"lat": -33.8688, "lon": 151.2093},
      "radius_meters": 500
    }
  ]
}
```

字段	类型	必需	描述
`type`	string	□	`"triggered"`
`imsi`	string	□	订阅者 IMSI
`method`	string	否	定位方法。默认：`cell`
`event_type`	string	□	`"entering"`、`"leaving"` 或 `"being_inside"`
`poll_interval_seconds`	integer	否	位置轮询之间的秒数。默认：30
`max_reports`	integer	否	最大触发报告。`0` = 无限
`areas`	array	□	区域定义的列表（圆形或多边形）

响应 **(201):**

```
{"status": "ok", "message": "Periodic session created"}
```

GET /api/deferred_location/:session_id

获取延迟会话的状态。

响应 **(200):**

```
{
  "status": "ok",
  "data": {
    "session_id": "a1b2c3d4-...",
    "type": "periodic",
    "imsi": "001010000000001",
    "status": "active",
    "remaining_reports": 7,
    "total_reports": 10
  }
}
```

DELETE /api/deferred_location/:session_id

取消活动的延迟会话。

响应 **(200):**

```
{"status": "ok", "message": "Session cancelled"}
```

状态	错误
400	缺少或无效的参数
404	会话未找到

CAP 警报

POST /api/cap

提交 CAP XML 警报以进行处理。警报被解析，多边形警告区域通过小区数据库解析为 TACs/LACs，警报根据配置排队等待批准或自动广播。

请求体

```
{
  "xml": "<alert xmlns=\"urn:oasis:names:tc:emergency:cap:1.2\">...</alert>"
}
```

参数	类型	必需	描述
`xml`	string	□	完整的 CAP v1.2 XML 警报文档

响应 **(201)**

```
{
  "status": "ok",
  "data": {
    "id": "a1b2c3d4-e5f6-...",
    "status": "pending",
    "source": "http_post",
    "received_at": "2025-01-15T10:30:00Z",
    "matched_cells": 42,
    "tacs": [100, 101],
    "lacs": [5001],
    "mcc": "001",
    "mnc": "01",
    "broadcast_params": {
      "message_id": 4370,
      "repetition_period": 30,
      "num_broadcasts": 10,
      "message_text": "Tornado Warning...",
      "event": "Tornado Warning",
      "severity": "Extreme",
      "urgency": "Immediate"
    }
  }
}
```

当 `require_approval` 为 `true` 时，`status` 字段为 `"pending"`，或在自动批准时为 `"sent"`。

错误响应

状态	原因
400	`"xml field is required"`
422	解析错误详细信息

GET /api/cap

列出所有状态的警报（待处理、活动、历史）。

响应 **(200)**

```
{
  "status": "ok",
  "data": {
    "pending": [...],
    "active": [...],
    "history": [...]
  }
}
```

GET /api/cap/:id

通过 ID 获取单个警报。

路径参数

参数	类型	描述
`id`	string	警报 UUID

响应 **(200)**

返回警报对象。

错误响应

状态	原因
404	`"Alert not found: <id>"`

PUT /api/cap/:id

批准或拒绝待处理的警报。

请求体

```
{
  "action": "approve",
  "operator": "operator1"
}
```

参数	类型	必需	描述
`action`	string	□	`"approve"` 或 `"reject"`
`operator`	string	否	审计轨迹的操作员名称（默认为 `"unknown"`）

响应 **(200)**

返回更新后的警报对象。

错误响应

状态	原因
400	`"action must be 'approve' or 'reject'"`
404	`"Alert not found: <id>"`

错误响应格式

所有错误响应遵循以下结构：

```
{
  "status": "error",
  "reason": "人类可读的错误描述"
}
```

常见的 HTTP 状态代码:

代码	意义
200	成功
201	创建（小区创建）
204	无内容（小区删除）
400	错误请求（缺少或无效的参数）
404	未找到（未知资源）
422	无法处理的实体（有效请求但无法满足）
500	内部服务器错误
503	服务不可用
504	网关超时（定位超时）

订阅者跟踪与虚拟驾驶测试

OmniLCS 提供两种定期位置监控功能：订阅者跟踪用于单个用户设备的持续监控，虚拟驾驶测试用于结合 GPS 定位和信号质量测量的多用户设备活动。

架构

订阅者跟踪

订阅者跟踪定期使用选定的定位方法定位单个用户设备。结果保存在 Mnesia 中，并通过 Web UI 和 REST API 提供。

Web UI

在侧边栏中导航到 订阅者跟踪。输入 IMSI、轮询间隔（最小 5 秒），并选择定位方法（小区、E-CID、GNSS、OTDOA）。点击 开始观察。

活动观察以表格形式显示，包含 IMSI、方法、间隔、定位次数、设备状态（在线/离线/错误）、最后定位时间和最后结果。点击一行查看定位历史。使用 导出 **CSV** 或 导出 **KML** 按钮下载数据。

REST API

方法	路径	描述
`GET`	`/api/tracking`	列出活动跟踪会话
`POST`	`/api/tracking`	开始跟踪。主体：`{"imsi": "...", "method": "gnss", "interval": 30}`
`GET`	`/api/tracking/:imsi`	获取某个 IMSI 的跟踪历史
`DELETE`	`/api/tracking/:imsi`	停止跟踪某个 IMSI
`GET`	`/api/tracking/:imsi/export/csv`	下载 CSV 导出
`GET`	`/api/tracking/:imsi/export/kml`	下载 KML 导出

设备状态

跟踪会话处理设备在线/离线转换：

状态	条件	行为
在线	`Engine.locate` 成功	定位与坐标存储
离线	没有连接的 MME 或用户设备不可达	记录错误，继续轮询会话
错误	由于其他原因定位失败	记录错误，继续轮询会话

会话在出现错误时不会停止。它会继续以配置的间隔进行轮询，直到明确取消。

虚拟驾驶测试

虚拟驾驶测试运行多 IMSI 活动，将 **GNSS** 定位（用于 GPS 坐标）与 **E-CID** 测量（用于 RSRP、RSRQ、服务小区、PCI、EARFCN）结合。这□信号质量测量的物理驾驶测试等效。

工作原理

在每个时间间隔，活动：

1. 对于活动中的每个 IMSI（最多 4 个并发）：
    - 通过 MME 向用户设备发送 LPP **GNSS** 请求以获取 GPS 坐标
    - 向用户设备发送 LPP **E-CID** 请求以获取信号测量
    - 将结果合并为单个测量记录
2. 将每个测量存储在 Mnesia 中
3. 记录到 InfluxDB（如果已配置）
4. 通过 PubSub 将结果广播到 Web UI

驾驶测试会话
E-SMLC 引擎
MME
UE
[每个间隔]
locate(imsi, method: :gnss)
Location-Request (IMSI)
Location-Response (correlation)
LPP RequestLocationInformation (GNSS)
NAS: LPP GNSS Request
NAS: LPP ProvideLocationInformation
LCS-AP Connection-Oriented-Info (LPP)
{:ok, %{latitude, longitude, altitude, ...}}
locate(imsi, method: :ecid)
Location-Request (IMSI)
Location-Response (correlation)
LPP RequestLocationInformation (ECID)
NAS: LPP ECID Request
NAS: LPP ECID ProvideLocationInformation
LCS-AP Connection-Oriented-Info (LPP)
{:ok, %{ecid_measurements: %{rsrp, rsrq, pci, ...}}}
合并 GNSS + ECID → 测量记录
保存到 Mnesia, 记录到 InfluxDB
驾驶测试会话
E-SMLC 引擎
MME
UE

Web UI

在侧边栏中导航到 虚拟驾驶测试。

1. 输入 活动名称（例如，“CBD覆盖测试”）
2. 输入 **IMSIs**（每行一个或用逗号分隔）
3. 设置 间隔（最小 10 秒）
4. 点击 开始活动

活动表显示名称、用户设备数量、间隔、轮询次数和最后轮询时间。点击活动行查看测量。测量表显示每次定位数据：

列	描述
时间	定位时间戳
IMSI	目标用户设备
纬度/经度	来自 GNSS 的 GPS 坐标
PCI	来自 E-CID 的物理小区 ID
RSRP	参考信号接收功率（用户设备报告）
RSRQ	参考信号接收质量（用户设备报告）
GNSS	GNSS 状态（正常/错误）
ECID	E-CID 状态（正常/错误）

RSRP 值以颜色编码：绿色（强），黄色（中），红色（弱）。

REST API

方法	路径	描述
`GET`	`/api/drive_test`	列出活动活动
`POST`	`/api/drive_test`	开始活动。主体：`{"name": "...", "imsis": ["..."], "interval": 30}`
`GET`	`/api/drive_test/:id`	获取活动测量。查询：`?limit=200&imsi=filter`
`DELETE`	`/api/drive_test/:id`	停止活动
`GET`	`/api/drive_test/:id/export/csv`	下载包含所有测量的 CSV
`GET`	`/api/drive_test/:id/export/kml`	下载带有信号质量颜色编码的 KML

示例 API 使用

开始活动：

```
curl -sk -X POST https://omnilcs:8445/api/drive_test \
  -H "Content-Type: application/json" \
  -d '{
    "name": "CBD Coverage Test",
    "imsis": ["001010000000001", "001010000000002"],
    "interval": 30
  }'
```

获取测量：

```
curl -sk https://omnilcs:8445/api/drive_test/<campaign_id>?
limit=100
```

导出 **KML**:

```
curl -sk
https://omnilcs:8445/api/drive_test/<campaign_id>/export/kml -o
coverage.kml
```

E-CID 测量

E-CID 方法通过 LPP（3GPP TS 36.355）直接从用户设备请求 RSRP、RSRQ 和 UE Rx/Tx 时间差。此方法无论 eNB LPPa 支持与否均有效，因为用户设备在本地执行测量。

测量字段

字段	描述	来源
`rsrp`	参考信号接收功率	用户设备测量（映射值，约 -44 到 -140 dBm）
`rsrq`	参考信号接收质量	用户设备测量（映射值，约 -3 到 -19.5 dB）
`pci`	物理小区 ID	服务小区标识符
`earfcn`	E-UTRAN 绝对无线电频率信道号	服务小区频率
`cell_global_id`	小区全球标识（PLMN + 小区 ID）	服务小区
`ue_rx_tx_time_diff`	用户设备 Rx-Tx 时间差	传播延迟估计

API 响应示例


```
{
  "status": "ok",
  "method": "ecid",
  "imsi": "001010000000001",
  "ecid_measurements": {
    "measurements": [
      {
        "pci": 373,
        "earfcn": 1825,
        "cell_global_id": {
          "cell_id": 4000,
          "plmn": {"mcc": "001", "mnc": "01"}
        },
        "rsrp": 40,
        "rsrq": 25,
        "ue_rx_tx_time_diff": 19
      }
    ]
  }
}
```


导出格式

CSV

以逗号分隔的值，带有标题行。包括所有测量列：IMSI、时间戳、坐标、信号测量和状态字段。适合导入电子表格工具或分析脚本。

KML

与 Google Earth / Google Maps 兼容。每个定位成为一个标记，包含：

- 坐标（纬度/经度/高度）
- 时间戳
- 包含 IMSI、RSRP、RSRQ、PCI 的描述
- 按 RSRP 信号强度（绿色/黄色/红色）颜色编码的图标

- 每个 IMSI 的轨迹 LineString 连接所有定位

在 Google Earth、Google Maps、QGIS 或任何 KML 兼容的 GIS 工具中打开。

InfluxDB 集成

当在 `runtime.exs` 中配置 InfluxDB 连接时，订阅者跟踪和虚拟驾驶测试都会将测量记录到 InfluxDB。

写入的测量

InfluxDB 测量	来源	字段
`subscriber_tracking`	跟踪会话	latitude, longitude, altitude, device_status
`drive_test`	驾驶测试活动	latitude, longitude, altitude, rsrp, rsrq, serving_pci, uncertainty

标签

标签	描述
`imsi`	目标用户设备 IMSI
`method`	使用的定位方法
`campaign_id`	驾驶测试活动标识符（仅限驾驶测试）

持久性

所有跟踪和驾驶测试数据存储在 Mnesia 的 disc_copies 表中，在应用程序重启后仍然有效。

Mnesia 表	键	内容
`:mnesia_tracking_history`	`{imsi, monotonic_time}`	单个跟踪定位
`:mnesia_tracking_config`	`imsi`	活动跟踪会话配置
`:mnesia_drive_test_measurements`	`{campaign_id, imsi, monotonic_time}`	驾驶测试测量
`:mnesia_drive_test_config`	`campaign_id`	活动配置

历史记录会自动修剪为每个 IMSI 1,000 条（跟踪）或每个活动 10,000 条（驾驶测试）。

OmniLCS Web 界面指南

OmniLCS 控制面板——一个通过 HTTPS 提供的实时 Web 界面，使用 **443** 端口。它基于 Phoenix LiveView 构建，提供无需页面重新加载的实时更新视图。所有页面每 2-3 秒自动刷新。

页面概述

导航栏按以下顺序呈现页面：

页面	路径	描述
仪表板	`/dashboard`	系统概述及关键指标
位置	`/location`	位置服务测试界面
SLs 接口	`/sls`	SLs (LCS-AP) 接口监控
小区数据库	`/cells`	小区数据库管理和导入
小区地图	`/map`	交互式小区地图，支持广播区域选择
Diameter	`/diameter`	Diameter 对等体监控
GMLC / Le	`/gmlc`	GMLC Le 接口监控，延迟会话管理
发送广播	`/send_broadcast`	跨 2G/3G/4G 的统一广播组合
活动广播	`/broadcasts`	监控和停止活动广播
CBC 2G	`/cbc`	2G CBSP 监控
CBC 3G	`/cbc3g`	3G SABP 监控
CBC 4G	`/cbc4g`	4G SBC-AP 广播组合和监控
CAP 警报	`/cap`	CAP 警报摄取、审批和广播监控
应用程序	`/application`	OTP 应用程序资源查看器（内置）
配置	`/configuration`	运行时配置查看器（内置）
日志	`/log`	实时日志查看器（内置）

仪表板

路径：`/dashboard`
刷新：每 2 秒

仪表板提供整个 OmniLCS 系统的高层次操作概述。

统计卡片

顶部有四个摘要卡片：

卡片	描述
活动会话	正在进行的位置会话数量
总请求	自启动以来处理的位置请求总数
成功率	完成请求与总请求的百分比
平均响应时间	平均定位持续时间（毫秒）

最近位置请求

显示 20 个最近位置会话的表格，包含以下列：

列	描述
IMSI	UE 标识符
方法	定位方法（小区、E-CID、GNSS、OTDOA、混合）
MME	发起请求的 MME 主机
时间	请求时间戳（HH:MM:SS）
状态	颜色编码徽章（活动、完成、错误、超时）

连接的对等体

显示处于“已连接”状态的 SLs (LCS-AP) 和 Diameter 对等体的列表，包括：

- 对等体主机名
- 接口类型（SLs 或 Diameter）

系统状态

子系统健康的指示灯：

指示器	绿色	黄色	红色
SLs 接口 (LCS-AP)	至少建立 1 个 MME SCTP 关联	--	无 MME 连接
Diameter 服务	至少 1 个对等体连接	--	无对等体连接
小区数据库	小区已加载	0 个小区	--

位置

路径：`/location`
刷新：每 2 秒

交互式位置服务测试界面，用于手动请求 UE 位置。

方法选择器

切换按钮选择定位方法：

方法	描述
小区	基于小区 ID，速度最快，精度最低
E-CID	使用 eNB 测量的增强小区 ID，通过 LPPa
GNSS	通过 LPP 的 GPS/GNSS，精度最高
OTDOA	基于 PRS 测量的多边定位
混合	结合方法并提供后备

每种方法显示描述面板，解释其工作原理。

请求表单

- **IMSI** 输入：输入要定位的 UE 的 IMSI（例如，`999990000000001`）
- 请求位置按钮：异步发送位置请求

请求作为后台任务运行。请求进行时会显示旋转指示器。

会话计数

摘要徽章显示：

- 总会话
- 活动会话
- 完成会话
- 错误计数
- 收到的��果

会话历史表

按创建时间（最新优先）排序的所有会话：

列	描述
IMSI	UE 标识符
方法	定位方法
MME	目标 MME 主机
时间	会话创建时间
持续时间	完成时间（毫秒或秒）
状态	活动、完成、错误、超时

完成的会话展开以显示：

- 坐标：纬度和经度
- 不确定性：位置不确定性（米）
- 置信度：置信度百分比
- 来源：实际使用的定位来源
- 地图：嵌入的 OpenStreetMap 显示位置

错误会话展开以显示错误原因。

通知

出现吐司通知：

- 请求已发起（信息）
- 位置已获取（成功，带坐标）
- 位置获取失败（错误，带原因）

通知在 5 秒后自动消失。

SLs 接口

路径：`/sls`
刷新：每 2 秒

监控 E-SMLC 和连接的 MMEs 之间的 SLs (LCS-AP) 接口。显示 SCTP 关联状态和连接的 MME 对等体。

接口状态

显示 LCS 应用协议 (LCS-AP) 接口状态:

- 协议参考 (3GPP TS 29.171)
- 监听配置 (PPID 29, 端口 9082)
- 连接的 MME 对等体数量
- 绿色/红色指示器

统计

卡片	描述
连接的 MMEs	活动 SCTP 关联到 MMEs 的数量
总关联	SCTP 关联总数

MME 对等体面板

显示连接的 MME SCTP 关联的列表，显示对等体主机名和连接状态。

小区数据库

路径：`/cells`
刷新：每 2 秒

管理用于小区 ID 定位和 OTDOA 计算的小区位置数据库。所有小区数据持久化到 Mnesia，并在应用程序重启后存活。

头部控件

控件	描述
小区数量	数据库中小区的总数
最后同步	上次 InfluxDB 同步的时间戳和结果
从 InfluxDB 同步	触发立即的 InfluxDB 小区同步
地图视图 / 表格视图	切换表格和地图显示
导入小区	打开特定供应商的导入模态
添加小区	打开手动小区输入表单

搜索

按小区 ID、PCI、EARFCN、小区名称或来源搜索小区。

表格视图

显示的列：

列	描述
小区 ID	唯一的小区标识符
PCI	物理小区标识 (LTE/NR) 或主扰频码 (UMTS)
EARFCN	无线频率信道号
纬度	小区纬度（十进制度）
经度	小区经度（十进制度）
RAT	无线接入技术：GSM、UMTS、LTE 或 NR
来源	数据来源徽章：**Huawei**（琥珀色，XLSX 导入）、**InfluxDB**（绿���，同步）或 **Manual**（蓝色）。徽章下方显示导入时间戳。
操作	编辑和删除按钮

导入小区

点击 导入小区 打开一个具有两步工作流的模态：

步骤 **1:** 供应商选择

选择要导入的小区数据的 NMS 供应商：

供应商	导入格式
Huawei	U2020 XLSX 导出或 CSV
Nokia	NetAct RAN 导出 CSV
Ericsson	ENM WCDMA 导出 CSV
ZTE	UMS 小区导出 CSV
其他 **/ JSON**	JSON 数组粘贴

步骤 **2:** 上传

每个供应商提供相应的上传界面。

Huawei XLSX（推荐）：拖放或选择一个或多个 Huawei U2020 小区报告 `.xlsx` 文件。支持多文件上传。例如，可以同时上传 UMTS 和 LTE 报告。文件在上传完成后自动导入。

RAT 类型（GSM、UMTS、LTE 或 NR）会根据每个文件的以下内容自动检测：

- 工作簿中的表名（例如，名为 "UMTS"、"LTE"、"GSM"、"NR" 的表）
- 规划数据表中的列标题（例如，`PScrambCode` 对于 UMTS，`DlEarfcn` 对于 LTE，`NRPCI` 对于 NR，`BCCH` 对于 GSM）

解析器读取 **Sheet1** 标签（带坐标的规划数据）并提取：

RAT	导入的关键字段
GSM	小区 ID、小区名称、LAC、BCCH���BSIC、经度、纬度、天线高度、方位角
UMTS	小区 ID、小区名称、LAC、SAC、RAC、UARFCN 下行、扩频码、经度、纬度、天线高度、方位角
LTE	小区 ID、小区名称、PCI、TAC、DL EARFCN、经度、纬度、天线高度、小区半径、带宽
NR	小区 ID、小区名称、NR PCI、TAC、NR DL EARFCN、经度、纬度、天线高度、小区半径、带宽

CSV（所有供应商）：上传从所选 NMS 导出的 CSV。第一行必须□列标题。列映射基于所选供应商格式自动完成。

JSON：粘贴小区对象的 JSON 数组。必填字段：`cell_id`。可选字段：`pci`、`earfcn`、`latitude`、`longitude`、`lac`、`tac`、`rat`。

重复处理

导入与现有条目具有相同小区 ID 的小区将覆盖现有数据。这意味着重新导入相同文件□安全的，并将更新任何更改的值。

数据持久性

所有小区数据存储在 Mnesia 中，使用 `disc_copies` 存储。小区在应用程序重启后持久存在，无需重新导入或重新同步。

添加 / 编辑小区表单

表单字段以网格形式组织：

字段	描述
小区 ID	唯一标识符（编辑时禁用）
PCI	物理小区标识（0-503）
EARFCN	无线频率信道号
纬度	十进制度
��度	十进制度
天线端口	1、2 或 4
PRS 带宽 (RBs)	6、15、25、50、75 或 100
PRS 配置索引	0-4095
CP 长度	正常或扩展

删除确认

点击小区行上的“Del”会显示确认模态以进行删除。

小区地图

路径：`/map`

交互式地图显示来自小区数据库的所有小区，使用 Leaflet 的深色主题 CartoDB 基础地图。小区根据其 RAT 类型以不同颜色的标记显示。

标记颜色

RAT	颜色
GSM (2G)	橙色
UMTS (3G)	蓝色
LTE (4G)	绿色
NR (5G)	紫色
未知	灰色
选中	琥珀色

在较低的缩放级别，标记使用 Leaflet MarkerCluster 进行聚类。点击标记会显示一个弹出窗口，显示小区的详细信息：小区 ID、名称、技术、LAC/SAC/TAC、PCI、EARFCN 和坐标。

广播区域选择

使用右上角的多边形或矩形绘图工具选择地理区域内的小区。当小区被选中时：

- 底部的选择栏显示所选小区的数量和唯一的 LAC(s)、SAC(s) 和 TAC(s)
- 出现一个 发送广播到选中 按钮，链接到发送广播页面，LACs、SACs 和 TACs 预填充

发送广播表单会根据所选区域代码自动选择适当的目标网络：

- 存在 LACs → 启用 2G (CBSP)
- 存在 SACs → 启用 3G (SABP)
- 存在 TACs → 启用 4G (SBc-AP)

Diameter

路径：`/diameter`
刷新：每 2 秒

监控剩余 Diameter 接口（SLg 和其他）的 Diameter 对等体连接。E-SMLC 到 MME 的 SLs 接口使用原生 LCS-AP 通过 SCTP，并在仪表板页面单独监控。

接口状态

两个面板显示接口健康状况：

接口	描述
SLg 接口 (TS 29.172)	GMLC 到 MME 通过 DRA, 应用 ID 16777264
其他 Diameter 接口	根据配置的其他 Diameter 应用

每个面板显示一个绿色/红色指示点和描述。

对等体摘要

顶部显示连接和断开连接的对等体计数。

对等体表

列	描述
对等体	Diameter 主机标识
领域	Diameter 领域
IP 地址	传输地址 (协议://ip:port)
状态	连接或断开连接徽章

点击对等体行展开以显示详细信息：

详细信息	描述
连接发起	OmniLCS □否发起连接
传输	SCTP 或 TCP
产品名称	远程对等体的 Diameter 产品名称
广告应用	对等体支持的应用 ID

CBC 2G

路径：`/cbc`
刷新：每 2 秒

监控 2G 小区播的 CBSP 接口。

接口信息

显示 CBSP 接口状态：

- 协议参考 (3GPP TS 48.049)
- 监听端口号
- 连接的对等体数量
- 绿色/红色指示器

统计卡片

卡片	描述
连接的 BSCs	状态为 `:connected` 的对等体数量
总连接	对等体总数（包括任何未连接的）
最近消息	最近消息缓冲区中的 CBSP 消息计数

连接的 **BSCs** 面板

显示连接的 CBSP 对等体的列表，包括：

- 对等体 IP 地址和端口
- 连接 ID
- 连接时间戳
- 连接状态徽章

最近 **CBSP** 消息面板

显示 20 条最近的 CBSP 消息的表格:

列	描述
时间	消息时间戳（HH:MM:SS）
对等体	源对等体的连接 ID
消息	可读消息类型（例如，“WRITE REPLACE COMPLETE”）
类型	颜色编码徽章（完成为 OK，失败为 FAIL，错误为 ERR，其他为 INFO）

消息通过对 `cbsp:connections` 和 `cbsp:messages` 主题的 PubSub 订阅接收。

CBC 3G

路径: `/cbc3g`
刷新: 每 2 秒

监控通过 Iu-BC 接口的 3G 小区广播的 SABP 接口（3GPP TS 25.419）。

接口信息

显示 SABP 接口状态:

- 协议参考 (3GPP TS 25.419)
- 监听端口号
- 连接的 RNC ��量
- 绿色/红色指示器

统计卡片

卡片	描述
连接的 RNCs	状态为 `:connected` 的 RNC 对等体数量
总连接	对等体总数（包括任何未连接的）
最近消息	最近消息缓冲区中的 SABP 消息计数

连接的 **RNCs** 面板

显示连接的 SABP 对等体的列表，包括：

- 对等体 IP 地址和端口
- 连接 ID
- 连接时间戳
- 连接状态徽章

最近 **SABP** 消息面板

显示 20 条最近的 SABP 消息的表格：

列	描述
时间	消息时间戳（HH:MM:SS）
对等体	源 RNC 的连接 ID
消息	可读消息类型（例如，“WRITE REPLACE COMPLETE”、“RESTART INDICATION”）
类型	颜色编码徽章（完成为 OK，失败为 FAIL，错误为 ERR，其他为 INFO）

消息通过对 `sabp:connections` 和 `sabp:messages` 主题的 PubSub 订阅接收。

发送广播

路径：`/send_broadcast`

刷新：实时

统一广播组合接口，用于在 2G (CBSP)、3G (SABP) 和 4G (SBc-AP) 网络上同时发送紧急警告或测试消息。选择一个或多个目标网络，广播将发送到每个选定网络上的所有连接对等体。

默认消息 ID 设置为 **4355 (ETWS 测试)** 以确保安全。

从小区地图预填充

从小区地图的区域选择导航时，表单会自动填充：

- 目标网络 根据区域参数启用（LACs → 2G, SACs → 3G, TACs → 4G）
- **LAC**、**SAC** 和 **TAC** 字段 预填充选定小区的值
- 小区范围 设置为“按 LAC”

这允许完整的工作流程：在地图上按地理位置选择小区，点击“发送广播到选中”，撰写消息并发送。

活动广播

路径：`/broadcasts`

刷新：每 3 秒 + 实时 PubSub

统一视图显示所有 2G (CBSP)、3G (SABP) 和 4G (SBc-AP) 网络上的活动小区广播。广播从发送开始跟踪，直到停止或过期。视图每 3 秒自动刷新，并在广播状态变化时实时更新。

摘要卡片

卡片	描述
总活动	所有网络的活动广播总数
2G (CBSP)	活动 2G 广播
3G (SABP)	活动 3G 广播
4G (SBC-AP)	活动 4G 广播

在任何广播行上使用停止按钮以在其各自网络上取消该广播。

CBC 4G

路径：`/cbc4g`

刷新：每 3 秒

功能齐全的 4G 小区广播管理接口，用于撰写、发送、监控和停止 SBC-AP 广播。

统计卡片

卡片	描述
连接的 MMEs	已建立的 SBC-AP SCTP 关联数量
活动广播	当前活动的 4G 广播数量
总发送	历史广播总数

撰写广播表单

字段	描述
消息文本	广播消息内容
消息 ID	16 位 CB 消息标识符（例如，4370 代表 CMAS）
序列号	16 位序列号
MCC	移动国家代码（例如，“313”）
MNC	移动网络代码（例如，“380”）
TACs	逗号分隔的跟踪区域代码
警告类型	无、地震、海啸、地震 + 海啸、测试、其他
重复（秒）	广播重复之间的秒数
广播次数	广播消息的总次数

点击“发送广播”对消息进行编码（GSM 7 位），构建 Write-Replace-Warning-Request PDU，并将其发送到所有连接的 MMEs。

连接的 **MMEs** 面板

显示 SBC-AP SCTP 关联的列表：

- MME 主机名
- IP 地址、端口和 SCTP 关联 ID
- 连接状态徽章（连接 / 状态名称）

活动广播面板

当前活动广播的表格：

列	描述
MSG ID	十六进制的消息标识符（例如，0x1112）
SERIAL	十六进制的序列号
MESSAGE	广播文本（截断）
STATUS	已发送、停止中、已停止、已确认
ACTION	停止按钮（对于已��送/已确认的广播）

点击“停止”发送 Stop-Warning-Request 以停止广播。

广播历史

所有广播的表格（最近 100 条）：

列	描述
MSG ID	十六进制的消息标识符
SERIAL	十六进制的序列号
MESSAGE	广播文本（截断）
TIME	发送时间戳（HH:MM:SS）
STATUS	最终状态徽章

状态更新通过对 `cbc:state` 和 `cbc:connections` 主题的 PubSub 订阅接收。

CAP 警报

路径：`/cap`
刷新：每 3 秒 + 实时 PubSub

来自外部警报机构的 CAP（公共警报协议）警报的操作员审批工作流。有关配置、馈送轮询和多边形解析的完整详细信息，请参见 CAP 警报操作指南。

统计卡片

卡片	描述
待审批	等待操作员操作的警报数量（当 > 0 时高亮琥珀色）
活动广播	当前正在广播的警报数量
总处理	历史警报总数
审批模式	当前设置：“手动”（require_approval: true）或“自动”

待处理警报面板

仅在 CAP 配置中 `require_approval` 为 `true` 时可见。

列	描述
TIME	警报接收时间
EVENT	警报事件类型（例如，“龙卷风警告”）
SEVERITY	警报严重性（极端、严重、中等、轻微）
CELLS	通过多边形解析匹配的小区数量
STATUS	待处理徽章（琥珀色）
ACTIONS	预览、批准、拒绝按钮

预览 展开以显示完整的警报描述、来源、匹配的 TACs/LACs、消息 ID 和 PLMN。

批准 触发通过 SBC-AP (4G)、SABP (3G) 和 CBSP (2G) 的即时广播。

拒绝 将警报移动到历史记录中作为拒绝。

活动广播面板

列	描述
EVENT	警报事件类型
MSG ID	CB 消息标识符
TACs	目标跟踪区域代码
STARTED	广播开始时间
STATUS	正在广播（蓝色）或已发送（绿色）

警报历史面板

列	描述
TIME	发送或接收时间戳
EVENT	警报事件类型
SEVERITY	警报严重性
CELLS	匹配的小区数量
TACs/LACs	目标区域代码
STATUS	已发送（绿色）、已拒绝（红色）或失败（红色）

当收到新的待处理警报时，会出现吐司通知。

GMLC / Le 接口

路径：`/gmlc`
刷新：每 3 秒 + 实时 PubSub

监控和管理 GMLC Le 接口和延迟位置会话���有关完整配置和 Diameter 协议的详细信息，请参见 GMLC & Le 接口操作指南。

统计卡片

卡片	描述
活动会话	总活动的延迟位置会话
定期	活动的定期位置会话数量
地理围栏	活动的触发/地理围栏会话数量

授权 LCS 客户端面板

列	描述
NAME	客户端身份（与 Diameter AVP 匹配）
TYPE	LCS 客户端类型（紧急服务、增值服务等）
ALLOWED METHODS	此客户端可以请求的定位方法
RATE LIMIT	每分钟请求次数（或“无限制”）

活动延迟会话面板

列	描述
SESSION ID	截断的会话 UUID
TYPE	会话类型徽章：定期（绿色）或触发（紫色）
IMSI	订阅者 IMSI
METHOD	定位方法
CLIENT	请求 LCS 客户端的名称
PROGRESS	定期：已完成/总报告。触发：剩余报告或“活动（无限制）”
LAST FIX	最近位置修正的时间
ACTIONS	取消按钮以终止会话

会话历史面板

显示最近 50 个完成或取消的会话，包括会话 ID、类型、IMSI、状态（完成/取消）和开始时间。

内置页面

这些页面由控制面板框架提供。

应用程序（`/application`）

OTP 应用程序资源查看器，显示：

- 正在运行的应用程序及其进程树
- 每个进程的内存使用情况
- 消息队列长度

配置（`/configuration`）

运行时配置查看器，显示：

- 所有应用程序环境变量
- 按应用程序分组

日志（`/log`）

实时日志查看器，显示：

- 来自 `ControlPanel.Logger` 后端的日志消息
- 可按日志级别筛选
- 自动滚动并具有暂停功能