

OmniSGW □作指南

OmniSGW - 服务网关 **(SGW)**

□ Omnitouch □□□□□□

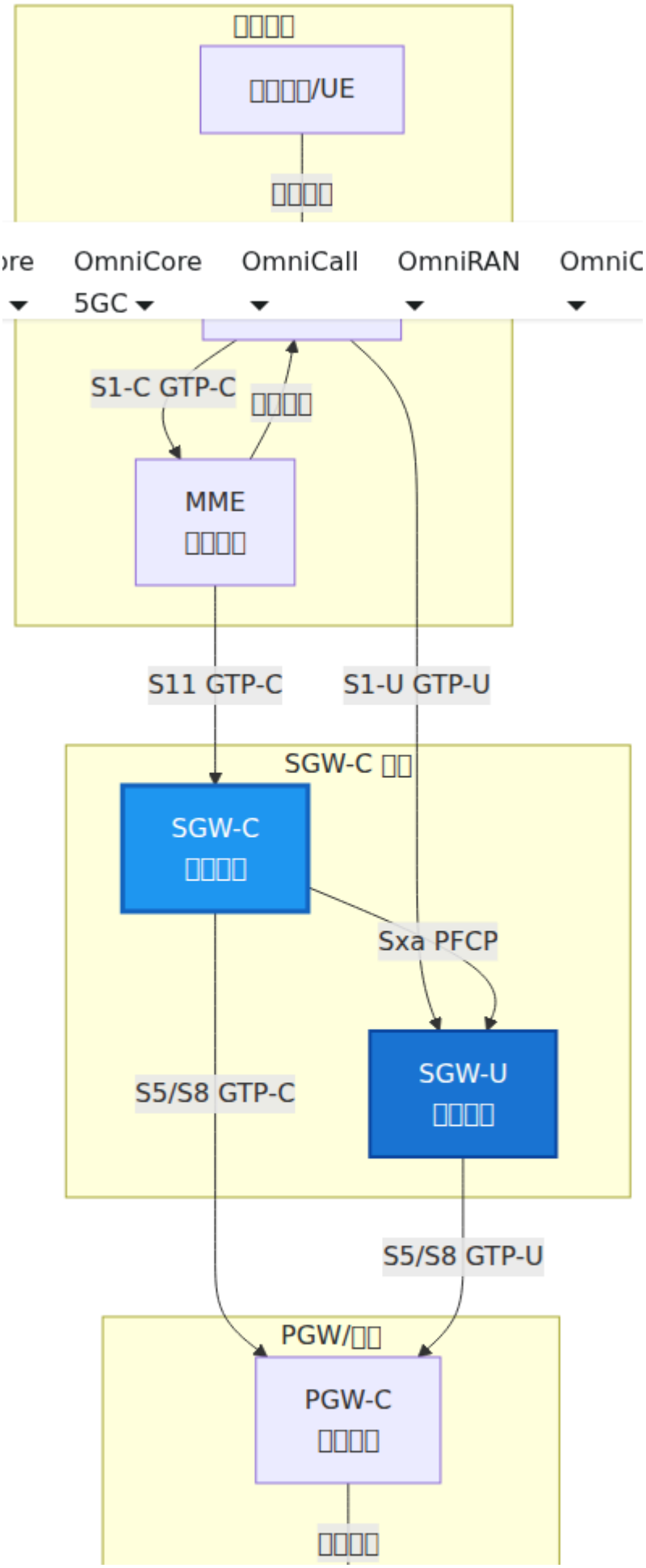
目录

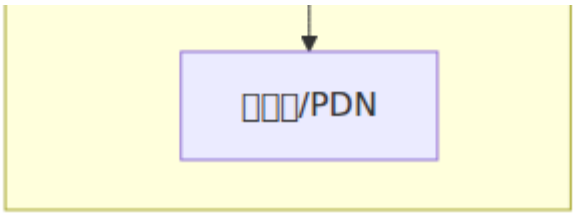
1. □述
2. 架构
3. 网络接口
4. 关键□念
5. 入门
6. 配置
7. Web UI - 实时操作仪表板
8. 监控与指标
9. 详细文档
10. 附加资源

概述

OmniSGW 是 Omnitouch 网络服务开发的高性能服务网关 (SGW) 实现，适用于 3GPP LTE 演进分组核心 (EPC) 网络。它管理 UE 移动性和承载管理的功能，包括：

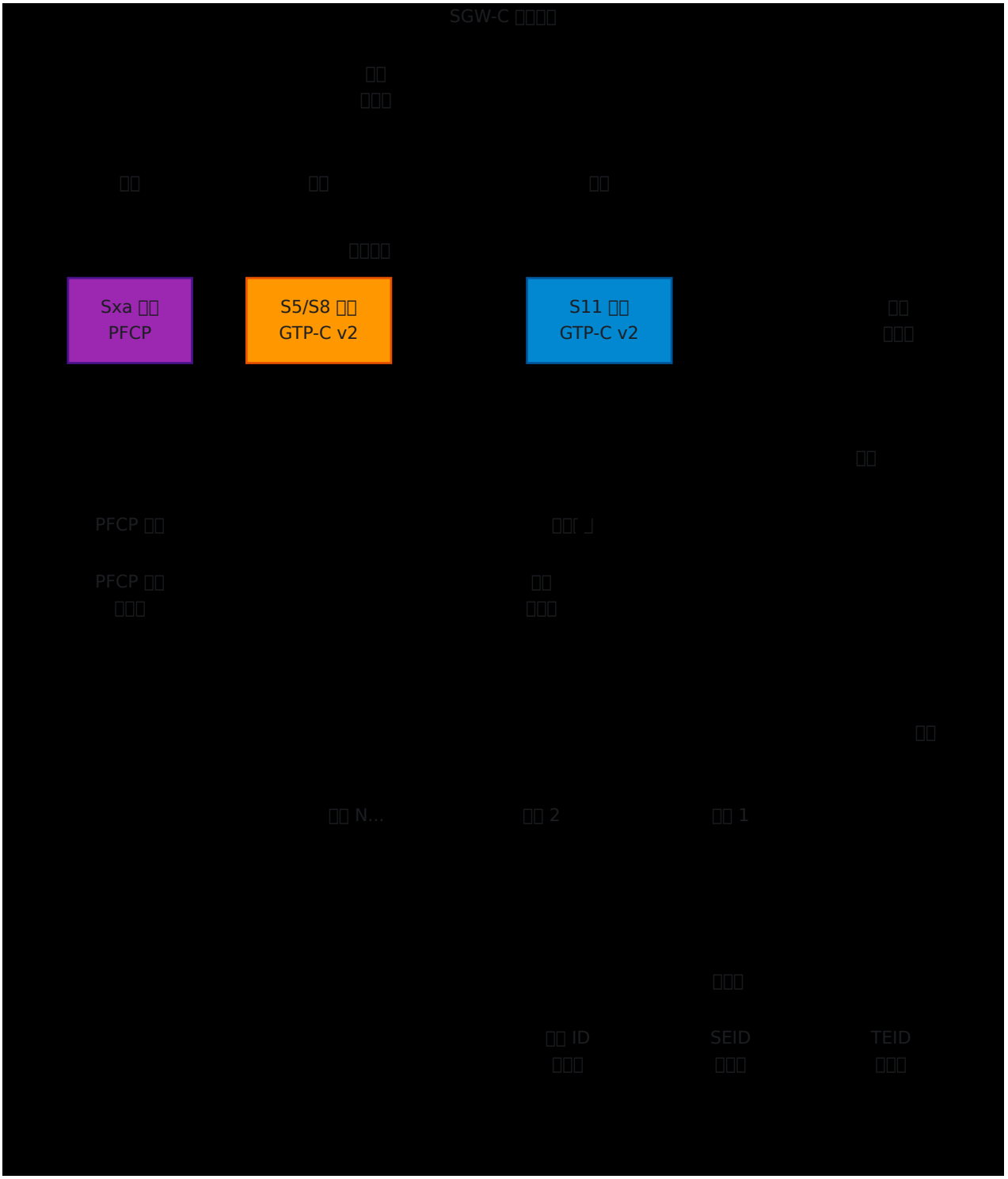
- 会话管理 - 创建、修改和终止 UE (用户设备) 数据会话
- 移动协调 - 处理 eNodeB 之间的切换和数据转发
- 承载管理 - 为不同的 QoS 需求创建和修改专用承载
- 计费信息 - 跟踪会话事件以进行离线计费
- 用户���面协调 - 控制 SGW-U (用户平面) 进行数据包转发


□□□□
□□□□/UE
□□□□
ore
OmniCore
5GC
OmniCall
OmniRAN
OmniC
S1-C GTP-C
□□□□
MME
□□□□
S11 GTP-C
S1-U GTP-U
SGW-C □□
SGW-C
□□□□
Sxa PFCP
SGW-U
□□□□
S5/S8 GTP-C
S5/S8 GTP-U
PGW/□□
PGW-C
□□□□
□□□□



SGW-C 的功能

- 接受来自 **MME** 的会话请求 通过 S11 接口 (GTP-C)
- 与 **PGW-C** 协调 通过 S5/S8 接口 (GTP-C) 进行 PDN 连接
- 管理承载生命周期 包括创建、修改和删除
- 在 **SGW-U** 中编程转发规则 通过 Sxa 接口 (PFCP)
- 通过管理 **eNodeB** 之间的切换来处理 **UE** 移动性
- 为挂起会话提供下行数据分页
- 跟踪计费信息 以支持离线计费系统



架构

组件概述

SGW-C
Sxa 代理
PFCP
S5/S8 代理
GTP-C v2
S11 代理
GTP-C v2
PFCP
PFCP
N...
2
1
ID
SEID
TEID

过程架构

SGW-C 基于 Elixir/OTP 构建，并使用监督过程架构：

- 应用程序监督者 - 管理所有组件的顶级监督者
- 协议代理 - 处理出协议消息 (S11, S5/S8, Sxa)
- 会话进程 - 每个活动 UE 会话一个 GenServer
- 注册表 - 跟踪分配的资源 (TEID、SEID、计费 ID 等)
- **PFCP** 节点管理器 - 维护与 SGW-U 对等体的 PFCP 关联

每个组件都受到监督，并在故障时会自动重启，以确保系统的可靠性。

可以通过 Web UI 应用程序页面监控实时系统健康指标：

网络接口

SGW-C 实现了三个主要的 3GPP 接口：

S11 接口 (GTP-C v2)

目的： MME 和 SGW-C 之间的控制平面信令

协议：基于 UDP 的 GTP-C 版本 2

关键消息：

- 创建会话请求/响应
- 删除会话请求/响应
- 修改承载请求/响应
- 创建承载请求/响应
- 删除承载请求/响应
- 下行数据通知/确认

配置：参见 S11 接口文档

Sxa 接口 (PFCP)

目的：SGW-C 和 SGW-U 之间的控制平面信令

协议：基于 UDP 的 PFCP (数据包转发控制协议)

关键消息：

- 关联设置请求/响应
- 会话建立请求/响应
- 会话修改请求/响应
- 会话删除请求/响应
- 会话报告请求/响应
- 心跳请求/响应

配置：参见 PFCP/Sxa 接口文档

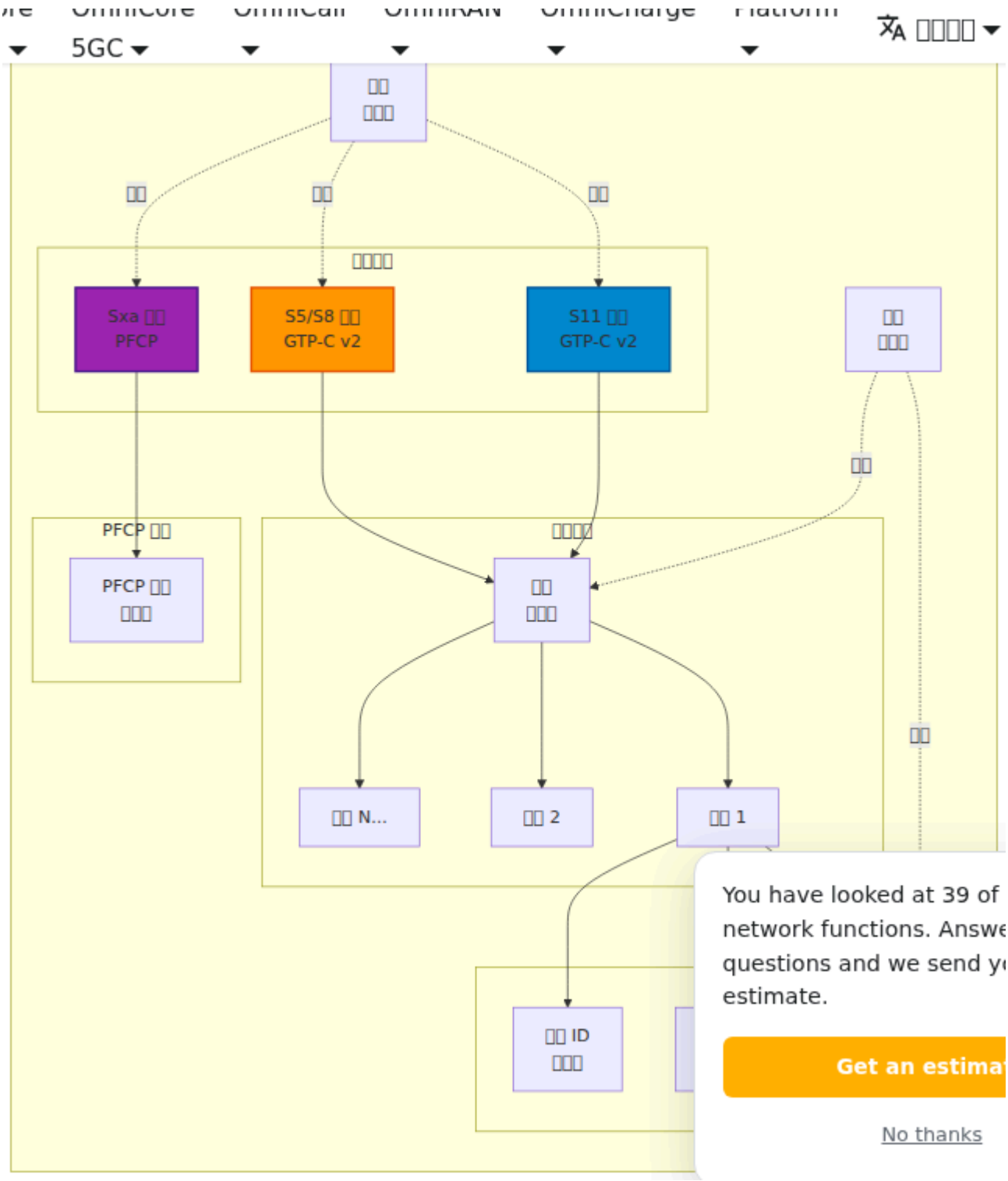
S5/S8 接口 (GTP-C v2)

目的：SGW-C 和 PGW-C 之间的控制平面信令，用于 PDN 连接

协议：基于 UDP 的 GTP-C 版本 2

关键消息：

- 创建会话请求/响应
- 删除会话请求/响应
- 修改承载请求/响应



- 创建承载请求/响应
- 删除承载请求/响应

配置：参见 S5/S8 接口文档

关键概念

UE 会话

UE 会话代表一个连接到网络的活动移动设备。每个会话管理：

- **IMSI**（国际移动用户身份）- 唯一的用户标识符
- **GUTI**（全球唯一临时标识符）- 来自 MME 的临时 UE 标识符
- **MSISDN** - 手机号码
- **TAI**（跟踪区域标识符）- 当前位置区域
- 会话 **TEID** - S11 和 S5/S8 的隧道端点标识符
- 活动承载 - 关联数据承载的列表

PDN 连接

PDN（数据包数据网络）连接代表 UE 通过特定 PGW-C 的数据连接。每个会话具有：

- **APN**（接入点名称）- 标识外部网络
- 计费 **ID** - 用于 SGW 和 PGW 之间计费的唯一标识符
- **TEID**（隧道端点 ID）- S5/S8 接口隧道标识符
- **SEID**（会话端点 ID）- Sxa 接口会话标识符
- 默认承载 - 每个 PDN 连接时创建
- 专用承载 - 针对特定 QoS 需求的额外承载

承��上下文

承载代���具有特定 QoS 特性的流量流：

- 默认承载 - 每个 PDN 连接时创建，适用于尽力而为的流量
- 专用承载 - 针对特定服务需求（语音、视频等）的额外承载
- **EBI**（EPS 承载 ID）- 每个会话中每个承载的唯一标识符
- **QoS** 参数 - QCI（QoS 类标识符）、ARP（分配与保留优先级）、比特率（MBR, GBR）

PFCP 规则

SGW-C 为 SGW-U 编程数据包处理规则：

- **PDR**（数据包检测规则）- 匹配数据包（上行/下行）
- **FAR**（转发动作规则）- 指定转发行为
- **QER**（QoS 执行规则）- 强制比特率限制
- **BAR**（缓冲动作规则）- 控制切换期间的数据包缓冲

有关详细信息，请参见 Sxa 接口文档。

移动性与切换

SGW-C 支持 UE 在 eNodeB 之间的移动性：

- 同 **MME** 切换 - 在同一 MME 内切换（无 SGW 更改）
- 跨 **MME** 切换 - 在 MMEs 之间切换，伴随 SGW 迁移
- 数据转发 - 在切换期间缓冲和转发数据
- 跟踪区域更新 - 在区域之间移动时 UE 重新注册

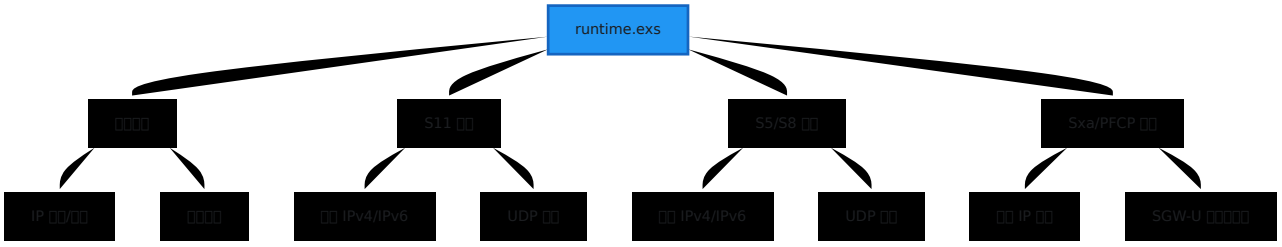
入门

先决条件

- Elixir ~1.16
- Erlang/OTP 26+
- 与 MME、SGW-U 和 PGW-C 的网络连接
- 对 LTE EPC 架构的理解

验证□作

检查日志以确认成功启动：



```
[info] □动 OmniSGW...
[info] 在 127.0.0.40:42068 □动指标导出器
[info] 在 127.0.0.10 □动 S11 代理
[info] 在 127.0.0.15 □动 S5/S8 代理
[info] 在 127.0.0.20 □动 Sxa 代理
[info] □动 PFCP 节点管理器
[info] OmniSGW 成功□动
```

访问指标在 `http://127.0.0.40:42068/metrics`（配置的地址）。

配置

所有运行时配置在 `config/runtime.exs` 中定义。配置分为几个部分：

配置概述

快速配置参考

部分	目的	文档
metrics	Prometheus 指标导出器	监控指南
s11	GTP-C 接口到 MME	S11 配置
s5s8	GTP-C 接口到 PGW-C	S5/S8 配置
sxa	PFCP 接口到 SGW-U	Sxa 配置

有关详细信息，请参见 完整配置指南。

Web UI - 实时□作仪表板

OmniSGW 包含一个内置的 **Web UI** 用于实时监控和操作，提供系统状态的即时可见性，无需命令行工具或指标查询。

访问 Web UI

```
http://<omnisgw-ip>:<web-port>/
```

可用页面：

页面	URL	目的	刷新率
UE 会话	`/ue_sessions`	查看所有活动 UE 会话和承载	2 秒
PFCP 会话	`/pfcp_sessions`	查看与 SGW-U 的 PFCP 会话	2 秒
SGW-U 状态	`/sgwu_status`	监控 PFCP 对等体关联	2 秒
日志	`/logs`	实时日志流	实时

主要功能

实时更新：

- 所有页面自动刷新（无需手动重新加载）
- 来自 OmniSGW 进程的实时数据流
- 颜色编码的状态指示器（绿色/红色）

搜索与过滤：

- 按 IMSI、GUTI、电话号码搜索会话
- 无需页面重新加载即可即时过滤

可扩展详细信息：

- 点击任何行以查看完整会话详细信息
- 检查所有活动承载和 QoS 参数
- 查看对等体配置和能力

无需身份验证（���部使用）：

- 直接从管理网络访问
- 设计用于 NOC/运营团队使用
- 仅绑定到管理 IP 以提高安全性

作工作流程

会话故障排除：

1. 用户报告连接问题
2. 打开 UE 会话页面
3. 按 IMSI 或电话号码搜索
4. 验证会话是否存在并具有正确的：
   - 跟踪区域
   - 活动承载及其 QoS
   - 隧道端点已建立
   - 正确的 PGW-C 关联
5. 如果未找到会话 → 检查日志以获取拒绝原因

系统健康检查：

1. 打开 SGW-U 状态页面 → 验证所有 SGW-U 对等体“已关联”
2. 打开 UE 会话 → 检查活动会话计数与容量
3. 监控 APN 之间的承载分布

容量监控：

- 观察 UE 会话计数
- 与许可/预期容量进行比较
- 确定高峰使用时间
- 监控按服务类型的分布

Web UI 与指标

使用 **Web UI** 进行:

- 单个会话和承载详细信息
- 实时对等体状态
- 快速健康检查
- 故障排除特定用户
- 验证配置

使用 **Prometheus** 指标进行:

- 历史趋势
- 报警和通知
- 容量规划图
- 性能分析
- 长期监控

最佳实践：同时使用两者 - Web UI 用于即时操作，Prometheus 用于趋势和警报。

监控与指标

除了 Web UI，OmniSGW 还暴露了与 Prometheus 兼容的指标以供监控:

可用指标

- 会话指标
  - `teid_registry_count` - 活动 S11/S5S8 TEID
  - `seid_registry_count` - 活动 PFCP 会话
  - `charging_id_registry_count` - 活动计费 ID
  - `active_ue_sessions` - 总活动 UE 会话
  - `active_bearers` - 所有会话中的总活动承载
- 消息指标

    - `s11_inbound_messages_total` - 在 S11 上接收到的 GTP-C 消息
    - `s5s8_inbound_messages_total` - 在 S5/S8 上接收到的 GTP-C 消息
    - `sxa_inbound_messages_total` - 接收到的 PFCP 消息
    - 消息处理持续时间分布

- 错误指标

    - `s11_inbound_errors_total` - S11 协议错误
    - `s5s8_inbound_errors_total` - S5/S8 协议错误
    - `sxa_inbound_errors_total` - Sxa 协议错误

访问指标

指标通过 HTTP 在配置的端点上暴露：

```
curl http://127.0.0.40:42068/metrics
```

有关仪表板设置和警报的详细信息，请参见 监控与指标指南。

详细文档

本节提供 OmniSGW 文档的全面□述。文档按主题和用例组织。

文档结构

```
OmniSGW 文档
├── OPERATIONS.md （本指南）
│
└── docs/
    ├── 配置与设置
    │   ├── configuration.md          完整的 runtime.exs 参考
    ⍰⍰
    ├── 网络接口
    │   ├── sxa-interface.md          Sxa/PFCP （SGW-U 通信）
    │   ├── s11-interface.md          S11 （MME 通信）
    │   └── s5s8-interface.md         S5/S8 （PGW-C 通信）
    │
    └── 操作
        ├── session-management.md     UE 会话生命周期
        ├── bearer-management.md      承载操作
        ├── cdr-format.md             离线计费记录
        └── monitoring.md             Prometheus 指标与警报
```

按主题分类的文档

□ 入门

文档	描述	目的
OPERATIONS.md	主要操作指南（本文件）	□述和快速入门

⚙ 配置

文档	描述
configuration.md	完整的 runtime.exs 配置参考

□ 网络接口

文档	描述
sxa-interface.md	PFCP/Sxa 接口到 SGW-U
s11-interface.md	GTP-C S11 接口到 MME
s5s8-interface.md	GTP-C S5/S8 接口到 PGW-C

□ □作与监控

文档	描述
session-management.md	UE 会话生命周期和操作
session-management.md	承载创建、修改、删除
cdr-format.md	离线计费数据记录格式
monitoring.md	Prometheus 指标、Grafana 仪表板、警报

阅读路径

对于网络运营商

1. OPERATIONS.md - □述（本文件）
2. configuration.md - 设置
3. monitoring.md - 监控
4. session-management.md - 日常操作

对于网络工程师

1. OPERATIONS.md - 架构□述（本文件）
2. sxa-interface.md - 用户平面控制
3. s11-interface.md - 移动管理
4. s5s8-interface.md - PDN 连接
5. session-management.md - 会话生命周期

6. session-management.md - 承载操作

对于配置与部署

1. configuration.md - 完整参考
2. monitoring.md - 设置监控

附加资源

3GPP 规范

规范	标题
TS 29.274	GTP-C v2 (S11 和 S5/S8 接口)
TS 29.244	PFCP (Sxa 接口)
TS 32.251	分组交换域计费
TS 32.298	CDR 编码
TS 23.401	EPC 架构

数据计费数据记录（CDR）格式

SGW-C 的离线计费

OmniSGW □ Omnitouch □□□□□□

目录

1. □述
2. CDR 文件格式
3. CDR 字段
4. CDR 事件
5. 文件结构
6. 配置
7. CDR 生成流程
8. 字段详细信息
9. 示例
10. 集成

概述

数据 **CDR（**计费数据记录**）**格式为服务网关控制平面（SGW-C）提供离线计费能力。CDR 被生成以记录承载会话事件、数据使用情况和用户信息，以便于计费和分析。

该通用格式与 PGW-C CDR 兼容，确保 EPC 基础设施中计费记录的一致性。

主要特点

- 基于 **CSV** 的格式 - 简单、易读的逗号分隔值
- 基于事件的记录 - 捕获承载的开始、更新和结束事件
- 流量计量 - 记录上行和下行数据使用情况

- 自动轮换 - 基于时间间隔的可配置文件轮换
- 符合 **3GPP** 标准 - 遵循 3GPP TS 32.251 (PS 域计费) 和 TS 32.298 (CDR 编码)

用例

用例	描述
离线计费	生成后付费计费的 CDR
分析	分析用户使用模式
审计跟踪	跟踪所有承载会话事件
容量规划	监控网络资源利用率
故障排除	调试会话和承载问题

CDR 文件格式

文件命名约定

```
<epoch_timestamp>
```

示例：

```
1726598022
```

文件名是文件创建时的 Unix 纪元时间戳（以秒为单位）。

文件位置

默认目录：

- SGW-C: `/var/log/sgw_c/cdrs/`

可通过 CDR 报告器配置中的 `directory` 参数进行配置。

文件头

每个 CDR 文件以包含元数据的多行头部开始：

```
# 数据 CDR 文件：
# 文件开始时间：HH:MM:SS (unix_timestamp)
# 文件结束时间：HH:MM:SS (unix_timestamp)
# 网关名称：<gateway_name>
#
epoch,imsi,event,charging_id,msisdn,ue_imei,timezone_raw,plmn,tac,eci
```

头字段：

- 文件开始时间 - CDR 文件创建的时间（可读和 Unix 时间戳）
- 文件结束时间 - 文件轮换将发生的时间（可读和 Unix 时间戳）
- 网关名称 - SGW-C 实例的标识符
- 列标题 - 数据记录的 CSV 字段名称

CDR 字段

字段摘要

位置	字段名称	类型	描述
0	epoch	整数	事件时间戳（Unix 纪元秒）
1	imsi	字符串	国际移动用户身份
2	event	字符串	CDR 事件类型（例如，"default_bearer_start"）
3	charging_id	整数	承载的唯一计费标识符
4	msisdn	字符串	移动站 ISDN 号码（电话号码）
5	ue_imei	字符串	国际移动设备身份
6	timezone_raw	字符串	UE 时区（保留，目前为空）
7	plmn	整数	公共陆地移动网络标识符
8	tac	整数	跟踪区域代码
9	eci	整数	E-UTRAN 小区标识符
10	sgw_ip	字符串	SGW-C S5/S8 控制平面 IP 地址
11	ue_ip	字符串	UE IP 地址（IPv4|IPv6 格式）
12	pgw_ip	字符串	PGW-C S5/S8 控制平面 IP 地址
13	apn	字符串	接入点名称
14	qci	整数	QoS 类标识符

位置	字段名称	类型	描述
15	octets_in	整数	下行数据量（字节）
16	octets_out	整数	上行数据量（字节）

CDR 事件

事件类型

CDR 为三种类型的事件生成：

事件类型	格式	描述	生成时机
承载开始	`<type>_bearer_start`	承载建立	发送创建会话响应
承载更新	`<type>_bearer_update`	会话期间的使用报告	来自用户平面的定期使用报告
承载结束	`<type>_bearer_end`	承载终止	删除会话请求/响应

承载类型：

- `default` - 默认承载（每个 PDN 连接一个）
- `dedicated` - 专用承载（每个 PDN 连接零个或多个）

事件示例

```
default_bearer_start       - 默认承载已建立
default_bearer_update      - 默认承载使用更新
default_bearer_end         - 默认承载已终止
dedicated_bearer_start     - 专用承载已建立
dedicated_bearer_update    - 专用承载使用更新
dedicated_bearer_end       - 专用承载已终止
```

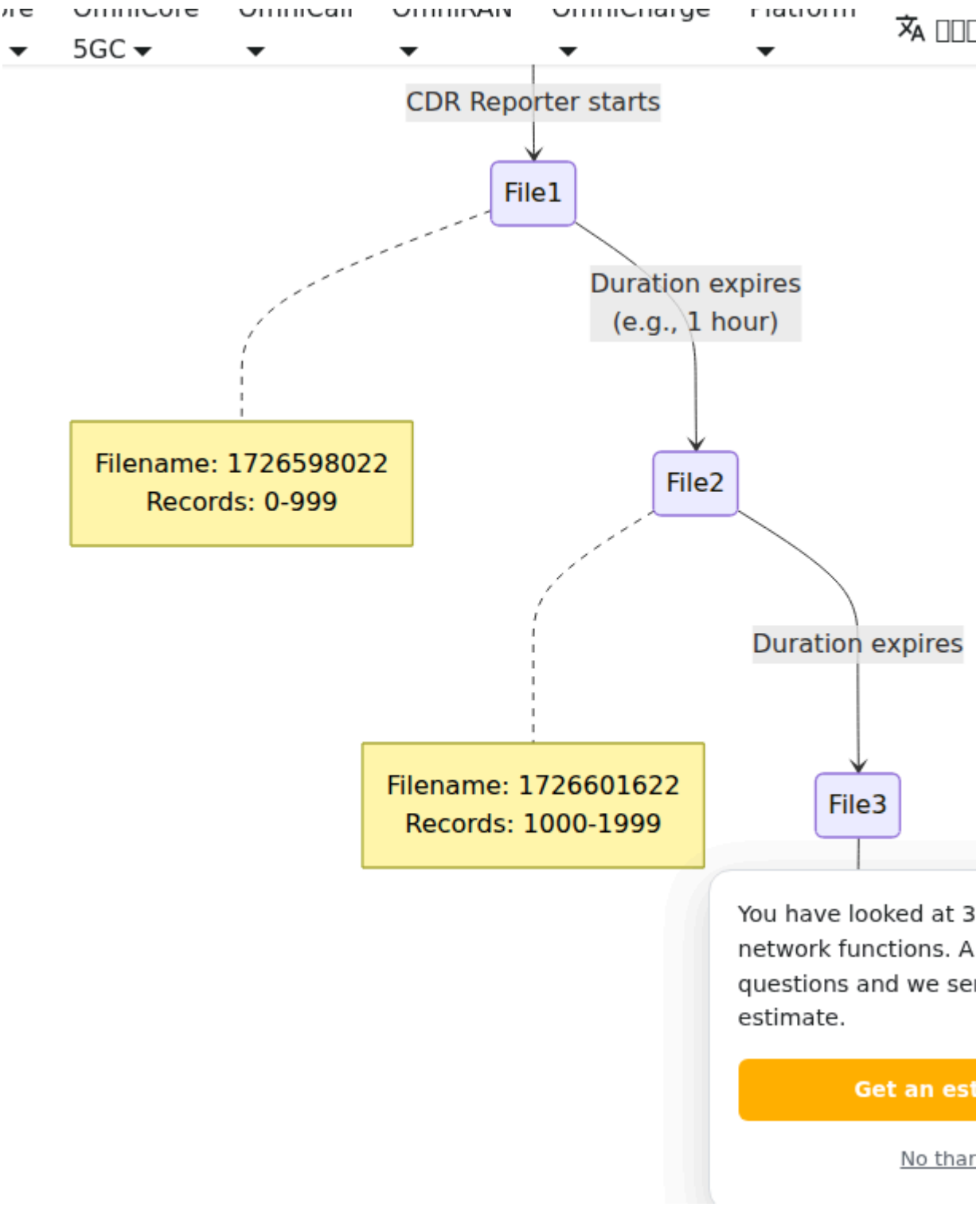
文件结构

示例 **CDR** 文件

```
# 数据 CDR 文件:
# 文件开始时间: 18:53:42 (1726598022)
# 文件结束时间: 19:53:42 (1726601622)
# 网关名称: sgw-c-prod-01
# epoch,imsi,event,charging_id,msisdn,ue_imei,timezone_raw,plmn,tac,e
1726598022,310260123456789,default_bearer_start,12345,15551234567,123
1726598322,310260123456789,default_bearer_update,12345,15551234567,12
1726598622,310260123456789,default_bearer_update,12345,15551234567,12
1726598922,310260123456789,default_bearer_end,12345,15551234567,12345
```

文件轮换

CDR 文件根据配置的持续时间自动轮换:



轮换过程：

1. 关闭当前 CDR 文件
2. 创建带有当前时间戳的新文件
3. 将头部写入新文件
4. 继续将 CDR 记录到新文件

配置

配置参数

参数	类型	描述	默认值	推荐值
`gateway_name`	字符串	SGW-C 实例标识符	-	使用主机名或实例 ID
`duration`	整数	文件轮换间隔（毫秒）	-	3600000 (1 小时)
`directory`	字符串	CDR 输出目录路径	-	`/var/log/sgw_c/cdrs`

配置示例

生产环境：

- **gateway_name:** "sgw-c-prod-01"
- **duration:** 3,600,000 毫秒 (1 小时轮换)
- **directory:** "/var/log/sgw_c/cdrs"

开发环境：

- **gateway_name:** "sgw-c-dev"
- **duration:** 300,000 毫秒 (5 分钟轮换用于测试)
- **directory:** "/tmp/sgw_c_cdrs"

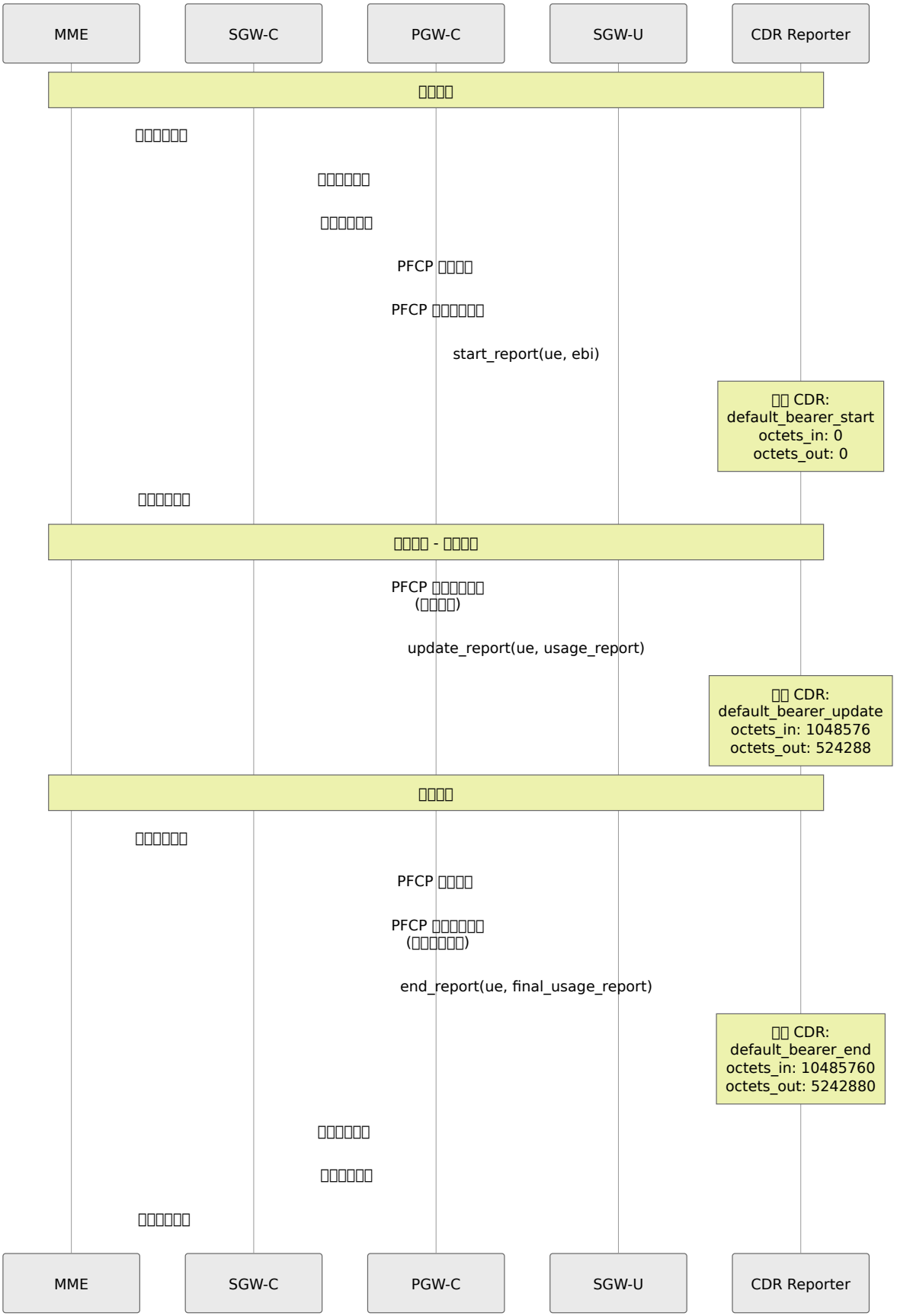
高流量：

- **gateway_name:** "sgw-c-prod-heavy"
- **duration:** 1,800,000 毫秒 (30 分钟轮换)
- **directory:** "/mnt/fast-storage/cdrs"

CDR 生成流程

承载生命周期 **CDR** 事件

SGW-C CDR 生成**:**



MME
SGW-C
PGW-C
SGW-U
CDR Reporter
会话建立
创建会话请求
创建会话请求
创建会话响应
PFCP 会话建立
PFCP 会话建立响应
start_report(ue, ebi)
生成 CDR:
default_bearer_start
octets_in: 0
octets_out: 0
创建会话响应
会话活动 - 数据流动
PFCP 会话报告请求
(使用报告)
update_report(ue, usage_report)
生成 CDR:
default_bearer_update
octets_in: 1048576
octets_out: 524288
会话终止
删除会话请求
PFCP 会话删除
PFCP 会话删除响应
(最终使用报告)
end_report(ue, final_usage_report)
生成 CDR:
default_bearer_end
octets_in: 10485760
octets_out: 5242880
删除会话请求
删除会话响应
删除会话响应
MME
SGW-C
PGW-C
SGW-U
CDR Reporter

CDR 生成事件

1. 承载开始**:**

- 何时**:** 发送创建会话响应
- 目的**:** 记录承载建立时的零使用
- **octets_in:** 0
- **octets_out:** 0

2. 承载更新**:**

- 何时**:** 从用户平面接收到 PFCP 会话报告请求
- 目的**:** 记录增量数据使用
- **octets_in:** 自承载开始以来的累计下行字节
- **octets_out:** 自承载开始以来的累计上行字节

3. 承载结束**:**

- 何时**:** 接收到 PFCP 会话删除响应（带有最终使用）
- 目的**:** 记录会话终止前的最终数据使用
- **octets_in:** 最终总下行字节
- **octets_out:** 最终总上行字节

字段详细信息

1. epoch (时间戳)

类型**:** Unix 纪元时间戳（秒）

描述**:** CDR 事件发生的时间

示例**:**

```
1726598022  → 2025-09-17 18:53:42 UTC
```

---

2. imsi (用户身份)

类型：字符串（最多 15 位数字）

格式：MCCMNC + MSIN

描述：唯一标识用户的国际移动用户身份

示例：

```
310260123456789
  └─┬─┘└─┬─┘└─────┬─────┘
   MCC  MNC      MSIN
   (310)(260) (123456789)
```

来源：UE 上下文，在创建会话请求中接收

---

3. event (CDR 事件类型)

类型：字符串

格式：`<bearer_type>_bearer_<event>`

值：

- `default_bearer_start`
- `default_bearer_update`
- `default_bearer_end`
- `dedicated_bearer_start`
- `dedicated_bearer_update`
- `dedicated_bearer_end`

确定：

- 如果 EBI (EPS 承载 ID) 等于 LBI (链接承载 ID): `default`
- 如果 EBI 不等于 LBI: `dedicated`

来源：承载上下文（EBI 与 LBI 比较）

---

4. charging_id（计费标识符）

类型：无符号 32 位整数

描述：用于跨网络元素的计费关联的唯一标识符

示例：

```
12345
```

来源：由 PGW-C 分配，在创建会话响应中接收

用途：

- 关联 SGW 和 PGW 之间的计费事件
- 用于 Diameter Gy/Gz 计费
- 每个承载唯一

---

5. msisdn（电话号码）

类型：字符串（E.164 格式）

描述：移动站 ISDN 号码（用户的电话号码）

格式：国家代码 + 国内号码

示例：

```
15551234567
 └┘└─────┘
  CC  National
  (1) (5551234567)
```

来源：UE 上下文，通常通过 MME 从 HSS 获取

---

6. ue_imei (设备身份)

类型：字符串（15 位数字）

格式：TAC (8) + SNR (6) + Spare (1)

描述：国际移动设备身份（设备标识符）

示例：

```
123456789012345
└──────┘└────┘└
   TAC   SNR S
```

来源：UE 上下文，从 MME 接收

---

7. timezone_raw (UE 时区)

类型：字符串（当前保留/为空）

描述：保留字段，用于 UE 时区信息

当前状态：未填充（CSV 中为空字段）

未来用途：可能包括时区偏移和夏令时标志

示例：

```
,    (空字段)
```

---

8. plmn (网络标识符)

类型：整数（遗留格式）

描述：公共陆地移动网络标识符，编码为小端十六进制

编码过程：

```
MCC: 505, MNC: 57
  ↓
"50557"
  ↓
交换对: "055570"
  ↓
十六进制转十进制: 0x055570 = 349552
```

示例：

```
349552  → MCC: 505, MNC: 57
```

来源：从 MME 获取的 UE 位置信息

注意：这是向后兼容的遗留编码格式

---

9. tac（跟踪区域代码）

类型：无符号 16 位整数

描述：跟踪区域代码标识 UE 所在的跟踪区域

范围：0 - 65535

示例：

```
1234
```

来源：UE 位置信息，在创建会话请求中从 MME 接收

用途：

- 标识移动管理区域
- 用于寻呼和位置更新

- TAI (跟踪区域标识) 的一部分

---

10. eci (E-UTRAN 小区标识符)

类型: 无符号 28 位整数

描述: E-UTRAN 小区标识符唯一标识为 UE 提供服务的小区

格式: eNodeB ID (20 位) + 小区 ID (8 位)

范围: 0 - 268,435,455

示例:

```
5678
```

来源: 从 MME 获取的 UE 位置信息

用途:

- 标识特定的小区塔和扇区
- 用于切换和移动管理
- 细粒度位置信息

---

11. sgw_ip (SGW 控制平面 IP)

类型: 字符串 (IPv4 或 IPv6 地址)

描述: SGW-C 的 S5/S8 控制平面 IP 地址 (F-TEID)

格式: 点分十进制 (IPv4) 或 冒号-十六进制 (IPv6)

示例:

```
10.0.0.15        (IPv4)
2001:db8::15     (IPv6)
```

来源：本地配置，分配给 S5/S8 接口

---

12. ue_ip (UE IP 地址)

类型：字符串（IPv4|IPv6 格式）

描述：分配给 UE 的 PDN 连接的 IP 地址

格式：`<ipv4>|<ipv6>`

示例：

```
172.16.1.100|                          (仅 IPv4)
|2001:db8::1                           (仅 IPv6)
172.16.1.100|2001:db8::1              (双栈)
```

来源：PGW-C 的 PDN 地址分配 (PAA)

注意：

- 空 IPv4：未分配 IPv4 地址
- 空 IPv6：未分配 IPv6 地址
- 两者均存在：双栈 PDN 连接

---

13. pgw_ip (PGW 控制平面 IP)

类型：字符串（IPv4 或 IPv6 地址）

描述：PGW-C 的 S5/S8 控制平面 IP 地址 (远程 F-TEID)

格式：点分十进制 (IPv4) 或冒号-十六进制 (IPv6)

示例：

```
10.0.0.20        (IPv4)
2001:db8::20     (IPv6)
```

来源：从 PGW-C 的创建会话响应中接收

---

14. apn (接入点名称)

类型：字符串（最多 100 个字符）

描述：标识外部网络 (PDN) 的接入点名称

格式：类似 DNS 的标签格式

示例：

```
internet
ims
mms
enterprise.corporate
```

来源：从 MME 的创建会话请求中接收

用途：

- 确定连接到哪个外部网络
- 驱动策略和计费规则
- 可能决定 IP 地址池

---

15. qci (QoS 类标识符)

类型：无符号 8 位整数

描述：QoS 类标识符定义承载的服务质量

范围：1 - 9 (标准化), 128-254 (运营商特定)

标准化 **QCI** 值：

QCI	资源类型	优先级	数据包延迟	数据包丢失	示例服务
1	GBR	2	100 ms	10^-2	对话语音
2	GBR	4	150 ms	10^-3	对话视频
3	GBR	3	50 ms	10^-3	实时游戏
4	GBR	5	300 ms	10^-6	非对话视频
5	Non-GBR	1	100 ms	10^-6	IMS 信令
6	Non-GBR	6	300 ms	10^-6	视频（缓冲）
7	Non-GBR	7	100 ms	10^-3	语音、视频、游戏
8	Non-GBR	8	300 ms	10^-6	视频（缓冲）
9	Non-GBR	9	300 ms	10^-6	默认承载

示例：

```
9  →  默认承载（最佳努力）
```

来源：PGW-C 的承载 QoS 参数

---

16. octets_in (下行流量)

类型：无符号 64 位整数

描述：在下行方向（网络 → UE）传输的字节数

单位：字节

示例：

```
1048576  → 1 MB 下行
```

来源：来自 SGW-U 的 PFCP 流量测量

注意：

- 对于 `update` 事件是累计的
- 对于 `end` 事件是最终总数
- 对于 `start` 事件始终为 0

---

17. octets_out (上行流量)

类型：无符号 64 位整数

描述：在上行方向（UE → 网络）传输的字节数

单位：字节

示例：

```
524288  → 512 KB 上行
```

来源：来自 SGW-U 的 PFCP 流量测量

注意：

- 对于 `update` 事件是累计的
- 对于 `end` 事件是最终总数
- 对于 `start` 事件始终为 0

示例

示例 1：基本会话与单次更新

时间线：

1. 承载已建立
2. 5 分钟后：使用更新（10 MB 下行，5 MB 上行）
3. 会话终止

CDR 输出：

```
# 数据 CDR 文件：
# 文件开始时间：10:00:00 (1726570800)
# 文件结束时间：11:00:00 (1726574400)
# 网关名称：sgw-c-01
# epoch,imsi,event,charging_id,msisdn,ue_imei,timezone_raw,plmn,tac,e
1726570800,310260111111111,default_bearer_start,10001,15551111111,111
1726571100,310260111111111,default_bearer_update,10001,15551111111,11
1726571400,310260111111111,default_bearer_end,10001,15551111111,11111
```

示例 2：双栈会话与多次更新

时间线：

1. 双栈承载已建立（IPv4 + IPv6）
2. 多次使用更新
3. 会话终止

CDR 输出：

```
1726570800,310260222222222,default_bearer_start,10002,15552222222,222
1726571100,310260222222222,default_bearer_update,10002,15552222222,22
1726571400,310260222222222,default_bearer_update,10002,15552222222,22
1726571700,310260222222222,default_bearer_update,10002,15552222222,22
1726572000,310260222222222,default_bearer_end,10002,15552222222,22222
```

示例 3：专用承载的会话

时间线：

1. 默认承载已建立（QCI 9）
2. 为视频创建专用承载（QCI 6）
3. 两个承载的使用更新
4. 删除专用承载
5. 终止默认承载

CDR 输出：

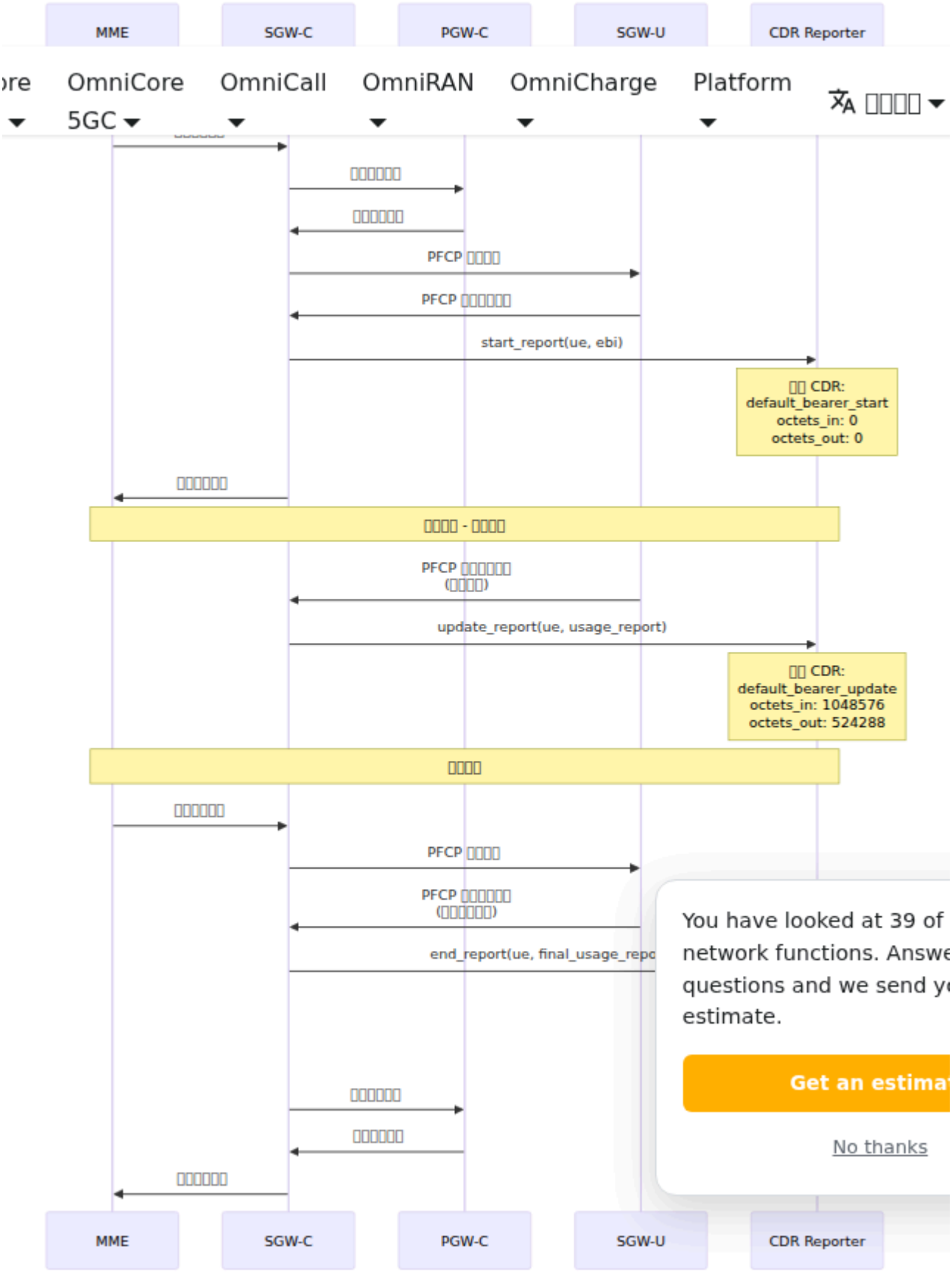
```
1726570800,310260333333333,default_bearer_start,10003,15553333333,333
1726571100,310260333333333,dedicated_bearer_start,10004,15553333333,3
1726571400,310260333333333,default_bearer_update,10003,15553333333,33
1726571400,310260333333333,dedicated_bearer_update,10004,15553333333,
1726571700,310260333333333,dedicated_bearer_end,10004,15553333333,333
1726572000,310260333333333,default_bearer_end,10003,15553333333,33333
```

分析：

- 默认承载 (10003) 承载背景流量 (10 MB 下行，4 MB 上行)
- 专用承载 (10004) 承载视频流量 (200 MB 下行，2 MB 上行)
- 不同的 QCI 值 (9 与 6) 反映不同的 QoS 处理

□□

CDR □□□□


MME
SGW-C
PGW-C
SGW-U
CDR Reporter
ore
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
PFCP
PFCP
start_report(ue, ebi)
CDR:
default_bearer_start
octets_in: 0
octets_out: 0
PFCP
update_report(ue, usage_report)
CDR:
default_bearer_update
octets_in: 1048576
octets_out: 524288
PFCP
PFCP
end_report(ue, final_usage_repo
You have looked at 39 of
network functions. Answe
questions and we send y
estimate.
Get an estima
No thanks
MME
SGW-C
PGW-C
SGW-U
CDR Reporter

CDR 收集方法

1. 基于文件的收集**:**

```
# 监控 CDR 目录 (SGW-C)
inotifywait -m /var/log/sgw_c/cdrs/ -e close_write | while read
path action file; do
    # 文件轮换完成，处理 CDR
    process_cdr "$path$file"
done
```

2. 实时流**:**

```
# 尾随并流式传输到处理管道
tail -F /var/log/sgw_c/cdrs/* | process_cdr_stream
```

相关文档

- 会话管理 - 会话生命周期
- Sxa 接口 - 来自 SGW-U 的使用报告
- 监控指南 - 指标和警报

3GPP 参考

- TS 32.251 - 分组交换 (PS) 域计费
- TS 29.274 - 3GPP 演进分组系统 (EPS); GTP-C 协议
- TS 29.244 - CP 和 UP 节点之间的接口 (PFCP)
- TS 32.298 - CDR 编码

CDR 格式 - *SGW-C □□□□□□□*

□ Omnitouch □□□□□□

□□□□: 1.0 □□□□: 2025-12-10

SGW-C 配置指南

完整的 **runtime.exs** 参考

OmniSGW □ Omnitouch □□□□□□

目录

1. □述
2. 配置结构
3. 指标配置
4. S11 接口配置
5. S5/S8 接口配置
6. Sxa 接口配置
7. CDR 配置
8. 部署示例

概述

所有 OmniSGW 运行时配置通过 `config/runtime.exs` 管理。此文件在启动时加载并控制：

- 网络接口绑定和端口
- 对等连接（MME、PGW-C、SGW-U）
- 指标和监控
- CDR 生成
- 操作参数

配置不会被编译到二进制文件中 - 对 `runtime.exs` 的更改在重启时生效，无需重新编译。

通过 Web UI 配置页面查看当前运行时配置：

□□□□

□□□□

```
# config/runtime.exs
import Config

config :sgw_c,
  metrics: %{ ... },
  s11: %{ ... },
  s5s8: %{ ... },
  sxa: %{ ... },
  cdr: %{ ... }
```

指标配置

基本配置

```
config :sgw_c,
  metrics: %{
    # 指标导出器 HTTP 绑定
    metrics_bind_address: "127.0.0.40",
    metrics_port: 42068,

    # 指标轮询间隔（毫秒）
    poll_interval_ms: 10000
  }
```

生产配置

```
config :sgw_c,
  metrics: %{
    # 绑定到管理网络接口（非公共）
    metrics_bind_address: System.get_env("MGT_IP") || "10.0.0.40",
    metrics_port: 42068,

    # 为响应式仪表板频繁轮询
    poll_interval_ms: 5000
  }
```

访问指标

```
# 以 Prometheus 格式导出指标
curl http://10.0.0.40:42068/metrics

# 常见指标：
# - teid_registry_count: 活动的 S11/S5S8 TEID
# - seid_registry_count: 活动的 PFCP 会话
# - s11_inbound_messages_total: S11 消息计数
# - sxa_inbound_messages_total: Sxa 消息计数
```

有关详细的指标参考、Prometheus 仪表板和警报配置，请参见 监控与指标指南。

S11 接口配置

基本配置

```
config :sgw_c,
  s11: %{
    # S11 的本地 IPv4 地址（MME 接口）
    local_ipv4_address: "10.0.0.10",

    # 可选：本地 IPv6 地址（用于双栈）
    local_ipv6_address: nil,

    # 可选：覆盖默认端口
    local_port: 2123,

    # 消息超时（毫秒）
    message_timeout_ms: 5000,

    # 重试配置
    max_retries: 3,
    retry_backoff_ms: 1000
  }
```

网络接口选择

```
# 单一接口（推荐）
config :sgw_c,
  s11: %{
    local_ipv4_address: "10.0.0.10"  # S11 的单一接口
  }

# 双接口（控制和平面网络分开）
config :sgw_c,
  s11: %{
    local_ipv4_address: "10.0.0.10"     # 控制平面网络
  },
  sxa: %{
    local_ip_address: "10.1.0.20"       # 用户平面网络
  }
```

消息时序配置

```
config :sgw_c,
  s11: %{
    # 对于高延迟网络（> 100ms RTT）
    message_timeout_ms: 10000,
    max_retries: 5,
    retry_backoff_ms: 2000,

    # 对于低延迟网络（< 50ms RTT）
    message_timeout_ms: 3000,
    max_retries: 2,
    retry_backoff_ms: 500
  }
```

S5/S8 接口配置

基本配置

```
config :sgw_c,
  s5s8: %{
    # S5/S8 的本地 IPv4 地址（PGW 接口）
    local_ipv4_address: "10.0.0.15",

    # 可选：本地 IPv6 地址
    local_ipv6_address: nil,

    # 可选：覆盖默认端口
    local_port: 2123,

    # PGW-C 对等体
    pgw_peers: [
      %{
        ip_address: "10.0.0.20",
        name: "pgw-c-primary"
      },
      %{
        ip_address: "10.0.0.21",
        name: "pgw-c-secondary"
      }
    ],

    # 消息超时
    message_timeout_ms: 5000,
    max_retries: 3,
    retry_backoff_ms: 1000
  }
```

PGW 对等体配置

```
# 单一 PGW
config :sgw_c,
  s5s8: %{
    pgw_peers: [
      %{
        ip_address: "10.0.0.20",
        name: "pgw-c-prod"
      }
    ]
  }

# 冗余 PGW（负载均衡）
config :sgw_c,
  s5s8: %{
    pgw_peers: [
      %{ip_address: "10.0.0.20", name: "pgw-c-1"},
      %{ip_address: "10.0.0.21", name: "pgw-c-2"},
      %{ip_address: "10.0.0.22", name: "pgw-c-3"}
    ]
  }

# 冗余 PGW（主动-备用）
config :sgw_c,
  s5s8: %{
    pgw_peers: [
      %{ip_address: "10.0.0.20", name: "pgw-c-primary"},
      %{ip_address: "10.0.0.21", name: "pgw-c-backup"}
    ]
  }
```

Sxa 接口配置

基本配置

```
config :sgw_c,
  sxa: %{
    # Sxa 接口的本地 IP 地址
    local_ip_address: "10.0.0.20",

    # 可选：覆盖默认端口
    local_port: 8805,

    # SGW-U 对等体
    peers: [
      %{
        ip_address: "10.0.0.30",
        node_id: "sgw-u-1.example.com"
      }
    ],

    # 心跳间隔（秒）
    heartbeat_interval_s: 20,

    # 会话超时（毫秒）
    session_timeout_ms: 5000,

    # 重试
    max_retries: 3
  }
```

SGW-U □□□□□

```
# □□ SGW-U
config :sgw_c,
  sxa: %{
    peers: [
      %{
        ip_address: "10.0.0.30",
        node_id: "sgw-u-prod-01"
      }
    ]
  }

# □□ SGW-U
config :sgw_c,
  sxa: %{
    peers: [
      %{
        ip_address: "10.0.0.30",
        node_id: "sgw-u-prod-01"
      },
      %{
        ip_address: "10.0.0.31",
        node_id: "sgw-u-prod-02"
      }
    ]
  }
```

心跳配置

```
# 快速检测（激进）
config :sgw_c,
  sxa: %{
    heartbeat_interval_s: 10,
    max_retries: 2
  }

# 正常检测（平衡）
config :sgw_c,
  sxa: %{
    heartbeat_interval_s: 20,
    max_retries: 3
  }

# 慢速检测（宽松）
config :sgw_c,
  sxa: %{
    heartbeat_interval_s: 40,
    max_retries: 5
  }
```

CDR 配置

基本配置

```
config :sgw_c,
  cdr: %{
    # CDR 中的网关标识符
    gateway_name: "sgw-c-prod-01",

    # 文件轮换间隔（毫秒）
    rotation_interval_ms: 3600000,  # 1 小时

    # CDR 文件的输出目录
    directory: "/var/log/sgw_c/cdrs"
  }
```

生产配置

```
config :sgw_c,
  cdr: %{
    # 使用主机名或部署中的实例 ID
    gateway_name: System.get_env("HOSTNAME") || "sgw-c-prod-01",

    # 每小时轮换以便于管理
    rotation_interval_ms: 3600000,

    # 使用快速存储以存放 CDR
    directory: System.get_env("CDR_DIR") || "/var/log/sgw_c/cdrs"
  }
```

高流量配置

```
config :sgw_c,
  cdr: %{
    gateway_name: "sgw-c-prod-high-vol",

    # 更频繁地轮换以管理文件大小
    rotation_interval_ms: 1800000,  # 30 分钟

    # 使用专用快速存储
    directory: "/mnt/fast-ssd/sgw_c/cdrs"
  }
```

□□□□

□□□□□□□□□

```
import Config

config :sgw_c,
  metrics: %{
    metrics_bind_address: "127.0.0.40",
    metrics_port: 42068,
    poll_interval_ms: 10000
  },
  s11: %{
    local_ipv4_address: "10.0.0.10",
    local_port: 2123,
    message_timeout_ms: 5000,
    max_retries: 3,
    retry_backoff_ms: 1000
  },
  s5s8: %{
    local_ipv4_address: "10.0.0.10",
    pgw_peers: [
      %{ip_address: "10.0.0.20", name: "pgw-c-prod"}
    ],
    message_timeout_ms: 5000,
    max_retries: 3,
    retry_backoff_ms: 1000
  },
  sxa: %{
    local_ip_address: "10.0.0.10",
    peers: [
      %{ip_address: "10.0.0.30", node_id: "sgw-u-prod-01"}
    ],
    heartbeat_interval_s: 20,
    session_timeout_ms: 5000,
    max_retries: 3
  },
  cdr: %{
    gateway_name: "sgw-c-prod-01",
    rotation_interval_ms: 3600000,
```

```
    directory: "/var/log/sgw_c/cdrs"
  }
```

□□□□□□□□□□

```
import Config

sgw_s11_ip = System.get_env("SGW_S11_IP") || "10.0.0.10"
sgw_s5s8_ip = System.get_env("SGW_S5S8_IP") || "10.0.0.15"
sgw_sxa_ip = System.get_env("SGW_SXA_IP") || "10.0.0.20"
mgt_ip = System.get_env("MGT_IP") || "10.0.0.40"

config :sgw_c,
  metrics: %{
    metrics_bind_address: mgt_ip,
    metrics_port: 42068,
    poll_interval_ms: 5000
  },
  s11: %{
    local_ipv4_address: sgw_s11_ip,
    message_timeout_ms: 5000,
    max_retries: 3,
    retry_backoff_ms: 1000
  },
  s5s8: %{
    local_ipv4_address: sgw_s5s8_ip,
    pgw_peers: [
      %{ip_address: "10.0.0.20", name: "pgw-c-1"},
      %{ip_address: "10.0.0.21", name: "pgw-c-2"},
      %{ip_address: "10.0.0.22", name: "pgw-c-3"}
    ],
    message_timeout_ms: 5000,
    max_retries: 3,
    retry_backoff_ms: 1000
  },
  sxa: %{
    local_ip_address: sgw_sxa_ip,
    peers: [
      %{ip_address: "10.0.0.30", node_id: "sgw-u-1"},
      %{ip_address: "10.0.0.31", node_id: "sgw-u-2"},
      %{ip_address: "10.0.0.32", node_id: "sgw-u-3"}
    ],
    heartbeat_interval_s: 20,
    session_timeout_ms: 5000,
    max_retries: 3
  },
```

```
  cdr: %{
    gateway_name: System.get_env("HOSTNAME") || "sgw-c-prod-01",
    rotation_interval_ms: 3600000,
    directory: "/var/log/sgw_c/cdrs"
  }
```

高流量承载级

```
import Config

# 从环境加载所有设置（生产中必需）
sgw_s11_ip = System.fetch_env!("SGW_S11_IP")
sgw_s5s8_ip = System.fetch_env!("SGW_S5S8_IP")
sgw_sxa_ip = System.fetch_env!("SGW_SXA_IP")
mgt_ip = System.fetch_env!("MGT_IP")
hostname = System.get_env("HOSTNAME")

# 从环境解析 PGW 对等体（JSON 格式）
pgw_peers_env = System.get_env("PGW_PEERS", "[]")
{:ok, pgw_peers} = Jason.decode(pgw_peers_env, keys: :atoms)

# 从环境解析 SGW-U 对等体
sgwu_peers_env = System.get_env("SGWU_PEERS", "[]")
{:ok, sgwu_peers} = Jason.decode(sgwu_peers_env, keys: :atoms)

config :sgw_c,
  metrics: %{
    metrics_bind_address: mgt_ip,
    metrics_port: 42068,
    poll_interval_ms: 5000
  },
  s11: %{
    local_ipv4_address: sgw_s11_ip,
    message_timeout_ms: 5000,
    max_retries: 3,
    retry_backoff_ms: 1000
  },
  s5s8: %{
    local_ipv4_address: sgw_s5s8_ip,
    pgw_peers: pgw_peers,
    message_timeout_ms: 5000,
    max_retries: 3,
    retry_backoff_ms: 1000
  },
  sxa: %{
    local_ip_address: sgw_sxa_ip,
    peers: sgwu_peers,
    heartbeat_interval_s: 20,
    session_timeout_ms: 5000,
```

```
    max_retries: 3
  },
  cdr: %{
    gateway_name: hostname,
    rotation_interval_ms: 1800000,  # 30 分钟轮换
    directory: "/mnt/fast-ssd/sgw_c/cdrs"
  }
```

环境变量参考

必需变量

变量	描述	示例
`SGW_S11_IP`	S11 接口 IP	`10.0.0.10`
`SGW_S5S8_IP`	S5/S8 接口 IP	`10.0.0.15`
`SGW_SXA_IP`	Sxa 接口 IP	`10.0.0.20`
`MGT_IP`	指标绑定地址	`10.0.0.40`

可选变量

变量	描述	默认值
`HOSTNAME`	CDR 的网关名称	系统主机名
`PGW_PEERS`	PGW 对等体的 JSON 数组	`[]`
`SGWU_PEERS`	SGW-U 对等体的 JSON 数组	`[]`
`CDR_DIR`	CDR 输出目录	`/var/log/sgw_c/cdrs`

示例部署

```
export SGW_S11_IP="10.0.0.10"
export SGW_S5S8_IP="10.0.0.15"
export SGW_SXA_IP="10.0.0.20"
export MGT_IP="10.0.0.40"
export HOSTNAME="sgw-c-prod-01"
export PGW_PEERS='[{"ip_address":"10.0.0.20","name":"pgw-c-1"}]'
export SGWU_PEERS='[{"ip_address":"10.0.0.30","node_id":"sgw-u-
1"}]'

mix run --no-halt
```

验证

启动时检查配置

监控启动日志:

```
mix run --no-halt 2>&1 | grep -E "S11|S5/S8|Sxa|Metrics"

# 预期输出:
# [info] Starting SGW-C...
# [info] Starting Metrics Exporter on 10.0.0.40:42068
# [info] Starting S11 Broker on 10.0.0.10
# [info] Starting S5/S8 Broker on 10.0.0.15
# [info] Starting Sxa Broker on 10.0.0.20
# [info] OmniSGW successfully started
```

验证活动配置

```
# 检查指标是否可访问
curl http://10.0.0.40:42068/metrics | head -20

# 验证 S11 端口是否在监听
netstat -an | grep 2123

# 检查日志中的 S11 对等体连接
tail -f /var/log/sgw_c/sgw_c.log | grep "S11"
```

常见配置问题

"地址已在使用中"

问题：启动时端口绑定失败

解决方案：

```
# 查找使用端口的进程
lsof -i :2123

# 杀死现有进程或使用不同的端口
killall sgw_c
# 或
config :sgw_c, s11: %{local_port: 2124}
```

"连接被拒绝" 到 PGW

问题： S5/S8 无法到达 PGW-C

解决方案：

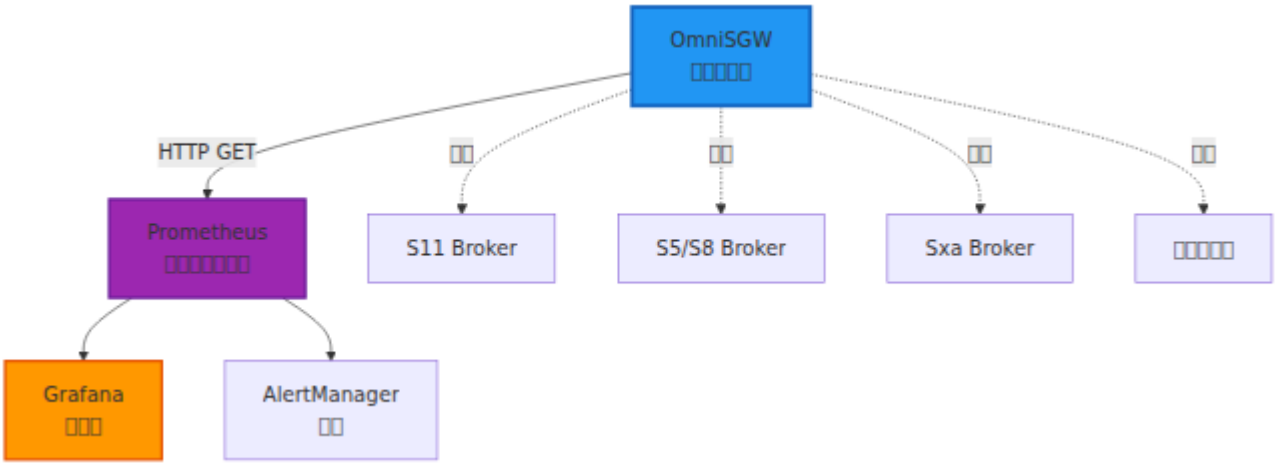
```
# 验证 PGW IP
ping 10.0.0.20

# 检查防火墙规则
iptables -L | grep 2123

# 测试连接性
nc -u -v 10.0.0.20 2123
```

"无法到达 SGW-U"

问题: Sxa 关联失败

解决方案:

```
# 验证 SGW-U 是否可达
ping 10.0.0.30

# 检查 PFCP 端口
netstat -an | grep 8805

# 验证 PFCP 端口是否开放
iptables -L | grep 8805
```



监控与指标指南

Prometheus 指标、**Grafana** 仪表板和警报

OmniSGW □ Omnitouch □□□□□□

目录

1. □述
2. 指标导出器
3. 可用指标
4. Prometheus 配置
5. Grafana 仪表板
6. 警报规则
7. 故障排除

概述

OmniSGW 提供与 Prometheus 兼容的指标，以全面监控网络操作、会话管理和系统健康。

指标架构

指标导出器

访问指标

指标在配置的 HTTP 端点上公开:

```
# 默认端点（如果已配置）
curl http://127.0.0.40:42068/metrics

# 导出到文件
curl http://127.0.0.40:42068/metrics > metrics.txt

# 实时监控
watch -n 5 'curl -s http://127.0.0.40:42068/metrics | head -30'
```

有关指标端点配置（绑定地址、端口和轮询间隔），请参见 配置指南。

指标格式

指标采用 Prometheus 文本格式:

```
# HELP teid_registry_count 分配的 TEID 总数
# TYPE teid_registry_count gauge
teid_registry_count 1234

# HELP s11_inbound_messages_total 入站 S11 消息的总数
# TYPE s11_inbound_messages_total counter
s11_inbound_messages_total{message_type="create_session_request"}
5432
s11_inbound_messages_total{message_type="delete_session_request"}
5100
s11_inbound_messages_total{message_type="modify_bearer_request"}
12000
```

可用指标

会话管理指标

活动会话:

```
teid_registry_count
├── 描述: 活动 S11/S5S8 TEID 分配
├── 类型: Gauge
├── 范围: 0 到最大许可容量
└── 示例: 1234 (1234 个活动会话)

seid_registry_count
├── 描述: 活动 PFCP 会话(每个 SGW-U 对等体)
├── 类型: Gauge
├── 标签: peer_ip
└── 示例: seid_registry_count{peer_ip="10.0.0.30"} 1234

active_ue_sessions
├── 描述: 总活动 UE 会话
├── 类型: Gauge
└── 示例: 5000

active_bearers
├── 描述: 总活动承载(默认 + 专用)
├── 类型: Gauge
└── 示例: 5500 (1 默认 + 每个会话 0.1 专用)

charging_id_registry_count
├── 描述: 活动计费 ID
├── 类型: Gauge
└── 示例: 5000
```

消息计数器

S11 (MME 接口):

```
s11_inbound_messages_total
├── 类型: Counter (递增)
├── 标签: message_type
├── 值:
│   ├── create_session_request
│   ├── delete_session_request
│   ├── modify_bearer_request
│   ├── create_bearer_request
│   ├── delete_bearer_request
│   ├── release_access_bearers_request
│   ├── downlink_data_notification
│   └── echo_request
└── 示例:
s11_inbound_messages_total{message_type="create_session_request"}
5432
```

S5/S8 (PGW 接口):

```
s5s8_inbound_messages_total
├── 类型: Counter
├── 标签: message_type
├── 值: (与 S11 请求类型相同)
└── 示例:
s5s8_inbound_messages_total{message_type="create_session_request"}
4500
```

Sxa (SGW-U 接口):

```
sxa_inbound_messages_total
├── 类型: Counter
├── 标签: message_type
├── 值:
│   ├── session_establishment_request
│   ├── session_modification_request
│   ├── session_deletion_request
│   ├── session_report_request
│   ├── association_setup_request
│   └── heartbeat_request
└── 示例:
sxa_inbound_messages_total{message_type="session_report_request"}
67000
```

性能指标

消息延迟:

```
s11_inbound_duration_seconds
├── 类型: Histogram (带桶)
├── 描述: S11 消息处理时间
├── 百分位: _count, _sum, _bucket
└── 示例: s11_inbound_duration_seconds_bucket{le="0.1"} 5000

s5s8_inbound_duration_seconds
├── 类型: Histogram
├── 描述: S5/S8 消息处理时间

sxa_inbound_duration_seconds
├── 类型: Histogram
├── 描述: Sxa 消息处理时间
```

PFCP 关联:

```
pfcp_association_status
├── 类型: Gauge
├── 值: 1 (已关联) 或 0 (未关联)
├── 标签: peer_ip, node_id
└── 示例: pfcp_association_status{peer_ip="10.0.0.30"} 1

pfcp_heartbeat_latency_ms
├── 类型: Gauge
├── 描述: 心跳往返时间
├── 标签: peer_ip
└── 示例: pfcp_heartbeat_latency_ms{peer_ip="10.0.0.30"} 15
```

错误指标

协议错误:

```
s11_inbound_errors_total
├── 类型: Counter
├── 标签: error_type
├── 值:
│   ├── parse_error
│   ├── validation_error
│   ├── timeout
│   └── other
└── 示例: s11_inbound_errors_total{error_type="timeout"} 12

s5s8_inbound_errors_total
├── 类型: Counter
├── 描述: S5/S8 错误

sxa_inbound_errors_total
├── 类型: Counter
├── 描述: Sxa 错误
```

创建会话失败:

```
create_session_response_cause
├── 类型: Counter
├── 标签: cause_code
├── 值: (3GPP 原因代码)
├── 示例:
│   ├── cause_code="0": 成功
│   ├── cause_code="16": 无可用资源
│   ├── cause_code="25": 语义错误
│   └── cause_code="49": 无匹配规则
```

Prometheus 配置

安装

```
# 下载 Prometheus
wget
https://github.com/prometheus/prometheus/releases/download/v2.45.0/pr
2.45.0.linux-amd64.tar.gz
tar xzf prometheus-2.45.0.linux-amd64.tar.gz
cd prometheus-2.45.0.linux-amd64
```

配置文件

prometheus.yml:

```yaml
global:
  scrape_interval: 15s
  evaluation_interval: 15s
  external_labels:
    monitor: 'sgw-c-prod'

scrape_configs:
  - job_name: 'sgw-c'
    static_configs:
      - targets: ['127.0.0.40:42068']
        labels:
          instance: 'sgw-c-prod-01'

  - job_name: 'sgw-c-backup'
    static_configs:
      - targets: ['127.0.0.41:42068']
        labels:
          instance: 'sgw-c-prod-02'

alerting:
  alertmanagers:
    - static_configs:
        - targets: ['127.0.0.50:9093']
```

□□ Prometheus

```
./prometheus --config.file=prometheus.yml \
             --storage.tsdb.path=/var/lib/prometheus \
             --web.console.libraries=consoles \
             --web.console.templates=console_templates
```

□□ Prometheus

```
http://localhost:9090
```

Grafana 仪表板

安装

```
# Docker（最简单）
docker run -d \
  --name=grafana \
  -p 3000:3000 \
  -e GF_SECURITY_ADMIN_PASSWORD=admin \
  grafana/grafana
```

添加数据源

1. 打开 **Grafana:** http://localhost:3000
2. 配置 → 数据源
3. 添加 → **Prometheus**
4. **URL:** http://prometheus:9090

仪表板：会话概览

面板：

```
行 1:
├── 活动会话 (Gauge)
├── 活动承载 (Gauge)
├── S11 消息/秒 (图表)
└── S5/S8 消息/秒 (图表)

行 2:
├── Sxa 消息/秒 (图表)
├── S11 延迟 p95 (图表)
├── S5/S8 延迟 p95 (图表)
└── Sxa 延迟 p95 (图表)

行 3:
├── S11 错误/分钟 (图表)
├── S5/S8 错误/分钟 (图表)
├── Sxa 错误/分钟 (图表)
└── PFCP 关联 (状态)
```

仪表板：接口健康

面板：

```
行 1:
├── S11 对等体状态 (状态)
├── S5/S8 对等体状态 (状态)
├── SGW-U 对等体状态 (状态列表)
└── 系统负载 (Gauge)

行 2:
├── S11 消息速率 (图表)
├── S5/S8 消息速率 (图表)
├── Sxa 消息速率 (图表)
└── 错误速率 (图表)

行 3:
├── 消息延迟直方图 (热图)
├── 会话创建速率 (图表)
├── 会话终止速率 (图表)
└── 承载创建速率 (图表)
```

仪表板：容量规划

面板：

```
行 1:
├── 会话与容量 (Gauge + 阈值)
├── 承载与容量 (Gauge + 阈值)
├── PFCP 会话分布 (柱状图)
└── 按 APN 的会话 (饼图)

行 2:
├── 会话增长趋势 (图表)
├── 承载增长趋势 (图表)
├── 峰值会话时间 (热图)
└── 会话持续时间分布 (直方图)
```

仪表板查询示例

活动会话：

```
teid_registry_count
```

会话创建速率：

```
rate(s11_inbound_messages_total{message_type="create_session_request'
[5m])
```

S11 延迟（**95th** 百分位）：

```
histogram_quantile(0.95,
rate(s11_inbound_duration_seconds_bucket[5m]))
```

错误速率：

```
rate(s11_inbound_errors_total[5m]) +
rate(s5s8_inbound_errors_total[5m]) +
rate(sxa_inbound_errors_total[5m])
```

PFCP 关联状态:

```
pfcp_association_status{peer_ip=~"10.0.0.3[0-2]"}
```

警报规则

警报规则文件

sgw-c-alerts.yml:

```yaml
groups:
  - name: sgw-c-alerts
    interval: 30s
    rules:
      # 话容量警报
      - alert: SGWCapacityHigh
        expr: (teid_registry_count / 100000) > 0.8
        for: 5m
        annotations:
          summary: "SGW 会话容量超过 80%"
          description: "会话: {{ $value }} / 100000"

      # 接口健康警报
      - alert: S11PeerDown
        expr: absent(s11_inbound_messages_total) > 0
        for: 2m
        annotations:
          summary: "S11 接口不可达"

      - alert: PGWPeerDown
        expr: create_session_response_cause{cause_code="49"} > 100
        for: 2m
        annotations:
          summary: "PGW-C 对等体不可达"

      - alert: SGWUAssociationDown
        expr: pfcp_association_status == 0
        for: 1m
        annotations:
          summary: "SGW-U 关联丢失"
          description: "对等体: {{ $labels.peer_ip }}"

      # 消息延迟警报
      - alert: S11LatencyHigh
        expr: histogram_quantile(0.95,
rate(s11_inbound_duration_seconds_bucket[5m])) > 1
        for: 5m
        annotations:
          summary: "S11 延迟超过 1 秒"
          description: "p95: {{ $value }}s"

      - alert: S5S8LatencyHigh
        expr: histogram_quantile(0.95,
```

```
rate(s5s8_inbound_duration_seconds_bucket[5m])) > 1
        for: 5m
        annotations:
          summary: "S5/S8 延迟超过 1 秒"

      # 错误率警报
      - alert: S11ErrorRate
        expr: rate(s11_inbound_errors_total[5m]) > 10
        for: 3m
        annotations:
          summary: "高 S11 错误率"
          description: "{{ $value }} errors/sec"

      - alert: SessionEstablishmentFailure
        expr: rate(create_session_response_cause{cause_code!="0"}
[5m]) > 20
        for: 3m
        annotations:
          summary: "高会话建立失败率"
          description: "{{ $value }} failures/sec"
```

配置 AlertManager

alertmanager.yml:

```yaml
global:
  resolve_timeout: 5m

route:
  receiver: 'sgw-alerts'
  group_by: ['alertname', 'instance']
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 12h

receivers:
  - name: 'sgw-alerts'
    webhook_configs:
      - url: 'http://slack-webhook-url'
    email_configs:
      - to: 'noc@example.com'
        from: 'sgw-alerts@example.com'
        smarthost: 'smtp.example.com:587'
```

警报通知示例

Slack 集成:

```
 SGW 容量高
严重性： 警告
活动会话: 85,000 / 100,000 (85%)
时间: 2025-12-10 15:30:00 UTC
操作： 监控容量增加
```

电子邮件集成:

```
主题: [警报] S11 对等体不可达

SGW-C S11 接口在 2 分钟内未接收到消息。
这可能表明:
- MME 网络连接问题
- 需要重□ SGW-C
- S11 端口配置已更改

立即采取行动: 检查 S11 状态
```

故障排除

指标未出现

问题： 指标端点为空或 404

诊断：

```
# 检查指标端点是否可达
curl -v http://127.0.0.40:42068/metrics

# 检查日志中的指标导出器错误
tail -f /var/log/sgw_c/sgw_c.log | grep -i metric

# 验证配置
cat config/runtime.exs | grep metrics
```

解决方案：

1. 重启 SGW-C 进程
2. 验证指标 IP/端口未被防火墙阻塞
3. 检查绑定地址配置
4. 确保有足够的内存用于指标收集

特定接口缺少指标

问题：S11 指标显示但 S5/S8 或 Sxa 缺失

诊断：

1. 检查接口是否已配置
2. 验证接口是否处于活动状态
3. 监控日志以查找连接错误

解决方案：

- 验证对等体可达性
- 检查接口绑定
- 审查配置

高内存使用

问题：指标导出器消耗过多内存

诊断：

```
# 检查进程内存
ps aux | grep sgw_c | grep -v grep | awk '{print $6}'

# 监控随时间增长
watch -n 5 'ps aux | grep sgw_c'
```

解决方案：

1. 减少指标轮询间隔
2. 限制指标样本数量
3. 实施指标保留策略
4. 扩展到多个实例

最佳实践

指标收集

- 抓取间隔：15-30 秒以保持平衡
- 保留：15-30 天的指标存储
- 聚合：预聚合高基数指标
- 采样：对延迟使用百分位，而不是原始值

仪表板设计

- 上下文：包括时间范围、实例、对等体信息
- 分层：□览 → 详细 → 调试
- 警报：在容量图上使用可视阈值
- 关联：链接相关指标

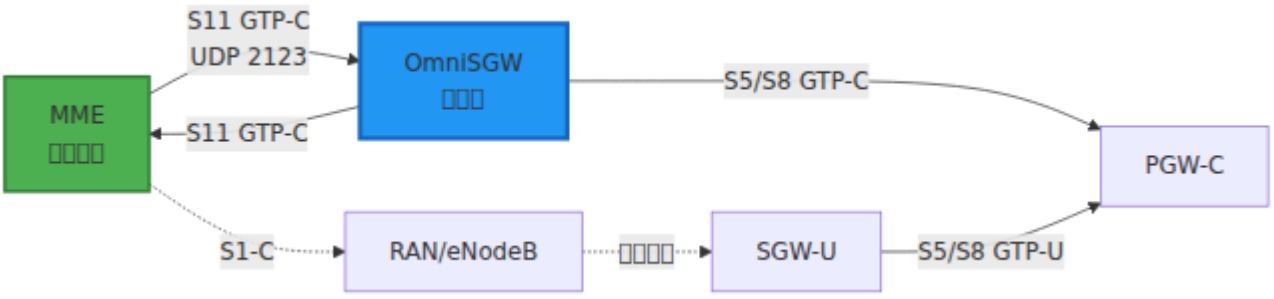
警报策略

- 层级：关键 → 警告 → 信息
- 升级：对关键警报进行呼叫
- 阈值调整：基线然后 +20% 作为警告
- 定期测试：每月测试警报路径

监控指南 - *OmniSGW □□□□□□□*

□ Omnitouch □□□□□□

文档版本：1.0 最后更新：2025-12-10



S11 接口文档

与 **MME** 的 **GTP-C** 通信

OmniSGW □ Omnitouch □□□□□□

目录

1. □述
2. 协议细节
3. 配置
4. 消息类型
5. 会话建立
6. 会话修改
7. 会话终止
8. 网络操作
9. 故障排除

概述

S11 接口 使用 **GTP-C v2** (GPRS 隧道协议 - 控制面) 协议将 OmniSGW 连接到 MME (移动管理实体)。该接口处理所有 UE 会话管理、承载操作和移动程序的控制面信令。

主要特性

- **GTP-C v2** 协议 - 符合标准的消息信令
- 基于 **TEID** 的路由 - 用于会话跟踪的隧道端点标识符
- 有状态会话管理 - 在消息之间保持 UE 上下文
- 切换支持 - 协调 MME 之间和 MME 内部的移动
- 承载□作 - 创建、修改和删除承载
- 下行数据通知 - 对挂起会话进行寻呼

协议细节

GTP-C 版本 **2**

- 协议：GTP-C v2 (3GPP TS 29.274)
- 传输：UDP
- 端口：2123（标准）
- 接口类型：控制面
- 方向：双向请求/响应

TEID（隧道端点标识符）

每个会话都有唯一的 TEID 用于路由消息：

- 本地 **TEID** - 由 OmniSGW 为来自 MME 的入站消息分配
- 远程 **TEID** - 由 MME 为发往 MME 的出站消息分配

```
消息路由：
  MME → SGW：在消息头中使用 OmniSGW 的本地 TEID
  SGW → MME：在消息头中使用 MME 的远程 TEID
```

消息格式

所有 S11 消息遵循 GTP-C v2 格式：

```
GTP-C 头部 (12-16 字节)
├── 版本 (3 位): 0x2 (GTP-C v2)
├── 夹带标志 (1 位)
├── TEID 标志 (1 位): 1 (TEID 存在)
├── 消息类型 (8 位): 标识消息类型
├── 消息长度 (16 位): 消息内容的长度
├── TEID (32 位): 隧道端点标识符
├──  序列号 (24 位): 用于请求/响应匹配
└── 备用 (8 位): 始终为 0

消息内容 (可变)
├── 信息元素 (IE)
│   ├── IE 类型 (8 位)
│   ├── 长度 (16 位)
│   └── 值 (可变)
└── ... 更多 IE
```

配置

基本配置

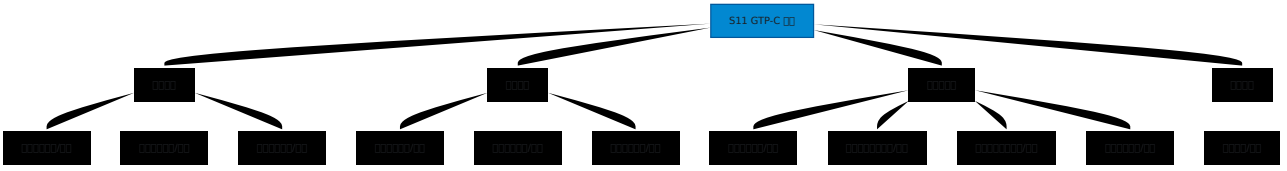
```
# config/runtime.exs
config :sgw_c,
  s11: %{
    # S11 接口的本地 IPv4 地址
    local_ipv4_address: "10.0.0.10",

    # 可选：本地 IPv6 地址（用于双栈）
    local_ipv6_address: nil,

    # 可选：覆盖默认端口
    local_port: 2123,

    # 消息超时
    message_timeout_ms: 5000,

    # 重试配置
    max_retries: 3,
    retry_backoff_ms: 1000
  }
```

网络要求

防火墙规则：

```
# 允许来自 MME 网络的 GTP-C（入站）
iptables -A INPUT -p udp --dport 2123 -s <mme_network>/24 -j
ACCEPT

# 允许发往 MME 的出站 GTP-C
iptables -A OUTPUT -p udp --dport 2123 -d <mme_network>/24 -j
ACCEPT
```

路由：



```
# 确保到 MME 网络的路由
ip route add <mme_network>/24 via <gateway_ip> dev eth0
```

网络测试：

```
# 测试与 MME 的连通性（使用 GTP 心跳）
# 检查日志中的 "S11 Broker connected" 消息

# 监控活动的 S11 会话
curl http://127.0.0.40:42068/metrics | grep teid_registry_count
```

消息类型

S11 消息概述

会话建立消息

创建会话请求 **(S11)**

方向： MME → OmniSGW

目的：建立新的 UE 会话（初始附着或 PDN 连接）

关键信息元素：

IE 名称	类型	描述
IMSI	二进制	国际移动用户身份
MSISDN	BCD	手机号码
MEI	二进制	移动设备身份
RAT 类型	枚举	无线接入技术（EUTRAN）
承载上下文	分组	默认承载配置
UE 时区	日期时间	UE 当前时区
ULI	分组	用户位置信息（TAI, ECGI）
服务网络	PLMN	MCC/MNC
APN	字符串	接入点名称

响应：创建会话响应

IE 名称	类型	描述
原因	枚举	请求结果（成功/失败）
承载上下文	分组	分配的承载信息及 TEID
PDN 地址分配	分组	从 PGW 分配的 IP 地址
APN 限制	枚举	APN 限制

会话修改消息

修改承载请求 **(S11)**

方向： MME → OmniSGW（请求由 MME 发起）

目的：在活动会话期间修改承载参数

关键信息元素：

IE 名称	类型	描述
MEI	二进制	移动设备标识符
ULI	分组	更新的用户位置信息
UE 时区	日期时间	更新的时区
TAI	TAI	跟踪区域标识符
ECGI	ECGI	E-UTRAN 小区全局标识符

响应：修改承载响应

IE 名称	类型	描述
原因	枚举	修改结果
承载上下文	分组	更新的承载参数

承载管理消息

创建承载请求/响应

方向：可以由 MME 或 SGW 发起

目的：为需要 QoS 的服务激活专用承��

触发场景：

- 语音服务激活

- 视频流请求
- 在线游戏激活

删除承载请求/响应

方向：可以由 MME 或 SGW（通过 PGW）发起

目的：在不再需要时停用专用承载

移动消息

释放接入承载请求/响应

方向： MME → OmniSGW

目的：在无线断开时挂起所有承载（寻呼场景）

效果：

- 承载保持在上下文中但挂起
- 用户面转发暂停
- 在 SGW-U 中启动数据缓冲
- UE 可以通过服务请求恢复

修改接入承载请求/响应

方向： OmniSGW → MME 或 MME → OmniSGW

目的：在切换或恢复期间更新承载接入

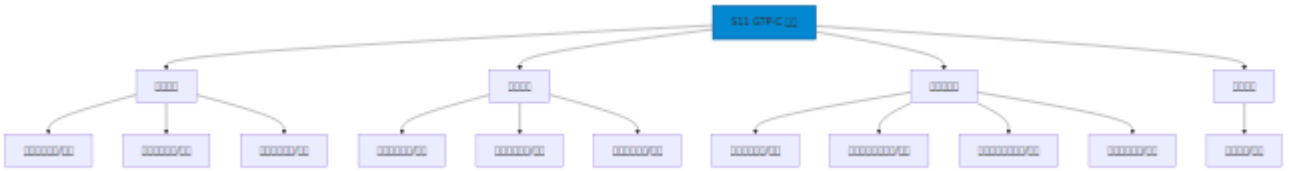
寻呼消息

下行数据通知 **(S11)**

方向： PGW-C → OmniSGW → MME

目的：通知 MME 有待处理的下行数据用于挂起的 UE

关键信息元素：



IE 名称	类型	描述
EBI	整数	EPS 承载 ID
IMSI	二进制	订阅者身份

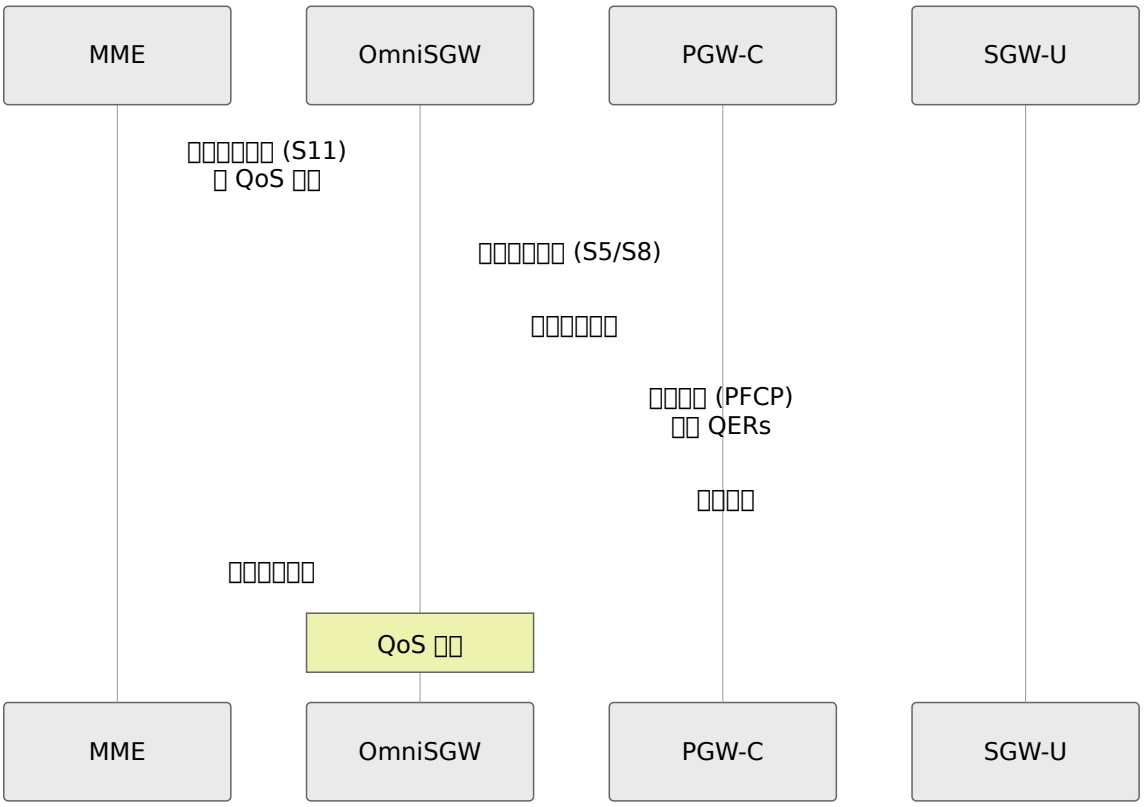
响应：下行数据确认

会话建立

UE 初始附着流程

状态转换：

```
[UE 未连接]
      ↓ (附着请求)
[创建到 PGW 的会话]
      ↓ (PGW 响应)
[建立用户面]
      ↓ (PFCP 会话活动)
[会话活动]
```



会话修改

承载 **QoS** 修改

MME
OmniSGW
PGW-C
SGW-U
修改承载请求 (S11)
新 QoS 参数
修改承载请求 (S5/S8)
修改承载响应
会话修改 (PFCP)
更新 QERs
修改响应
修改承载响应
QoS 更新
MME
OmniSGW
PGW-C
SGW-U

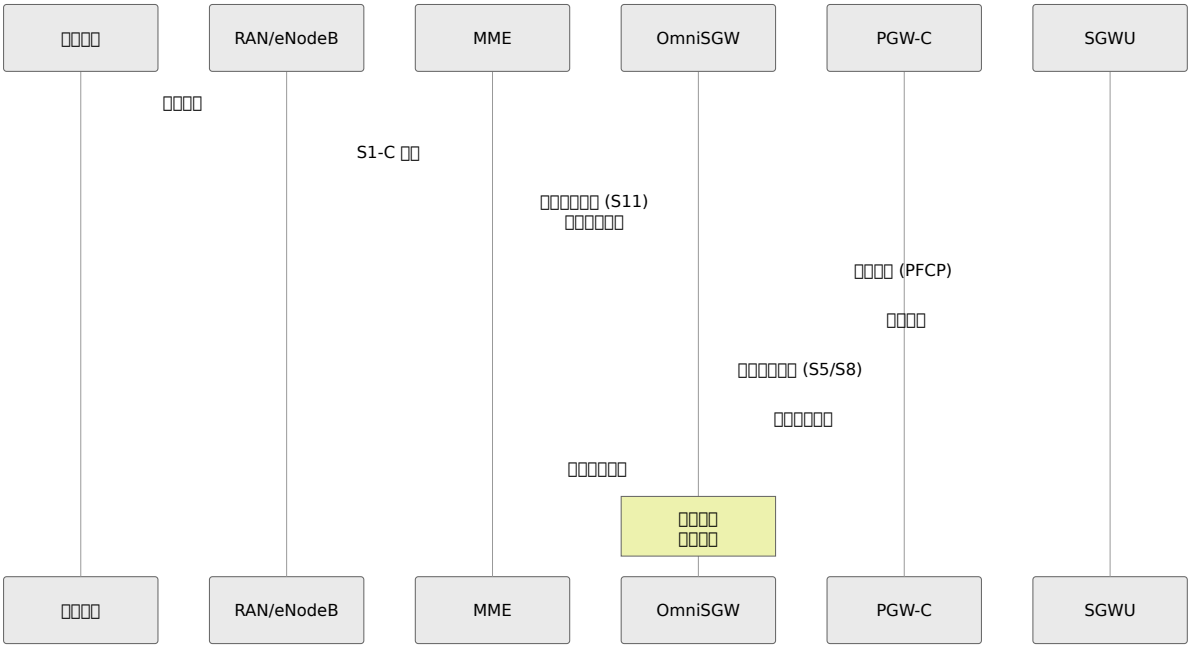
跟踪区域更新 **(TAU)**

无 **SGW** 更改的 **TAU**:

- MME 更新 UE 位置
- ULI/TAI 通过修改承载发送到 SGW
- SGW 更新本地 UE 上下文
- 无需会话迁移

有 **SGW** 更改的 **TAU**:

- 旧 SGW 从 MME 接收释放接入承载
- 新 SGW 接收创建会话请求
- 从旧 SGW 到新 SGW 的数据转发
- 转发完成后，旧 SGW 释放会话



会话终止

正常会话终止

移动设备
RAN/eNodeB
MME
OmniSGW
PGW-C
SGWU
分离请求
S1-C 消息
删除会话请求 (S11)
所有承载标志
会话删除 (PFCP)
删除响应
删除会话请求 (S5/S8)
删除会话响应
删除会话响应
会话终止
资源释放
移动设备
RAN/eNodeB
MME
OmniSGW
PGW-C
SGWU

状态转换:

```
[会话活动]
    ↓ (删除会话请求)
[释放用户面]
    ↓ (PFCP 会话已删除)
[通知 PGW]
    ↓ (PGW 会话已删除)
[会话终止]
```

网络操作

消息流监控

实时监控 S11 消息活动:

```
# 监控 S11 消息计数器
watch -n 1 'curl -s http://127.0.0.40:42068/metrics | grep
s11_inbound'

# 输出示例:
#
s11_inbound_messages_total{message_type="create_session_request"}
1245
#
s11_inbound_messages_total{message_type="delete_session_request"}
1200
# s11_inbound_messages_total{message_type="modify_bearer_request"}
3450
```

会话检查

查看活动会话及其 S11 状态:

```
Web UI → UE 会话页面

对于每个会话:
  - IMSI 和 GUTI
  - 当前 TAI (跟踪区域)
  - 本地 TEID (用于 S11)
  - 远程 TEID (来自 MME)
  - 带有 QoS 参数的承载列表
  - 相关的 PGW-C
```

切换监控

跟踪切换活动:

```
# 计算修改承载请求 (指示切换)
curl -s http://127.0.0.40:42068/metrics | grep
modify_bearer_request_total

# 监控切换延迟
# 检查日志中的 "TAU with SGW change" 消息
```

故障排除

会话建立失败

问题：创建会话请求被拒绝

诊断：

1. 检查 Web UI → UE 会话以获取拒绝原因
2. 验证指标：`s11_inbound_errors_total`
3. 检查日志以获取特定原因代码

常见原因及解决方案：

原因	理由	解决方案
16	无可用资源	检查 SGW-U 容量，PFCP 会话计数
25	IE 中的语义错误	验证请求中的承载上下文
49	PGW 无法访问	检查 S5/S8 与 PGW-C 的连通性
65	不支持的 APN	验证 APN 配置

消息路由问题

问题："消息路由到未知 TEID"

诊断：

```
# 检查 TEID 注册
curl -s http://127.0.0.40:42068/metrics | grep teid_registry_count

# 验证 TEID 分配
# Web UI → UE 会话 → 按 IMSI 搜索
```

常见原因：

- 会话已释放，但延迟消息仍在到达
- 使用不同 TEID 的重复创建会话
- 来自不同 MME 实例的消息，具有相同 TEID

切换问题

问题：切换失败或数据丢失

诊断：

1. 监控指标中的修改承载请求/响应
2. 检查日志中的 "切换" 或 "TAU" 消息
3. 检查切换期间 PFCP 会话状态

解决方案：

- 验证 SGW-U 在切换窗口期间处于活动状态
- 检查已安装的数据转发规则
- 监控释放接入承载的时机

性能问题

问题：高 S11 消息延迟

检查的指标：

```
# 消息处理持续时间
curl -s http://127.0.0.40:42068/metrics | grep
s11_inbound_duration_seconds

# 会话计数
curl -s http://127.0.0.40:42068/metrics | grep active_ue_sessions

# 承载计数
curl -s http://127.0.0.40:42068/metrics | grep active_bearers
```

优化步骤：

1. 减少不必要的修改承载操作
2. 监控并优化 PFCP 会��建立时间
3. 通过多个 SGW-C 实例进行横向扩展
4. 在高峰负载期间检查 CPU 和内存使用情况

有关全面的指标信息、Prometheus 设置和仪表板配置，请参见 监控与指标指南。

最佳实践

配置

- 端口绑定：将 S11 绑定到管理网络接口以提高安全性
- 超时：根据网络 RTT 设置适当的消息超时
- 重试：在可靠性和网络负载之间取得平衡

□作

- 会话限制：监控与容量以防止过载
- 对等监控：跟踪 MME 连接状态
- 错误跟踪：对持续的 S11 错误率增加进行警报
- 优雅关闭：在维护前排空会话

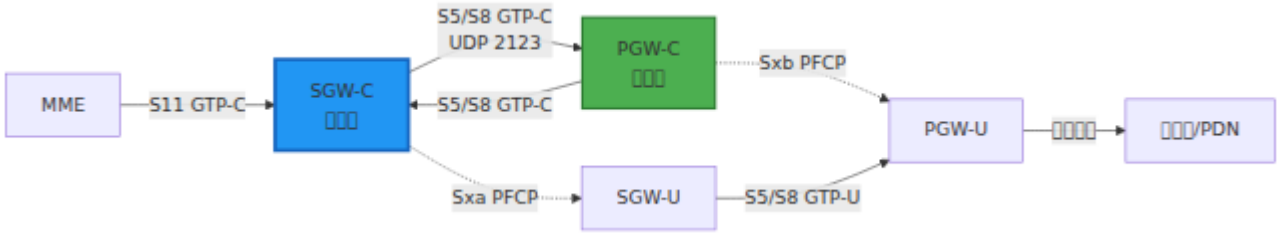
安全

- 网络隔离： S11 应位于隔离的网络段
- 访问控制：限制 S11 端口仅对授权的 MME IP
- 监控：对意外的对等连接进行警报

消息参考摘要

消息	方向	频率	优先级
创建会话请求/响应	MME → SGW	会话创建	高
删除会话请求/响应	MME → SGW	会话结束	高
修改承载请求/响应	MME ↔ SGW	QoS 更改, TAU	中
创建承载请求/响应	MME ↔ SGW	承载激活	中
删除承载请求/响应	MME ↔ SGW	承载停用	中
释放接入承载请求/响应	MME → SGW	寻呼挂起	高
修改接入承载请求/响应	MME ↔ SGW	移动恢复	高
下行数据通知/确认	SGW → MME	数据寻呼	中
回声请求/响应	MME ↔ SGW	路径监控	低

S11 接口 - *MME □ SGW-C □□□□□*



S5/S8 接口文档

与 **PGW-C** 的 **GTP-C** 通信

OmniSGW □ Omnitouch □□□□□□

目录

1. □述
2. 协议细节
3. 配置
4. 会话建立
5. 会话修改
6. 会话终止
7. 消息类型
8. 网络操作
9. 故障排除

概述

S5/S8 接口 使用 **GTP-C v2**（GPRS 隧道协议 - 控制面）协议将 OmniSGW 连接到 PGW-C（数据包网关控制平面）。该接口处理网关之间的 PDN 会话管理信令。

主要特性

- **GTP-C v2** 协议 - 符合标准的信令
- 基于 **TEID** 的会话路由 - 用于跟踪的隧道端点标识符
- **PDN** 连接管理 - 创建/修改/删除 PDN 连接
- 承载管理 - 默认和专用承载操作
- 计费 **ID** 交换 - 跨网关的协调计费
- **IP** 地址分配 - 从 PGW 池中为 UE 提供 IP

协议细节

GTP-C 版本 **2**

- 协议: GTP-C v2 (3GPP TS 29.274)
- 传输: UDP
- 端口: 2123 (标准)
- 接口类型: 控制面
- 方向: 双向请求/响应

TEID（隧道端点标识符）

每个 PDN 会话在两个方向上都有唯一的 TEID:

- **SGW TEID** - 由 SGW-C 为来自 PGW 的 S5/S8 消息分配
- **PGW TEID** - 由 PGW-C 为来自 SGW 的 S5/S8 消息分配

消息流:

```
SGW-C → PGW-C: 在头部使用 PGW-C 的 TEID
PGW-C → SGW-C: 在头部使用 SGW-C 的 TEID
```

计费 **ID**

计费 ID 对于计费协调至关重要:

- 生成者： PGW-C 在创建会话响应期间
- 传递给： SGW-C 以生成 CDR
- 用于：关联 SGW 和 PGW CDR 之间的离线费用
- 格式： 32 位整数，每个 PDN 连接唯一

配置

基本配置

```
# config/runtime.exs
config :sgw_c,
  s5s8: %{
    # S5/S8 接口的本地 IPv4 地址
    local_ipv4_address: "10.0.0.15",

    # 可选：本地 IPv6 地址
    local_ipv6_address: nil,

    # 可选：覆盖默认端口
    local_port: 2123,

    # PGW-C 对等体
    pgw_peers: [
      %{
        ip_address: "10.0.0.20",
        name: "pgw-c-primary"
      },
      %{
        ip_address: "10.0.0.21",
        name: "pgw-c-secondary"
      }
    ],

    # 消息超时
    message_timeout_ms: 5000,
    max_retries: 3,
    retry_backoff_ms: 1000
  }
```

网络要求

防火墙规则：

```
# 允许来自 PGW-C 网络的 GTP-C
iptables -A INPUT -p udp --dport 2123 -s <pgw_network>/24 -j
ACCEPT

# 允许向 PGW-C 的出站 GTP-C
iptables -A OUTPUT -p udp --dport 2123 -d <pgw_network>/24 -j
ACCEPT
```

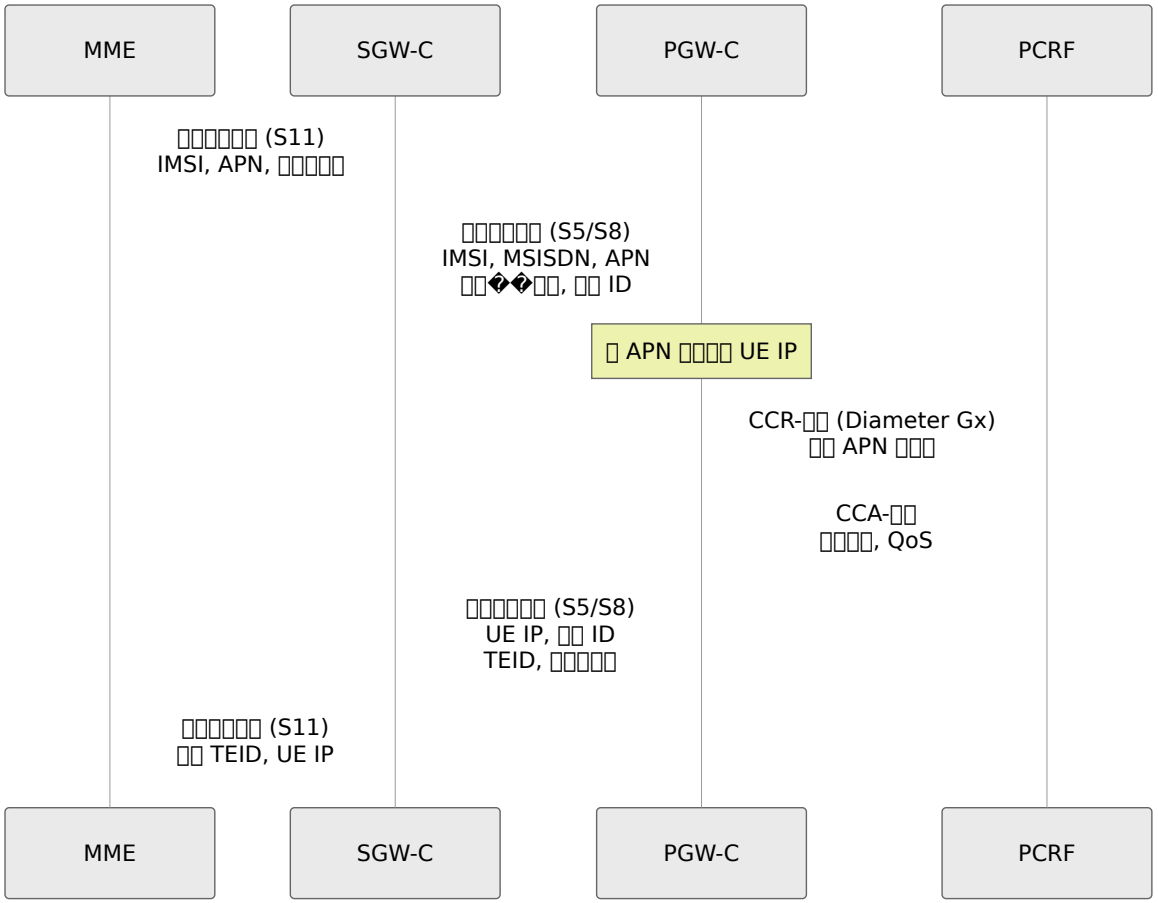
路由：

```
# 确保到 PGW-C 网络的路由
ip route add <pgw_network>/24 via <gateway_ip> dev eth0
```

会话建立

初始 **PDN** 连接请求

当 MME 通过 S11 请求 PDN 连接时，SGW-C 通过 S5/S8 转发到 PGW-C。



MME
SGW-C
PGW-C
PCRF
创建会话请求 (S11)
IMSI, APN, 承载上下文
创建会话请求 (S5/S8)
IMSI, MSISDN, APN
承载��下文, 计费 ID
从 APN 池中分配 UE IP
CCR-初始 (Diameter Gx)
请求 APN 的策略
CCA-初始
策略规则, QoS
创建会话响应 (S5/S8)
UE IP, 计费 ID
TEID, 承载上下文
创建会话响应 (S11)
承载 TEID, UE IP
MME
SGW-C
PGW-C
PCRF

创建会话请求 **(SGW-C → PGW-C)**

关键信息元素：

IE 名称	来源	描述
IMSI	MME	移动用户身份
MSISDN	MME	手机号码
MEI	MME	移动设备身份
承载上下文	MME	承载配置 (QCI, ARP)
APN	MME	接入点名称 (internet, ims, mms)
服务网络	MME	PLMN 代码 (MCC/MNC)
RAT 类型	MME	无线接入技术 (EUTRAN)
ULI	MME	用户位置信息 (TAI, ECGI)
计费 ID	SGW	SGW 生成的计费参考

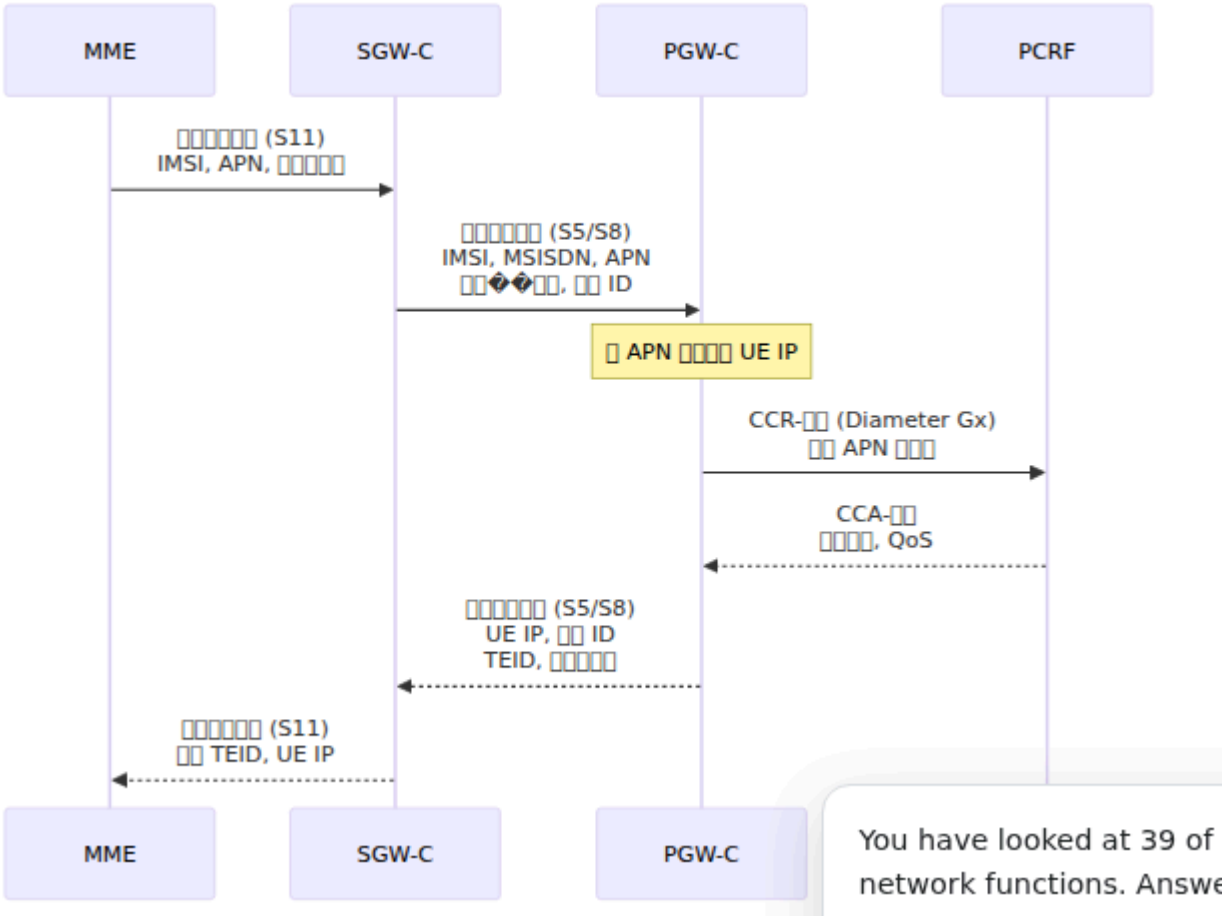
创建会话响应 (PGW-C → SGW-C)

关键信息元素:

IE 名称	来源	描述
原因	PGW	成功/失败指示
承载上下文	PGW	分配的承载与 TEID
PDN 地址分配	PGW	分配的 UE IP 地址
APN 限制	PGW	此 APN 的策略
计费 ID	PGW	PGW 生成的计费 ID
TEID	PGW	为 S5/S8 隧道分配

响应代码

原因代码	���述	恢复
0	请求已接受	会话已建立
16	无可用资源	拒绝给 MME，用户操作
25	IE 中的语义错误	检查消息格式
49	没有匹配规则	PGW-C 策略不匹配
64	找不到上下文	会话已存在
65	响应中的语义错误	PGW 配置错误
72	缺少/不正确的强制 IE	消息不完整



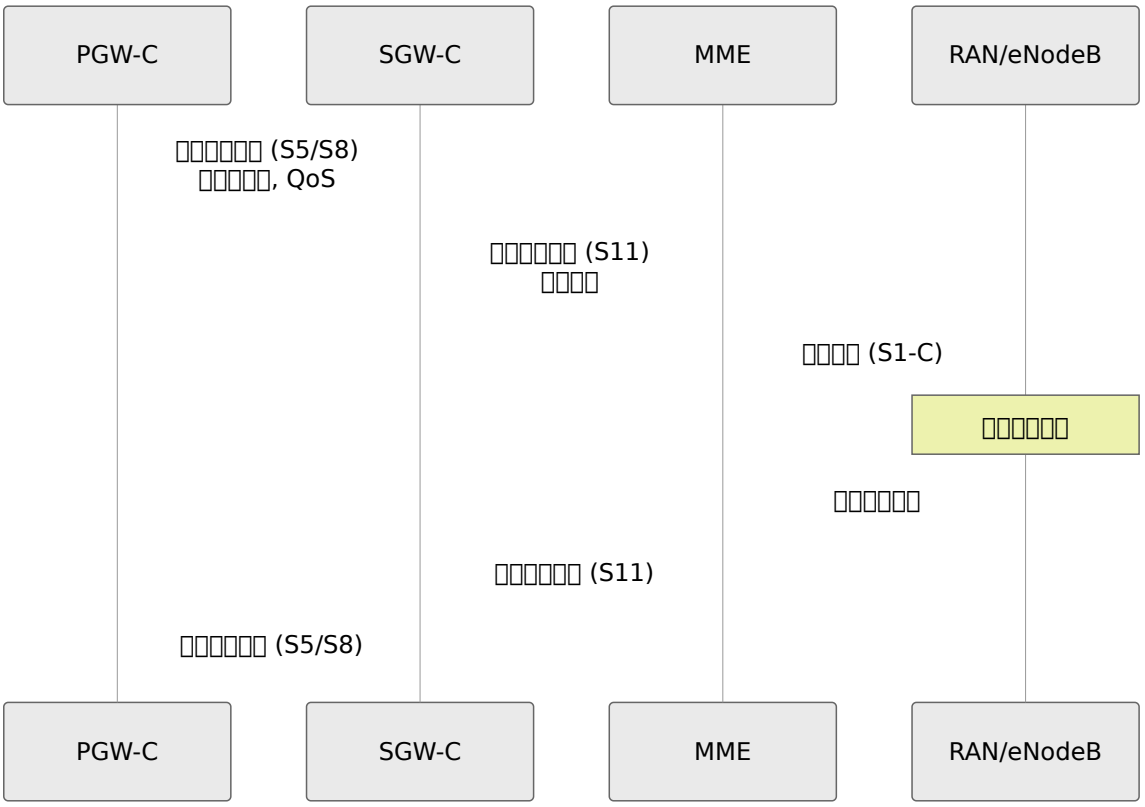
会话修改

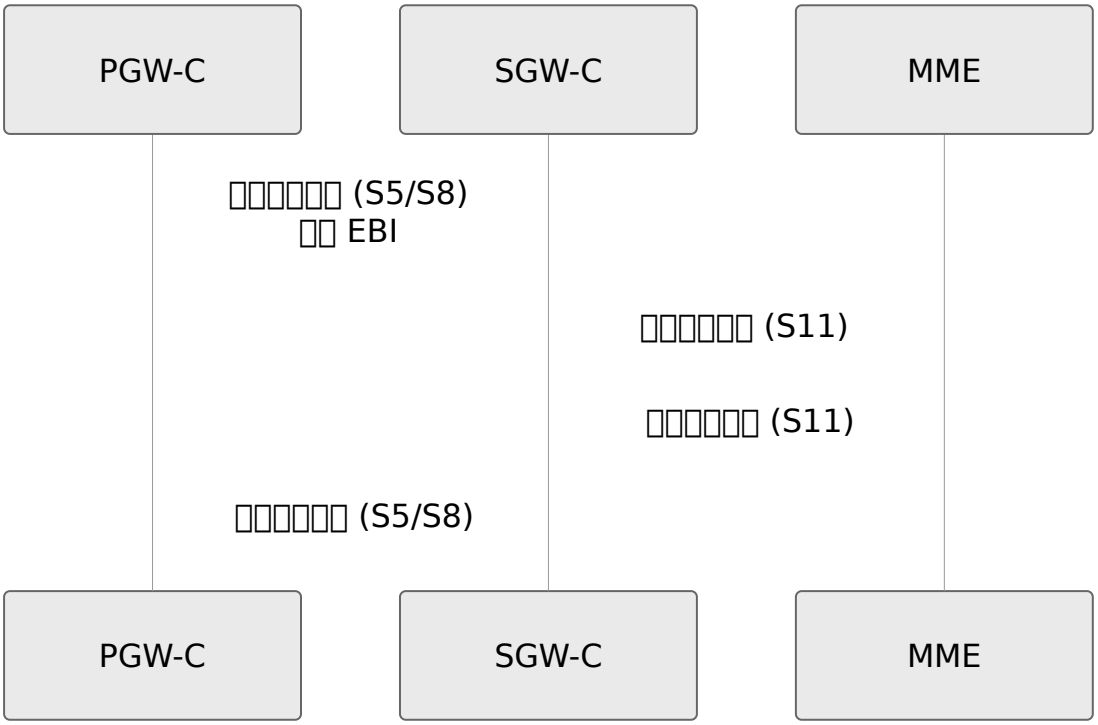
承载 **QoS** 修改

当 MME 通过 S11 请求 QoS 更改时，SGW-C 通过 S5/S8 传播到 PGW-C。

创建承载（专用承载）

PGW-C 可以通过 S5/S8 请求专用承载激活:



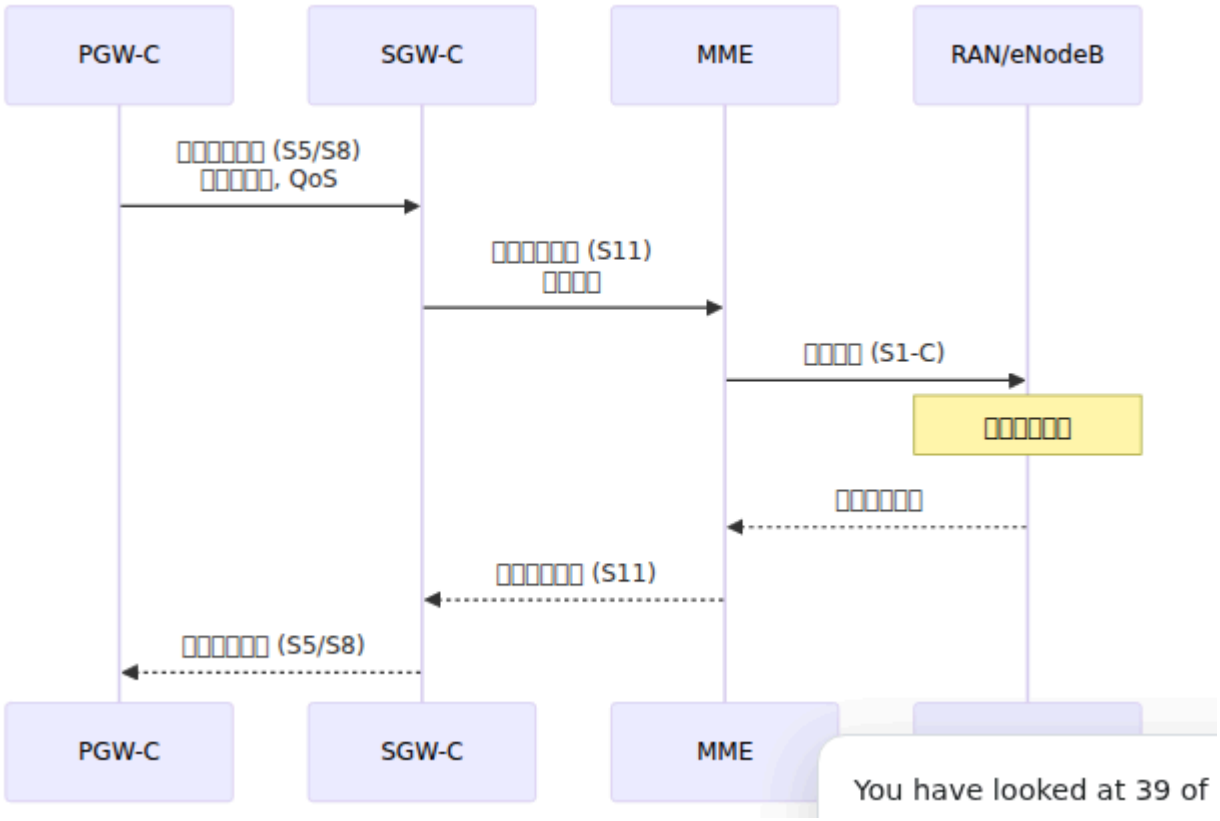


PGW-C
SGW-C
MME
RAN/eNodeB
创建承载请求 (S5/S8)
承载上下文, QoS
创建承载请求 (S11)
承载信息
创建承载 (S1-C)
激活专用路径
创建承载接受
创建承载响应 (S11)
创建承载响应 (S5/S8)
PGW-C
SGW-C
MME
RAN/eNodeB

删除承载（专用承载）

当不再需要专用承载时：

PGW-C
SGW-C
MME
删除承载请求 (S5/S8)
承载 EBI
删除承载请求 (S11)
删除承载响应 (S11)
删除承载响应 (S5/S8)
PGW-C
SGW-C
MME

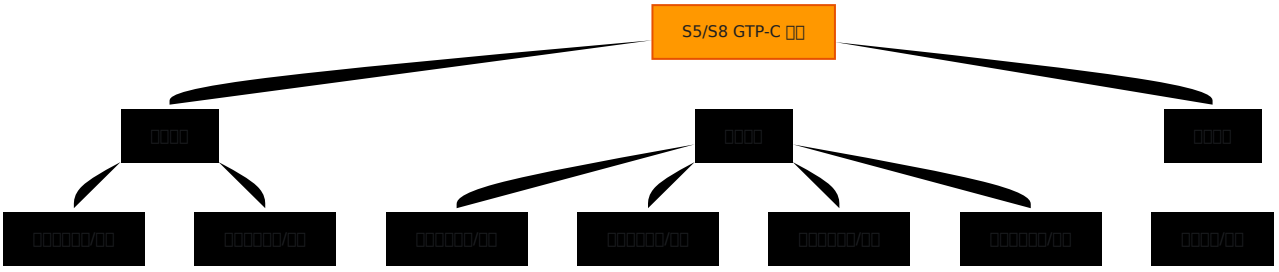


会话终止

正常 **PDN** 断开连接

状态转换：

```
[PDN 连接]
    ↓ (来自 MME 的删除会话请求)
[释放 PGW 会话]
    ↓ (收到 PGW 删除响应)
[释放 SGW 资源]
    ↓ (TEID 释放，CDR 记录)
[PDN 断开]
```



消息类型

S5/S8 消息摘要

消息细节

创建会话请求/响应

- 触发条件：初始附加，PDN 连接请求
- 频率：每个 UE 每个 PDN 连接约 1 次
- 方向：双向

删除会话请求/响应

- 触发条件：脱离，PDN 断开连接
- 频率：每个 PDN 连接终止约 1 次
- 方向：双向

修改承载请求/响应

- 触发条件： QoS 更改，承载修改
- 频率：可变（每个会话 0 到多次）
- 方向：双向

创建/删除承载请求/响应

- 触发条件：专用承载激活/停用
- 频率：可变（每个会话 0 到多次）
- 方向：双向

回声请求/响应

- 触发条件：路径/对等体监控
- 频率[illegible] 定期（建议每分钟至少 1 次）
- 方向：双向

网络[illegible]作

对等体监控

监控 PGW-C 连接性：

```
# 检查活动的 S5/S8 TEID
curl -s http://127.0.0.40:42068/metrics | grep s5s8_teid

# 监控 S5/S8 消息流
curl -s http://127.0.0.40:42068/metrics | grep
s5s8_inbound_messages_total

# 预期：创建/删除/修改消息的稳定流
```

PDN 会话验证

检查活动的 PDN 连接：

```
Web UI → UE 会话页面
└── 对于每个 UE 会话：
    ├── 关联的 PGW-C 对等体
    ├── 计费 ID（来自 PGW）
    ├── UE IP 地址（来自 PGW）
    ├── 带有 QoS 的承载列表
    └── S5/S8 TEID 对
```

消息流检查

跟踪 S5/S8 消息活动：

```
# 计算创建会话操作
curl -s http://127.0.0.40:42068/metrics | grep
create_session_request_total

# 监控承载修改
curl -s http://127.0.0.40:42068/metrics | grep modify_bearer

# 检查错误率
curl -s http://127.0.0.40:42068/metrics | grep
s5s8_inbound_errors_total
```

PGW 选择策略

如果配置了多个 PGW-C 对等体:

```
选择逻辑:
├── 负载均衡: 在对等体之间轮询
├── 粘性: 相同 APN 始终使用相同 PGW
├── 主动-备用: 对等体不可用时故障转移
└── 自定义: 特定于应用的逻辑
```

监控分布:

```
# 每个 PGW 对等体的会话
curl -s http://127.0.0.40:42068/metrics | grep session_by_pgw_peer
```

故障排除

会话建立失败

问题: "PGW 拒绝创建会话请求"

诊断:

```
# 检查原因代码
curl -s http://127.0.0.40:42068/metrics | grep
create_session_response_cause

# 检查 PGW 连接性
curl -s http://127.0.0.40:42068/metrics | grep s5s8_peer_status
```

常见原因及解决方案:

原因	理由	解决方案
16	无资源	检查 PGW 容量，IP 池耗尽
25	语义错误	验证承载上下文是否符合 PGW 预期
49	没有匹配规则	检查 PGW 的 APN 配置
72	缺少 IE	验证 MME 是否发送所需字段

承载□作失败

问题: "修改承载请求失败"

诊断:

1. 检查修改承载的错误率指标
2. 检查 QoS 参数的有效性
3. 验证 PGW 是否可达

解决方案:

- 减少 QoS 修改频率
- 验证 PGW 策略内的 QoS 值
- 检查 PGW 是否存在 PCRF/策略问题

消息超时问题

问题："S5/S8 消息超时"

指标：

```
# 消息延迟
curl -s http://127.0.0.40:42068/metrics | grep
s5s8_inbound_duration_seconds

# 超时计数
curl -s http://127.0.0.40:42068/metrics | grep s5s8_timeout_total
```

解决方案：

- 如果网络 RTT 较高，增加 message_timeout_ms
- 检查网络拥塞
- 验证 PGW CPU/内存可用性
- 监控数据包丢失

计费 ID 不匹配

问题："CDR 中的计费 ID 不匹配"

诊断：

- 验证 PGW 是否返回有效的计费 ID
- 检查 CDR 日志中是否缺少计费 ID
- 比较 SGW 和 PGW CDR

解决方案：

- 确保 PGW 在所有响应中发送计费 ID
- 在 CDR 记录中优雅地处理缺少的计费 ID

有关详细的指标参考和 Prometheus 仪表板设置，请参阅 监控与指标指南。

最佳实践

配置

- **PGW** 冗余：配置多个 PGW-C 对等体以实现故障转移
- 负载分配：使用轮询实现负载均衡
- 超时：根据 WAN RTT 适当设置（典型：5-10 秒）
- 重试： 2-3 次重试，采用指数回退

□作

- 对等体健康：监控回声响应时间
- **APN** 路由：将 SGW APN 配置与 PGW APN 匹配
- 错误跟踪：对持续的 S5/S8 错误率发出警报
- 容量规划：监控 PGW 的 IP 池使用情况

会话管理

- 会话限制：跟踪并发会话与 PGW 容量
- 承载计数：监控默认 + 专用承载分布
- **QoS** 验证：验证 PGW 接受的 QoS 参数
- 计费：验证收到并记录的计费 ID

与其他接口的集成

S11 ↔ S5/S8 协调

```
S11 消息流（来自 MME）
     ↓
SGW-C 会话处理
     ↓
S5/S8 消息（发送到 PGW-C）
     ↓
等待响应
     ↓
S11 响应（返回给 MME）
```

S5/S8 ↔ Sxa 协调

```
S5/S8 创建会话响应（来自 PGW）
     ↓
提取承载/QoS 信息
     ↓
Sxa 会话建立（发送到 SGW-U）
     ↓
等待用户平面准备就绪
     ↓
完成 S11 创建会话响应
```

会话管理指南

UE 会话生命周期和□作

OmniSGW □ Omnitouch □□□□□□

目录

1. □述
2. 会话生命周期
3. 会话状态
4. 承载操作
5. 移动性处理
6. 切换程序
7. 操作程序
8. 会话检查
9. 故障排除

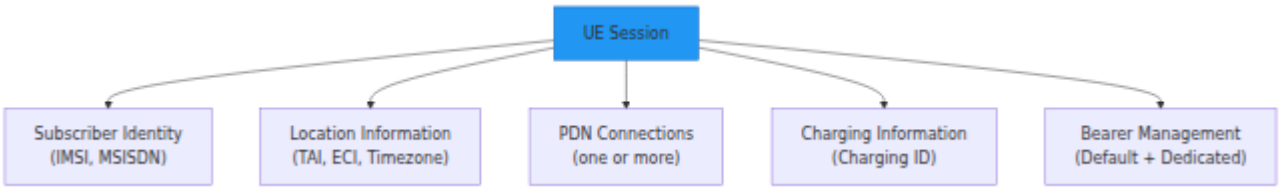
概述

UE 会话表示一个连接到网络的活动移动设备。SGW-C 维护会话上下文并在以下实体之间进行协调：

- **MME** - 移动管理实体（通过 S11）
- **PGW-C** - 数据包网关控制平面（通过 S5/S8）
- **SGW-U** - 用户平面转发（通过 Sxa）

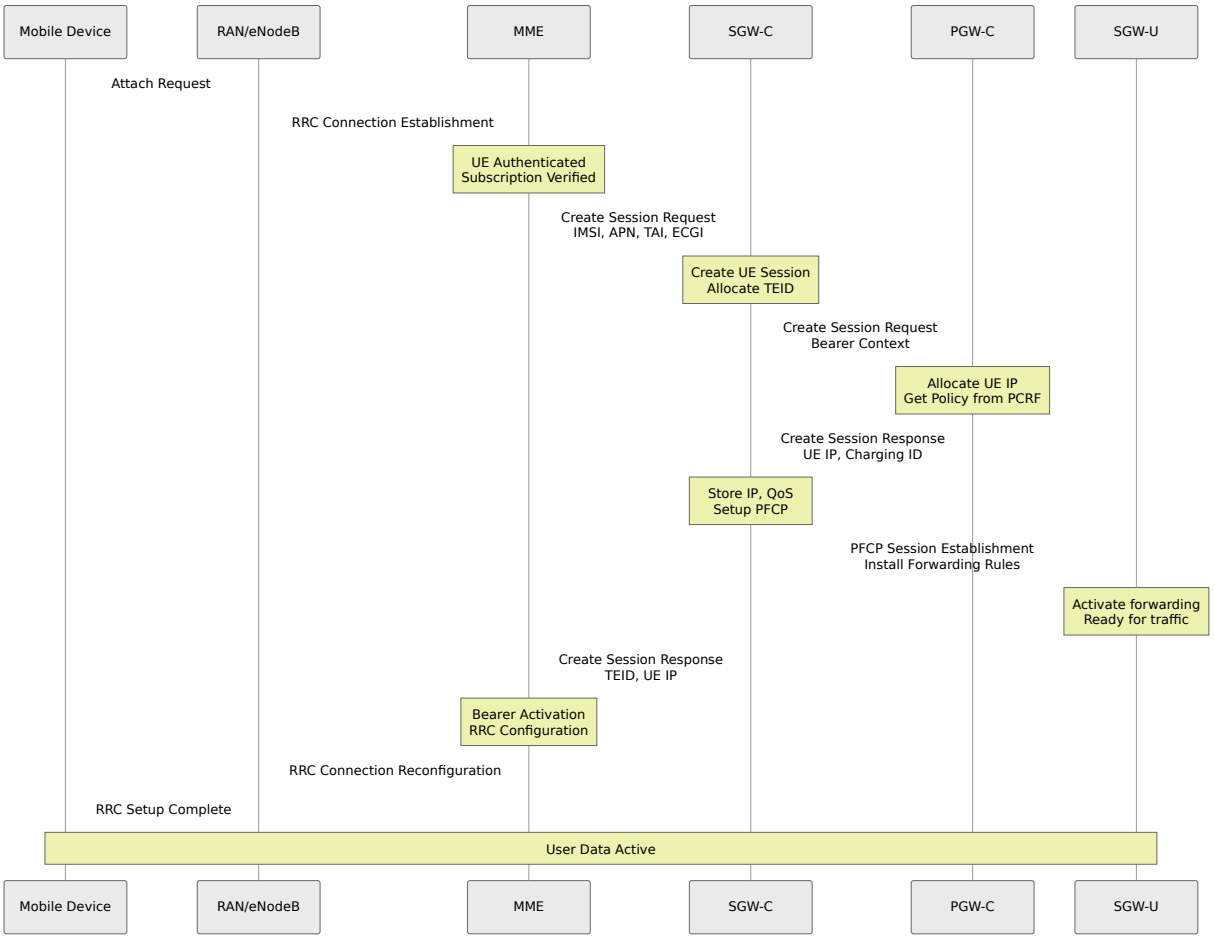
每个会话都有一个唯一的 IMSI（用户身份），并可能包含一个或多个 PDN 连接。

□□□□

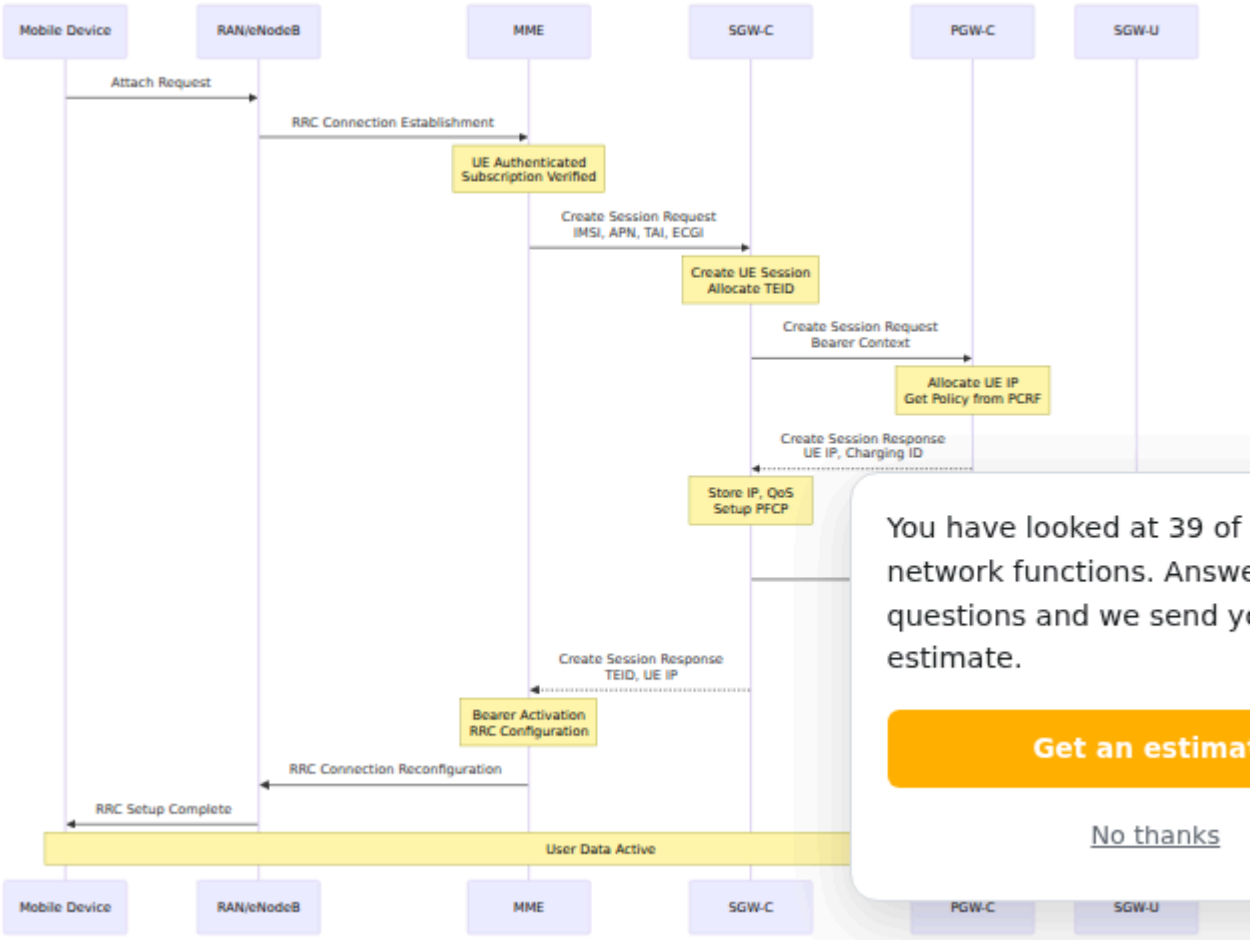

UE Session
Subscriber Identity
(IMSI, MSISDN)
Location Information
(TAI, ECI, Timezone)
PDN Connections
(one or more)
Charging Information
(Charging ID)
Bearer Management
(Default + Dedicated)

□□□□□□

□□□□□UE □□□


Mobile Device
RAN/eNodeB
MME
SGW-C
PGW-C
SGW-U
Attach Request
RRC Connection Establishment
UE Authenticated
Subscription Verified
Create Session Request
IMSI, APN, TAI, ECGI
Create UE Session
Allocate TEID
Create Session Request
Bearer Context
Allocate UE IP
Get Policy from PCRF
Create Session Response
UE IP, Charging ID
Store IP, QoS
Setup PFCP
PFCP Session Establishment
Install Forwarding Rules
Activate forwarding
Ready for traffic
Create Session Response
TEID, UE IP
Bearer Activation
RRC Configuration
RRC Connection Reconfiguration
RRC Setup Complete
User Data Active
Mobile Device
RAN/eNodeB
MME
SGW-C
PGW-C
SGW-U

□□□□□UE □□□



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

□□□□

UE □□□□□

```
[No Session]
    ↓ (Create Session Request from MME)
[Creating Session - PGW]
    ↓ (Create Session Response from PGW)
[Creating Session - User Plane]
    ↓ (PFCP Session Establishment Response)
[Session Active]
    ↓ (Modify Bearer Request or bearer changes)
[Session Modifying]
    ↓ (Modification Complete)
[Session Active]
    ↓ (Delete Session Request or network error)
[Session Terminating]
    ↓ (All responses received, CDR logged)
[Session Terminated]
```

关键状态变量

```
Session State:
├── IMSI: Mobile subscriber identity
├── GUTI: Temporary ID from MME
├── Location:
│   ├── TAI: Current tracking area
│   ├── ECI: Current cell
│   └── Timezone: UE timezone
├── PDN Connections: Array of PDN connection contexts
│   ├── APN: Access Point Name
│   ├── TEID (S11): To MME
│   ├── TEID (S5/S8): To PGW-C
│   ├── Charging ID: From PGW-C
│   ├── UE IP: From PGW-C
│   ├── PGW-C Address: S5/S8 peer
│   └── Bearers: Default + Dedicated
│       ├── EBI: Bearer ID
│       ├── QCI: QoS class
│       ├── ARP: Priority
│       ├── GBR: Guaranteed rate
│       └── MBR: Maximum rate
└── Charging: Charging ID, event log
```

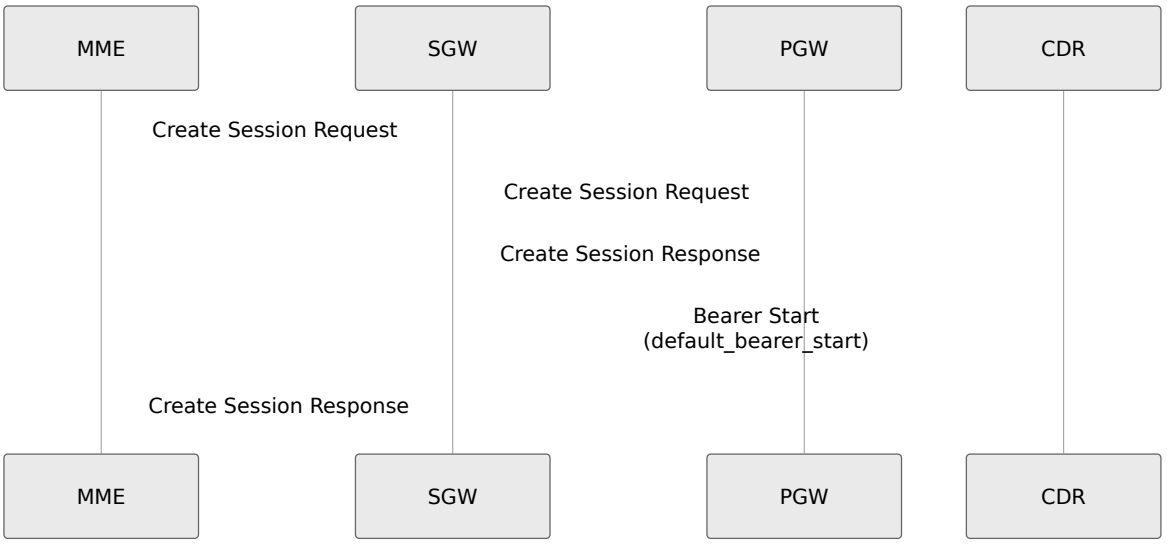
承载□作

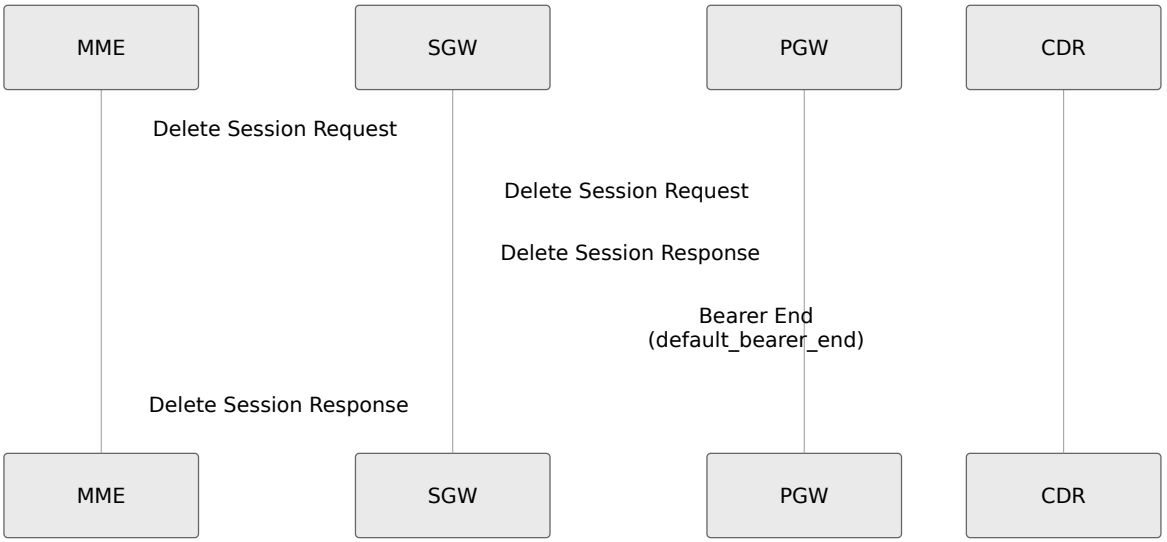
默认承载

每个 PDN 连接都会创建默认承载：

- **QoS：** 通常为 QCI 9（最佳努力）
- 生命周期：与 PDN 连接相同
- 流量：承载所有未被专用承载匹配的流量
- 强制性：每个 PDN 连接必须有默认��载

承载开始事件：





承载结束事件：

专用承载

专用承载为特定服务提供优质的 QoS：

- 激活：由应用程序或网络策略请求
- **QoS：** QCI 1-8（各种服务类型）
- 生命周期：可以短于 PDN 连接
- 可选：每个 PDN 连接可以有零个或多个

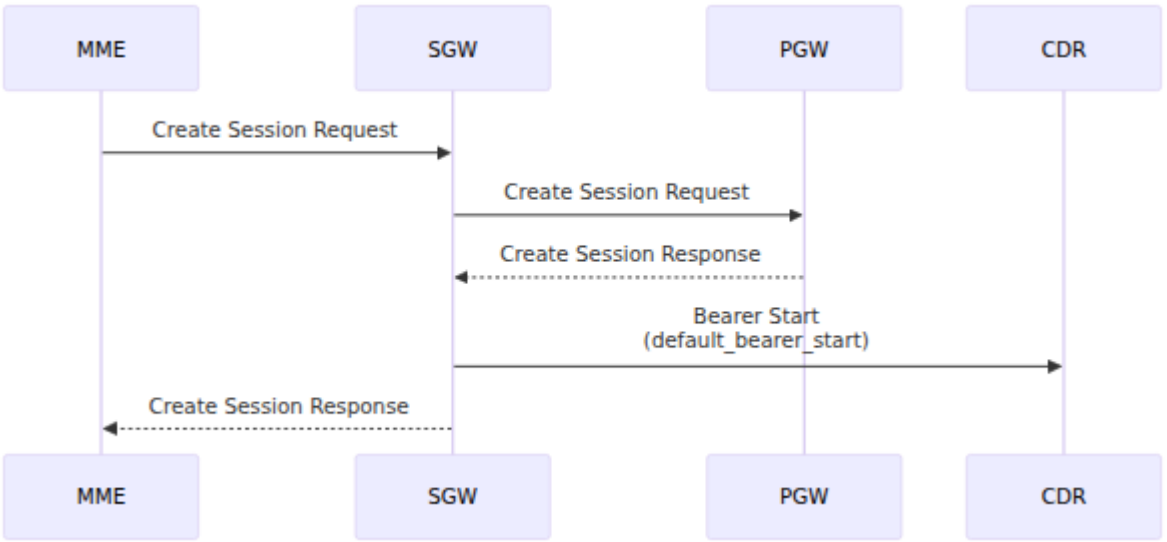
专用承载激活：

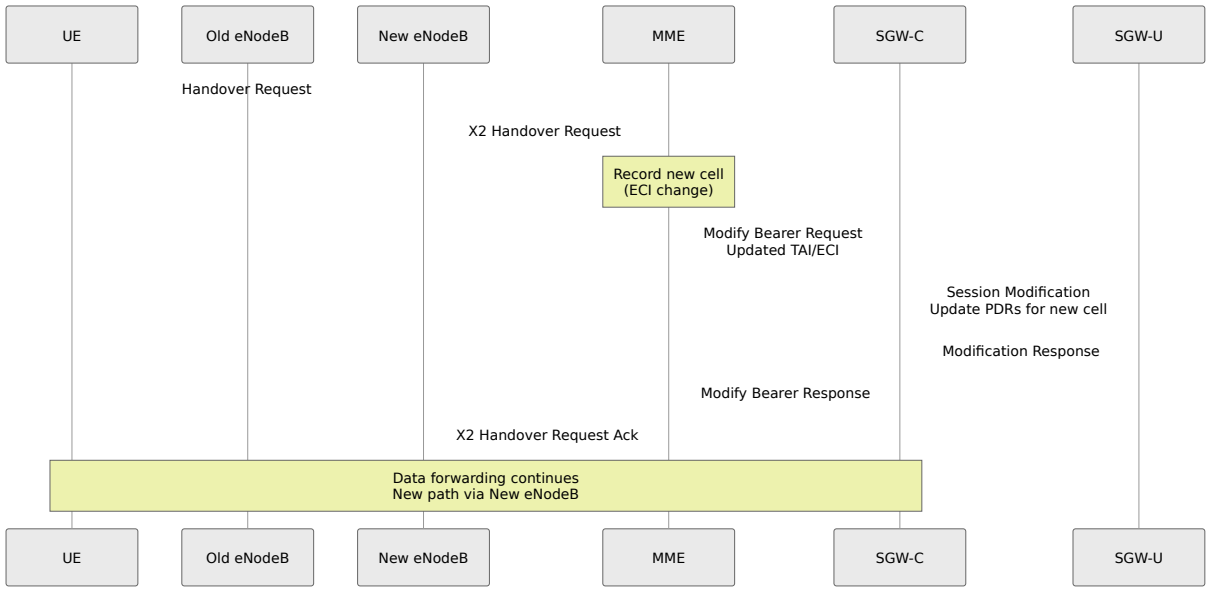
```
Application Trigger
    ↓
PGW-C Policy Decision (via PCRF)
    ↓
Create Bearer Request (S5/S8)
    ↓
SGW forwards to MME (S11)
    ↓
MME activates bearer on RAN
    ↓
Create Bearer Response back through SGW to PGW
```

□□□□□□:

```
Network or Application Decision
    ↓
Delete Bearer Request (S5/S8)
    ↓
SGW forwards to MME (S11)
    ↓
MME deactivates bearer on RAN
    ↓
Delete Bearer Response back through SGW to PGW
```

□□ QoS □□





移动性处理

内部 **MME** 切换（无 **SGW** 更改）

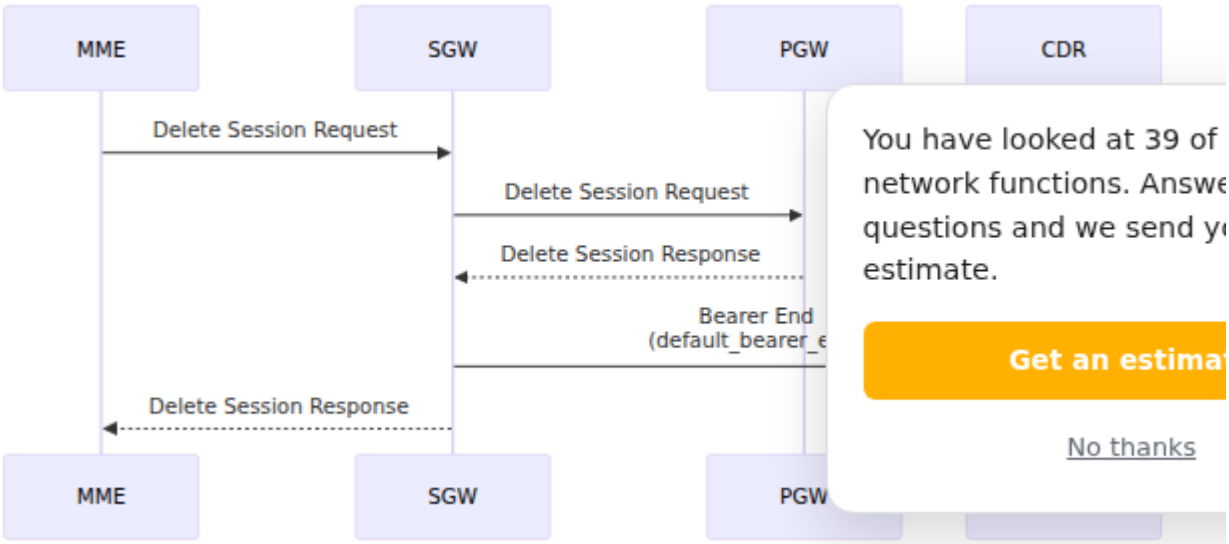
场景：UE 在同一 MME 区域内移动

会话影响：

- 会话保持活动状态
- TEID 保持不变
- 会话中的位置更新
- CDR 继续使用相同的 Charging ID

跨 **MME** 切换（带 **SGW** 更改）

场景：UE 移动到不同的 MME，需要新的 SGW



会话影响：

- 旧会话终止，CDR 记录为“切换”指示
- 新会话创建，使用相同的 Charging ID
- 数据转发保持连接
- 用户平面通过新的 SGW-U 重新路由

跟踪区域更新（**TAU**）

无 **SGW** 更改的 **TAU:**

```
UE 更新位置
    ↓
MME 发送 TAU Accept
    ↓
MME 更新 SGW 以获取新位置
    ↓
SGW 修改会话（TAI, ECI）
    ↓
无服务中断
```

带 **SGW** 更改的 **TAU:**

- 类似于跨 MME 切换
- 会话迁移到新 SGW
- CDR 在旧 SGW 和新 SGW 之间协调

切换程序

准备阶段

在切换完成之前：

1. 选择新 **SGW-U** - 选择转发路径
2. 安装 **PDR** - 安装新的转发规则
3. 缓冲激活 - 启用正在传输的数据包的缓冲
4. 信令协调 - S11/S5/S8 消息交换

数据转发阶段

在切换过渡期间：

- 旧 **SGW-U** 中的缓冲 - 数据包暂时保留
- 新 **SGW-U** 中的缓冲 - 准备接收
- **GTP** 隧道 - 数据从旧路径转发到新路径
- 数据包排序 - 保持顺序

完成阶段

在切换完成后：

1. 缓冲清除 - 释放缓��的数据包
2. 路径切换 - 流量切换到新路径
3. 旧路径清理 - 释放旧的转发规则
4. 会话更新 - 更新位置和 TEID

作程序

会话检查

通过 Web UI 监控活动会话:

1. 打开 http://<sgw-ip>:<port>/ue_sessions
2. 查看所有活动 UE 会话
3. 按 IMSI、GUTI 或电话号码搜索
4. 点击会话以查看详细信息:
   - 位置 (TAI, ECI)
   - 活动承载和 QoS
   - PGW-C 关联
   - TEID 对信息
   - Charging ID

UE 会话述显示所有活动会话及其关键标识符:

点击任何会话以查看包括 TEID、位置、承载和 PDN 连接的详细信息:

有关 Web UI 导航和访问说明，请参见 操作指南。

指标监控

跟踪会话指标：

```
# 计算活动会话
curl -s http://10.0.0.40:42068/metrics | grep active_ue_sessions

# 计算活动承载
curl -s http://10.0.0.40:42068/metrics | grep active_bearers

# 按 APN 监控
curl -s http://10.0.0.40:42068/metrics | grep sessions_by_apn

# 监控消息速率
curl -s http://10.0.0.40:42068/metrics | grep
s11_inbound_messages_total
```

有关可用指标、Prometheus 仪表板和警报设置的完整参考，请参见 监控与指标指南。

优雅的会话终止

要干净地终止会话：

1. 通过 **API** 触发：请求会话删除
2. 等待完成：监控会话状态

3. 验证清理：检查指标
4. 审查 **CDR**：确认最终记录

会话限制

监控容量：

```
# 检查当前负载
curl -s http://10.0.0.40:42068/metrics | \
  grep -E "active_ue_sessions|active_bearers" | \
  awk '{print $NF}'

# 在达到 80% 的许可容量时发出警报
# 达到限制时优雅处理
```

故障排除

会话无法建立

症状：创建会话请求失败

诊断：

1. 检查指标以获取原因代码
2. 检查 S11 错误日志
3. 验证 PGW 连接
4. 检查 Charging ID 可用性

常见原因：

原因	解决方案
PGW 无法访问	验证 S5/S8 网络连接
没有可用 IP	检查 PGW IP 池状态
APN 未配置	验证 PGW 中的 APN
没有可用 SGW-U	确保 SGW-U 关联处于活动状态
策略不匹配	检查 PGW 策略配置

会话意外掉线

症状：活动会话在没有删除请求的情况下终止

诊断：

1. 检查日志中的错误消息
2. 监控 SGW-U 心跳状态
3. 检查 PGW 连接
4. 审查指标以���找错误峰值

常见原因：

原因	解决方案
SGW-U 崩溃	重启 SGW-U，监控日志
网络断开	检查接口状态
PGW 故障	切换到备份 PGW
消息超时	增加超时，检查 RTT

切换失败

症状：切换丢失数据包或完全失败

诊断：

1. 监控修改承载消息
2. 检查 PFCP 规则更新
3. 验证数据转发设置
4. 检查承载缓冲

常见原因：

原因	解决方案
缓冲禁用	在 PFCP 规则中启用 BAR
PDR 未更新	验证发送的 PFCP 修改
转发路径中断	检查到新 SGW-U 的路由
时机过于紧凑	增加切换超时

消息延迟高

症状： S11/S5S8 消息处理缓慢

诊断：

```
# 检查消息延迟
curl -s http://10.0.0.40:42068/metrics | \
  grep "inbound_duration_seconds"

# 检查队列深度
curl -s http://10.0.0.40:42068/metrics | \
  grep queue_depth

# 检查系统负载
top -n1 | head -1
```

解决方案:

1. 如果网络 RTT 高，则增加消息超时
2. 在多个 SGW-C 实例之间负载均衡
3. 监控并减少消息速率
4. 检查是否有卡住的会话

CDR 生成问题

症状: CDR 丢失或不完整

诊断:

1. 检查 CDR 目录是否存在
2. 验证写入权限
3. 检查磁盘空间
4. 审查日志中的生成错误

解决方案:

```
# 监控 CDR 生成
tail -f /var/log/sgw_c/cdrs/*

# 检查文件权限
ls -la /var/log/sgw_c/cdrs/

# 确保目录可写
chmod 755 /var/log/sgw_c/cdrs/
```

有关完整的 CDR 字段参考和集成细节，请参见 CDR 格式指南。

最佳实践

会话管理

- 监控容量：跟踪与许可限制的比较
- 阈值警报：在 70-80% 容量时触发
- 优雅降级：在维护前排空
- 健康检查：监控对等连接

切换□作

- 快速故障转移：配置激进的心跳
- 优雅切换：确保启用缓冲
- 路径冗余：多个 SGW-U 对等体
- 测试：定期进行切换模拟

计费协调

- 验证 **Charging ID**：确保 PGW 分配
- **CDR** 验证：比较 SGW 和 PGW CDR
- 事件关联：在网关之间关联 CDR 事件
- 归档：长期 CDR 存储

Sxa 接口文档

PFCP 与 **SGW-U** 的通信

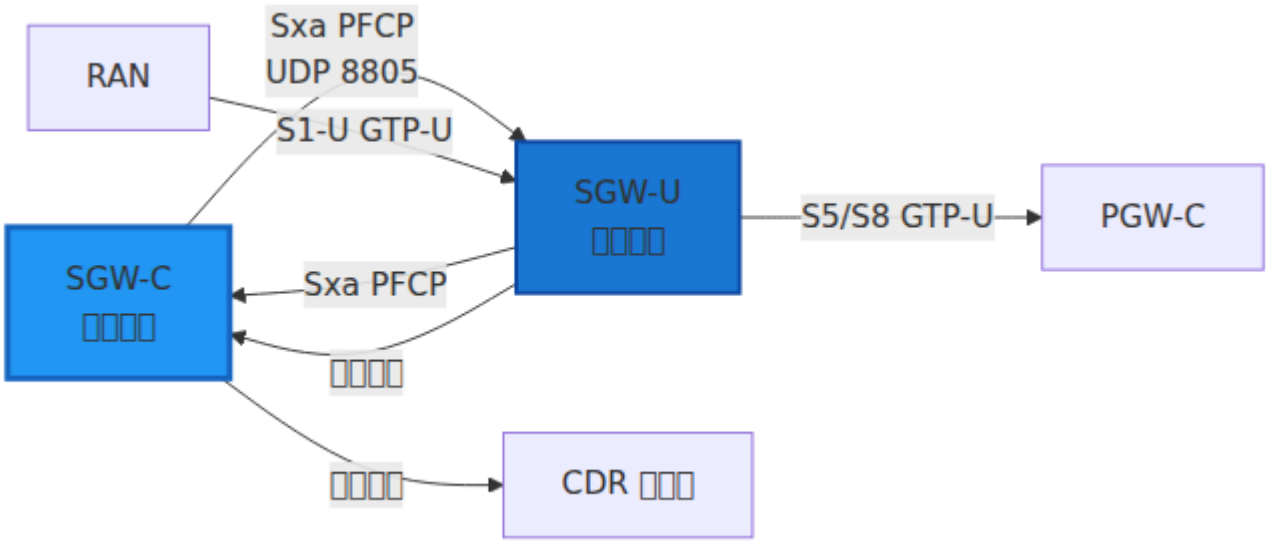
OmniSGW □ Omnitouch □□□□□□

目录

1. □述
2. 协议详细信息
3. 配置
4. PFCP 关联
5. 会话管理
6. PFCP 规则
7. 使用报告
8. 网络操作
9. 故障排除

概述

Sxa 接口 使用 **PFCP**（分组转发控制协议）协议将 OmniSGW 连接到 SGW-U（服务网关用户平面）。该接口控制用户平面数据包的转发、QoS 执行和使用报告。



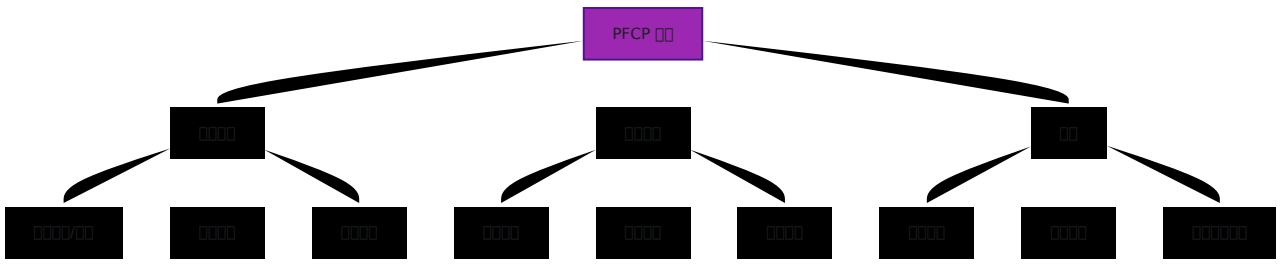
主要特性

- **PFCP v1.0** - 符合标准的分组转发控制
- 基于 **SEID** 的会话跟踪 - 用于关联的会话端点标识符
- 数据包检测规则 - 灵活的数据包匹配用于上行/下行
- 转发动作规则 - 控制数据包路由和封装
- **QoS** 执行 - 每个承载的比特率限制和优先级
- 使用计量 - 用于计费和分析的流量跟踪
- 缓冲控制 - 移动事件期间的自动缓冲

协议详细信息

PFCP 版本 1.0

- 协议: PFCP v1.0 (3GPP TS 29.244)
- 传输: UDP
- 端口: 8805 (标准)
- 接口类型: 控制平面
- 关联模型: CP 和 UP 形成持久关联



SEID（会话端点标识符）

每个会话都有唯一的 SEID 用于跟踪：

- **CP SEID** - 由 SGW-C 分配，用于上行消息到 SGW-U
- **UP SEID** - 由 SGW-U 分配，用于下行消息到 SGW-C

消息路由：

```
SGW-C → SGW-U: 使用 SGW-U 的 UP SEID
SGW-U → SGW-C: 使用 SGW-C 的 CP SEID
```

消息类型概述

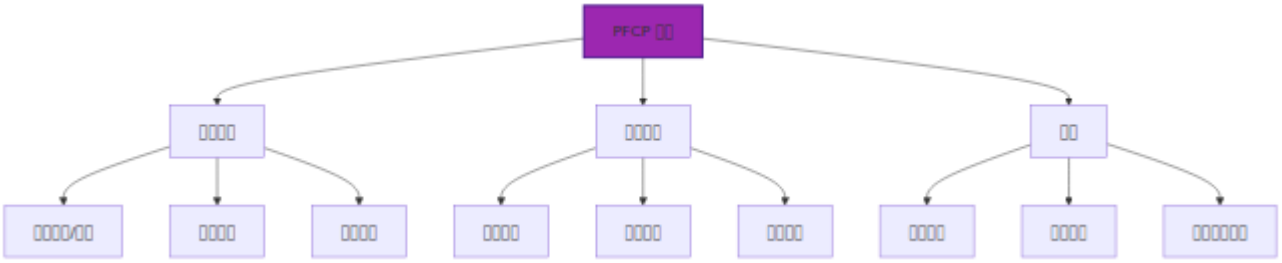
配置

基本配置

```
# config/runtime.exs
config :sgw_c,
  sxa: %{
    # Sxa 接口的本地 IP 地址
    local_ip_address: "10.0.0.20",

    # 可选：覆盖默认端口
    local_port: 8805,

    # 要连接的 SGW-U 对等体
    peers: [
      %{
        ip_address: "10.0.0.30",
        node_id: "sgw-u-1.example.com"
      },
      %{
        ip_address: "10.0.0.31",
        node_id: "sgw-u-2.example.com"
      }
    ],

    # 关联心跳间隔（秒）
    heartbeat_interval_s: 20,

    # 会话建立超时（毫秒）
    session_timeout_ms: 5000,

    # 会话操作的最大重试次数
    max_retries: 3
  }
```

网络要求

防火墙规则：



```
# 允许来自 SGW-U 网络的 PFCP
iptables -A INPUT -p udp --dport 8805 -s <sgwu_network>/24 -j
ACCEPT

# 允许向 SGW-U 的出站 PFCP
iptables -A OUTPUT -p udp --dport 8805 -d <sgwu_network>/24 -j
ACCEPT
```

路由:

```
# 确保到 SGW-U 网络的路由
ip route add <sgwu_network>/24 via <gateway_ip> dev eth0
```

网络��试:

```
# 测试 PFCP 连接性
# 检查日志中的 "关联建立完成" 消息

# 监控活动 PFCP 会话
curl http://127.0.0.40:42068/metrics | grep seid_registry_count
```

PFCP 关联

关联生命周期

在建立任何会话之前，SGW-C 和 SGW-U 必须形成 PFCP 关联。

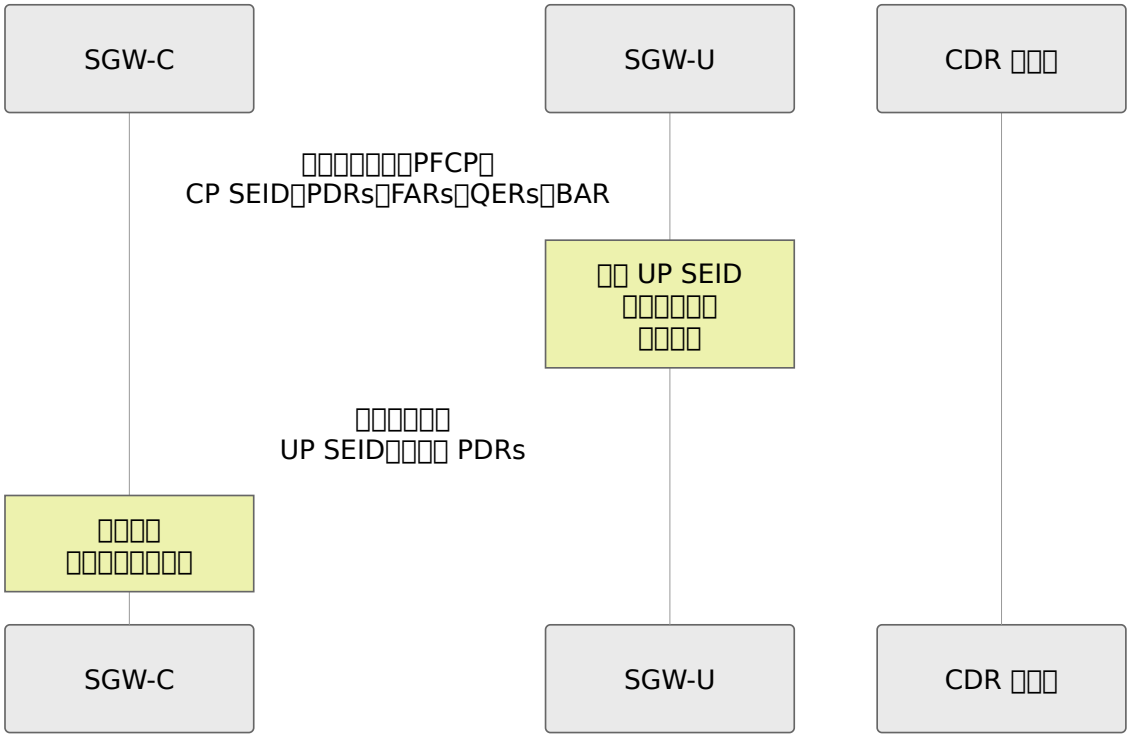
关联状态机

```
[断开连接]
      ↓ (建立请求)
[关联中]
      ↓ (建立响应 OK)
[已关联]
      ↓ (会话创建)
[会话活动]
      ↓ (心跳失败)
[重新关联中]
      ↓ (建立 OK 或超时)
[已关联或断开连接]
```

恢复处理

如果 PFCP 关联丢失并恢复:

1. 恢复检测:
    - 心跳超时触发恢复
    - 发送新的关联建立请求
    - 检查恢复时间戳
2. 会话恢复:
    - 会话可能可恢复，也可能不可恢复
    - 查询 SGW-U 以获取现有会话
    - 如有必要，重新建立丢失的会话
3. 数据转发:
    - 在恢复期间，SGW-U 中的用户平面缓冲
    - PDR 在明确删除之前保持活动
    - 在故障转移期间最小化数据包丢失



会话管理

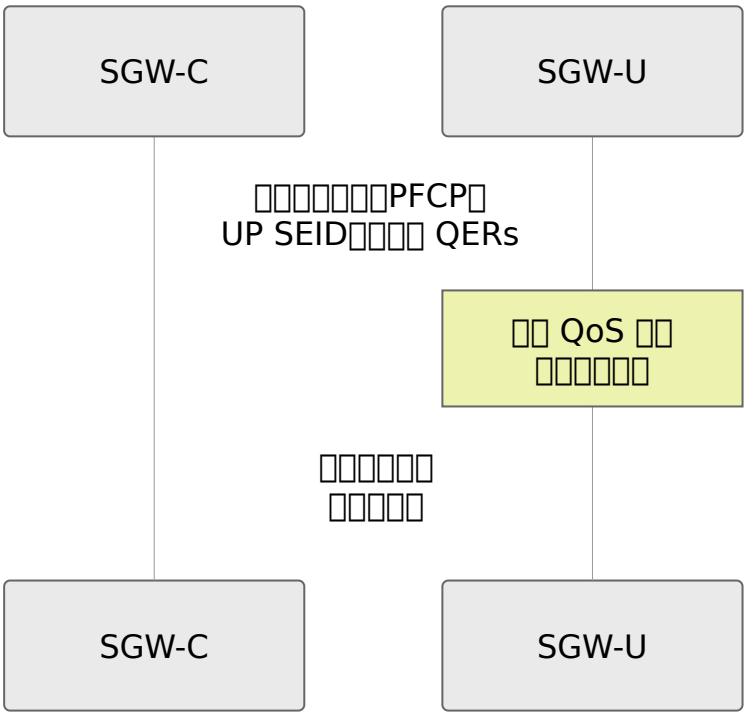
会话建立

触发：来自MME的创建会话请求（在S11接收）

SGW-C
SGW-U
CDR记录器
会话建立请求（PFCP）
CP SEID, PDRs, FARs, QERs, BAR
分配UP SEID
安装转发规则
分配缓冲
会话建立响应
UP SEID, 创建的PDRs
会话活动
用户平面准备就绪
SGW-C
SGW-U
CDR记录器

发送的信息元素：

元素	描述
CP SEID	由SGW-C为此会话分配
PDRs	数据包检测规则（见下文）
FARs	转发动作规则
QERs	QoS执行规则
BAR	移动的缓冲动作规则
创建PDR	响应的规则标识符



会话状态：

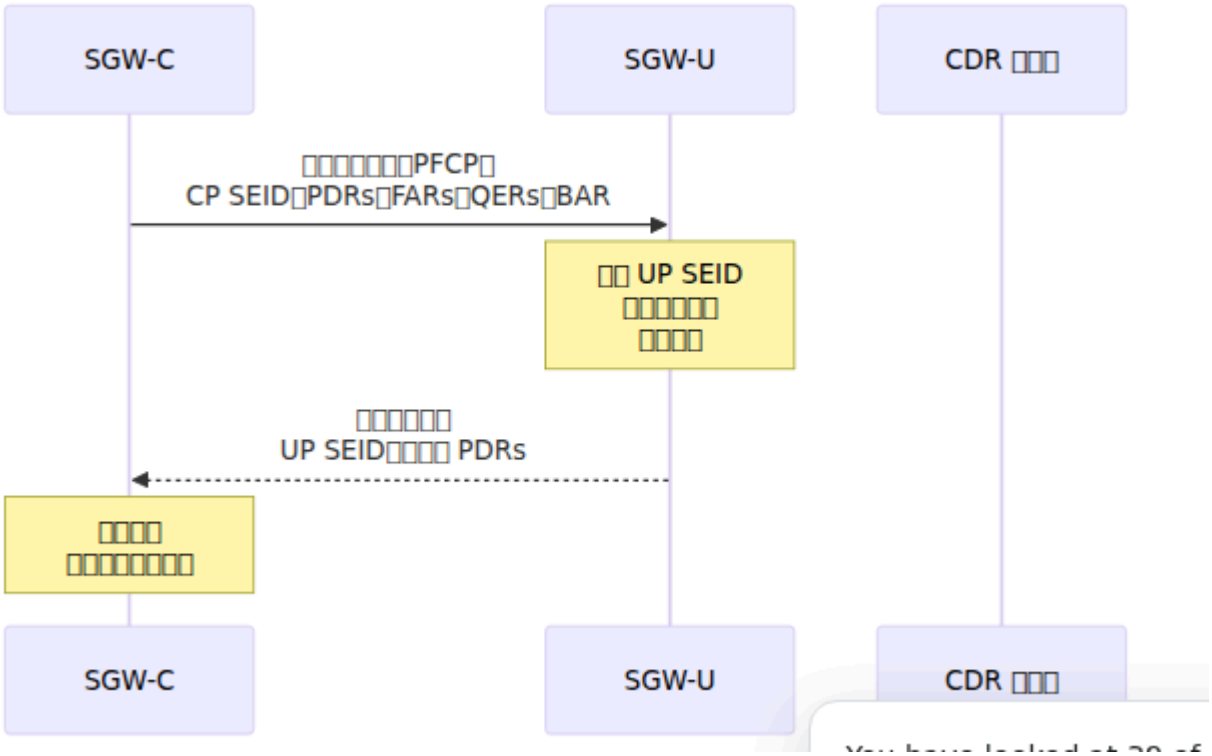
```
[无会话]
    ↓ （建立请求）
[建立中]
    ↓ （建立响应）
[会话活动]
```

会话修改

触发：来自 MME 的修改承载请求（QoS 变更，切换）

SGW-C
SGW-U
会话修改请求（PFCP）
UP SEID，更新的 QERs
更新 QoS 参数
重新配置缓冲
会话修改响应
更新的规则
SGW-C
SGW-U

常见修改：



修改	更新的元素	原因
QoS 变更	QERs	承载升级/降级
切换	PDRs, FARs	eNodeB 变更, SGW-U 选择
承载添加	新 PDR, FAR, QER	专用承载激活
承载删除	移除 PDR, FAR, QER	承载停用

会话删除

触发：来自 MME 的删除会话请求（分离）

会话状态：

```
[会话活动]
      ↓ (删除请求)
[删除中]
      ↓ (删除响应)
[会话终止]
```

PFCP □□

PDR□□□

示例用例 - 默认承载：

- 检测：来自/到 UE IP 的所有数据包
- 动作：通过 PDN 转发（PGW-U 方向）
- QoS：按承载应用

示例用例 - 专用承载：

- 检测：匹配特定流的数据包（端口范围，协议）
- 动作：通过专用路径转发
- QoS：优质费率（GBR）

FAR（转发动作规则）

指定如何处理匹配的数据包。

转发动作：

动作	描述	用例
转发	将数据包发送到目标网络	正常转发
缓冲	暂时存储数据包	在移动/寻呼期间
丢弃	丢弃数据包	策略执行，防火墙
复制	将数据包发送到多个目标	合法拦截

封装选项：

- **GTP-U** - 添加 GTP-U 隧道头（S1-U，S5/S8）
- 以太网 - 添加以太网头（用于直接互连）
- **IPv4** - 纯 IPv4 转发（用于互联网突破）
- **IPv6** - 纯 IPv6 转发

示例 - **UE** 到互联网：

PDR 匹配：源接口 = 接入，UE IP = 10.45.0.50
FAR 动作：
- 转发 = 是
- 外部头封装 = 无（直接互联网）
- 转发参数 = 互联网网关

QER（QoS 执行规则）

对每个承载执行比特率限制。

QoS 参数：

参数	类型	描述
QCI	整数	QoS 类标识符（1-9）
MBR（最大比特率）	比特率	允许的最大速率
GBR（保证比特率）	比特率	最低保证速率
ARP	整数	分配和保留优先级（1-15）

QoS 类（QCI）：

QCI	服务类型	比特率示例
1	���音（GBR）	MBR: 64 kbps
2	视频通话（GBR）	MBR: 256 kbps
3	实时游戏（GBR）	MBR: 50 kbps
4	非 GBR	GBR: 128 kbps, MBR: 256 kbps
5	IMS 信令	GBR: 100 kbps, MBR: 256 kbps
6	视频流媒体	MBR: 10 Mbps
7	带视频的语音（GBR）	GBR: 64 kbps, MBR: 384 kbps
8	网页浏览	MBR: 5 Mbps
9	电子邮件	MBR: 3 Mbps

示例 - 默认承载（**QCI 9**）：

```
QCI: 9（尽力而为）
MBR: 100 Mbps（依赖于站点）
GBR: 无（非 GBR）
ARP: 15（最低优先级）
```

示例 - 专用语音承载（**QCI 1**）：

```
QCI: 1（语音）
MBR: 128 kbps（上行 + 下行）
GBR: 64 kbps（保证）
ARP: 1（最高优先级）
```

BAR（缓冲动作规则）

控制在移动事件期间的数据包缓冲。

缓冲场景：

1. 切换缓冲：
    - UE 在 eNodeB 之间切换
    - 移动时缓冲数据包
    - 切换完成后释放
2. 寻呼缓冲：
    - UE 处于空闲状态（挂起的承载）
    - 下行数据到达
    - 缓冲直到 UE 重新激活
3. **SGW** 迁移：
    - 在与 SGW 变更的跨 MME 切换期间
    - 旧 SGW 缓冲并转发到新 SGW
    - 保持有序交付

BAR 配置：

```
BAR 设置：
├── 缓冲超时：保持数据包的时间
├── 数据包计数阈值：最大缓冲数据包
└── 下行数据报告触发
    └── 当数据到达时向 CP 发送通知
```



使用报告

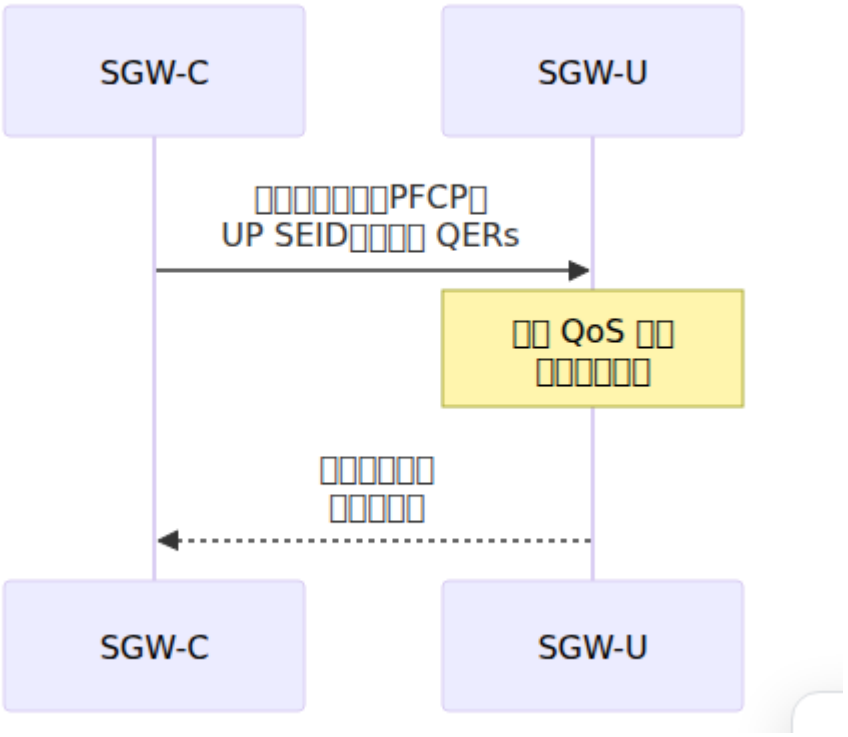
使用报告消息

SGW-U 向 SGW-C 发送使用报告以进行计费和分析。

使用报告触发

报告在以下情况下发送：

触发	条件
时间段	每 N 秒定期报告
流量阈值	转发 N 字节后
持续时间阈值	转发 N 秒后
会话结束	当会话被删除时
修改	当规则被更新时
即时报告	在修改消息中请求



使用报告字段

```
使用报告:
├── 使用报告触发: 导致此报告的原因
├── UR-SEQN: 用于排序的序列号
├── 每个承载的使用信息:
│   ├── EBI: 承载标识符
│   ├── 流量测量
│   │   ├── UL: 上行字节
│   │   ├── DL: 下行字节
│   │   ├── 总计: 总字节
│   │   └── 数据包: 总数据包
│   ├── 持续时间测量: 活动秒数
│   └── 第一个/最后一个数据包的时间: 时间戳
└── 查询 UR: 请求即时报告
```

CDR 生成流程

网络□作

PFCP 关联监控

监控活动的 PFCP 关联:

```
# 检查关联状态
curl -s http://127.0.0.40:42068/metrics | grep pfcp_association

# 预期输出:
# pfcp_association_status{peer_ip="10.0.0.30"} 1  (已关联)
# pfcp_association_status{peer_ip="10.0.0.31"} 1  (已关联)

# Web UI → SGW-U 状态页面
# 显示所有对等体的 "已关联" 状态和恢复信息
```

会话指标

监控活动的 PFCP 会话:

```
# 计数活动会话
curl -s http://127.0.0.40:42068/metrics | grep seid_registry_count

# 监控每个 SGW-U 的分布
curl -s http://127.0.0.40:42068/metrics | grep seid_by_peer

# 使用率（字节/秒）
curl -s http://127.0.0.40:42068/metrics | grep usage_octets_rate
```

消息流监控

跟踪 PFCP 消息活动:

```
# 监控所有 PFCP 消息
watch -n 1 'curl -s http://127.0.0.40:42068/metrics | grep sxa_inbour

# 示例输出:
#
sxa_inbound_messages_total{message_type="session_establishment_respor
5432
#
sxa_inbound_messages_total{message_type="session_modification_respons
12100
# sxa_inbound_messages_total{message_type="session_report_request"} 0
```

规则安装验证

检查规则是否在 SGW-U 中正确安装:

```
# 监控会话建立的成功/失败
curl -s http://127.0.0.40:42068/metrics | grep
sxa_session_establishment

# 检查 PDR 安装问题
curl -s http://127.0.0.40:42068/metrics | grep pdr_installation

# 查找超时
curl -s http://127.0.0.40:42068/metrics | grep sxa_timeout_total
```

故障排除

关联失败

问题: "关联建立失败"

诊断:

1. 检查网络连接: `ping <sgwu_ip>`
2. 验证端口: `netstat -an | grep 8805`

3. 检查日志以获取错误详细信息

常见原因和解决方案:

原因	症状	解决方案
网络不可达	设置超时	验证到 SGW-U 的路由
端口被阻塞	连接被拒绝	检查防火墙规则
SGW-U 关闭	无响应	重启 SGW-U 进程
节点 **ID** 不匹配	设置被拒绝	验证配置

会话建立失败

问题: "会话建立失败"

诊断:

```
# 检查指标
curl -s http://127.0.0.40:42068/metrics | grep seid_registry_count

# 检查日志以获取具体错误
tail -f /var/log/sgw_c/sgw_c.log | grep "会话建立"
```

常见原因:

原因	错误消息	解决方案
SGW-U 无资源	"资源不足"	检查 SGW-U 容量
无效 **PDR**	"缺少强制 IE"	验证规则定义
SEID 冲突	"SEID 已存在"	检查会话重复
超时	"会话建立超时"	增加超时或检查 SGW-U

使用报告问题

问题: "使用报告缺失"

诊断:

```
# 检查报告计数
curl -s http://127.0.0.40:42068/metrics | grep
session_report_request_total

# 监控 CDR 生成
tail -f /var/log/sgw_c/cdrs/<timestamp>
```

解决方案:

- 验证 SGW-U 心跳是否活动
- 检查会话报告触发配置
- 确保 CDR 目录权限正确
- 监控 SGW-U 缓冲溢出

性能问题

问题: PFCP 消息延迟高

检查的指标:

```
# 消息处理持续时间
curl -s http://127.0.0.40:42068/metrics | grep
sxa_inbound_duration_seconds

# 每个对等体的会话负载
curl -s http://127.0.0.40:42068/metrics | grep seid_by_peer

# 队列深度
curl -s http://127.0.0.40:42068/metrics | grep pfcp_queue_depth
```

优化步骤：

1. 在多个 SGW-U 对等体之间进行负载均衡
2. 如果网络丢包，增加心跳超时
3. 监控并减少规则复杂性
4. 通过增加 SGW-C 实例进行水平扩展

有关完整的指标参考、仪表板配置和警报设置，请参见 监控与指标指南。

最佳实践

配置

- 心跳间隔：设置为 20-30 秒以确保可靠检测
- 会话超时：根据网络 RTT 设置为 5-10 秒
- 最大重试次数： 2-3 次，以平衡可靠性和延迟
- 对等体选择：在所有 SGW-U 对等体之间分配负载

□作

- 对等体冗余：配置多个 SGW-U 实例以实现故障转移
- 优雅重载：支持在线软件升级
- 会话排水：在维护之前迁移会话
- 监控：跟踪关联恢复频率

故障排除

- 保留日志：保留 PFCP 消息跟踪以便调试
- 关联：将 S11 消息与 PFCP 会话操作关联
- 基线指标：建立正常性能基线
- 测试故障场景：练习 SGW-U 故障转移程序
