API 文档说明

目录

- 错误代码说明
- 状态码列表

错误代码说明

400 错误响应

```
{
  "error": "Invalid JSON format"
}
```

说明：

- JSON 格式错误
- 请求参数缺失
- 参数类型错误

404 未找到

```
{
  "error": "Resource not found"
}
```

说明：

- □□□/□□□□/□□□□□
- URL □□ ID □□□

422 □□□□□□□□

```
{
  "errors": {
    "imsi": ["has already been taken"],
    "key_set_id": ["does not exist"]
  }
}
```

□□□

- □□□□
- □□□□□□□□
- □□□□□□□

500 □□□□□□□□

```
{
  "error": "Internal server error"
}
```

□□□

- □□□□□□□
- □□□□□□□□□

□□□□□□

API Request

OmniRAN

Invalid JSON

400 Bad Request

Valid

Authorized?

No

401 Unauthorized

Yes

Resource Exists?

No

404 Not Found

Yes

Data Valid?

No

422 Validation Error

Yes

Process Request

Database OK?

Error

Success

500 Server Error   200/201 Success

← 返回 API 文档

API 参考文档

概述

- 开发者快速入门
- 服务器访问 IP 列表

开发人员指南

下面的示例向你展示了如何构建请求。这些示例使用通用的命令行工具，应该适用于大多数平台。

你可能需要 安装命令行 `jq` 来对 JSON 进行格式化 `apt-get install jq` 或 `brew install jq` 都应该可以使用。

- 发送请求
- APN 推送通知
- EPC 推送通知
- 管理订阅

```bash
# 1. 创建密钥集
KEY_SET_ID=$(curl -k -X POST
https://hss.example.com:8443/api/key_set \
  -H "Content-Type: application/json" \
  -d '{
    "ki": "0123456789ABCDEF0123456789ABCDEF",
    "opc": "FEDCBA9876543210FEDCBA9876543210",
    "authentication_algorithm": "milenage",
    "amf": "8000",
    "sqn": 0
  }' | jq -r '.response.id')

# 2. 创建 APN QoS 配置文件
APN_QOS_ID=$(curl -k -X POST
https://hss.example.com:8443/api/apn/qos_profile \
  -H "Content-Type: application/json" \
  -d '{
    "name": "默认互联网 QoS",
    "allocation_retention_priority": 8,
    "apn_ambr_dl_kbps": 50000,
    "apn_ambr_ul_kbps": 25000,
    "pre_emption_capability": true,
    "pre_emption_vulnerability": true,
    "qci": 9
  }' | jq -r '.response.id')

# 3. 创建 APN 标识符
APN_ID=$(curl -k -X POST
https://hss.example.com:8443/api/apn/identifier \
  -H "Content-Type: application/json" \
  -d '{
    "apn": "internet",
    "ip_version": "ipv4v6"
  }' | jq -r '.response.id')

# 4. 创建 APN 配置文件
APN_PROFILE_ID=$(curl -k -X POST
https://hss.example.com:8443/api/apn/profile \
  -H "Content-Type: application/json" \
  -d "{
    \"apn_identifier_id\": $APN_ID,
    \"apn_qos_profile_id\": $APN_QOS_ID,
    \"name\": \"默认的 APN\"
```

```bash
  }" | jq -r '.response.id')

# 5. □□ EPC □□□□
EPC_PROFILE_ID=$(curl -k -X POST
https://hss.example.com:8443/api/epc/profile \
  -H "Content-Type: application/json" \
  -d "{
    \"apn_profiles\": [$APN_PROFILE_ID],
    \"name\": \"□□□□□□\",
    \"network_access_mode\": \"packet_only\",
    \"tracking_area_update_interval_seconds\": 600,
    \"ue_ambr_dl_kbps\": 100000,
    \"ue_ambr_ul_kbps\": 50000
  }" | jq -r '.response.id')

# 6. □□□□
SUBSCRIBER_ID=$(curl -k -X POST
https://hss.example.com:8443/api/subscriber \
  -H "Content-Type: application/json" \
  -d "{
    \"imsi\": \"001001123456789\",
    \"key_set_id\": $KEY_SET_ID,
    \"epc_profile_id\": $EPC_PROFILE_ID
  }" | jq -r '.response.id')

echo "□□□□□□□□ID: $SUBSCRIBER_ID"
```

□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□

1. □□□□ (□□□) - □□□□□□□□
2. □□□□□□□□□ (EPC □□□□) - □□□□□□□□□□□□
3. **APN** □□ (APN □□□□) - □□ QoS □□□□
4. □□□□ (□□) - □□□□□□□□

□□□□□

- □□□□□□□□□□□ MSISDN □□
- □□□□□□□□□□□□ IMS □□□□
- □□□□□□□□□□ □□□□□□

- 物联网卡 SIM。如欲开始使用 SIM

卡，请

- 在 MSISDN 上下 - 用于建立外拨连接
- 在数据会话中 - 用于发送数据

设备专用的 IP 地址

在数据会话期间，为设备分配 IP 地址的方法。

注意： 此功能仅适用于"专用型" APN 上具有动态 IPv4 地址的 IoT 网络设备。

```bash
# □□□□□□□□□□ jq (apt-get install jq □ brew install jq)

# 1. □□□□□
KEY_SET_ID=$(curl -k -X POST
https://hss.example.com:8443/api/key_set \
  -H "Content-Type: application/json" \
  -d '{
    "ki": "0123456789ABCDEF0123456789ABCDEF",
    "opc": "FEDCBA9876543210FEDCBA9876543210",
    "authentication_algorithm": "milenage",
    "amf": "8000",
    "sqn": 0
  }' | jq -r '.response.id')

# 2. □□ APN QoS □□□□
APN_QOS_ID=$(curl -k -X POST
https://hss.example.com:8443/api/apn/qos_profile \
  -H "Content-Type: application/json" \
  -d '{
    "name": "IoT □□□□",
    "allocation_retention_priority": 8,
    "apn_ambr_dl_kbps": 10000,
    "apn_ambr_ul_kbps": 5000,
    "pre_emption_capability": false,
    "pre_emption_vulnerability": false,
    "qci": 9
  }' | jq -r '.response.id')

# 3. □□ APN □□□
APN_ID=$(curl -k -X POST
https://hss.example.com:8443/api/apn/identifier \
  -H "Content-Type: application/json" \
  -d '{
    "apn": "internet",
    "ip_version": "ipv4"
  }' | jq -r '.response.id')

# 4. □□ APN □□□□
APN_PROFILE_ID=$(curl -k -X POST
https://hss.example.com:8443/api/apn/profile \
  -H "Content-Type: application/json" \
  -d "{
    \"apn_identifier_id\": $APN_ID,
```

```bash
      \"apn_qos_profile_id\": $APN_QOS_ID,
      \"name\": \"IoT □□□ APN\"
    }" | jq -r '.response.id')

# 5. □ APN □□□□□ IP
STATIC_IP_ID=$(curl -k -X POST
https://hss.example.com:8443/api/epc/static_ip \
    -H "Content-Type: application/json" \
    -d "{
      \"apn_profile_id\": $APN_PROFILE_ID,
      \"ipv4_static_ip\": \"100.64.1.100\"
    }" | jq -r '.response.id')

# 6. □□ EPC □□□□
EPC_PROFILE_ID=$(curl -k -X POST
https://hss.example.com:8443/api/epc/profile \
    -H "Content-Type: application/json" \
    -d "{
      \"apn_profiles\": [$APN_PROFILE_ID],
      \"name\": \"IoT □□□□\",
      \"network_access_mode\": \"packet_only\",
      \"tracking_area_update_interval_seconds\": 600,
      \"ue_ambr_dl_kbps\": 10000,
      \"ue_ambr_ul_kbps\": 5000
    }" | jq -r '.response.id')

# 7. □□ MSISDN□□□□□□□
MSISDN_ID=$(curl -k -X POST
https://hss.example.com:8443/api/msisdn \
    -H "Content-Type: application/json" \
    -d '{
      "msisdn": "14155551000"
    }' | jq -r '.response.id')

# 8. □□□□□□□ IP □□□
SUBSCRIBER_ID=$(curl -k -X POST
https://hss.example.com:8443/api/subscriber \
    -H "Content-Type: application/json" \
    -d "{
      \"imsi\": \"001001999999999\",
      \"key_set_id\": $KEY_SET_ID,
      \"epc_profile_id\": $EPC_PROFILE_ID,
      \"msisdns\": [$MSISDN_ID],
      \"static_ips\": [$STATIC_IP_ID]
```

```
    }" | jq -r '.response.id')

echo "IoT 设备激活完成："
echo "  订户 ID: $SUBSCRIBER_ID"
echo "  IMSI: 001001999999999"
echo "  MSISDN: 14155551000"
echo "  分配 IPv4: 100.64.1.100（来自 'internet' APN 池）"
```

它的作用：

此脚本创建一个完全配置的 IoT 设备订户：

1. 创建订户 (步骤一) - 使用其凭据
2. **APN** 配置 (APN 配置对象) - "互联网"接入点
3. 请求 **IP** 地址 (分配 IP) - 分配的 IPv4 地址 100.64.1.100
4. 设置网络访问参数 (EPC 配置对象) - 用于 IoT 连接的参数
5. 分配号码 (MSISDN) - 设备的电话号码
6. 激活订户 (激活) - 使其处于活动状态

结果：

设备现在已激活并连接到"互联网" APN，并具有静态分配的 IP 地址 `100.64.1.100`，无需使用 DHCP 或协商。

您还可以：

- 添加更多 APN 或分配 IP（查看更多 APN 配置步骤 2-5
- 启用语音和短信功能（查看 IMS 配置对象
- 设置使用限制和配额（查看使用配额对象
- 创建物理 SIM（查看创建物理 SIM

另请参阅

- 分配 IP 地址 - 管理订户的 IP 地址
- 网络访问参数 - 配置订户 IP 会话参数
- 将 MSISDN 分配 - 管理电话号码分配

← 返回 API 参考

OmniHSS API □□

← □□□□□□

□□

- API □□
- □□□□
- □□□□
- MSISDN □□
- SIM □□
- □□□□□
- □□□□□□
- □□ IP □□
- □□□□
- EIR □□
- □□□□□
- □□□□
- API □□□□

API □□

□□ URL

```
https://[hostname]:8443/api
```

□□□□

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

- **□□:** □□□ HTTPS
- **□□:** 8443□□□□□

□□: □□ API □□□□□□□□□□□□□□"□□" JSON □□□□□

□□□□:

```
{
  "name": "value",
  "field": "value"
}
```

□□□□□□□□□□□□:

```
{
  "subscriber": {
    "name": "value",
    "field": "value"
  }
}
```

□□:

```
# ✓ □□
curl -X POST https://hss.example.com:8443/api/ims/profile \
  -H "Content-Type: application/json" \
  -d '{"name": "default", "ifc_template": "..."}'

# ✗ □□
curl -X POST https://hss.example.com:8443/api/ims/profile \
  -H "Content-Type: application/json" \
  -d '{"ims_profile": {"name": "default", "ifc_template": "..."}}'
```

□□□□

□□□□□□□ JSON□□□□□□□□□□

□□□□:

```
{
  "status": "success",
  "response": { ... }
}
```

错误响应:

```
{
  "status": "error",
  "response": {
    "invalid_fields": {
      "field_name": "error message"
    }
  }
}
```

HTTP 状态码

状态	名称	适用场景
200	OK	成功的 GET、PUT、DELETE
201	已创建	成功的 POST
400	错误请求	请求格式错误
404	未找到	资源不存在
422	无法处理的实体	验证失败
500	内部服务器错误	服务器端错误

API □□□□

Client | API | Validation | Database

HTTPS □□ (JSON)

□□□□

alt

　[□□□□]

　□□

　400/422 □□□□

　[□□□□]

□□□□

alt

　[□□□□□]

　□□

　500 □□□□

　[□□]

　□□

　200/201 □□□□

Client | API | Validation | Database

□□□□

□□□□

□□□□□□□□□□□□□

□□: `GET /api/subscriber`

□□□□:

字段	类型	说明
enabled	boolean	是否启用该用户
ims_enabled	boolean	是否启用 IMS 服务

示例请求:

```
curl -k https://hss.example.com:8443/api/subscriber
```

示例响应:
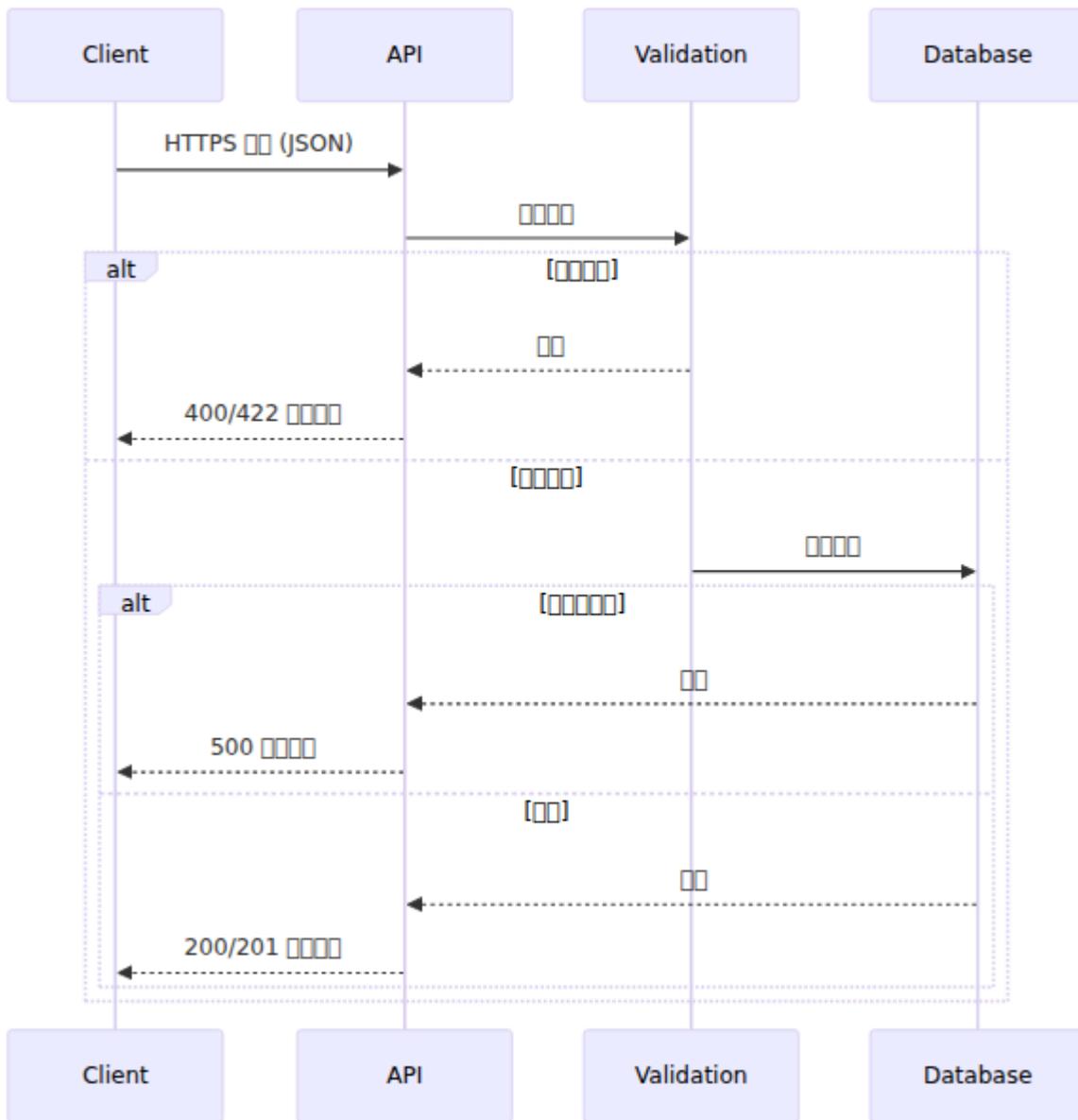
```json
{
  "data": [
    {
      "id": 1,
      "imsi": "001001123456789",
      "enabled": true,
      "ims_enabled": true,
      "sim_id": 1,
      "key_set_id": 1,
      "epc_profile_id": 1,
      "ims_profile_id": 1,
      "roaming_profile_id": 1,
      "custom_attributes": {},
      "inserted_at": "2025-10-15T10:30:00Z",
      "updated_at": "2025-10-15T10:30:00Z"
    }
  ]
}
```

通过 ID 获取用户

通过数据库 ID 获取特定用户。

路由: `GET /api/subscriber/:id`

示例请求:

字段	类型	描述
id	integer	签约用户的 ID

请求示例:

```
curl -k https://hss.example.com:8443/api/subscriber/1
```

通过 IMSI 查询用户

通过 IMSI 查询用户。

路由: GET /api/subscriber/imsi/:imsi

路径参数:

字段	类型	描述	说明
imsi	string	国际移动用户识别码	14-15 位数字

请求示例:

```
curl -k
https://hss.example.com:8443/api/subscriber/imsi/001001123456789
```

响应说明: 返回与 IMSI 对应的签约用户详细信息。

通过 MSISDN 查询用户

通过电话号码查询用户。

路由: GET /api/subscriber/msisdn/:msisdn

请求示例:

参数	类型	描述	格式
msisdn	string	移动台 ISDN 号码	1-15 位数字 (E.164)

请求示例:

```
curl -k
https://hss.example.com:8443/api/subscriber/msisdn/14155551234
```

响应格式: 响应与根据主键进行查询时完全相同。

创建订户

本节介绍如何。

方法: `POST /api/subscriber`

请求体:

```json
{
  "subscriber": {
    "imsi": "001001123456789",
    "enabled": true,
    "ims_enabled": true,
    "sim_id": 1,
    "key_set_id": 1,
    "epc_profile_id": 1,
    "ims_profile_id": 1,
    "roaming_profile_id": 1,
    "custom_attributes": {
      "note": "测试订户"
    }
  }
}
```

字段说明:

- `imsi` - 必需，14-15 位数字字符串

- `key_set_id` - □□□□□□□ □□□
- `epc_profile_id` - □□□□□□□ EPC □□□□

□□□□:

- `enabled` - □□: true
- `ims_enabled` - □□: true
- `sim_id` - □□ SIM □
- `ims_profile_id` - □□ IMS □□□□□IMS □□□□□
- `roaming_profile_id` - □□ □□□□□□□□□□□□□□□
- `msisdns` - MSISDN ID □□□□□□□□
- `static_ips` - □□ APN □□□□ □□ IP ID □□
- `custom_attributes` - □□□□□□

□□:

- □□□□□□□□ - □□□□□□
- □ MSISDN □□ - □□□□□□□□□□□
- □□ IP □□ - □□□□ IP □□□□ APN

□□□□:

```
curl -k -X POST https://hss.example.com:8443/api/subscriber \
  -H "Content-Type: application/json" \
  -d '{
    "subscriber": {
      "imsi": "001001123456789",
      "key_set_id": 1,
      "epc_profile_id": 1
    }
  }'
```

□□□□:

```
□□□□

□□□□□?
  ├─ 否 → 失败: □□□□□□
  └─ 是 → EPC □□□□□□?
            ├─ 否 → 失败: □□□ EPC □□□□
            └─ 是 → IMSI □□?
                      ├─ 是 → 失败: IMSI □□□
                      └─ 否 → □□□□
                               □□□□□□□□
                               201 □□□
```

□□□□

□□□□□□□□

□□: `PUT /api/subscriber/:id`

□□□□:

□□	□□	□□
`id`	integer	□□□□□□ ID

□□□:

```
{
  "subscriber": {
    "enabled": false,
    "ims_enabled": false,
    "epc_profile_id": 2,
    "custom_attributes": {
      "note": "□□□□"
    }
  }
}
```

□□□□□:

- `enabled` - □□/□□□□□□□
- `ims_enabled` - □□/□□ IMS □□
- `sim_id` - □□ SIM □ □□
- `key_set_id` - □□ □□□□□□□□□□
- `epc_profile_id` - □��� □□□□□□□□□
- `ims_profile_id` - □□ □□□□□□□□□
- `roaming_profile_id` - □□ □□□□
- `msisdns` - □□□□□□□□□ □□□□
- `static_ips` - □□□□□□ APN □ □□ IP

- `custom_attributes` - □□□□□□□□

□□□□:

- `imsi` - □□□□ IMSI□□□□□□□□□□□

□□:

- □□□□□□ - □□□□□□□□□□

□□□□:

```
curl -k -X PUT https://hss.example.com:8443/api/subscriber/1 \
  -H "Content-Type: application/json" \
  -d '{
    "subscriber": {
      "enabled": false
    }
  }'
```

□□□□:

- □□□□□□: `{"enabled": false}`
- □□□□□□□: `{"ims_enabled": false}`
- □□□□□□□□: `{"epc_profile_id": 2}` (□ EPC □□□□)
- □□□□□□: `{"roaming_profile_id": 3}` (□ □□□□)

□□□□

□□□□□□□□□□

□□: `DELETE /api/subscriber/:id`

□□□□:

□□	□□	□□
`id`	integer	□□□□□ ID

示例响应:

```
curl -k -X DELETE https://hss.example.com:8443/api/subscriber/1
```

说明: 此操作将永久删除订户及其所有关联的PDN 配置。此操作无法撤销。请谨慎使用 IMSI。

注意: 此操作将删除以下内容:

- 订户记录 - 核心订户身份信息
- SIM - 身份验证凭据和密钥
- 配置文件 - 关联的配置文件分配
- MSISDNs - 所有关联的电话号码

取消订户位置（网络发起）

此端点触发取消位置（CLR）过程，以从当前服务网络中分离订户 MME。

端点: `POST /api/subscriber/cancel_location`

请求体:

```
{
  "imsi": "001001123456789"
}
```

参数:

字段	类型	必填	说明
`imsi`	string	是	要取消位置的 IMSI（14-15 位数字）

响应示例:

```
curl -k -X POST
https://hss.example.com:8443/api/subscriber/cancel_location \
  -H "Content-Type: application/json" \
  -d '{"imsi": "001001123456789"}'
```

□□□□ (200 OK):

```
{
  "data": {
    "message": "□□□□□□□□□□□",
    "imsi": "001001123456789",
    "destination_host": "mme01.operator.com",
    "destination_realm": "epc.operator.com"
  }
}
```

□□□�� (404 □□□):

```
{
  "error": "□□□□□□□□□□□□ MME □□"
}
```

□□:

- □□□□□□ MME □□ S6a CLR (`subscriber_state.last_seen_mme`)
- □□ `Cancellation-Type: subscription_withdrawal`□□□□□□□□□
- □□ `CLR-Flags: {s6a_indicator: 1, reattach_required: 1}`□UE □□□□□□□
- □□□□□□□□□ `last_seen_mme` □ null□□□□□ 404
- □□□□□ **IMSI** □□□ **MSISDN**□□□□□□□/SIM□

□□□□:

- □□□□: □□□□□□□□
- □□□□: □□□□□□□□□□
- □□□□: □□□□□ MME □□□□□□□
- □□: □□□□□□□□□□□□□□□□
```

- **□□: □□□□□□□□□□□□**

**□ IMSI □□:**

□□ CLR □□□□ MSISDN □□□□:

1. **□□ MSISDN□□□□ IMSI:**

```
// □□□□□ IMSI 001001123456789 □ MSISDNs ["+1234567890",
"+9876543210"]
POST /api/subscriber/cancel_location
{"imsi": "001001123456789"}


// □□: □□□□□ CLR□□□□ MSISDN □□□□□□□□□□
```

2. **□□ IMSIs□□□□□□□□:**

```
// □□□□□□□□□□□ MSISDN □□□□□ IMSIs□□□□□□□□□□
// □□ A: IMSI 001001111111111, MSISDN "+1234567890"
// □□ B: IMSI 001001222222222, MSISDN "+1234567890"

POST /api/subscriber/cancel_location
{"imsi": "001001111111111"}

// □□: □□□□ A □□□□□□□□ B □□□□□
```

□□□□:

- **□□ IMSI:** CLR □□□□ IMSI □□□□□□□□ MSISDN
- **□□:** CLR □□□□□□□□□□□□□□□□ CLR □□□□□□□□□ MME □□□□□
- **□□□□ MME □□:** □□ MME □□□□□□□□□□□ CLR□□□□ HSS □□□□
- **□□:** □□□□ IMSI □□□□□□□□□□

□□□□:

- □□□□□□□□□□□□□
- □ IMSI □□□
- S6a □□□□

MSISDN □□

MSISDN□□□ □ MSISDN □□□

□□ MSISDN

□□□□□□□□□□□

□□: `GET /api/msisdn`

□□□□:

```
curl -k https://hss.example.com:8443/api/msisdn
```

□□ MSISDN

□□□□□□□□□□□

□□: `GET /api/msisdn/:id`

□□□□:

```
curl -k https://hss.example.com:8443/api/msisdn/1
```

□□ MSISDN

□□□□□□□□□□□□□

□□: `POST /api/msisdn`

□□□:

```json
{
  "msisdn": {
    "msisdn": "14155551234"
  }
}
```

约束:

- 长度为 1-15 位数字
- 必须唯一
- 应符合 E.164 格式（国家代码 + 号码）

示例请求:

```
curl -k -X POST https://hss.example.com:8443/api/msisdn \
  -H "Content-Type: application/json" \
  -d '{
    "msisdn": {
      "msisdn": "14155551234"
    }
  }'
```

将 MSISDN 关联到签约

创建独立的号码记录后，您可以通过更新签约记录将其关联到特定的用户签约。

将 MSISDN 关联:

□□□□

□□□□□?

□ → 检查: □□□□□□

□ → EPC □□□□□□?

□ → 检查: □□□□ EPC □□□□

□ → IMSI □□?

□ → 检查: IMSI □□□

□ → □□□□ → □□□□□□□□□ → 201 □□□

请求体 以 MSISDN 或者 IMSI 查询 可以得到不同的结果

删除 MSISDN

删除指定的分配记录

方法: `DELETE /api/msisdn/:id`

请求示例:

```
curl -k -X DELETE https://hss.example.com:8443/api/msisdn/1
```

SIM 管理

SIM 管理端点处理 SIM 卡记录，包括 ICCID、关联的认证密钥、PIN/PUK 码和 OTA 配置。SIM 卡创建和更新操作由开
通模块处理。

注意:

- 与 IMSI 关联 - 每个分配的 SIM 卡必须关联到

列出 SIM

获取所有 SIM 记录

方法: `GET /api/sim`

请求示例:

```
curl -k https://hss.example.com:8443/api/sim
```

获取 SIM

获取特定 SIM 记录

端点: `GET /api/sim/:id`

请求示例:

```
curl -k https://hss.example.com:8443/api/sim/1
```

创建 SIM

创建一个新的 SIM 卡条目。

端点: `POST /api/sim`

请求体:

```json
{
  "sim": {
    "iccid": "8991101200003204510",
    "sim_vendor": "Gemalto",
    "batch_name": "2025-Q1-Batch-01",
    "is_esim": false,
    "pin1": "1234",
    "pin2": "5678",
    "puk1": "12345678",
    "puk2": "87654321",
    "adm1": "admin-code-1",
    "kic": "0123456789ABCDEF0123456789ABCDEF",
    "kid": "FEDCBA9876543210FEDCBA9876543210"
  }
}
```

必需字段:

- `iccid` - 19-20 位数字标识符

可选的其他字段:

- `sim_vendor` - 供应商名称
- `batch_name` - 批次名称
- `is_esim` - eSIM 卡布尔标志

- `pin1`, `pin2` - 初始化的 PIN 密码
- `puk1`, `puk2` - PIN 解锁密钥
- `adm1-adm10` - 管理密钥
- `kic`, `kid` - OTA 空中下载密钥（如果适用）

示例请求:

```
curl -k -X POST https://hss.example.com:8443/api/sim \
  -H "Content-Type: application/json" \
  -d '{
    "sim": {
      "iccid": "8991101200003204510",
      "sim_vendor": "Gemalto"
    }
  }'
```

更新 SIM

更新 SIM 卡信息。

端点: `PUT /api/sim/:id`

示例请求:

```
curl -k -X PUT https://hss.example.com:8443/api/sim/1 \
  -H "Content-Type: application/json" \
  -d '{
    "sim": {
      "batch_name": "更新后的批次名"
    }
  }'
```

删除 SIM

从数据库 SIM 卡信息。

端点: `DELETE /api/sim/:id`

说明: 每个密钥集可以关联到多个 SIM。

密钥集管理

密钥集包含了生成 Milenage 认证向量所需的所有密钥参数，包括Ki、OPC/OP、AMF、SQN等。**注意 🔒** 密钥信息属于敏感数据。

内容:

- **密钥管理** - 创建、查看、更新和删除密钥集

获取密钥集列表

获取所有密钥集的列表。

请求: `GET /api/key_set`

示例请求:

```
curl -k https://hss.example.com:8443/api/key_set
```

获取密钥集

获取指定密钥集。

请求: `GET /api/key_set/:id`

示例请求:

```
curl -k https://hss.example.com:8443/api/key_set/1
```

示例响应:

```json
{
  "data": {
    "id": 1,
    "ki": "0123456789ABCDEF0123456789ABCDEF",
    "opc": "FEDCBA9876543210FEDCBA9876543210",
    "op": null,
    "amf": "8000",
    "sqn": 0,
    "authentication_algorithm": "milenage",
    "ota_counter": 0
  }
}
```

创建密钥集

创建一个新的密钥集条目。

端点: `POST /api/key_set`

请求体:

```json
{
  "key_set": {
    "ki": "0123456789ABCDEF0123456789ABCDEF",
    "opc": "FEDCBA9876543210FEDCBA9876543210",
    "amf": "8000",
    "sqn": 0,
    "authentication_algorithm": "milenage"
  }
}
```

字段说明:

- `ki` - 128 位密钥（32 个十六进制字符）
- `opc` 或 `op`：OPC 密钥或 OP 密钥
- `authentication_algorithm` - 通常设置为 "milenage"

响应示例:

- `amf` - □□: "8000"
- `sqn` - □□: 0
- `ota_counter` - □□: 0

□□□□:

- □□□□□□□□□□□□□□□□□
- Ki□OPC□OP: 32 □□□□□□□□□128 □□
- AMF: 4 □□□□□□□□□16 □□

□□□□:

```
curl -k -X POST https://hss.example.com:8443/api/key_set \
  -H "Content-Type: application/json" \
  -d '{
    "key_set": {
      "ki": "0123456789ABCDEF0123456789ABCDEF",
      "opc": "FEDCBA9876543210FEDCBA9876543210",
      "authentication_algorithm": "milenage"
    }
  }'
```

□□□□: □□□□□□□□□□□□□□□□□□□□□□□□□ API □□□□

□□□□□

□□□□□□□□□□

□□: `PUT /api/key_set/:id`

□□: □□□□ □□ □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□: □□□

□□□□□

□□□□□□□□□□

□□: `DELETE /api/key_set/:id`

注意: 请确保您的配置符合 □□ 的规定，以避免任何潜在的法律问题。

□□□□□□□

EPC □□□□

EPC□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ □□ □□□□□。

□□ EPC □□□□

□□: `GET /api/epc/profile`

□□ EPC □□□□

□□: `GET /api/epc/profile/:id`

□□ EPC □□□□

□□: `POST /api/epc/profile`

□□□:

```
{
  "apn_profiles": [],
  "name": "□□□□□□",
  "network_access_mode": "□□□□",
  "tracking_area_update_interval_seconds": 600,
  "ue_ambr_dl_kbps": 100000,
  "ue_ambr_ul_kbps": 50000
}
```

□□:

名称	描述	单位	约束值
name	□□□□□□	必填	□□□□□
ue_ambr_dl_kbps	□□□□□□	Kbps	10000-1000000
ue_ambr_ul_kbps	□□□□□□	Kbps	5000-500000
network_access_mode	□□□□	字符串	"□□□□" 或 "□□□□□□"
tracking_area_update_interval_seconds	TAU □□□	秒	600□□□□
apn_profiles	APN □□ □□ ID □□	数组	[] □ [1, 2, 3]

□□□□:

```
curl -k -X POST https://hss.example.com:8443/api/epc/profile \
  -H "Content-Type: application/json" \
  -d '{
    "apn_profiles": [],
    "name": "□□ 100Mbps",
    "network_access_mode": "□□□□",
    "tracking_area_update_interval_seconds": 600,
    "ue_ambr_dl_kbps": 100000,
    "ue_ambr_ul_kbps": 50000
  }'
```

□□:

- □□□□□□ - □□□□□□□□□□□□□
- □□□□□□ - □□□□□□□ EPC □□□□

更新 **EPC** 配置文件

请求: `PUT /api/epc/profile/:id`

描述: 按 EPC 配置文件的唯一标识符更新其内容。请 求正文的结构与创建配置文件相同。

删除 **EPC** 配置文件

请求: `DELETE /api/epc/profile/:id`

描述: 按唯一标识符删除 指定 的配置文件。

IMS 配置文件

IMS（IP 多媒体子系统）配置文件用于管理与语音和多媒体服务相关的初始过滤条件（IFC）。这些配置允许 IMS 网络根 据预 定义的规则处理服务请求。

获取 **IMS** 配置文件

请求: `GET /api/ims/profile`

创建 **IMS** 配置文件

请求: `POST /api/ims/profile`

请求体:

```
{
  "name": "默认 VoLTE",
  "ifc_template": "<IMS-XML-模板-占位符>"
}
```

字段说明:

- `name` - 此配置文件的描述性名称。
- `ifc_template` - 包含 Liquid 模板语法的 IFC（初始过滤条件）XML 内容

IFC 模板变量:

IFC 模板中可用的 Liquid 变量取决于具体的实施和上下文。

□□	□□	□□□
`{{ imsi }}`	□□ IMSI	`001001123456789`
`{{ msisdns }}`	MSISDN □□□□□□□□□	`["14155551234", "14155555678"]`
`{{ mcc }}`	□□□□□□	`001`
`{{ mnc }}`	□□□□□□	`001`

□□□□□□□□:

IFC □□□□ **Liquid** □□□□□□□ Jinja2□□□□□□□ IMS □□□□ □□ □□□

1. □□: □□ IMS □□□□□□□□□□□□□□□□□□□□ `{{ imsi }}` □ `{% for msisdn in msisdns %}`
2. □□: API □□□□□□□□□□□□□□□□□□□□□□□□□□□ XML □□
3. □□□□□: □□□□□ IMS □□□□MAA/SAA□□□□HSS□
   - □□□□□ IMS □□□□
   - □□□□□□□□□□□□□□□□□
     - `{{ imsi }}` → □□□ IMSI
     - `{{ msisdns }}` → □□□□□□□□
     - `{{ mcc }}` → □□□□□□□□□□
     - `{{ mnc }}` → □□□□□□□□□□
   - □□ Cx/Diameter □□□□ XML □□□ S-CSCF

□□□□:

```
<!-- □□□□□□ -->
{{ imsi }}

<!-- □□□□□□ -->
{% for msisdn in msisdns %}
  <MSISDN>{{ msisdn }}</MSISDN>
{% endfor %}

<!-- □□□□ -->
{{ imsi }}@ims.mnc{{ mnc }}.mcc{{ mcc }}.3gppnetwork.org
```

IFC □□□□:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<IMSSubscription>
<PrivateID>{{ imsi }}@ims.mnc{{ mnc }}.mcc{{ mcc
}}.3gppnetwork.org</PrivateID>
<ServiceProfile>
{% for msisdn in msisdns %}
<PublicIdentity>
<Identity>sip:{{ msisdn }}@ims.mnc{{ mnc }}.mcc{{ mcc
}}.3gppnetwork.org</Identity>
<Extension>
<IdentityType>0</IdentityType>
</Extension>
</PublicIdentity>
<PublicIdentity>
<Identity>tel:{{ msisdn }}</Identity>
<Extension>
<IdentityType>0</IdentityType>
</Extension>
</PublicIdentity>
{% endfor %}
<InitialFilterCriteria>
<Priority>10</Priority>
<TriggerPoint>
<ConditionTypeCNF>0</ConditionTypeCNF>
<SPT>
<ConditionNegated>0</ConditionNegated>
<Group>0</Group>
<Method>REGISTER</Method>
</SPT>
</TriggerPoint>
<ApplicationServer>
<ServerName>sip:as.ims.mnc{{ mnc }}.mcc{{ mcc
}}.3gppnetwork.org</ServerName>
<DefaultHandling>0</DefaultHandling>
</ApplicationServer>
</InitialFilterCriteria>
</ServiceProfile>
</IMSSubscription>
```

调用示例 (curl):

```
curl -k -X POST https://hss.example.com:8443/api/ims/profile \
  -H "Content-Type: application/json" \
  -d '{
    "name": "default",
    "ifc_template": "<?xml version=\"1.0\" encoding=\"UTF-8\"?>
<IMSSubscription><ServiceProfile>...</ServiceProfile>
</IMSSubscription>"
  }'
```

请求示例 (Python):

```python
import requests

response = requests.post(
    "https://hss.example.com:8443/api/ims/profile",
    json={
        "name": "default",
        "ifc_template": ifc_template_string
    },
    verify=False  # 仅测试环境使用
)
```

成功响应 (201 创建):

```json
{
  "status": "success",
  "response": {
    "id": 1,
    "name": "default",
    "ifc_template": "<?xml version=\"1.0\" encoding=\"UTF-8\"?
>..."
  }
}
```

说明:

- API 存储 IFC 模板,但不会验证其 XML
- 模板可以包含占位符,供后续替换
- `name` 字段应在系统内唯一
```

参考:

- 初始过滤条件 - IFC 概念和结构的详细信息
- 服务触发 - IMS 中的服务激活
- 共享 IFC 集合 - 优化存储

APN 配置管理

APN（接入点名称）配置定义移动设备如何连接到数据网络。

1. **APN 标识符** - 连接 APN 的名称和 IP 版本
2. **APN QoS 配置信息** - 服务质量参数配置
3. **APN 核心设置** - 网络选择、QoS设置以及 EPC 集成选项

系统将 **PCRF 集成** 进行全面的政策控制，**QoS 管理**来保障服务质量 以及 **网络选择策略** 来优化 APN 连接管理。

获取 APN 标识符

请求: `GET /api/apn/identifier`

更新 APN 标识符

请求: `POST /api/apn/identifier`

请求体:

```
{
  "apn": "internet",
  "ip_version": "ipv4v6"
}
```

IP 版本值:

- `"ipv4"` - 仅 IPv4
- `"ipv6"` - 仅 IPv6
- `"ipv4v6"` - IPv4v6双栈模式
- `"ipv4_or_ipv6"` - IPv4 或 IPv6，任选其一模式

查询 APN QoS 配置列表

请求: `GET /api/apn/qos_profile`

创建 APN QoS 配置列表

请求: `POST /api/apn/qos_profile`

请求体:

```json
{
  "name": "□□□□□□□",
  "allocation_retention_priority": 8,
  "apn_ambr_dl_kbps": 50000,
  "apn_ambr_ul_kbps": 25000,
  "pre_emption_capability": false,
  "pre_emption_vulnerability": true,
  "qci": 9
}
```

查询 APN 配置列表

请求: `GET /api/apn/profile`

创建 APN 配置列表

请求: `POST /api/apn/profile`

请求体:

```json
{
  "apn_identifier_id": 1,
  "apn_qos_profile_id": 1,
  "name": "□□□ APN"
}
```

参数说明:

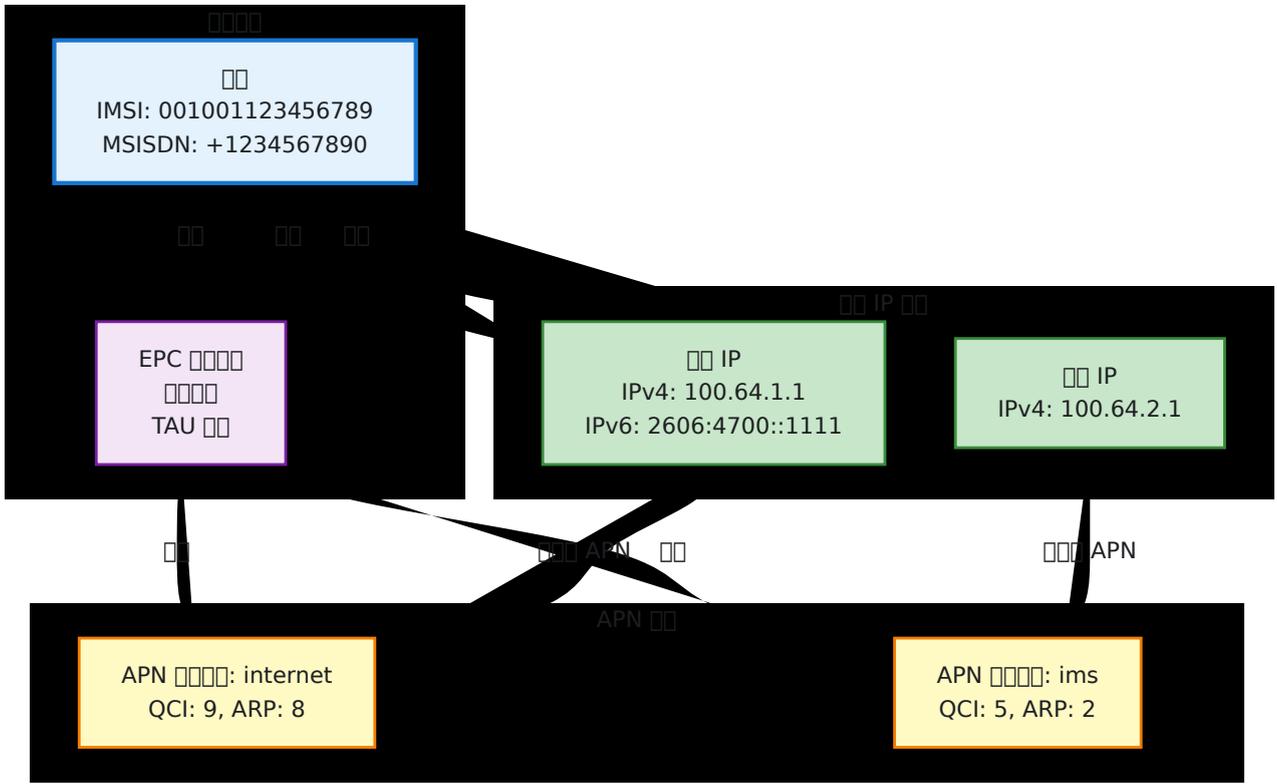- `apn_identifier_id` - □□□□□□□□ APN □□□
- `apn_qos_profile_id` - □□□□□□□□ APN QoS □□□□

另请参阅:

- **用户设备数据库** - 管理 APN 分配和订阅信息
- **EPC 节点状态** - APN 策略如何影响 EPC 节点行为

默认 IP 网络

默认 IP 网络定义了默认 APN，包括网络地址以及用于 APN 分配给用户的 IPv4 和/或 IPv6 地址池的网关或 DHCP 服务器的地址信息。

图表:

配置步骤如下所示:

订阅数据模型 - APN 关联静态地址:

关键在于 S6a 订阅数据中的 APN-配置 AVP 结构，它将用户与其服务绑定：

配置详情:

1. 上下文标识符: 每个 APN 需要唯一标识（例如：0, 1, 2…）
2. 接入点名: 直接来自 `apn_identifier.apn`（例如：“internet”或“ims”）
3. **PDN 类型**: 从 `apn_identifier.ip_version` 映射（ipv4=0，ipv6=1，ipv4v6=2，ipv4_or_ipv6=3）
4. **QoS 配置**: 直接来自 `apn_qos_profile` 表
5. **AMBR 配置**: 转换从 1000（kbps → bps 的倍数）
6. 静态的 IP 分配: 按用户查找+APN 特定的静态的 IP 的覆盖
   - 优先顺序: `subscriber.static_ips` → 按 `apn_profile_id` 过滤 → 静态 IP
   - 静态 IP 类型取决于 `apn_identifier.ip_version`
7. **VPLMN-地址-允许**: 通常设成 0（国内用）- 否则表示漫游 IP的分配被允许
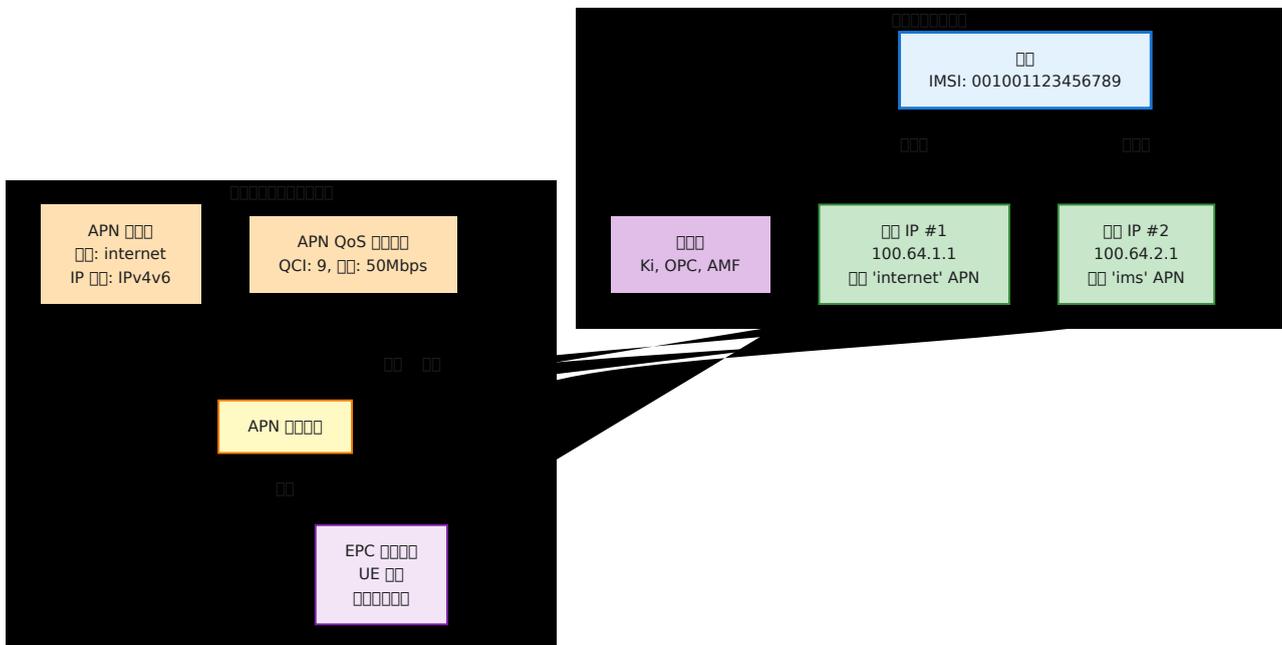
可视化表示:

关键概念:

- 每 APN 配置: 每个需要有 IP 需求的不同的 APN 的配置
- 每用户每 APN 的静态 IP: 用户可以 APN 指定接收的静态 IP 地址
- IPv4 和 IPv6 支持: 静态 IP 可以是 IPv4、或 IPv6 或两者
- 静态 IP 的优先: 静态 IP 配置优先于 动态 IP 的地址的分配
  - 如果有 IPv4 或 IPv6 的静态的地址被配置给该用户用于该特定的 APN 使用
  - 匹配的字段从对应的 IP 列读取出
  - 值从 `ipv4_static_ip` 和 `ipv6_static_ip` 列读取（如果值被配置过）
- 静态优先级: 静态配置的 IP 的优先于动态的

□□□□:

- IoT □□□□□ IP □□
- □□□□□□□□□□□□□□□ IP □□□□□□□□
- □□□□ IP □□□□□□□□□
- □□□ IP □□□□□□□□
- □□ IP □□□□□□□□□

□□□□ IP

□□□□□□ IP □□□

□□: `GET /api/epc/static_ip`

□□□□:

```
curl -k https://hss.example.com:8443/api/epc/static_ip
```

□□□□:

```json
{
  "data": [
    {
      "id": 1,
      "apn_profile_id": 5,
      "ipv4_static_ip": "100.64.1.1",
      "ipv6_static_ip": "2606:4700:4700::1111",
      "apn_profile": {
        "id": 5,
        "name": "□□□ APN",
        "apn_identifier": {
          "apn": "internet",
          "ip_version": "ipv4v6"
        }
      },
      "inserted_at": "2025-11-15T10:30:00Z",
      "updated_at": "2025-11-15T10:30:00Z"
    }
  ]
}
```

□□□□ IP

□□□□□□□ IP □□□□

□□: `GET /api/epc/static_ip/:id`

□□□□:

□□	□□	□□
`id`	integer	□□ IP □□□□ ID

□□□□:

```
curl -k https://hss.example.com:8443/api/epc/static_ip/1
```

配置静态 IP

为 APN 配置文件设置静态 IP 地址。

端点: `POST /api/epc/static_ip`

请求体:

```json
{
  "static_ip": {
    "apn_profile_id": 5,
    "ipv4_static_ip": "100.64.1.1",
    "ipv6_static_ip": "2606:4700:4700::1111"
  }
}
```

必需参数:

- `apn_profile_id` - 要配置的目标 APN 配置文件
- 以下至少一个 `ipv4_static_ip` 或 `ipv6_static_ip`

可选参数:

- `ipv4_static_ip` - IPv4 静态地址（点分十进制格式）
- `ipv6_static_ip` - IPv6 静态地址（冒号格式）

IP 地址要求:

- IPv4: 点分十进制格式（例如 `100.64.1.1`）
- IPv6: 标准冒号��制格式（例如 `2606:4700:4700::1111`）
- IPv4 和 IPv6 地址可以同时配置，IP 地址应 符合以下：
  - 必须是有效的 IP 地址格式
  - 分配的 IP 地址应在运营商分配的范围内匹配 APN
  - 静态地址分配需要预先在系统中配置

响应示例:

类型	IPv4	IPv6	配置
仅 IPv4	✓	-	{"ipv4_static_ip": "100.64.1.1"}
仅 IPv6	-	✓	{"ipv6_static_ip": "2606:4700:4700::1111"}
双栈	✓	✓	同时配置两个字段

配置示例:

仅 IPv4 静态 IP:

```
curl -k -X POST https://hss.example.com:8443/api/epc/static_ip \
  -H "Content-Type: application/json" \
  -d '{
    "static_ip": {
      "apn_profile_id": 5,
      "ipv4_static_ip": "100.64.1.1"
    }
  }'
```

仅 IPv6 静态 IP:

```
curl -k -X POST https://hss.example.com:8443/api/epc/static_ip \
  -H "Content-Type: application/json" \
  -d '{
    "static_ip": {
      "apn_profile_id": 6,
      "ipv6_static_ip": "2606:4700:4700::1111"
    }
  }'
```

双栈静态 IP:

```
curl -k -X POST https://hss.example.com:8443/api/epc/static_ip \
  -H "Content-Type: application/json" \
  -d '{
    "static_ip": {
      "apn_profile_id": 5,
      "ipv4_static_ip": "100.64.1.1",
      "ipv6_static_ip": "2606:4700:4700::1111"
    }
  }'
```

响应示例 (201 已创建):

```
{
  "data": {
    "id": 1,
    "apn_profile_id": 5,
    "ipv4_static_ip": "100.64.1.1",
    "ipv6_static_ip": "2606:4700:4700::1111",
    "inserted_at": "2025-11-15T10:30:00Z",
    "updated_at": "2025-11-15T10:30:00Z"
  }
}
```

说明:

- 静态 IP 唯一性验证 - 同一地址不可重复分配
- APN 配置绑定 - 关联 APN 规则

更新静态 IP

更新现有的静态 IP 分配。

接口: `PUT /api/epc/static_ip/:id`

请求示例:

□□	□□	□□
id	integer	□□ IP □□□ ID

□□□:

```
{
  "static_ip": {
    "ipv4_static_ip": "100.64.1.2",
    "ipv6_static_ip": "2606:4700:4700::1112"
  }
}
```

□□□□:

- `ipv4_static_ip` - □□ IPv4 □□
- `ipv6_static_ip` - □□ IPv6 □□
- `apn_profile_id` - □□ APN □□

□□□□:

- `id` - □□□□□□

□□: □□□□□□□ IP □□□□□□□□□□□□ PDN □□□□□□ PDN □□□□□□□□□ IP□□□□□□□□□□□□□□□

□□□□:

```
curl -k -X PUT https://hss.example.com:8443/api/epc/static_ip/1 \
  -H "Content-Type: application/json" \
  -d '{
    "static_ip": {
      "ipv4_static_ip": "100.64.1.2"
    }
  }'
```

□□□□ IP

□□□□ IP □□□

接口: `DELETE /api/epc/static_ip/:id`

请求参数:

参数	类型	说明
`id`	integer	静态 IP 记录的 ID

请求示例:

```
curl -k -X DELETE https://hss.example.com:8443/api/epc/static_ip/1
```

说明:

- 永久删除 IP 分配
- 释放该 APN 的关联（APN 将不再分配此地址）
- 用户再次连接 IP 时将从动态池中获得新的 IP
- 该操作仅IP 映射，不影响订户

注意: 如果用户当前正在使用此 IP，删除操作仍会成功，但现有 PDN 连接可能继续使用 IP，直到会话结束才会释放。下次连接 将使用新的地址。

静态 IP 批量操作

如需管理 IP 地址池或配置多个订户，使用批量操作可更 高效 地处理。

批量分配:

1. 创建静态 **IP**：为 一批订户 IP。
2. 使用 `static_ips` 字段的批量请求。

批量分配静态 **IP** 请求体:

```bash
# 第 1 步: 为 "internet" APN 创建静态 IP
STATIC_IP_ID=$(curl -k -X POST \
https://hss.example.com:8443/api/epc/static_ip \
  -H "Content-Type: application/json" \
  -d '{
    "static_ip": {
      "apn_profile_id": 5,
      "ipv4_static_ip": "100.64.1.1",
      "ipv6_static_ip": "2606:4700:4700::1111"
    }
  }' | jq -r '.data.id')

# 第 2 步: 创建关联静态 IP 的用户
curl -k -X POST https://hss.example.com:8443/api/subscriber \
  -H "Content-Type: application/json" \
  -d "{
    \"subscriber\": {
      \"imsi\": \"001001123456789\",
      \"key_set_id\": 1,
      \"epc_profile_id\": 1,
      \"static_ips\": [$STATIC_IP_ID]
    }
  }"
```

为现有用户分配多个静态 IP:

```bash
curl -k -X PUT https://hss.example.com:8443/api/subscriber/1 \
  -H "Content-Type: application/json" \
  -d '{
    "subscriber": {
      "static_ips": [1, 2]
    }
  }'
```

重要说明 IP约束与 APN：

系统强制执行每个用户每 IP协议、每个 IP 版本仅一个 APN。

```bash
# □ "internet" APN □□□□ IP
INTERNET_IP=$(curl -k -X POST
https://hss.example.com:8443/api/epc/static_ip \
  -H "Content-Type: application/json" \
  -d '{
    "static_ip": {
      "apn_profile_id": 5,
      "ipv4_static_ip": "100.64.1.1"
    }
  }' | jq -r '.data.id')

# □ "ims" APN □□□□ IP
IMS_IP=$(curl -k -X POST
https://hss.example.com:8443/api/epc/static_ip \
  -H "Content-Type: application/json" \
  -d '{
    "static_ip": {
      "apn_profile_id": 6,
      "ipv4_static_ip": "100.64.2.1"
    }
  }' | jq -r '.data.id')

# □□□ IP □□□□□
curl -k -X POST https://hss.example.com:8443/api/subscriber \
  -H "Content-Type: application/json" \
  -d "{
    \"subscriber\": {
      \"imsi\": \"001001123456789\",
      \"key_set_id\": 1,
      \"epc_profile_id\": 1,
      \"static_ips\": [$INTERNET_IP, $IMS_IP]
    }
  }"
```

□□□□:

- ✓ □□: □□□□ APN □□□□□□ IP
- ✗ □□: □□□□ APN □□□□□□ IP

□□□□ - □□ **APN:**

```
# 为订阅者分配 IP 并将其绑定到 APN（静态）
curl -k -X POST https://hss.example.com:8443/api/subscriber \
   -H "Content-Type: application/json" \
   -d '{
     "subscriber": {
       "imsi": "001001123456789",
       "static_ips": [1, 2]
     }
   }'

# 错误响应:
{
  "errors": {
    "static_ips": [
      "提供的静态 APN 数量与 IP 数量不匹配。您提供了两个静态 IP 100.64.1.1 绑定
internet 和 100.64.1.2 绑定 internet"
    ]
  }
}
```

参见:

- 订阅者 - 创建订阅
- 订阅者 - 更新订阅者
- 静态 IP 地址池 - 管理地址池

漫游配置

漫游配置允许您定义访问网络的规则和策略，包括 IMS 路由和网络选择。您可以 基于网络 MCC/MNC 配置漫游策略。

获取漫游配置

方法: `GET /api/roaming/profile`

更新漫游配置

方法: `POST /api/roaming/profile`

示例:

```json
{
  "roaming_profile": {
    "name": "默认漫游配置",
    "data_action_if_no_rules_match": "deny",
    "ims_action_if_no_rules_match": "deny",
    "roaming_rules": []
  }
}
```

取值:

- `"allow"` - 允许
- `"deny"` - 拒绝

字段说明:

- `data_action_if_no_rules_match` - 数据业 务（上网） 的默认动作
- `ims_action_if_no_rules_match` - IMS 业务的默认动作

获取漫游规则

请求: `GET /api/roaming/rule`

创建漫游规则

请求: `POST /api/roaming/rule`

请求体:

```
{
  "roaming_rule": {
    "name": "□□ AT&T",
    "mcc": "310",
    "mnc": "410",
    "data_action": "allow",
    "ims_action": "allow"
  }
}
```

□□:

- `mcc` - □□□□□□□3 □□□□
- `mnc` - □□□□□□□2-3 □□□□
- `data_action` - `"allow"` □ `"deny"` □□□□
- `ims_action` - `"allow"` □ `"deny"` IMS/□□□□

□□:

- □□□□ - □□□□□□□□
- □□□□ - □□□□□ Diameter □□□□□□□□

EIR □□

OmniHSS □□ S13 Diameter □□□□□□□□□□□□EIR□□□□□EIR □□□□ IMEI □□□□□□□□□□

□□□□ **EIR □□** □□□□□□□□□□□□□□□**S13 □□□□□ IMEI □□□**

□□ EIR □□

□□: `GET /api/eir/rule`

□□ EIR □□

□□: `POST /api/eir/rule`

示例:

```json
{
  "eir_rule": {
    "name": "苹果 iPhone 6",
    "imei_regex": "^35[0-9]{6}0[0-9]{7}$",
    "action": 1
  }
}
```

字段:

- `name` - 规则名称（显示用）
- `imei_regex` - 匹配 IMEI 的正则表达式
- `action` - 处理动作（0：允许；1：拒绝；2：）

动作值:

- `0` - 允许设备接入
- `1` - 拒绝设备接入
- `2` - 允许接入但上报告警

应用场景:

- 封禁被盗或丢失设备的 IMEI
- ▢▢限制特定型号（TAC 段）的设备接入
- 根据设备型号执行差异化的接入策略

相关:

- 配置指南 - S13 接口的 EIR 配置说明
- 配置指南 - OmniHSS EIR 配置

配置示例

下面是一个完整的配置文件示例：

- □□□□□ - API □□□□□□□
- □□□□ - □□□□□□□□□□
- **API** □□□□ - □□□□□□□□□

API 参考文档

状态检查

获取 API 服务状态

端点： `GET /api/status`

请求示例：

```
curl -k https://hss.example.com:8443/api/status
```

响应示例：

```
{
  "status": "ok"
}
```

说明： 用于健康检查和监控系统集成。

OmniHSS □□□□

□□

- □□□□
- □□□□
- Diameter □□
- □□□
- □□□
- □□□□
- □□□□

□□□□

OmniHSS □□ Elixir □ Erlang/OTP □□

□□□□

□□□□
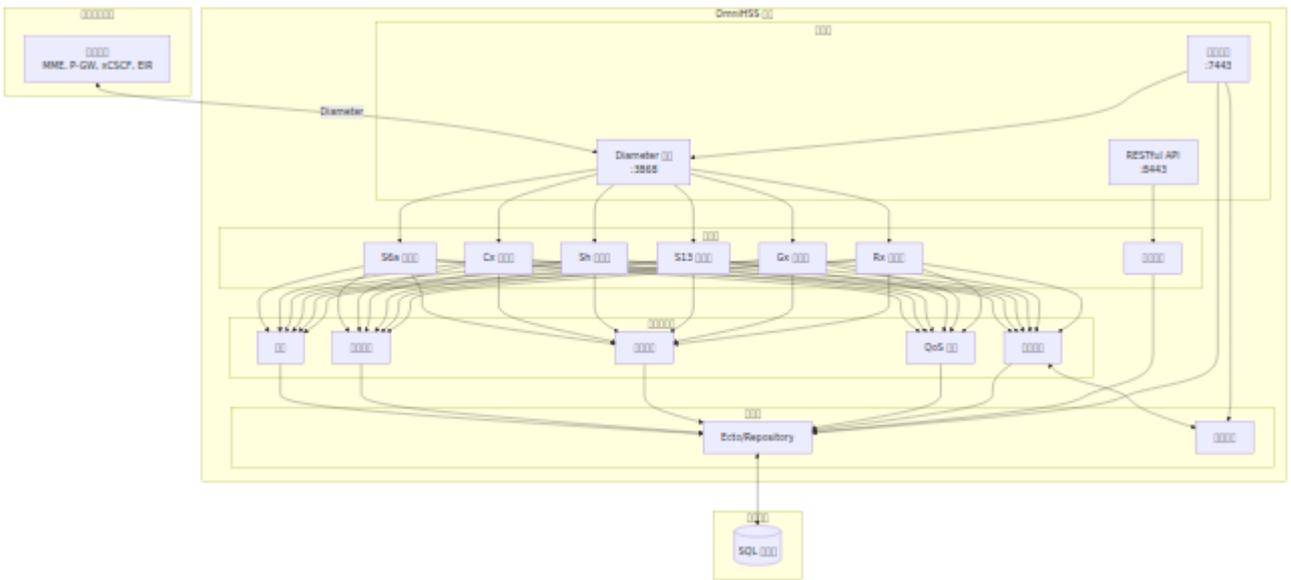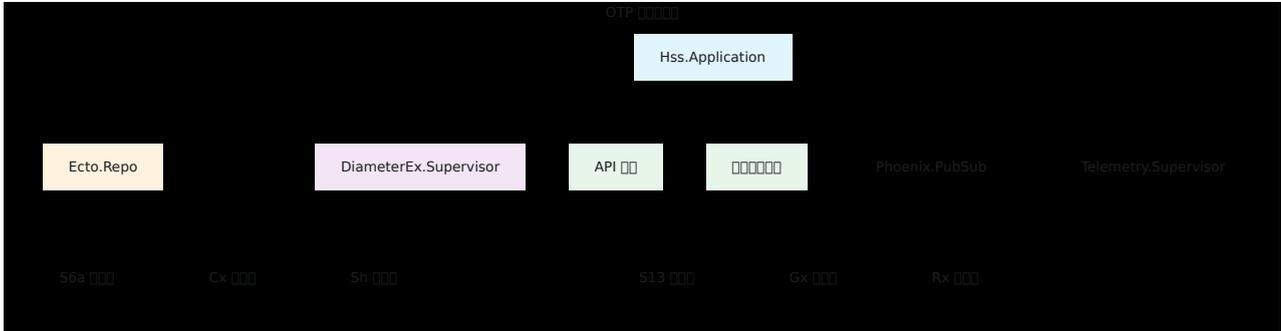
Diameter □□□□□

□□ Diameter □□□□S6a□Cx□Sh□S13□Gx□Rx□□□□□□□ DiameterEx □□□□□□□□□□□□□

1. **□ DiameterEx □□** - □□□□□□ Diameter □□ ID
2. □□□□ - □□ AVP□□□□□□□□□
3. □□□□□□ - □□□□□□□□□□□□□
4. □□□□ - □□ AVP □□ Diameter □□□□
5. □□□□ - □□□□□□ Diameter □□□□

Diameter 接口

Diameter 接口配置

OmniHSS 支持以下多种协议类型的 Diameter 接口

Diameter 接口:
:omnitouch_hss

监听器
TCP/SCTP :3868

路由器

S6a
应用 ID: 16777251

S13
应用 ID: 16777252

Cx
应用 ID: 16777216

Sh
应用 ID: 16777217

Gx
应用 ID: 16777238

Rx
应用 ID: 16777236

状态机状态

●

配置连接信息

Configured

尝试连接

Connecting

握手成功

Connected   握手失败   连接已建立

连接已断开   握手失败

收到合法 Diameter 消息   ◉   Down

检测到
连接中断

Diameter 🔲🔲🔲

:omnitouch_hss

S6a 🔲🔲 ID: 16777251	
S13 🔲🔲 ID: 16777252	
Cx 🔲🔲 ID: 16777216	
Sh 🔲🔲 ID: 16777217	
Gx 🔲🔲 ID: 16777238	
Rx 🔲🔲 ID: 16777236	

🔲🔲🔲
TCP/SCTP :3868

🔲🔲🔲

接口篇

S6a 接口 (LTE/EPC)

用于 LTE 网络中的用户认证与注册

MME

AIR ULR PUR NOR AIA   ULA PUA   NOA   CLR

S6a 服务器

MME 发起的消息

认证
向量

位置
更新

位置
更新

认证值
Milenage

签约
数据

EPC 配置
AMBR, APNs

清除位置
消息流

由 ULR 触发，或由 CLR
从 MME 发起，
通过 API 主动触发

Cx 接口 (IMS)

用于 IMS 网络的认证

□□□□□□

Configured

□□□□

Connecting

□□□□

Connected

□□□□ □□□□□

□□□□□□ □□□□

□□□□ Diameter □□

Down

□□□□
□□□□

Sh 接口 (IMS □□□□)

□ IMS □□□□□□□□□□□□□□□□□□□□□□□□

订阅者服务

UDR PUR SNR UDA PUA SNA PNR

Sh 接口层

Sh 数据
存储库

发布
订阅

数据存储
sh_repository_data

PubSub
队列

Gx 接口 (策略控制)

本节提供有关策略控制的信息，该控制用于支持 **PCRF 功能**。

P-GW

CCR-I   CCR-U   CCR-T   CCA-I   CCA-U   CCA-T   RAR

Gx 接口

PDN 连接
建立

QoS 参数

策略安装

APN 级别
AMBR, QCI

计费规则

Rx 接口 (IMS 联动)

针对 IMS 会话（例如 VoLTE 通话），应用功能（AF）会向 **PCRF 发送**

S13 接口 (EIR)

当网络需要根据终端的 IMEI（设备识别码）进行校验时，使用 **EIR 功能**

EIR

ECR ← → ECA

S13 □□□

EIR □□
□□

□□□□□
□□

| □□□ □□ | □□□ □□ | □□□ □□ | □□ □□ |

□□□

□□□□□□□

SUBSCRIBER

has — JOIN_SUB_MSISDN
uses — SIM
uses — KEY_SET
has — SUBSCRIBER_STATE
has — IMS_PROFILE
has — EPC_PROFILE
has — ROAMING_PROFILE

reference — MSISDN

has — PDN_SESSION
has — CIRCUIT_SESSION
include — JOIN_EPC_APN
include — JOIN_ROAM_RULE

has — LTE_CALL
uses — APN_PROFILE
reference — (JOIN_EPC_APN)
reference — ROAMING_RULE

has — APN_IDENTIFIER
has — APN_QOS_PROFILE

Ecto □□□□□

□□□□□

UDR PUR SNR UDA PUA SNA PNR

Sh □□□

Sh □□□
□□□□

□□
□□

□□□□□
sh_repository_data

PubSub
□□

协议流程对比

根据 Diameter 协议类型，执行不同的会话处理流程

Diameter 消息

协议判断

S6a AIR   S6a ULR   Gx SAR   Gx CCR-I

动作: 鉴权向量 + 生成密钥 + 返回结果

动作: 更新位置 + epc_profile + apns + 注册状态机

动作: 服务器地址 + ims_profile + msisdns

动作: 策略规则 + apn_profile + qos + 计费控制器

认证向量 生成完成

位置更新 处理完成

关联 IMS 用户 路由信息

策略决策 QoS 控制

□□□□

API □□

监督树结构

监督进程 - 端口 7443

Phoenix 端点

LiveView

PubSub

监督进程

接入网关
监督器

动态 Diameter
监督器

信令路由
进程

工作器

Ecto.Repo

连接监控器

Diameter 会话

配置缓存

□□□□

□□□□□

□□□
OmniHSS □□

BEAM VM
Erlang □□□

OmniHSS □□

□□□□□

□□

□□□□　　　□□□□　　　SQL □□□
　　　　　　　　　　　　　　　　□□/□□

HTTPS　　　　　　Diameter

□□□

□□□□
API :8443
□□□□ :7443

Diameter □□
:3868

□□□□□□□□

□□□□□□□□□□□□□□□□

□□□□□□

1. □□□

- Erlang/OTP □□□□□□□□□□□□□□
- □□□□ Diameter □□□□□□□□□□
- □□□□□□□□□□□□□□□□

2. □□□

- □□ Diameter □□□□□□□□□□□□□□□

- □□□□□□□□□□□□□
- □□□□□□□□□□□□

3. □□□

- □□ Diameter □□□□□□□□□
- □□□□□□□□□□□□□□□□
- □□□□□□□□□

4. □□

- □□□□□□□□□□□□□□
- □□□□□□□□□□□□□
- □□□□□□□□

5. □□□□

- □□□□□□□□□□□□
- □□□□□□□□□□□□□
- Diameter □□□□□□□
- □□□□□□□□□□□□

OmniHSS 用户手册

目录

- 产品功能概述
- 系统界面和布局
- 订阅者管理
- 数据卡管理
- Diameter 配置
- 网络配置
  - 本地 PLMN 配置
  - HSS 对等配置
- IMS 配置
- EIR 配置
- API 接口使用指南
- 故障排查指南

产品功能概述

OmniHSS 是一款高性能的归属用户

config/config.exs (□□□)

□□□□□□□□□□□□□□□□□□□□□

- □□□□□□□□□
- API □□□□
- □□□□

config/runtime.exs (□□□)

□□□□□□□□□□□□□□□□□□□□□□□□

- □□□□□□□□
- Diameter □□□□□
- □□ PLMN □□
- IMS S-CSCF □□
- □□□□□□

□□□□□□□□□

□□□□□□□□□□□□□□□□□□□ HSS □□□□
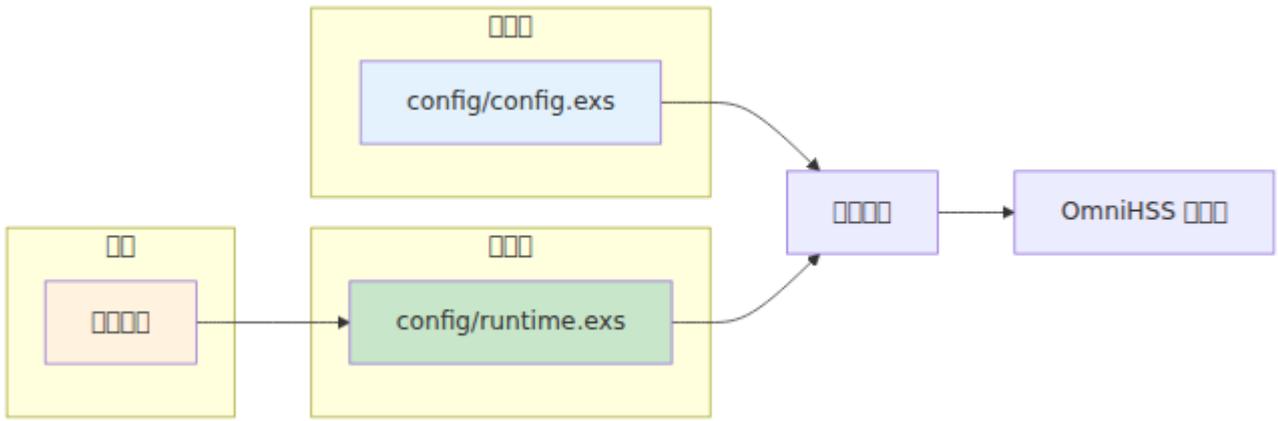
```
# config/runtime.exs

config :license_client,
  # □□□□□□ API □□□□□□□□□□□□□□□□
  license_server_api_urls:
["https://license.example.com:8443/api"],

  # □□□□□□□□
  licensee: "□□□□□□□",

  # □□□□□□
  product_name: "omnihss"
```

□□□□□□□□□

□□	□□	□□	□□
license_server_api_urls	□□□□ □□ URL □□	□	["https://10.0.0.1:8443/api"]
licensee	□□□□ □□□□ □	□	"ACME Telecom"
product_name	□□□□ □□□□ □	□	"omnihss"
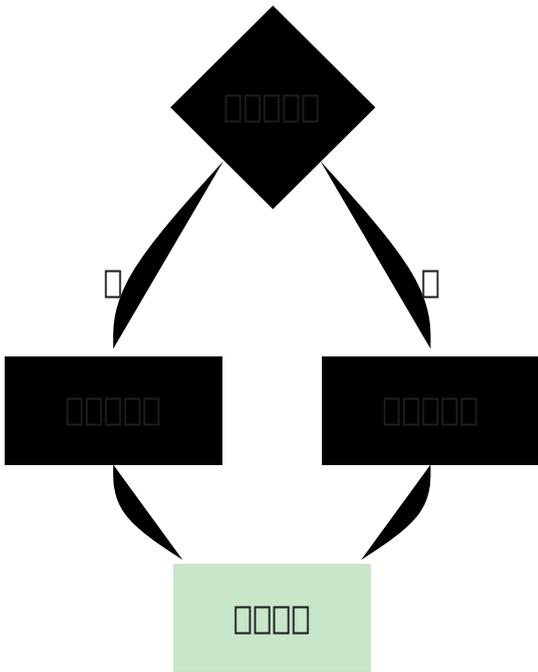
□□□□□

- □□□□□□□□□□□ HSS □□
- □□ HTTPS □□□□□□□□□□
- □□ URL □□□□□□□□□
- □□□□□□�□□□□□□□□

□□□□□

□□□□□

□□□□

□□□□□

□ □

□□□□□　□□□□□

□□□□

□□□□□□

OmniHSS □□□□□□□□□□

- □□□□□□□□□□□□□□□□□
- □□□□□ runtime.exs □□□□
- □□□□□□□□□□□□□□□□□□□□

□□□□□

□□□□□□□

```elixir
# config/runtime.exs

config :hss, Hss.Repo,
  # □□□□□□□
  username: System.get_env("DATABASE_USERNAME", "root"),
  password: System.get_env("DATABASE_PASSWORD", "password"),
  hostname: System.get_env("DATABASE_HOSTNAME", "localhost"),
  database: System.get_env("DATABASE_NAME", "omnihss"),

  # □□□□□
  pool_size:
String.to_integer(System.get_env("DATABASE_POOL_SIZE", "20")),

  # □□□□□□□□□
  timeout: 15_000,
  connect_timeout: 15_000,

  # □□□□
  show_sensitive_data_on_connection_error: false
```

□□□□□□□

□□	□□	□□	□□
`username`	SQL □□□□□□	`"root"`	□□□□□□□□□□□
`password`	SQL □□□□□	`"password"`	□□□□□□□□□□□□□
`hostname`	SQL □□□□□□□□	`"localhost"`	□□□□□ FQDN □ IP
`database`	□□□□□	`"omnihss"`	□□□□□□□□□□□□□
`pool_size`	□□□□□	`20`	□□□□□□□10-50 □□□□□

□□□□□

□□□□□
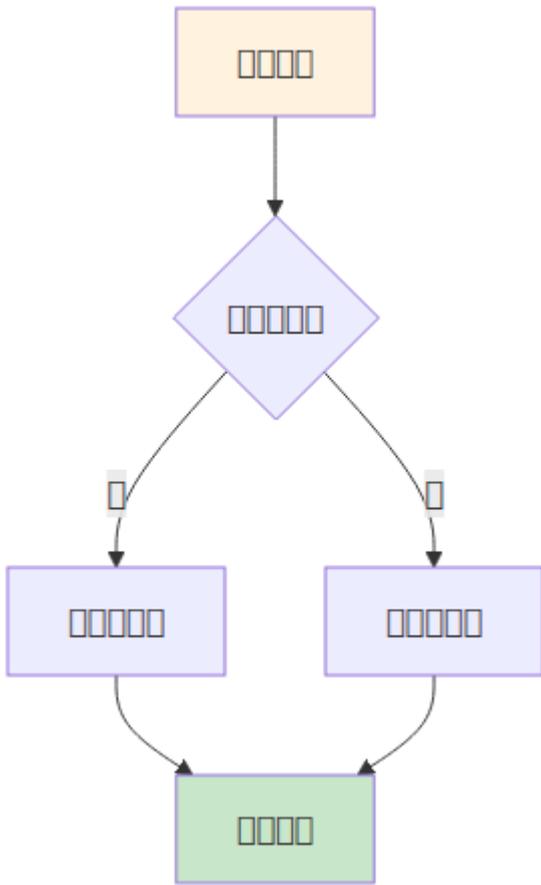
- 第 20 □□□□□
- □□"□□□□□"□□
- □□□□□□□□□□□□□□□ 10
- □□□□□□□ 4MB □□□
- □□□□□□□□□ SQL □□□□□

□□□□□□□□□□□

```
# config/runtime.exs - □□□□

config :hss, Hss.Repo,
  username: System.fetch_env!("DATABASE_USERNAME"),      # □□□□□
  password: System.fetch_env!("DATABASE_PASSWORD"),      # □□□□□
  hostname: System.get_env("DATABASE_HOSTNAME",
"db.internal.example.com"),
  database: System.get_env("DATABASE_NAME", "omnihss"),
  port: String.to_integer(System.get_env("DATABASE_PORT",
"3306")),
  pool_size:
String.to_integer(System.get_env("DATABASE_POOL_SIZE", "30")),
  ssl: true,
  ssl_opts: [
    cacertfile: "/etc/ssl/certs/mysql-ca.pem",
    verify: :verify_peer
  ]
```

Diameter 配置

Diameter 基本配置

```elixir
# config/runtime.exs

diameter_config = %{
  service_name: :omnitouch_hss,

  # 监听配置
  listen_ip: System.get_env("DIAMETER_LISTEN_IP", "10.7.25.186"),
  listen_port: String.to_integer(System.get_env("DIAMETER_LISTEN_PORT", "3868")),

  # Diameter 身份
  host: System.get_env("DIAMETER_HOST", "omnihss"),
  realm: System.get_env("DIAMETER_REALM", "epc.mnc001.mcc001.3gppnetwork.org"),

  # 产品信息
  product_name: "OmniHSS",
  vendor_id: 10415,  # 3GPP
  supported_vendor_ids: [5535, 10415],

  # 超时配置
  request_timeout: 5000,

  # 对等节点配置
  peers: [
    # 在此处配置对等节点
  ]
}

config :hss, :diameter, diameter_config
```

Diameter 🔲🔲🔲🔲

Diameter 🔲🔲 FQDN	🔲🔲🔲🔲
	🔲🔲
🔲🔲 🔲🔲	omnihss.epc.mnc001.mcc001.3gppnetwork.org
🔲🔲 🔲🔲	
omnihss epc.mnc001.mcc001.3gppnetwork.org	

🔲🔲🔲🔲🔲
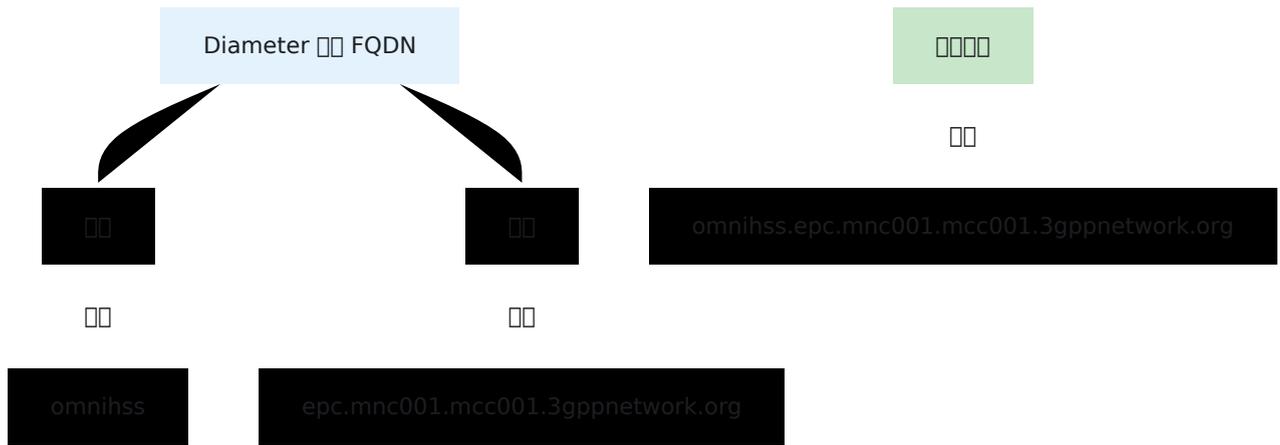
- 🔲🔲🔲 HSS 🔲🔲🔲🔲🔲🔲🔲🔲"omnihss"🔲"hss01"🔲
- 🔲🔲🔲 🔲🔲🔲 PLMN 🔲🔲🔲 Diameter 🔲🔲🔲🔲
  🔲🔲"epc.mnc001.mcc001.3gppnetwork.org"🔲
- 🔲🔲🔲🔲🔲 🔲🔲🔲 `{host}.{realm}`

🔲🔲 Diameter 🔲🔲🔲

🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲

```
# config/runtime.exs

peers: [
  # MME 口口口口口
  %{
    host: "mme01.epc.mnc001.mcc001.3gppnetwork.org",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    ip: "10.7.25.100",
    port: 3868,
    transport: :sctp,  # 口 :tcp
    applications: [:s6a]
  },

  # P-GW 口口口口口
  %{
    host: "pgw01.epc.mnc001.mcc001.3gppnetwork.org",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    ip: "10.7.25.101",
    port: 3868,
    transport: :sctp,
    applications: [:gx]
  },

  # I-CSCF 口口口口口
  %{
    host: "icscf01.ims.mnc001.mcc001.3gppnetwork.org",
    realm: "ims.mnc001.mcc001.3gppnetwork.org",
    ip: "10.7.25.102",
    port: 3868,
    transport: :tcp,
    applications: [:cx]
  }
]
```

口口口口口

口口口口口口��口口口 HSS 口口口口

```
# config/runtime.exs

diameter_config = %{
  # ... □□□□ ...
  peers: []  # □ - □□□□□□□□
}
```

Diameter □□□□□□□

□□□□□□
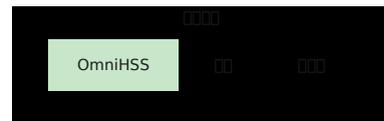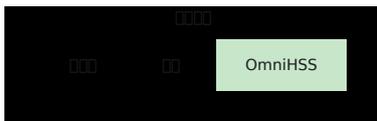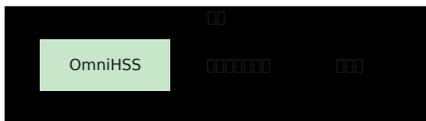
□□	□□	□□	□□
SCTP	□□□□□□□□□□□	□□□□□□□□□□□□□	□□□□ Diameter
TCP	□□□□□□□□□□□□□□	□□□□□□□□□	□□ SCTP □□□□□□

□□□□

□□ PLMN □□

□□ PLMN □□□□□□□□□□□

```
# config/runtime.exs

config :hss, :home_plmn, %{
  mcc: System.get_env("HOME_PLMN_MCC", "001"),  # □□□□□□
  mnc: System.get_env("HOME_PLMN_MNC", "001")   # □□□□□□
}
```

HSS □□□□

□□□□□□ HSS □□□□□□□□□

```
# config/runtime.exs

config :hss,
  # □□□□□□□□ Ecto □□□
  ecto_repos: [Hss.Repo],

  # MME □□□□ CLR□□□□□□□□□□
  send_clr_on_mme_change: true,

  # □□□□□□□□□□□ Diameter □□
  stop_diameter_on_database_failure: true,

  # □□□□□□□□□
  license_enforced: true,
  license_module: LicenseClient
```

HSS □□□□□

□□	□□	□□	□□
`ecto_repos`	□□□□□□ Ecto □□□□□	`[Hss.Repo]`	□□ □□ □□ □
`send_clr_on_mme_change`	□□□□□ MME □□□□ □□□□□	`true`	□□ □□ □□ □□ □□ □□ □
`stop_diameter_on_database_failure`	□□□□□□□□□ □□ Diameter □□	`true`	□□ □□ □□ □□ □□
`license_enforced`	□□□□□□□	`true`	□□ □□ □
`license_module`	□□□□□□□□□ □	`LicenseClient`	□□ □□

PLMN 标识组成

示例：

- AT&T（美国）：MCC=310，MNC=410
- Verizon（美国）：MCC=311，MNC=480
- Vodafone（英国）：MCC=234，MNC=15
- 测试网络：MCC=001，MNC=01

□□□□□□

```
# config/runtime.exs

# Diameter □□
listen_ip: System.get_env("DIAMETER_LISTEN_IP", "0.0.0.0"),  # □□□
□
# □□□□□□□
# listen_ip: "10.7.25.186",

# API □□
config :hss, HssWeb.Api.Endpoint,
  http: [
    ip: {0, 0, 0, 0},  # □□□□
    port: 8443
  ]

# □□□□□□□
config :hss, HssWeb.ControlPanel.Endpoint,
  http: [
    ip: {0, 0, 0, 0},  # □□□□
    port: 7443
  ]
```

□□□□□□□

```
                          ┌ 绑定地址 ◆ ┐

  0.0.0.0              监听 IP                127.0.0.1
  (□□□□)         (□□□192.168.1.10)        (□□□□□)

   □□                  □□□                   □□
  □□□□□               □□□□□                □□□□□□□
```

- 0.0.0.0 (□□□□)
- 监听 IP (□□□192.168.1.10)
- 127.0.0.1 (□□□□□)

IMS 配置

S-CSCF 选择配置
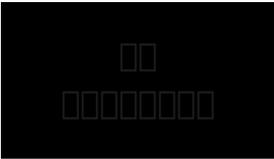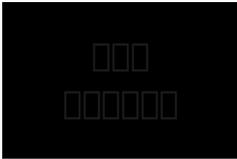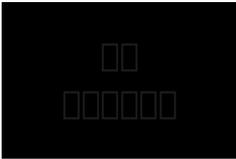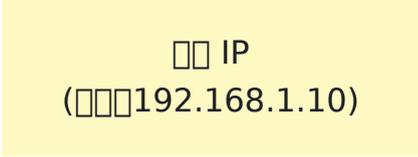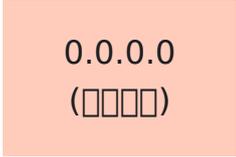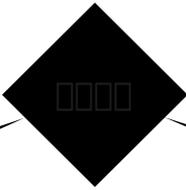
```
# config/runtime.exs

config :hss, :ims, %{
  scscf: %{
    # 选择策略:random_peer 或 :round_robin
    selection_method: :random_peer,

    # 可用 S-CSCF 服务器列表
    peers: [
      %{
        host:
"sip:scscf01.ims.mnc001.mcc001.3gppnetwork.org:5060",
        capabilities: []  # 可选能力集定义
      },
      %{
        host:
"sip:scscf02.ims.mnc001.mcc001.3gppnetwork.org:5060",
        capabilities: []
      }
    ]
  }
}
```

S-CSCF 工作流程

工作流程图

值	行为	特点
`:random_peer`	随机选择一个 S-CSCF	负载分布均匀
`:round_robin`	轮询选择 S-CSCF	负载分布均匀

IMS 服务器

配置的IMS 服务和 EPC 服务器的域名

```
# EPC 域名
"epc.mnc001.mcc001.3gppnetwork.org"

# IMS 域名
"ims.mnc001.mcc001.3gppnetwork.org"
```

EIR 配置

本节将 **EIR 配置** 涵盖了设备��份认证的核心参数。

未知设备处理策略

```
# config/runtime.exs

config :hss, :eir, %{
  # 定义未知设备的处理方式
  unknown_equipment_behaviour: :whitelist
  # 选项：
  #   :whitelist - 允许所有设备
  #   :blacklist - 阻止所有设备
  #   :greylist - 监控但允许所有设备
  #   :reject_unknown_equipment - 拒绝未知识别的设备
}
```

工作流程图

工作流程图

状态	含义	操作
`:whitelist`	已授权的设备 IMEI	允许接入网络
`:blacklist`	被禁止的设备 IMEI	拒绝接入
`:greylist`	需要监控的设备 IMEI	允许接入
`:reject_unknown_equipment`	未知设备的默认处理	拒绝接入

提示： 建议将已知 `:whitelist` 设备和需要监控的设备 `:greylist`，而将 `:blacklist` 用于明确禁止的设备

API 网关配置示例

API 端点配置

```elixir
# config/config.exs

config :hss, HssWeb.Api.Endpoint,
  url: [host: "localhost"],
  render_errors: [view: HssWeb.ErrorView, accepts: ~w(json)],
  pubsub_server: Hss.PubSub,

  # HTTPS 配置
  https: [
    port: 8443,
    cipher_suite: :strong,
    certfile: "priv/cert/omnitouch.crt",
    keyfile: "priv/cert/omnitouch.pem"
  ]
```

控制面板配置

```elixir
# config/config.exs

config :hss, HssWeb.ControlPanel.Endpoint,
  url: [host: "localhost"],
  render_errors: [view: HssWeb.ErrorView, accepts: ~w(html json)],
  pubsub_server: Hss.PubSub,
  live_view: [signing_salt: "some-secret"],

  # HTTPS 配置
  https: [
    port: 7443,
    cipher_suite: :strong,
    certfile: "priv/cert/omnitouch.crt",
    keyfile: "priv/cert/omnitouch.pem"
  ]
```

TLS 配置架构
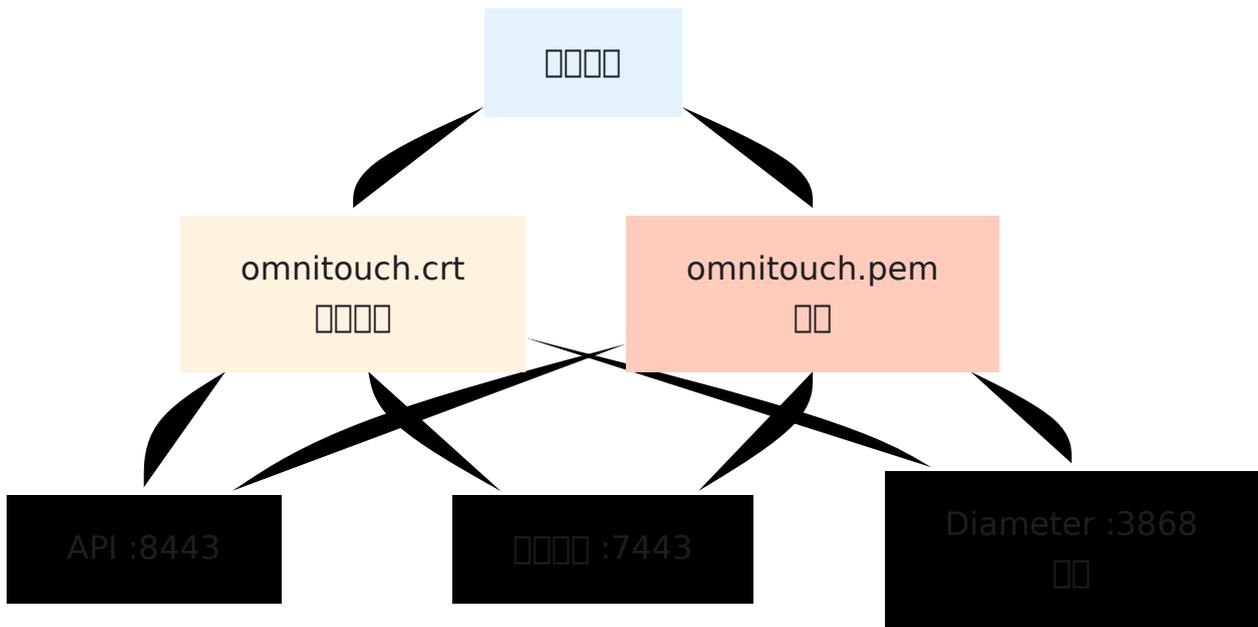
```
                    配置来源
                   /        \
        omnitouch.crt      omnitouch.pem
          证书文件             私钥
         /      \            /      \
   API :8443   管理接口 :7443   Diameter :3868
                                   接口
```

证书要求：

- 有效的 X.509 证书
- 正确的格式
- 包含完整的证书链信息
- CN 或 SAN 必须匹配主机名

配置示例：

```
https: [
  port: 8443,
  cipher_suite: :strong,
  certfile: System.get_env("TLS_CERT_FILE",
"/etc/ssl/certs/omnihss.crt"),
  keyfile: System.get_env("TLS_KEY_FILE",
"/etc/ssl/private/omnihss.key"),
  cacertfile: System.get_env("TLS_CA_FILE", "/etc/ssl/certs/ca-
bundle.crt")
]
```

□□□□□□□

□□□□□□


```
         □□□□
    ┌─────┴─────┐
    │           │
0.0.0.0      □□ IP        127.0.0.1
(□□□□)    (□□□192.168.1.10)   (□□□□□)
    │           │              │
   □□          □□□             □□
 □□□□□□□      □□□□□□□        □□□□□□□□□
```

□□□□□

□□□□

- ☐ □□□□□□□□□□□□□□□□
- ☐ □□ PLMN□MCC□MNC□
- ☐ Diameter □□□□□□
- ☐ Diameter □□ IP □□□
- ☐ API □□□□□□□ TLS □□
- ☐ □□□□□□□□□□□□ URL□□□□□□�□□□□□□□□
- ☐ HSS □□□□□send_clr_on_mme_change□stop_diameter_on_database_failure□

□□□□□□

- ☐ □□ Diameter □□□□□□□□□□□□□□□□
- ☐ □□□□□□□ Diameter □□□□□□ 3868□
- ☐ □□□□□□□ HTTPS □□□□□ 7443□8443□

- ☐ Diameter □□□ DNS □□

IMS □□□□□□□ IMS □□□

- ☐ □□ S-CSCF □□□□□
- ☐ □□ S-CSCF □□□□
- ☐ □□ IMS □□

□□□□

- ☐ □□ EIR □□
- ☐ □□□□□□□□
- ☐ □□□□□□□□

□□□□

□□□□□□□

1. **□□□□□**

   > □□□□□□□□□□□□□□

2. **□□□□□□□□**

   ```
   □□ https://[hostname]:7443
   □□□□□□□□□
   ```

3. **API □□□**

   ```
   curl -k https://[hostname]:8443/api/status
   ```

4. **Diameter □□□**

   ```
   □□□□□□□ Diameter □□
   □□□□□□□
   ```

5. □□□□□

□□□□□□□□□□□□
□□□□□□ SQL □□□

□□□□□□□□□□

```
# config/runtime.exs - □□□□□□□

import Config

#
========================================================================
# □□□□□□
#
========================================================================
config :hss, Hss.Repo,
  username: System.fetch_env!("DATABASE_USERNAME"),
  password: System.fetch_env!("DATABASE_PASSWORD"),
  hostname: System.get_env("DATABASE_HOSTNAME", "db.omnihss.internal'
  database: System.get_env("DATABASE_NAME", "omnihss"),
  port: String.to_integer(System.get_env("DATABASE_PORT", "3306")),
  pool_size: String.to_integer(System.get_env("DATABASE_POOL_SIZE", '
  timeout: 15_000,
  connect_timeout: 15_000,
  ssl: true,
  ssl_opts: [
    cacertfile: "/etc/ssl/certs/mysql-ca.pem",
    verify: :verify_peer
  ]

#
========================================================================
# □□□□□□□□□
#
========================================================================
config :license_client,
  license_server_api_urls: [System.get_env("LICENSE_SERVER_URL",
"https://license.example.com:8443/api")],
  licensee: System.get_env("LICENSE_ORGANIZATION", "□□□□"),
  product_name: "omnihss"

#
========================================================================
# □□ PLMN □ HSS □□□□
#
========================================================================
```

```elixir
config :hss,
  ecto_repos: [Hss.Repo],
  home_plmn: %{
    mcc: System.get_env("HOME_PLMN_MCC", "001"),
    mnc: System.get_env("HOME_PLMN_MNC", "001")
  },
  send_clr_on_mme_change: true,
  stop_diameter_on_database_failure: true,
  license_enforced: true,
  license_module: LicenseClient

#
# ======================================================================
# Diameter 配置
#
# ======================================================================
diameter_config = %{
  service_name: :omnitouch_hss,
  listen_ip: System.get_env("DIAMETER_LISTEN_IP", "10.7.25.186"),
  listen_port: String.to_integer(System.get_env("DIAMETER_LISTEN_PORT",
"3868")),
  host: System.get_env("DIAMETER_HOST", "omnihss01"),
  realm: System.get_env("DIAMETER_REALM",
"epc.mnc001.mcc001.3gppnetwork.org"),
  product_name: "OmniHSS",
  vendor_id: 10415,
  supported_vendor_ids: [5535, 10415],
  request_timeout: 5000,
  peers: [
    %{
      host: "mme01.epc.mnc001.mcc001.3gppnetwork.org",
      realm: "epc.mnc001.mcc001.3gppnetwork.org",
      ip: "10.7.25.100",
      port: 3868,
      transport: :sctp,
      applications: [:s6a]
    }
  ]
}

config :hss, :diameter, diameter_config

#
# ======================================================================
```

```
# IMS □□
#
====================================================================
config :hss, :ims, %{
  scscf: %{
    selection_method: :random_peer,
    peers: [
      %{host: "sip:scscf01.ims.mnc001.mcc001.3gppnetwork.org:5060"},
      %{host: "sip:scscf02.ims.mnc001.mcc001.3gppnetwork.org:5060"}
    ]
  }
}

#
====================================================================
# EIR □□
#
====================================================================
config :hss, :eir, %{
  unknown_equipment_behaviour: :whitelist
}

#
====================================================================
# API □□□□
#
====================================================================
config :hss, HssWeb.Api.Endpoint,
  http: [ip: {0, 0, 0, 0}, port: 8443],
  https: [
    port: 8443,
    cipher_suite: :strong,
    certfile: System.get_env("TLS_CERT_FILE", "/etc/ssl/certs/omnihss
    keyfile: System.get_env("TLS_KEY_FILE", "/etc/ssl/private/omnihss
  ],
  url: [host: System.get_env("API_HOST", "api.omnihss.internal"), por
8443]

#
====================================================================
# □□□□□□□□□
#
====================================================================
config :hss, HssWeb.ControlPanel.Endpoint,
```

```
    http: [ip: {0, 0, 0, 0}, port: 7443],
    https: [
      port: 7443,
      cipher_suite: :strong,
      certfile: System.get_env("TLS_CERT_FILE", "/etc/ssl/certs/omnihss
      keyfile: System.get_env("TLS_KEY_FILE", "/etc/ssl/private/omnihss
    ],
    url: [host: System.get_env("CP_HOST", "hss.omnihss.internal"), port
```

OmniHSS □□□□□□

□□

□□□□□□

OmniHSS □□ Diameter □□□□□□□□□□□□□□□□ Phoenix LiveView □□□□□□□□□□□□□□□□□□□□□□□□

□□□□

- □□□□ - □□□□□□□
- □□□□ - □□□□□□□□□□□□
- **Diameter □□** - □□□□□□□□□
- □□□□ - □□□□□□□
- □□□□□ - □□□□□□□□

访问地址

```
URL: https://[hostname]:7443
Protocol: 强制 HTTPS
Port: 7443（可配置）
Certificate: 在 config/config.exs 中配置
```
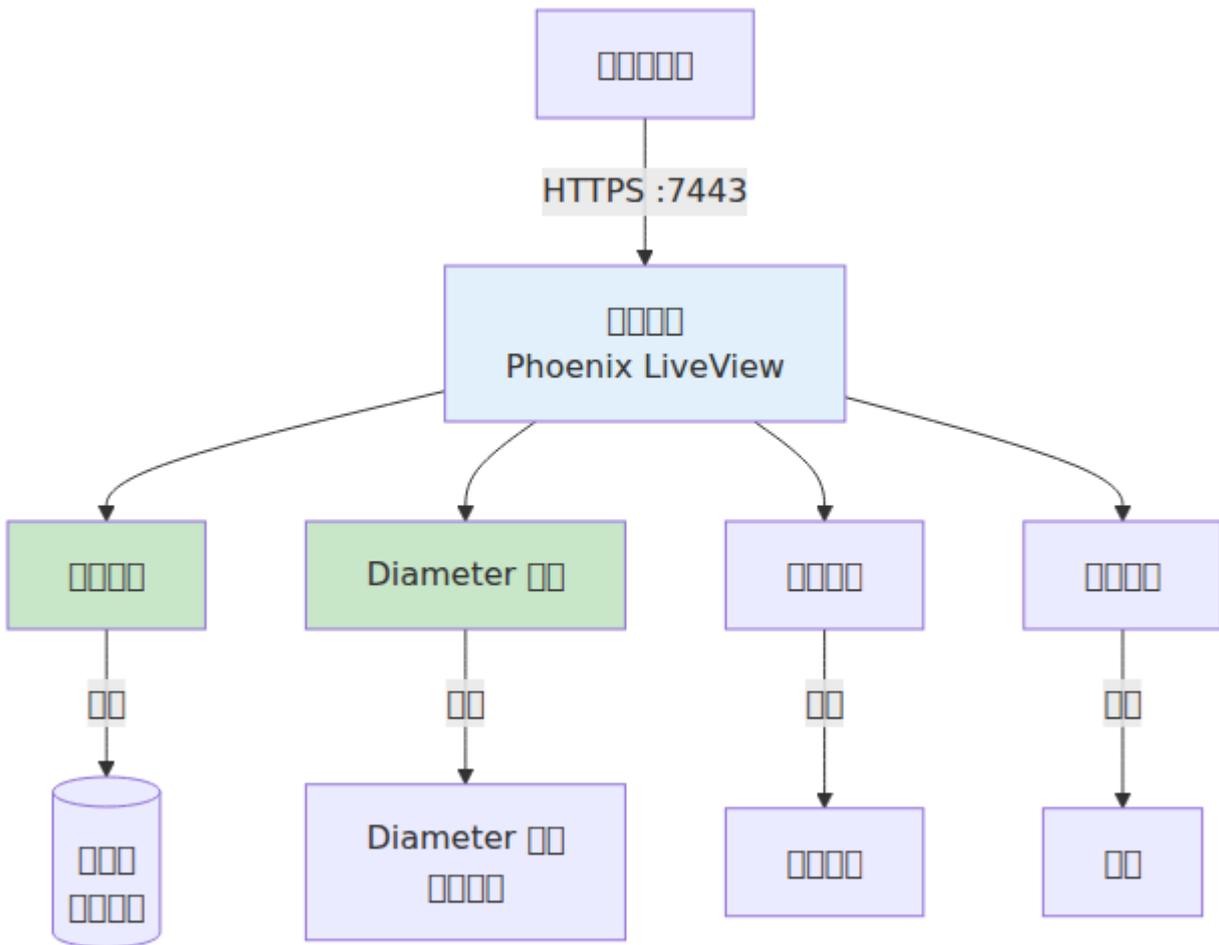
系统架构图

系统功能模块

用户认证

1. 支持的认证方式

2. □□□ `https://[hostname]:7443`
3. □□ TLS □□□□□□□□□□□
4. □□□□□□□□□□□□□□□

TLS □□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□ □□□□□□□□□□□□□□□□□□□

□□□□

- □□ **7443** □□□□□□□□□□□□□□□
- **HTTPS** □□□□□ - □□□ HTTP
- □□□□□ □□□□□□□□□□□ 7443

□□□□□

□□□□□□□□□□□□□□LiveView, WebSockets□□

- Chrome/Chromium□□□□
- Firefox
- Safari
- Edge

□□□ □□□ Internet Explorer□

□□□□

URL: `https://[hostname]:7443/overview`

□□□□□□□□□□□□□□□□□□□

□□□□

□□
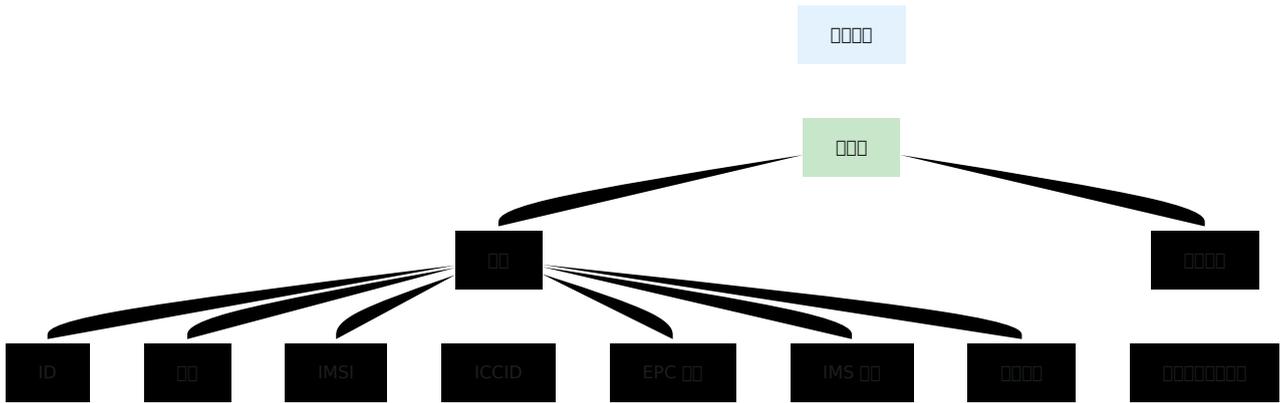
□	□□	□
ID	□□□□□□ ID	□□
□□	□□□□	✓□□□□/ ✗□□□□
IMSI	□□□□□□□□□	14-15 □□□
ICCID	SIM □ ID	19-20 □□□□□ "N/A"
EPC □□	□□□□□□□□□	□□□□□ ID
IMS □□	□□□□□□□□□	□□□□□ID □ "N/A"
□□□□	□□□□□□	□□□□□ID □ "N/A"

□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□

□□□□

位置：

- **MCC** - 移动国家代码（3 位数字）
- **MNC** - 移动网络代码（2-3 位数字）
- **TAC** - 跟踪区域代码
- **小区 ID** - 服务小区标识
- **eNodeB ID** - 基站标识符
- **ECI** - E-UTRAN 小区标识符

网络功能

信息：

- 服务节点 **MME** - 当前服务的 MME 标识符
- 服务节点标识 - MME 的 Diameter 标识
- **RAT** 类型 - 无线接入技术类型（"E-UTRAN"表示 LTE）
- 始发主机标识 - 发起 Diameter 请求的主机

IMS 信息

信息：

- 服务器 **S-CSCF** - 已分配的 S-CSCF SIP URI
- **IMS** 公共身份 - SIP URI（例如sip:+14155551234@ims.example.com）
- 首选呼叫 **P-CSCF** - 可选，从 HSS 到 P-CSCF
- 首选呼叫 **I-CSCF** - 可选，从 HSS 到 I-CSCF

网络功能

□□□

- **PDN □□ - □□□□□□□□□**
- □□□□ - □□ VoLTE □□□□□

□□□□□

| □□
□□□ | □□□
□□□□ MME | PDN □□
□□□□ | IMS □□□
□□□□□□□ | □□□
□□ VoLTE □□ |

□□□□□□□

- □□□ □□□□□□□□□ MME
- □□□□ □□□□□ MME □□□□□□□□□
- **PDN □□□** PDN □□□□ > 0
- **IMS □□□□** □□□ S-CSCF □□
- □□□□ □□□□□□ > 0

□□□□

□□□□□ **1** □ □□□□□□□□□□□□□□

□□□□□

- □□□□□□□□□□□□□□□□□□
- □□□□□□□
- □□□□□□

□□□□

1. □□□□□□
   - □□□□□□□□□

- ○ 支持多种协议的流程跟踪展示
- ○ 支持 IMS 注册流程

2. 注册流程

- ○ 注册请求的发起
- ○ 支持多种网络类型的注册流程展示
- ○ 鉴权流程展示
- ○ 注册成功的确认

3. 呼叫流程

- ○ 呼叫请求的发起
- ○ 支持 PDN 连接建立
- ○ 支持多种 VoLTE 流程

Diameter 网络

URL: `https://[hostname]:7443/diameter`

Diameter 网络用于查询 Diameter 网络的拓扑结构信息。

网络拓扑

配置

键	描述	值
主机名	Diameter 服务器地址	FQDN
领域	Diameter 领域	域名
IP:端口	网络地址	IP 地址和端口
协议	传输协议	TCP 或 SCTP
安全	传输安全	无加密 / 已加密

应用程序

应用程序与接口

每个应用程序对应于特定的功能：

主要功能

- 用户管理 - 与 HSS 进行身份验证
- 移动管理 - 支持用户移动性
- 应用 **ID** - 识别特定 Diameter 应用

应用 ID 示例：
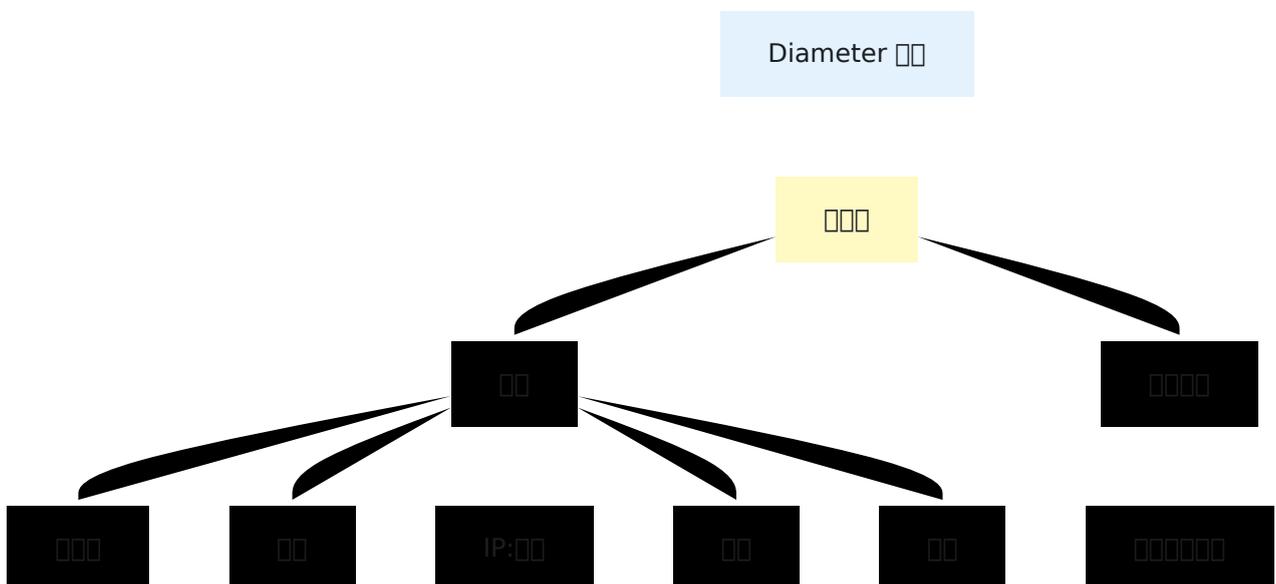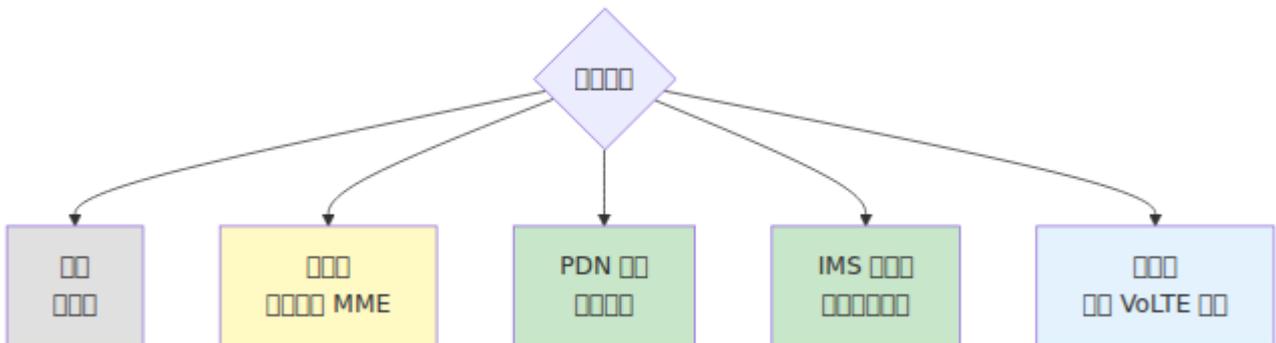
- 16777251 - S6a (MME)

- 16777238 - Gx (P-GW)
- 16777216 - Cx (I-CSCF, S-CSCF)
- 16777217 - Sh (□□□□□)
- 16777236 - Rx (P-CSCF)
- 16777252 - S13 (EIR □□□□□□□□□)

□□□□□□

```
OmniHSS                          Diameter Peer
```

[HSS □□□□□□□□]

CER (□□□□□□)

CEA (□□□□□□)

□□□□□□

[□□□□□□□□□]

CER

CEA

□□□□□□

[□□□□]

DWR (□□□□□□□)

DWA (□□□□□□□)

[□□□□]

DWR

□□□ - □□□□□□

□□□□□□

```
OmniHSS                          Diameter Peer
```

□□□□
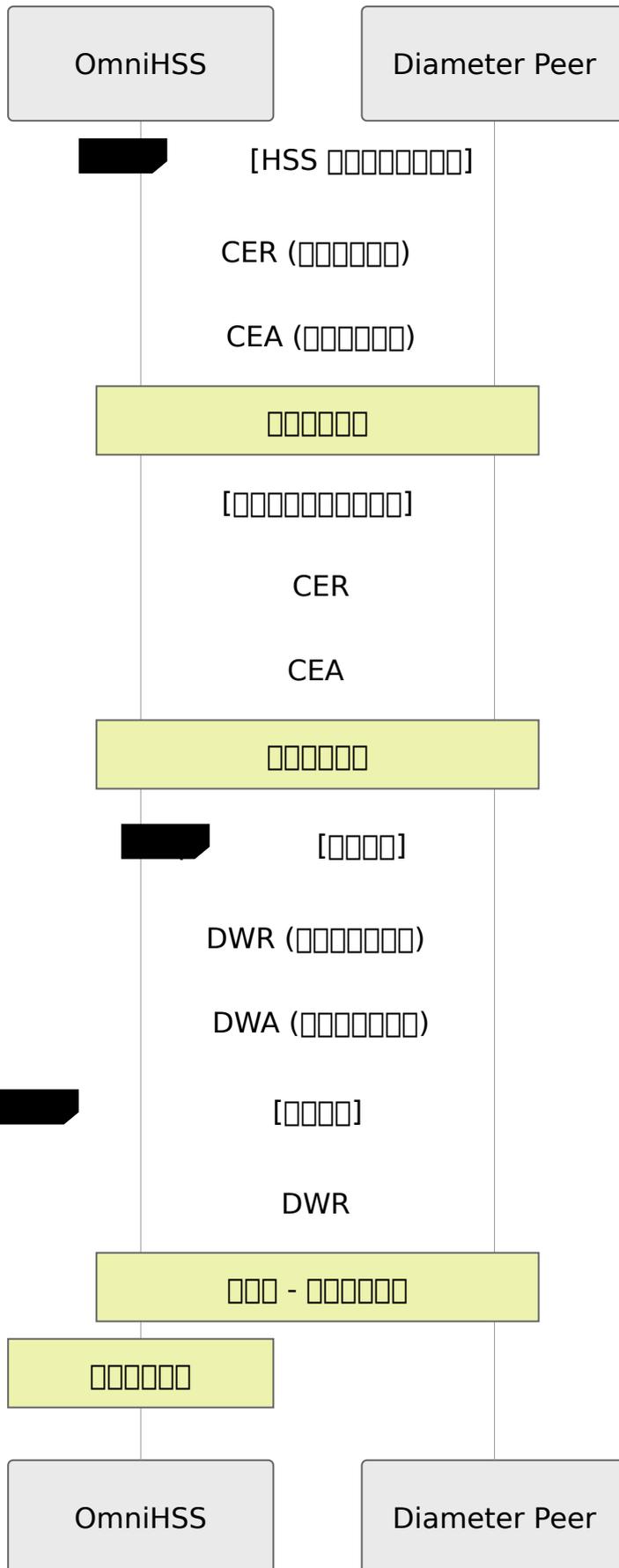
Diameter □□□ **1** □ □□□□□

□□□□

1. □□□□□

   - □□□□□□□□□□□□□
   - □□□□□□□□□□□
   - □□□□□□□

2. □□□□

   - □□□□□□□□□
   - □□□□□□□□TCP □ SCTP□
   - □□□□ ID □□□□□□
   - □□□□□□□□□□□

3. □□□□

   - □□□□□□□□□□
   - □□□□□□□
   - □□□□□□□□□

□□□□

□□□□□□□

□□□□□

1. □□□□□
2. □□□□□□□□□□
3. □□□□□□
4. Diameter □□□□□
5. □□□□□□□□□ TLS□

□□□□□□□

1. 确认网络连通：`ping [peer-ip]`
2. 确认端口是否开放：`telnet [peer-ip] 3868`
3. 检查防火墙配置
4. 确认 HSS 服务正常运行状态
5. 检查配置的 Diameter 参数与 HSS 一致

连接建立后断开问题

可能原因：

1. 心跳超时
2. 协议版本不兼容问题
3. 能力协商失败
4. Diameter 参数错误

排查和解决步骤：

1. 检查心跳配置
2. 确认协议版本兼容性设置
3. 查看能力协商日志
4. 确认对端的主机 ID 配置正确

系统管理

URL: `https://[hostname]:7443/application`

系统管理模块用于系统级的监控和管理。

功能

- 系统监控 - Erlang VM 的运行状态监控
- 日志管理 - 系统日志的查看管理
- 数据库管理 - OmniHSS 数据库管理
- **Erlang VM 管理** - 虚拟机节点管理

□□□□

□□□□

1. □□□□

   - □□□□□□□□□
   - □□□□□□□□□□□□
   - □□□□□□□□

2. □□□□

   - □□□□□□□
   - □□□□□□□□
   - □□□□□□□□

3. □□□□

   - □□□□□
   - □□□□□□
   - □□ Erlang VM □□

系统配置
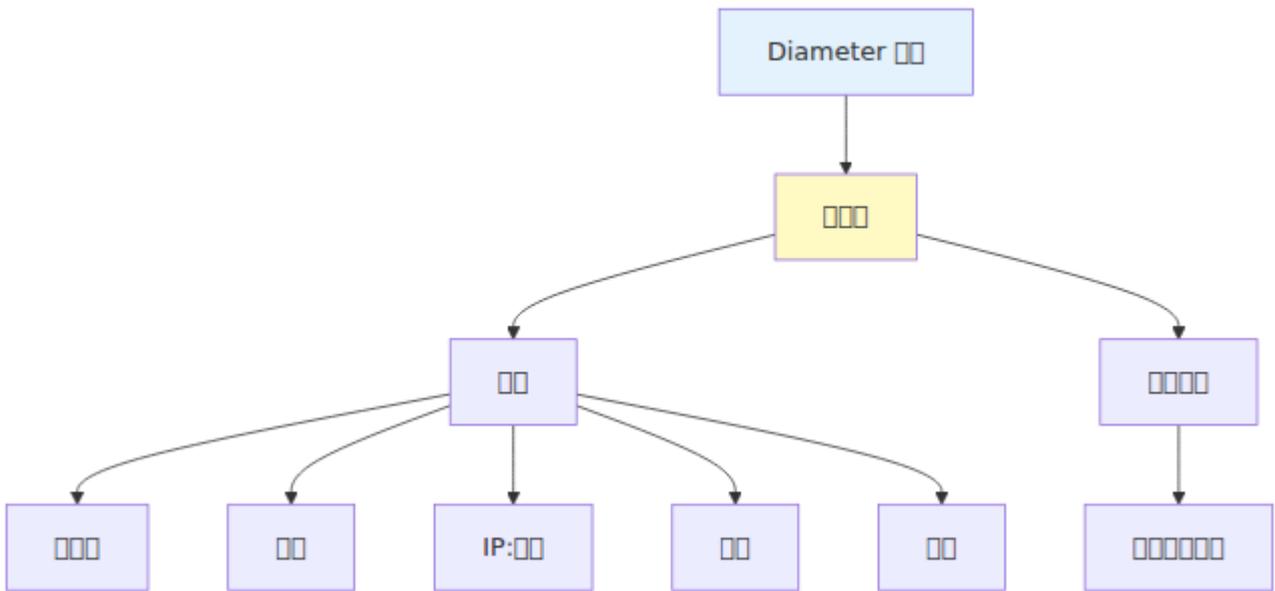
URL: `https://[hostname]:7443/configuration`

配置和管理 OmniHSS 系统的运行时参数。

权限

- 系统配置 - 查看所有配置参数
- 系统配置 - 修改基本配置
- 系统配置 - 修改高级配置

配置分类

操作流程

1. 加载配置

   - 读取 runtime.exs 中的初始配置
   - 应用环境变量覆盖
   - 加载 Diameter 栈配置

2. 修改配置

   - 选择配置项目
   - 输入新值并进行有效性验证
   - 保存并应用新配置

3. 验证

   - 检查配置项之间的依赖关系

- ○ □□□□□□□□□

□□□□□ □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□

□□□□□

□□□□□□□□□□□□□□□□□□□□□□□

□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

- **Ctrl+R / F5** - □□□□□□□□□□□□□□□□□□□□□□
- **Ctrl+F** - □□□□□□
- **Ctrl+T** - □□□□□□□□□□□□□□

□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□
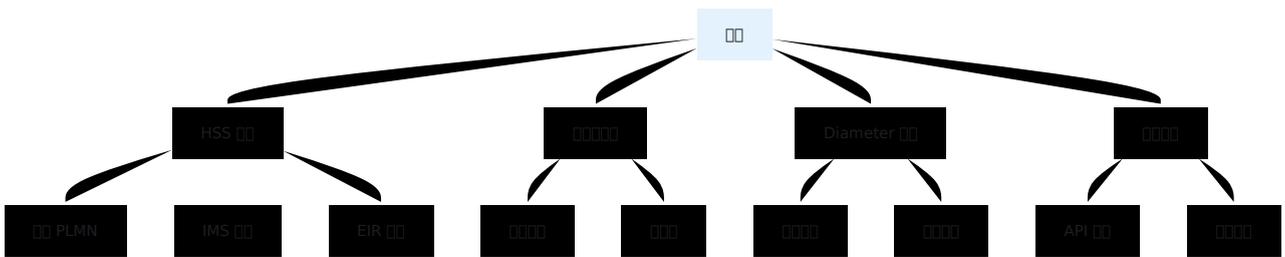
- □□ 1□□□□□□□□□□□□□
- □□ 2□Diameter □□□□□□□□□□
- □□ 3□□□□□□□□□□□□□

□□□□□□□□□□□□□□

□□□□□

□□□□□□□□□□□□□□□□□□□□□□�□�□�□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□ 1920x1080 □□□□□□□□□□□□□

□□□□□□

□□□□

1. □□□□

   - □□□□□□□□□□□
   - □□□□□□□□□□□□□
   - □□ Diameter □□ - □□□□□□□□□

2. □□□□

   - □□□□□□□□□□□□□□
   - □□□□□□□□
   - □□ Diameter □□□□□□□□□

3. □□□□

   - □□□□□□□□

- □□□□□□□□□□□□□□□
- □□□□□□

□□□□□□□□

OmniCharge | OmniRAN | Downloads | 🔤 □□□□ ▼

OmniHSS   Diameter Peer

CER (□□□□□□)

CEA (□□□□□□)

□□□□□□

[□□□□□□□□□□]

CER

CEA

□□□□□□

loop   [□□□□]

DWR (□□□□□□□)

DWA (□□□□□□□)

alt   [□□□□]

DWR

□□□ - □□□□□□

□□□□□□

OmniHSS   Diameter Peer

□□□□

□□□□□□□□□□□□□□□□
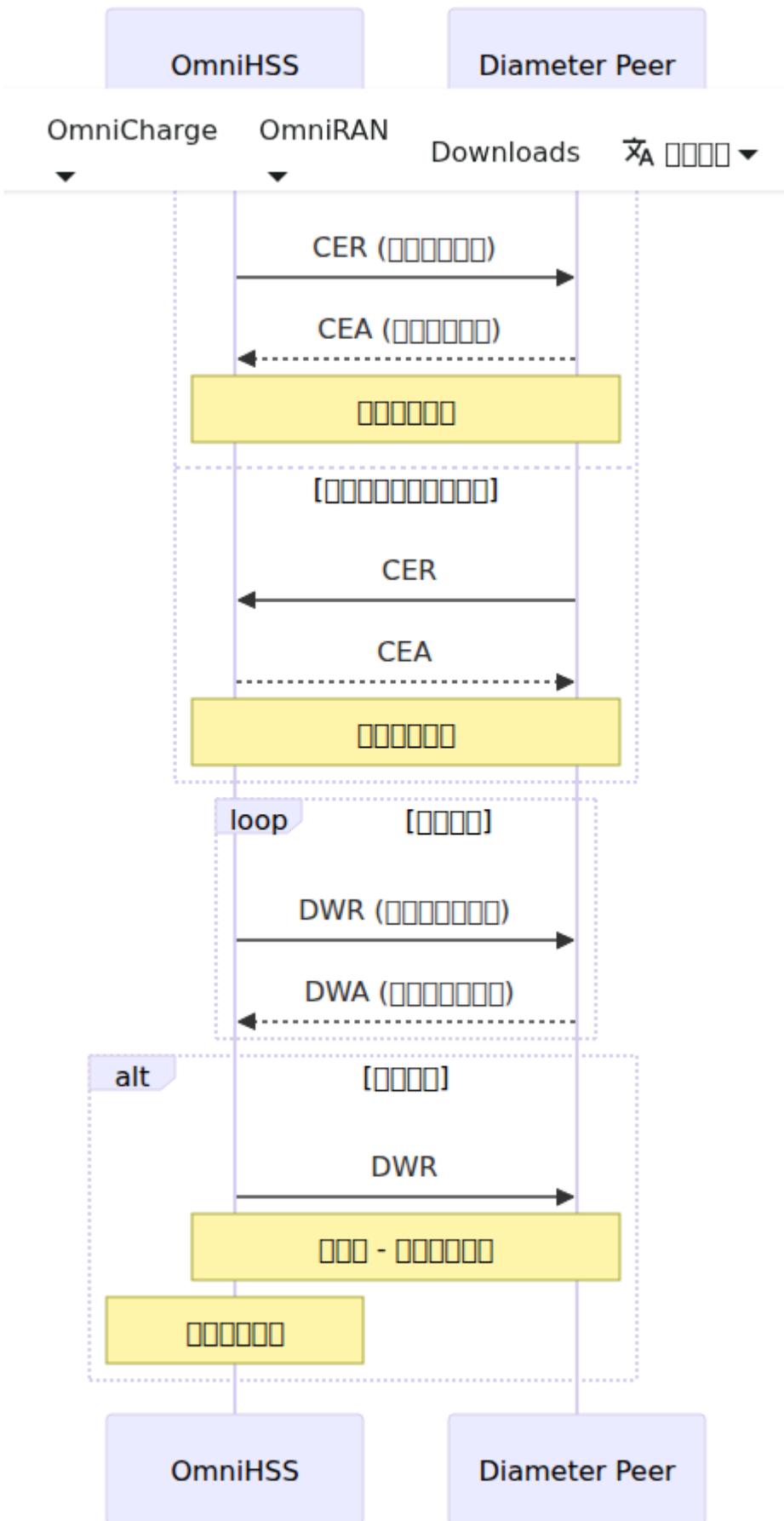
□□	□□	□□
□□□□ Diameter □□	1 □□	2+ □□□□□□□
□□□□	> 80%	> 90%
□□□□□□	> 5%	> 10%
□□□□	> 80% □□□	> 95% □□□

EIR□□□□□□□□

□□

HSS □□□□□□□ EIR□□□□□□□□□□□□□□□□□□□□□□□□□□EIR □□ IMEI□□□

□□□□

- **S13 □□**：□□ Diameter □□□□□□□□□□□
- **IMEI □□**：□□ IMEI/IMEISV □□□□□□□
- □□□□□□□□□□□□□ IMEI、IMEISV □ IMSI □□□□
- □□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□
- **REST API**：□□□ CRUD □□□□ EIR □□□□

□□

Diameter □□

□□	□□ ID	□□□	□□
S13	16,777,252	MME/SGSN	□□□□□□

□□□□□□□

EIR □□□□□□□□□□□□□□□□□

EIR_RULE		
int	id	PK
string	action	
string	regex	
timestamp	inserted_at	
timestamp	updated_at	

支持的动作：

- `whitelist` - 允许接入
- `blacklist` - 禁止接入
- `greylist` - 监控接入

正则表达式可以匹配 IMEI、IMEISV 或 IMSI 前缀

黑灰名单策略

类型	编号	状态	处理方式
白名单	0	设备已授权	允许正常接入
灰名单	1	设备可疑/待观察	允许但需监控
黑名单	2	设备被盗或非法	拒绝接入

S13 接口

消息类型：

设备身份检查请求（ECR）/ 设备身份检查应答（ECA）

方向：MME/SGSN → HSS（EIR）

（例：MME 发起设备身份检查，判断终端是否合法）

请求 AVPs：

- Session-Id
- Origin-Host, Origin-Realm
- Destination-Realm
- Auth-Session-State
- Terminal-Information
  - IMEI（15 位数字）
  - Software-Version（2 位软件版本号）
- User-Name（IMSI，可选）
- Vendor-Specific-Application-Id

EIR 处理：

1. 提取 IMEI、Software-Version，以及可选的 IMSI，进行检查。
2. 可选关联 IMSI：
   - 验证设备与用户是否匹配。
   - 检查是否存在设备与用户的绑定。
3. 检查设备身份的类型：
   - **IMEISV** 类型（IMEI + Software-Version 组合）
   - **IMEI** 类型（□□□ IMEI）
   - **IMSI** 类型（仅检查用户身份）
   - 其他扩展设备身份检查类型。
4. 返回检查结果

响应 AVPs：

- Session-Id（与请求相同）
- Result-Code: 2001（成功）
- Equipment-Status: 0（设备合法）/ 1（设备被盗）/ 2（设备灰名单）

错误处理：

- Experimental-Result: 5422（设备/用户身份不匹配）
- Experimental-Result: 5012（设备不合法）

身份检查流程

检查顺序

EIR 按以下顺序检查设备身份信息：

```
1. IMEISV（IMEI + Software-Version）
   ↓ 若没有，则检查
2. 纯 IMEI
   ↓ 若没有，则检查
3. IMSI（订阅者身份标识）
   ↓ 若没有，则检查
4. 无身份信息
```

检查流程

情况 1：IMEISV 存在

- 组合 IMEI + Software-Version："35979139461611" + "08" = "3597913946161108"
- 系统将在 EIR 的设备数据库中查找
- 根据查找结果返回状态："whitelist"、"blacklist"、"greylist"

情况 2：IMEI 存在但无软件版本

- 仅使用 IMEI："35979139461611"
- 系统将在 EIR 的设备数据库中查找
- 返回相应的设备状态信息

情况 3：IMSI 存在，但没有 IMEI 相关信息

- 使用订阅者 IMSI："999999876543210"
- 系统将在 EIR 数据���中进行查找
- 返回相应的订阅者状态信息
- 用于识别订阅者层面的限制信息

情况 4：既无设备信息也无订阅信息

- 类型参数：`eir_unknown_equipment_behaviour`
- 选项：
    - `:whitelist` - 允许，返回设备已知状态
    - `:blacklist` - 拒绝，返回设备已知状态
    - `:greylist` - 允许但跟踪，返回灰名单
    - `:reject_unknown_equipment` - 返回原因 5422，拒绝访问

设备标识格式及含义

示例	类型	说明
`"35979139461650"`	完整 IMEI	包含校验位的/十五位
`"3597913946165.*"`	IMEI 前缀匹配	十四位/正则匹配
`"3597913946161108"`	完整 IMEISV	包含软件版本号后缀
`"999999876543210"`	IMSI	用户标识而非设备标识
`"359791.*"`	TAC 前缀匹	型号分配码匹配

正常情况流程

场景 1：设备校验 - 识别并验证 IMEI

```
MME/SGSN              HSS                 Database
                      (EIR)
    |                   |                    |
    |    S13 ECR        |                    |
    |  (IMEI: 35979139461650,               |
    |   IMSI: 999999876543210)              |
    |------------------>|                    |
    |                   |                    |
    |           提取 IMSI 作为订阅者           |
    |                   |                    |
    |               设置上下文                 |
    |                   |                    |
    |             更新订阅者信息               |
    |            (last_seen_realm,           |
    |          last_received_message_type)   |
    |                   |                    |
    |                确认后                   |
    |                   |                    |
    |              查找 EIR 配置              |
    |                   |                    |
    |                返回配置                  |
    |                   |                    |
    |  提取 IMEISV "35979139461650"          |
    |            → 标准化                     |
    |                   |                    |
    |                   |                    |
    |  匹配 IMEI "35979139461650"            |
    |   → 配置条目 "3597913946165.*"         |
    |        → 状态: "whitelist"             |
    |                   |                    |
    |                   |                    |
    |    S13 ECA        |                    |
    |  (Result-Code: 2001,                  |
    |   Equipment-Status: 0)                 |
    |<------------------|                    |
  ┌─────────────┐      |                    |
  │ 设备已授权，  │      |                    |
  │ 会话继续进行  │      |                    |
  └─────────────┘      |                    |
MME/SGSN              HSS                 Database
                      (EIR)
```

示例 2：白名单 - 具体的 IMEI、通配符匹配

场景 3：设备检查 - 白名单模式下的未知设备

MME/SGSN	HSS (EIR)	Database	Configuration

S13 ECR
(IMEI: 9999999999999,
IMSI: 999999876543210)

检查 IMSI 是否在允许

列表中（空）

允许，因为列表为

空检查

查询 EIR 数据

库记录

查询 IMEISV "99999999999999"
→ 未找到

查询 IMEI "9999999999999"
→ 未找到

查询 IMSI "999999876543210"
→ 未找到

读取 unknown_equipment_behaviour

:whitelist

配置返回: "whitelist"

S13 ECA
(Result-Code: 2001,
Equipment-Status: 0)

> 未知设备被
> 允许（白名单行为）

MME/SGSN	HSS (EIR)	Database	Configuration

场景 4：被拒绝 - 未知设备且策略为拒绝

```
MME/SGSN                          HSS                Database        Configuration
                                 (EIR)

              S13 ECR
       (IMEI: 99999999999999,
        IMSI: 999999876543210)

                            根据 IMSI 查询订户

                                无订户记录

                            查询白名单数据库

                                 无结果

                              查�� EIR 数据

                                 库数据

       查询 IMEISV "99999999999999"
                 → 无结果

        查询 IMEI "99999999999999"
                 → 无结果

         查询 IMSI "999999876543210"
                 → 无结果

                        读取 unknown_equipment_behaviour

                          :reject_unknown_equipment

              S13 ECA
       (Experimental-Result: 5422,
 DIAMETER_ERROR_EQUIPMENT_UNKNOWN)
┌─────────────┐
│   设备未知,   │
│   服务被拒绝   │
└─────────────┘
MME/SGSN                          HSS                Database        Configuration
                                 (EIR)
```

范例 5：设备检查 - 白名单响应

OmniCharge   OmniRAN   Downloads   文A 选择语言▾   Omnitouch Website ⎘

S13 ECR
(IMEI: 9999999999999,
IMSI: 999999876543210)

查询 IMSI 黑名单状态

无黑名单

查询黑白名单状态

白名单

检查 EIR 配置

允许接入

检查 IMEISV "99999999999999"
→ 白名单

检查 IMEI "9999999999999"
→ 白名单

检查 IMSI "999999876543210"
→ 白名单

检查 unknown_equipment_behaviour

:whitelist

返回状态: "whitelist"

S13 ECA
(Result-Code: 2001,
Equipment-Status: 0)

设备已授权
允许继续接入网络

MME/SGSN   HSS
(EIR)   Database   Configuration

场景 6：灰名单 - IMEISV 匹配

```
   MME/SGSN              HSS              Database
                        (EIR)

      S13 ECR
(IMEI: 35979139461630,
 Software-Version: 08,
 IMSI: 999999876543210)

                    检查 IMSI 是否被排除

                         未被排除

                       根据配置决定

                        检查顺序

                       先检 EIR 列表

                        构造查询

                      构造 IMEISV:
                 "35979139461630" + "08"
                  = "3597913946163008"

                 查找 IMEISV "3597913946163008"
                  → 找到条目 "3597913946163008"
                       → 状态: "greylist"

      S13 ECA
(Result-Code: 2001,
 Equipment-Status: 2)
```

设备在灰名单中，
允许但监控

```
   MME/SGSN              HSS              Database
                        (EIR)
```

场景 7：设备未知 - IMSI 已知

Sequence diagram content:

MME/SGSN → **HSS (EIR)**: S13 ECR (IMEI: 99999999999999, IMSI: 999999876543210)

HSS (EIR) → Database: 查询 IMSI 对应的设备

Database --> HSS (EIR): 返回设备信息

HSS (EIR) → Database: 查询设备状态信息

Database --> HSS (EIR): 状态值

HSS (EIR) → Database: 查询�� EIR 规则

Database --> HSS (EIR): 匹配规则

HSS (EIR): 检查 IMEISV "99999999999999" → 不匹配

HSS (EIR): 检查 IMEI "99999999999999" → 不匹配

HSS (EIR): 检查 IMSI "999999876543210" → 不匹配

HSS (EIR) → Configuration: 读取 unknown_equipment_behaviour

Configuration --> HSS (EIR): :reject_unknown_equipment

HSS (EIR) --> MME/SGSN: S13 ECA (Experimental-Result: 5422, DIAMETER_ERROR_EQUIPMENT_UNKNOWN)

Note over MME/SGSN: 设备未知，拒绝该设备接入

REST API

EIR 规则管理

查询所有：`/api/eir/rule`

返回所有 EIR 规则

□□□

```
GET /api/eir/rule
```

□□□HTTP 200□□

```json
{
  "data": [
    {
      "id": 1,
      "action": "whitelist",
      "regex": "3597913946165.*",
      "inserted_at": "2025-01-15T10:30:00Z",
      "updated_at": "2025-01-15T10:30:00Z"
    },
    {
      "id": 2,
      "action": "blacklist",
      "regex": "35979139461640",
      "inserted_at": "2025-01-16T14:20:00Z",
      "updated_at": "2025-01-16T14:20:00Z"
    }
  ]
}
```

□□□□ **EIR** □□

□□□

```
GET /api/eir/rule/{id}
```

□□□HTTP 200□□

```
{
  "data": {
    "id": 1,
    "action": "whitelist",
    "regex": "3597913946165.*"
  }
}
```

创建 EIR 规则

请求：

```
POST /api/eir/rule
Content-Type: application/json

{
  "action": "blacklist",
  "regex": "35979139461640"
}
```

参数：

- `action`：规则类型，可为 "whitelist"、"blacklist" 或 "greylist"
- `regex`：匹配规则的正则表达式，用于匹配设备标识信息

成功（HTTP 201）：

```
{
  "data": {
    "id": 3,
    "action": "blacklist",
    "regex": "35979139461640"
  }
}
```

参数错误（HTTP 400）：

```json
{
  "errors": {
    "regex": ["□□□□"]
  }
}
```

修改 EIR 规则（部分）

请求：

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

{
  "action": "greylist"
}
```

响应（HTTP 200）：

```json
{
  "data": {
    "id": 3,
    "action": "greylist",
    "regex": "35979139461640"
  }
}
```

替换 EIR 规则

请求：

```
PUT /api/eir/rule/{id}
Content-Type: application/json

{
  "action": "whitelist",
  "regex": "359791394616.*"
}
```

响应（HTTP 200）：

```json
{
  "data": {
    "id": 3,
    "action": "whitelist",
    "regex": "359791394616.*"
  }
}
```

删除 EIR 规则

请求：

```
DELETE /api/eir/rule/{id}
```

响应：HTTP 204 无内容。

配置

Diameter 应用配置

S13 应用（`config/runtime.exs`）：

```elixir
%{
  application_name: :s13,
  application_dictionary: :diameter_gen_3gpp_s13,
  vendor_specific_application_ids: [
    %{vendor_id: 10415, auth_application_id: 16_777_252}
  ]
}
```

对端连接配置

在 `config/runtime.exs` 中配置对端连接的示例如下：

示例：

```
config :hss, :eir,
  unknown_equipment_behaviour: :whitelist
```

可选值：

- `:whitelist` - □□□□□□□□□□□□□□□
- `:blacklist` - □□□□□□□□□□□□
- `:greylist` - □□□□□□□□□□□□
- `:reject_unknown_equipment` - □□ Diameter □□ 5422□□□□

默认值：

- □□/□□□`:whitelist` - □□□□□□
- □□□□□□□`:whitelist` - □□□□□□□□□
- □□□□□□□`:greylist` - □□□□□□□□□□
- □□□□□□□`:reject_unknown_equipment` - □□□□□□□

□□□□

□□□□	□□	□□	□□
2001	□□	DIAMETER_SUCCESS	□□□□□□□
5422	□□	DIAMETER_ERROR_EQUIPMENT_UNKNOWN	□□□□□□□□□□□□□□
5012	□□	DIAMETER_ERROR_UNKNOWN	□□□□

□□

1. □□□□□□□

□□□□□□□□□□□□

示例：

```
POST /api/eir/rule
{
  "action": "blacklist",
  "regex": "35979139461640"  # 具体 IMEI
}
```

该设备将被禁止接入网络，无论使用哪张卡。

2. 按型号黑名单

阻止某一型号的所有设备接入：

示例：

```
POST /api/eir/rule
{
  "action": "whitelist",
  "regex": "359791394.*"  # 某品牌/型号的 TAC
}
```

仅允许该 TAC 前缀的设备接入网络。

3. 按用户卡黑名单

阻止某个特定用户（按其 SIM 卡的SIM 卡）：

示例：

```
POST /api/eir/rule
{
  "action": "blacklist",
  "regex": "999999876543210"  # IMSI
}
```

拦截特定用户使用的 SIM 卡接入核心网。

4. 灰名单管理场景

场景：将可疑设备加入观察名单

请求：

```
POST /api/eir/rule
{
  "action": "greylist",
  "regex": "35979139.*"  # 匹配特定 TAC 段的
}
```

系统将监控匹配这些设备的活动。

5. 精确封锁场景

场景：封锁特定设备号段的某个型号

请求：

```
POST /api/eir/rule
{
  "action": "blacklist",
  "regex": "359791394616.*05"  # IMEI 号段 + 软件版本 05
}
```

这允许针对特定 IMEI 号段中运行软件版本 "05" 的设备。

参考资料

协议栈

EIR 网元支持以下协议和接口：

- **S13 接口（核心）** - ECR/ECA 消息处理
- 鉴权接口支持 - 支持设备标识检查 IMEI/IMEISV/IMSI 组合

- **EIR □□□□□ - □□□□□□□□□**
- **REST API □□□ - □□□□□□**

□□□□□□□□□

□□□□□□□□□□□□□□□□□□□

1. **IMEISV □□□□□□□□□** IMEI + □□□□
2. **IMEI □□□□□□□** IMEI
3. **IMSI □□□□□** IMSI□□□□□□□
4. □□□□□□□□□□□□□□□

□□□□□□□□

- `whitelist` - □□□□□
- `blacklist` - □□□□□
- `greylist` - □□□□□□
- `reject_unknown_equipment` - □□□□

□□□□

IMEI □□

IMEI □□□□□□□□□□□EIR□

- □□□□□□□□□□□ IMEI □
- □□□□□□□□□□□□□□□
- □□ API □□□□□□□□□□□□□□

□□□□

EIR □□□□□□□□□□ ID□□□□□□□□□□□□□□

```
□□ 1: □□ "359791.*" □□ "whitelist"  (□□)
□□ 2: □□ "35979139461640" □□ "blacklist"  (□□)
```

本节介绍国际移动设备标识的结构，以及设备状态管理。

核心功能

设备标识寄存器的主要功能：

- 设备状态管理：支持 S13 ECR 流程
- REST API EIR 状态查询
- IMEI 生命周期与黑白名单管理

技术规范

- Diameter 接口 - S13 接口实现
- API 接口 - 标准化 API 接口
- 集成 - 对接 HSS 网元
- 数据模型 - 状态定义

一、IMEI 结构

IMEI 标准（15 位数字）

```
35 9791 394616 1
|  |    |         └ 校验位（Luhn 算法）
|  |    └ 序列号（6 位数字）
|  └ FAC（最终装配码，4 位数字）
└ TAC（型号分配码，共 8 位，前身为 RBI）
    |  └ RBI（报告机构标识，2 位数字）
    └ 制造商/型号（6 位数字）
```

IMEISV 结构（16 位示例）

```
35 9791 394616 1 08
|  |    |        |  └ 软件版本（2 位数字）
└ IMEI（15 位数字）
```

通配符

IMEI/IMEISV	代码	描述
359791394616108	3597913946161.*	匹配 TAC+FAC+序列号 359791394616* 的设备
359791394616140	35979139461614.	匹配 359791394616141-9 的任意设备
35979139461640	35979139461640	完整 IMEI 匹配
3597913946163008	3597913946163008	完整 IMEISV（IMEI + SV）匹配

OmniHSS 用户指南

目录

产品概述

OmniHSS 是一款面向电信运营商的归属用户服务器解决方案，提供统一的用户签约数据管理、认证授权和移动性管理功能。

□□□□

□□□□

□□□□
- APN □□□
- APN QoS □□□□
- □□□□
- EIR □□

□□ SIM □□□ MSISDN

□□□□
- □□□□
- PDN □□
- LTE □□
- □□□□

□□□□□□
- EPC □□□□
- IMS □□□□
- □□□□□□
- APN □□□□

□□□□

□□

□□□□□□□□□□□□□□□□□

SUBSCRIBER			
bigint	id	PK	
boolean	enabled		
boolean	ims_enabled		
string	imsi	UK	14-15 □□□
map	custom_attributes		

□□ SIM
□□ KEY_SET
□□ EPC_PROFILE
□□ IMS_PROFILE
□□ RDAMING_PROFILE
□□ SUBSCRIBER_STATE
□□ JOIN_SUB_MSISDN

□□:
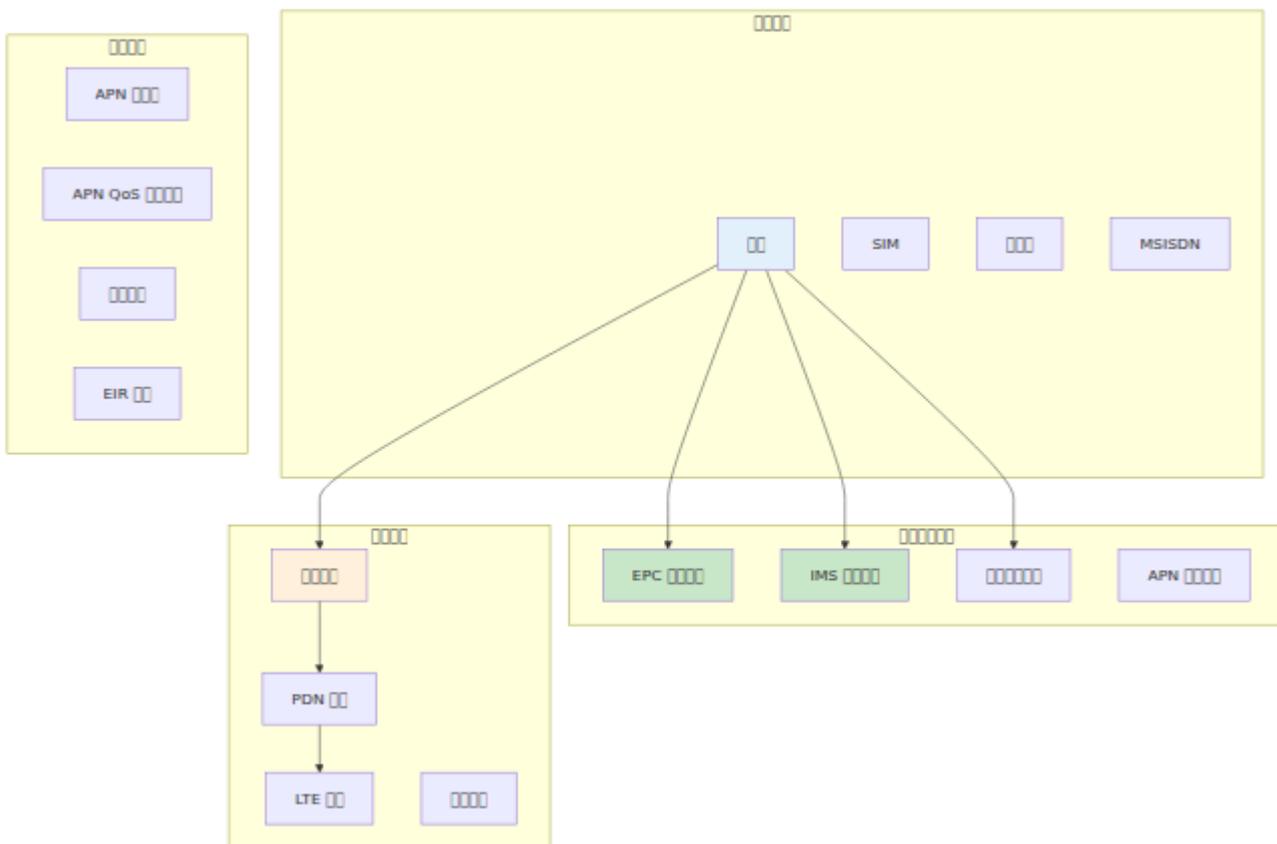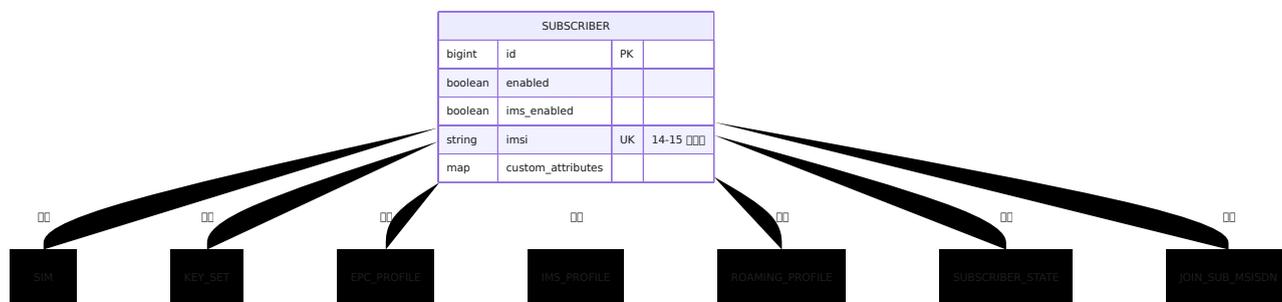
字段	类型	描述	备注
`id`	bigint	标识	自动生成
`enabled`	boolean	是否启用订户	默认: true
`ims_enabled`	boolean	IMS 是否启用	默认: true
`imsi`	string	国际移动用户标识	14-15 位数字字符串
`custom_attributes`	map	自定义属��属性	可选
`sim_id`	bigint	关联的 SIM	可选
`key_set_id`	bigint	关联的密钥集	可选
`epc_profile_id`	bigint	关联的 EPC 配置文件	可选
`ims_profile_id`	bigint	关联的 IMS 配置文件	可选
`roaming_profile_id`	bigint	关联的漫游配置文件	可选
`subscriber_state_id`	bigint	关联的订户状态	自动管理

注意事项:

- 每个订户都有一个唯一的 IMSI
- IMSI 是一个 14-15 位的数字字符串，用于识别订户
- 订户可以关联多个 MSISDN（电话号码）
- 订户状态自动管理，无需手动设置
- `enabled` 字段控制订户是否可以访问 IMS。
- `ims_enabled` 字段控制 IMS 功能

SIM

SIM 实体代表物理或虚拟的 SIM 卡。

字段:
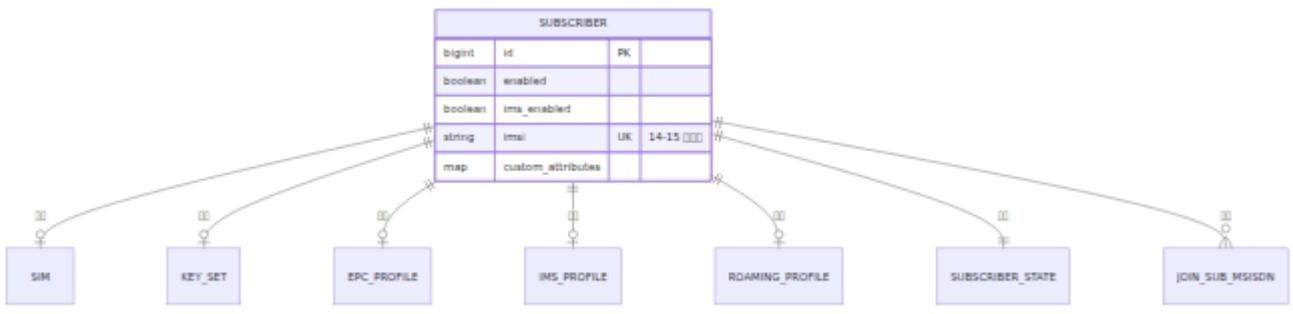
字段	类型	说明	是否加密
iccid	string	集成电路卡 ID	明文
sim_vendor	string	SIM 供应商	明文
batch_name	string	批次名称	明文
is_esim	boolean	是否为 SIM 卡片	明文
pin1, pin2	string	PIN 码	加密
puk1, puk2	string	PUK 码	加密
adm1 - adm10	string	管理密钥	高度加密
kic, kid	binary	OTA 传输密钥	高度加密

关键点:

- ICCID 唯一标识 SIM 卡
- 每个 SIM 卡对应一个订户和一组密钥
- PIN/PUK 码用于用户解锁 SIM 卡片
- ADM 密钥用于管理 SIM 卡片
- KIC/KID 用于 SIM OTA空中传输的安全保护

□□□

□□□□□□□□□□□□□□□□□

KEY_SET			
bigint	id	PK	
binary	ki		128 □
binary	opc		128 □
binary	op		128 □
binary	amf		16 □
bigint	sqn		48 □□□
string	authentication_algorithm		

□□□

□□:

□□	□□	□□	□□
`ki`	binary	□□□□	128 □□16 □□□
`opc`	binary	□□□□□□□□□□	128 □
`op`	binary	□□□□□□□□□□ OPC□	128 □
`amf`	binary	□□□□□□□	16 □□2 □□□
`sqn`	bigint	□□□□□□□	48 □
`authentication_algorithm`	string	□□□□	□□□ "milenage"
`ota_counter`	bigint	OTA □□□□□	□□

□□□:

- □□□□□□□□□□□□□
- Ki □□ SIM □□□□□□
- □□□□ OPC □ OP □□□□OPC □□□ OP □□□
- SQN □□□□□□□□□□□
- Milenage □□□□□□□□□□□

□□□□□□:
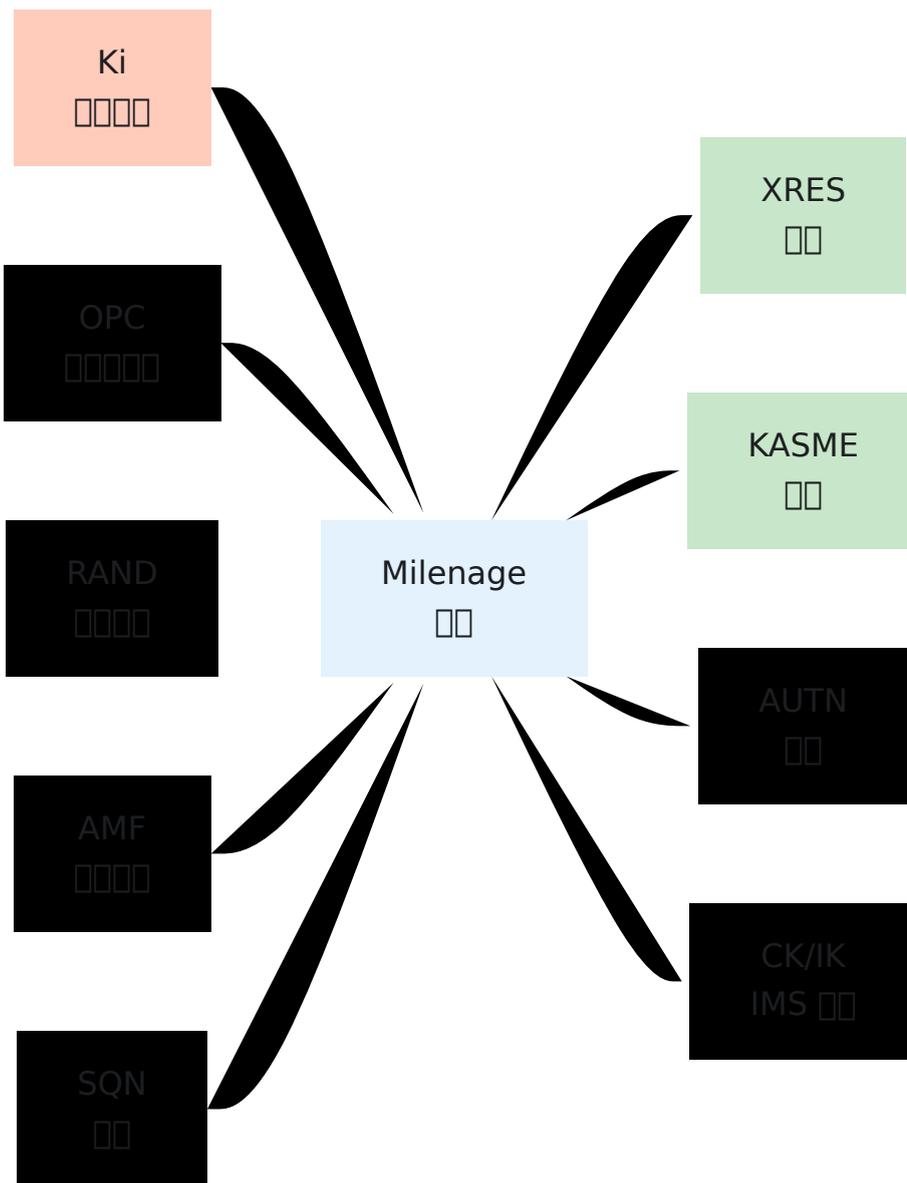
MSISDN

MSISDN □□□□□□□□□□□□

KEY_SET

bigint	id	PK	
binary	ki		128 □
binary	opc		128 □
binary	op		128 □
binary	amf		16 □
bigint	sqn		48 □□□
string	authentication_algorithm		

□□□

SUBSCRIBER

字段:

字段	类型	描述	约束
`msisdn`	string	国际 ISDN 号码	1-15 位数字（E.164 格式）

约束:
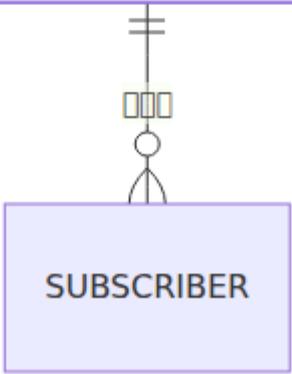
- MSISDN 是订阅者的电话号码
- 一个 MSISDN 对应一个订阅者
- 一个 MSISDN 对应一个订阅者记录
- 格式: 国家代码 + 国内号码，例如"+1 415-555-1234" 将存储为 "14155551234"。

□ MSISDN □□:

用户
IMSI: 001001123456789

MSISDN: 14155551001
□□□

MSISDN: 14155551002
□□□□

MSISDN: 14155551003
□□□□

□□□□□□

EPC □□□□

EPC □□□□□□ LTE □□□□□□□□□

说明:
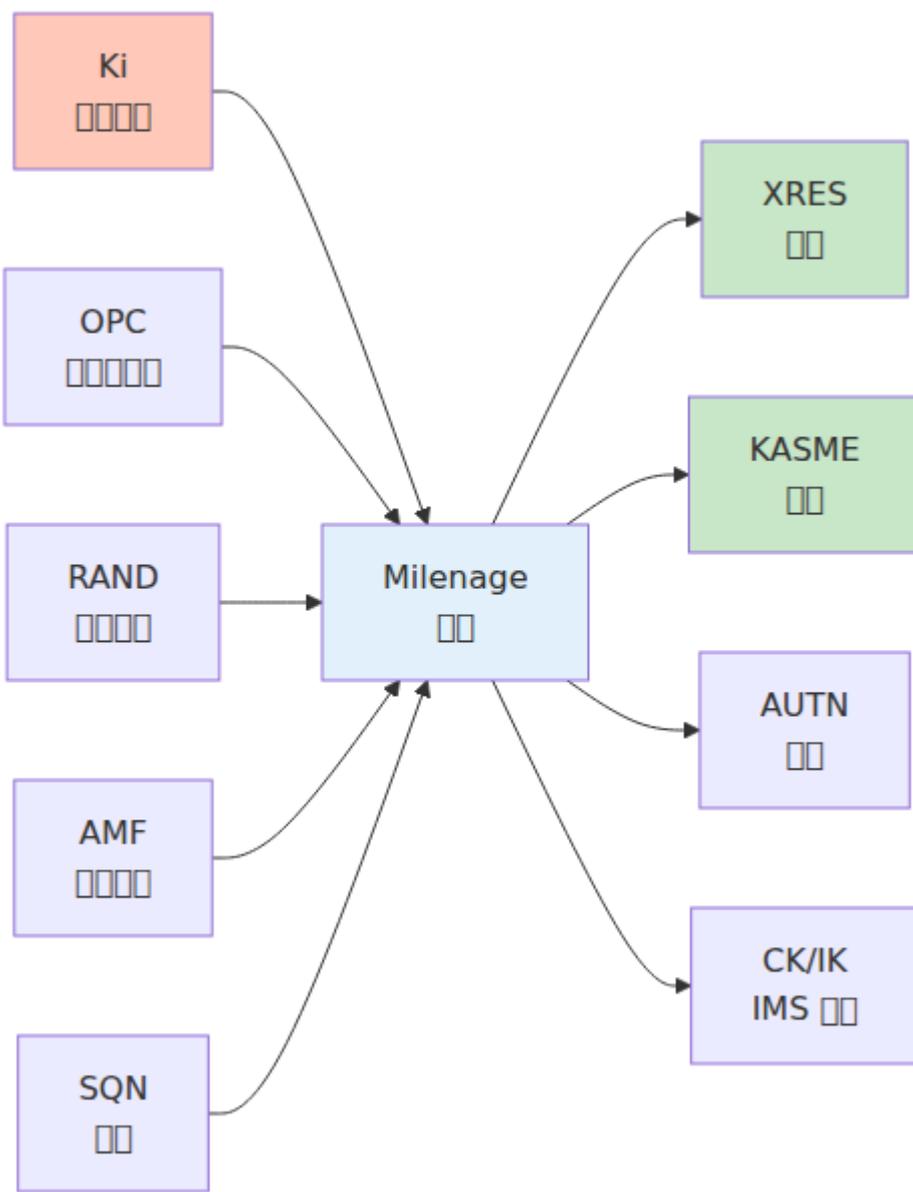
字段	类型	描述	单位
name	string	签约记录的名称	无
ue_ambr_dl_kbps	integer	下行聚合最大速率	Kbps
ue_ambr_ul_kbps	integer	上行聚合最大速率	Kbps
network_access_mode	string	接入模式	"仅分组" 或 "分组和电路域"
tracking_area_update_interval_seconds	integer	TAU 间隔	秒

接入模式说明:

AMBR 聚合最大比特率说明:
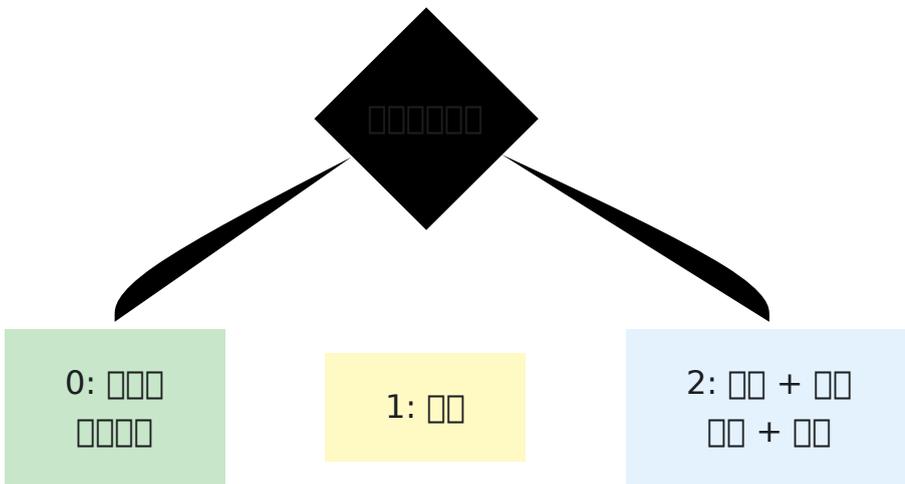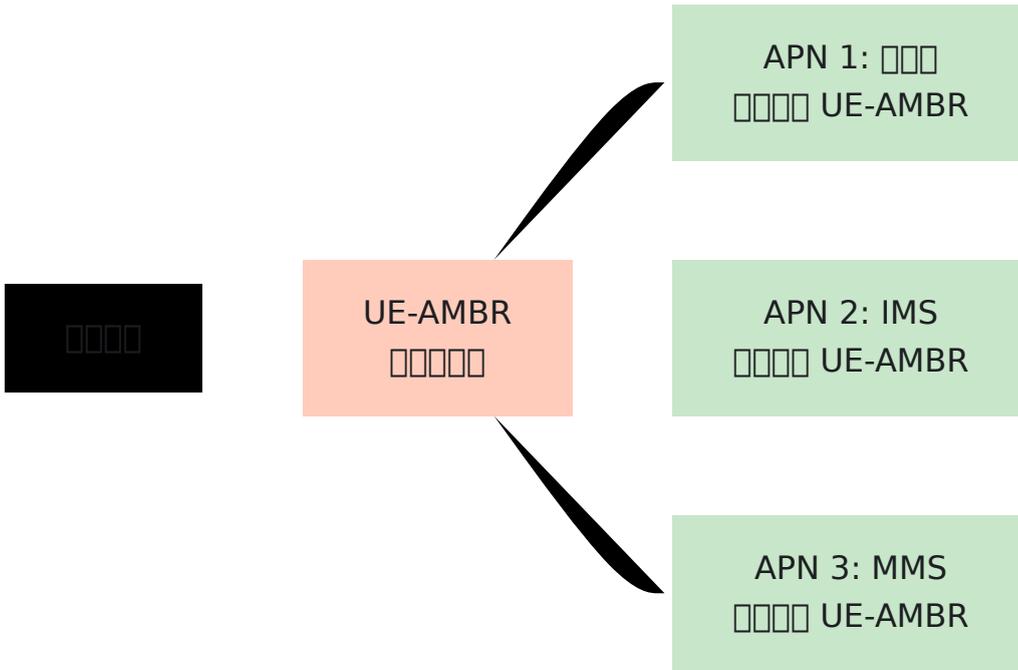
IMS □□□□

IMS □□□□□□□□/□□□□□□□□

□□:
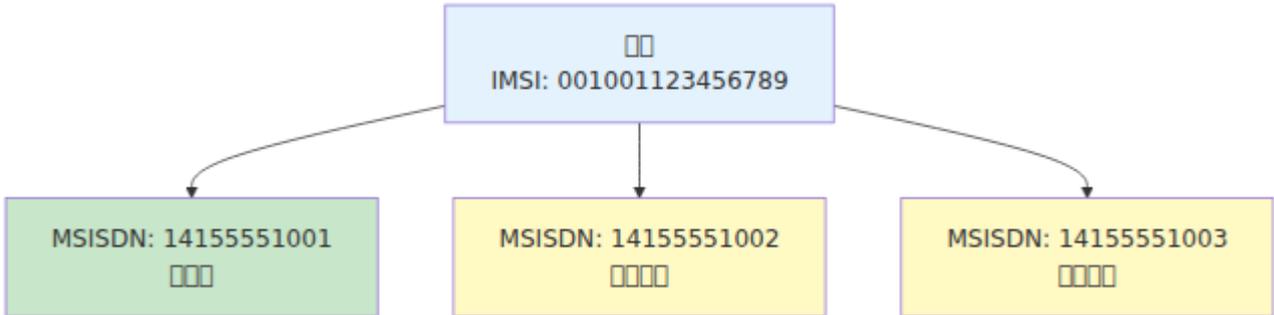
□□	□□	□□	□□
`name`	string	□□□□□□□	□□
`ifc_template`	text	□□□□□□ XML □□	□□□□ XML

IFC □□□□:
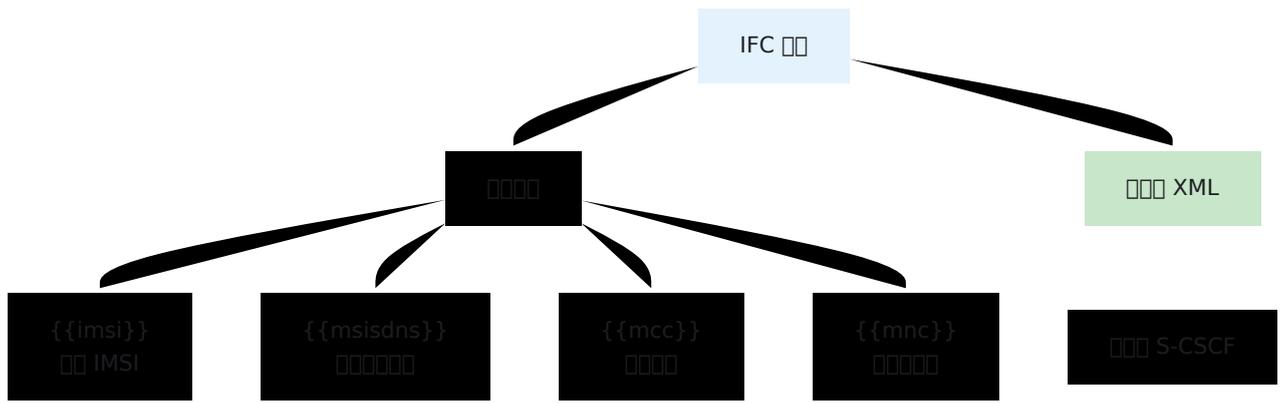
说明:

- IFC（初始过滤条件）决定哪些 IMS 请求需要处理
- 替换变量会动态生成配置
- 条件通配符支持灵活匹配
- 将 IMS 请求路由到正确的 S-CSCF

APN 配置管理

APN 配置表及其关联关系，构成数据会话的基础：

关系说明:

APN 标识符
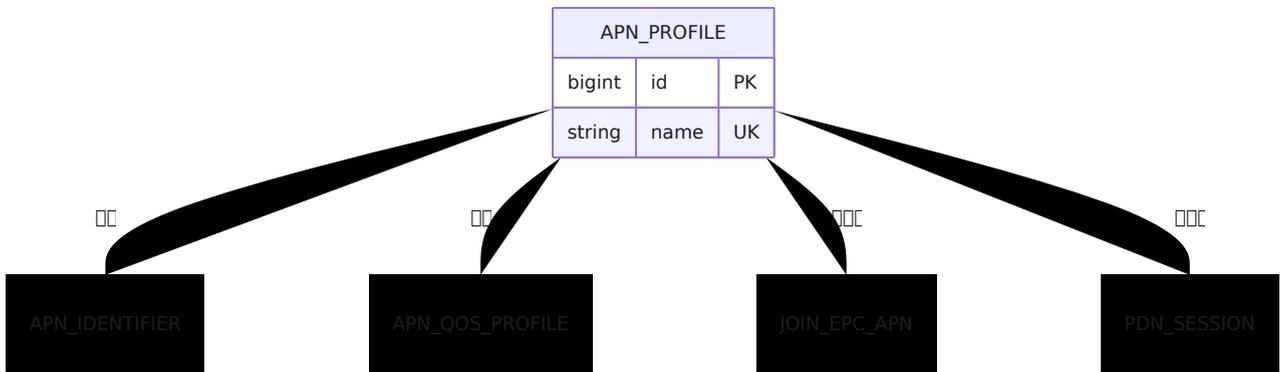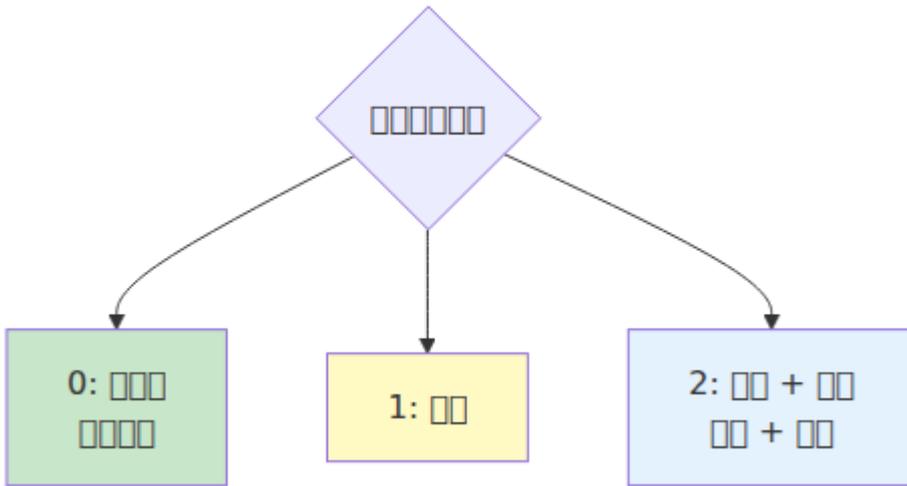
字段	类型	含义	示例
`apn`	string	APN 名称	"internet", "ims", "mms"
`ip_version`	string	IP 协议版本	见下表
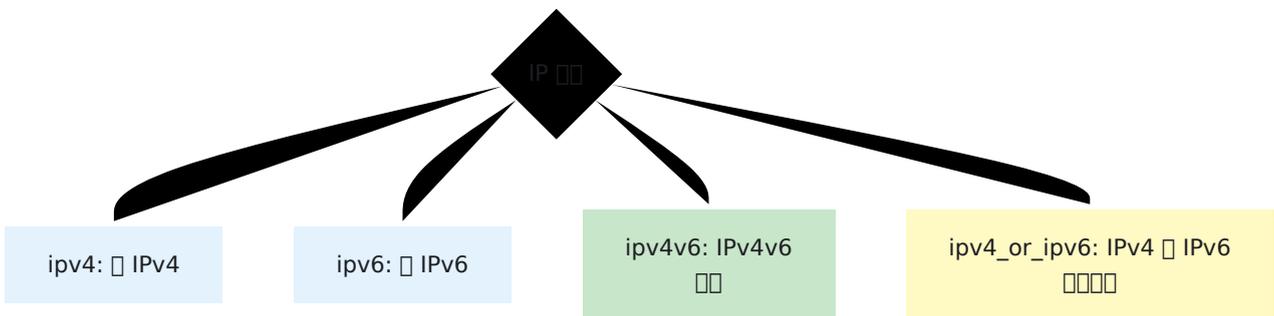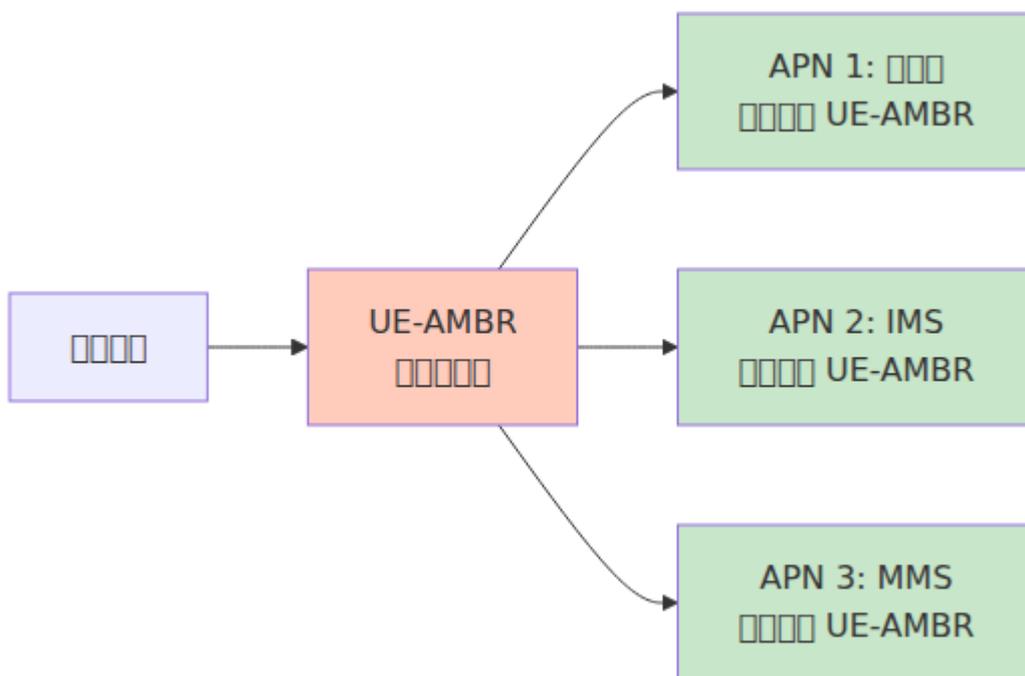
IP 协议版本:

APN QoS 配置说明

QoS 参数:

参数	含义	范围	默认配置
`qci`	QoS 类别标识符	1-9	QCI 9（尽力而为）
`allocation_retention_priority`	ARP 优先级	1-15	8（中等优先级）
`apn_ambr_dl_kbps`	APN 下行带宽	0+	无限
`apn_ambr_ul_kbps`	APN 上行带宽	0+	无限
`pre_emption_capability`	抢占其他承载	true/false	false
`pre_emption_vulnerability`	可被抢占	true/false	true

QCI 表:

QCI

QCI 1-4
□□□□□
□□□□□

QCI 5-9
□□□
□□□□

QCI 1: □□□□   QCI 2: □□□□   QCI 4: □□□   QCI 5: IMS □□   QCI 9: □□□□□

□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□
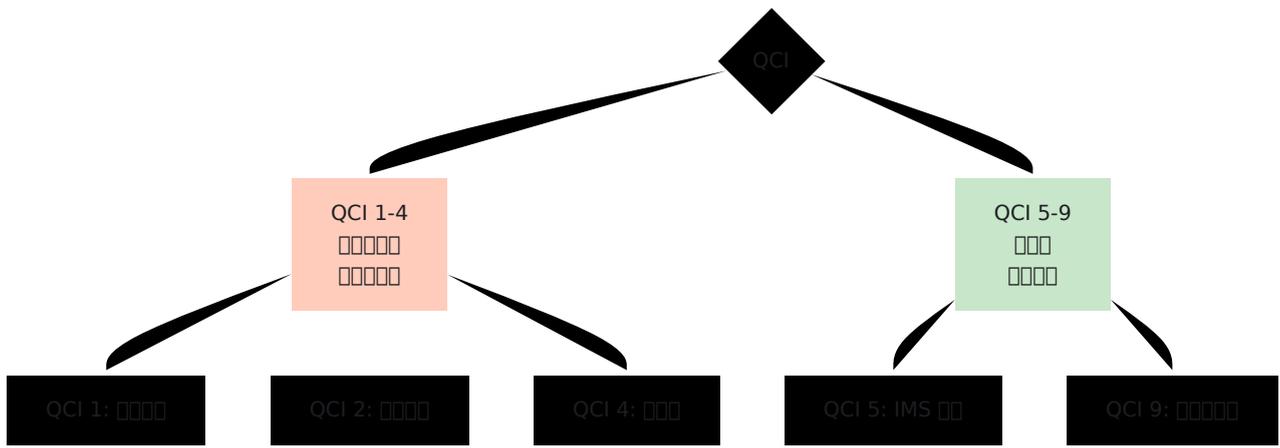
ROAMING_PROFILE			
bigint	id	PK	
string	name	UK	
string	data_action_if_no_rules_match		□□□□□
string	ims_action_if_no_rules_match		□□□□□

□□   □□□

JOIN_ROAM_RULE   SUBSCRIBER

□□

ROAMING_RULE

□□□□:

```
                        ┌──────────┐
                        │  IFC □□  │
                        └──────────┘
                       /            \
              ┌──────────┐      ┌───────────┐
              │  □□□□   │      │  □□□ XML  │
              └──────────┘      └───────────┘
            /    |    |    \           │
┌──────────┐ ┌──────────┐ ┌────────┐ ┌──────────┐ ┌────────────┐
│{{imsi}}  │ │{{msisdns}}│ │{{mcc}} │ │{{mnc}}   │ │ □□□ S-CSCF │
│□□ IMSI   │ │□□□□□□   │ │□□□□  │ │□□□□□   │ └────────────┘
└──────────┘ └──────────┘ └────────┘ └──────────┘
```

□□�□□:
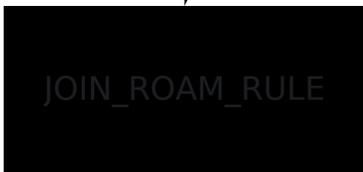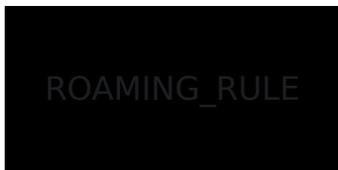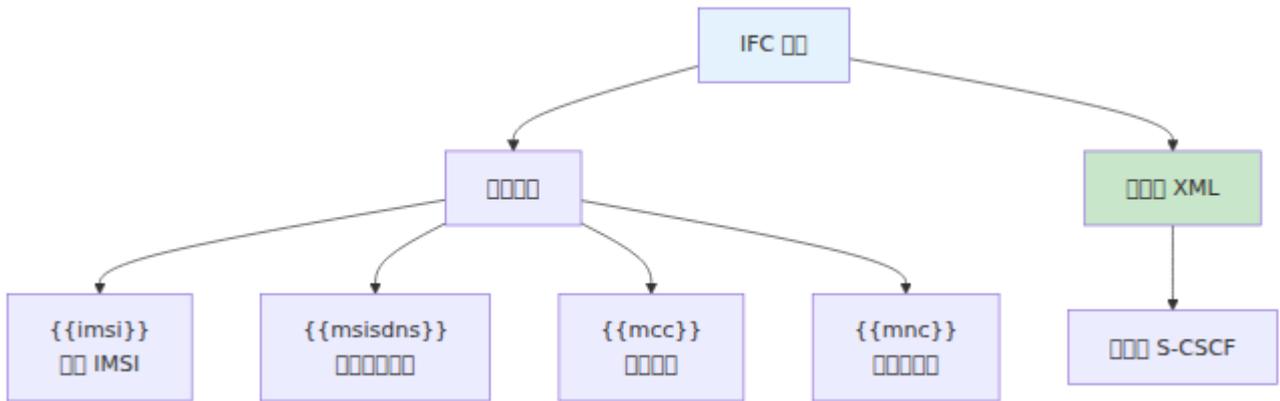
□□□□□□
MCC: 310, MNC: 410

□□□□?

□□□□   □□□□

□□□□□□   □□□□□□

□□   □□   □□□□=□□□□□   □□□□=□□□□□

🟥🟥🟥🟥

🟥🟥🟥🟥

🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥🟥

🟥🟥🟥🟥:

🟥🟥🟥🟥:
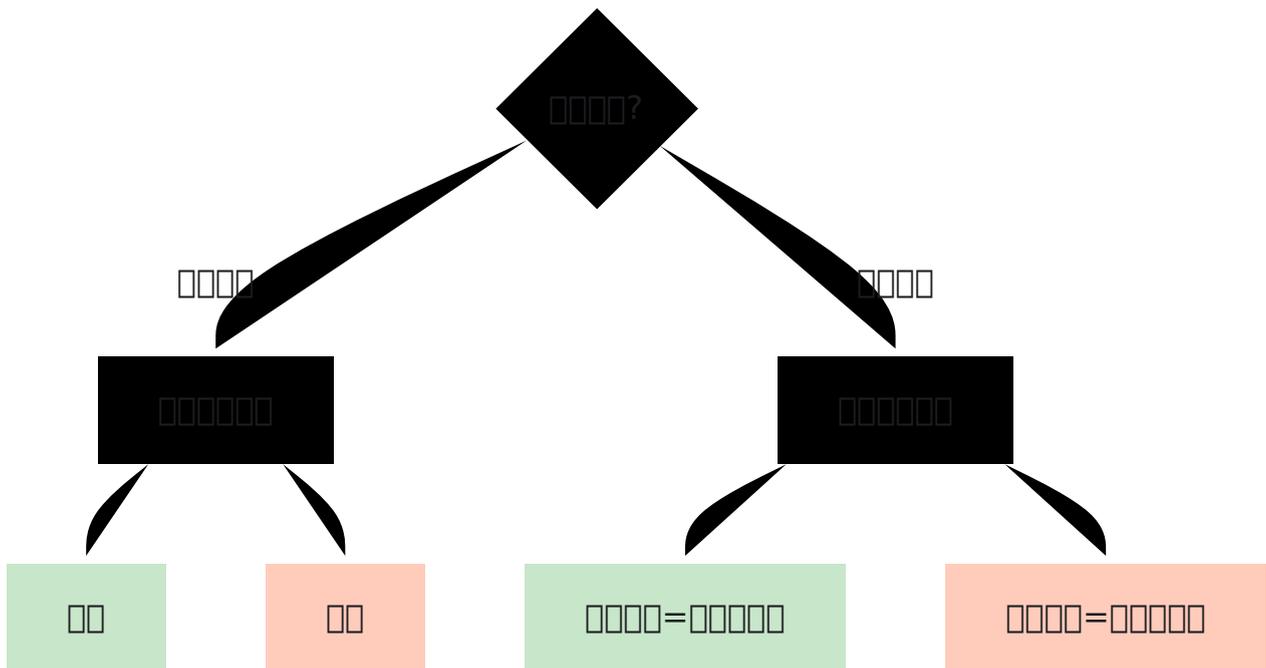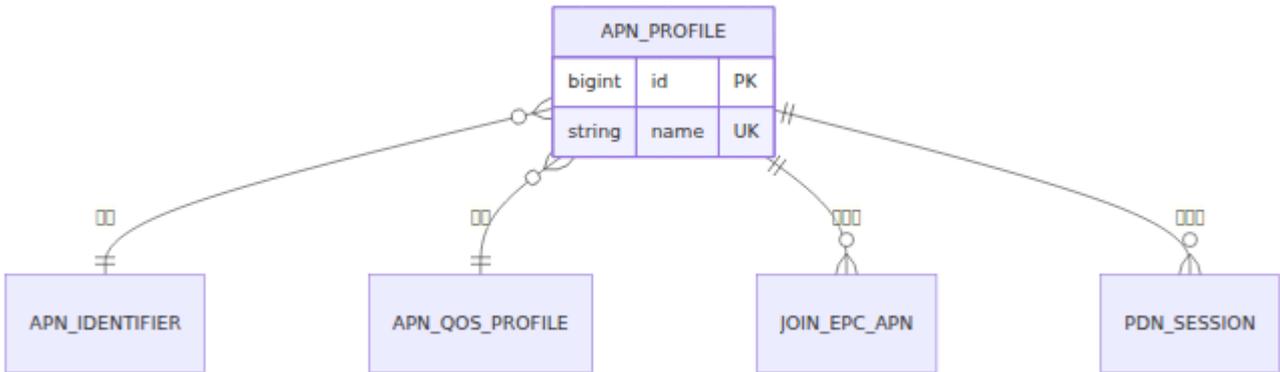
- `last_seen_mcc`, `last_seen_mnc` - 🟥🟥🟥🟥🟥
- `last_seen_tac` - 🟥🟥🟥🟥🟥🟥
- `last_seen_cell_id` - 🟥🟥 ID
- `last_seen_enodeb_id` - eNodeB ID
- `last_seen_eci` - E-UTRAN 🟥🟥🟥🟥🟥

🟥🟥🟥🟥:

- `last_seen_mme` - 🟥🟥🟥🟥🟥🟥 MME
- `last_seen_realm` - MME 🟥🟥🟥🟥
- `last_seen_rat_type` - 🟥🟥🟥🟥🟥🟥LTE🟥5G 🟥🟥

IMS 🟥🟥:

- `assigned_scscf` - 🟥🟥🟥🟥🟥🟥 S-CSCF
- `ims_public_identity` - SIP URI🟥🟥🟥🟥 sip:+14155551234@ims.example.com🟥
- `sh_repository_data` - 🟥🟥🟥 IMS 🟥🟥🟥🟥🟥🟥

🟥🟥🟥:

- `last_seen_at` - □□□□□□□□□□□
- □□ `last_*_at` □□□□□□□□□□□

PDN □□

PDN □□□□□□□□□□□□□□□

PDN_SESSION		
bigint	id	PK
string	pgw_session_id	
integer	rat_type	
string	ip_address	
string	assigned_pgw_host	
boolean	emergency	
boolean	roaming	
datetime	created_at	
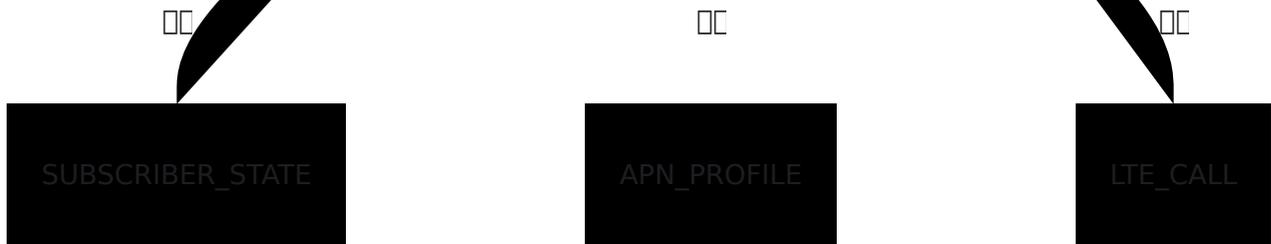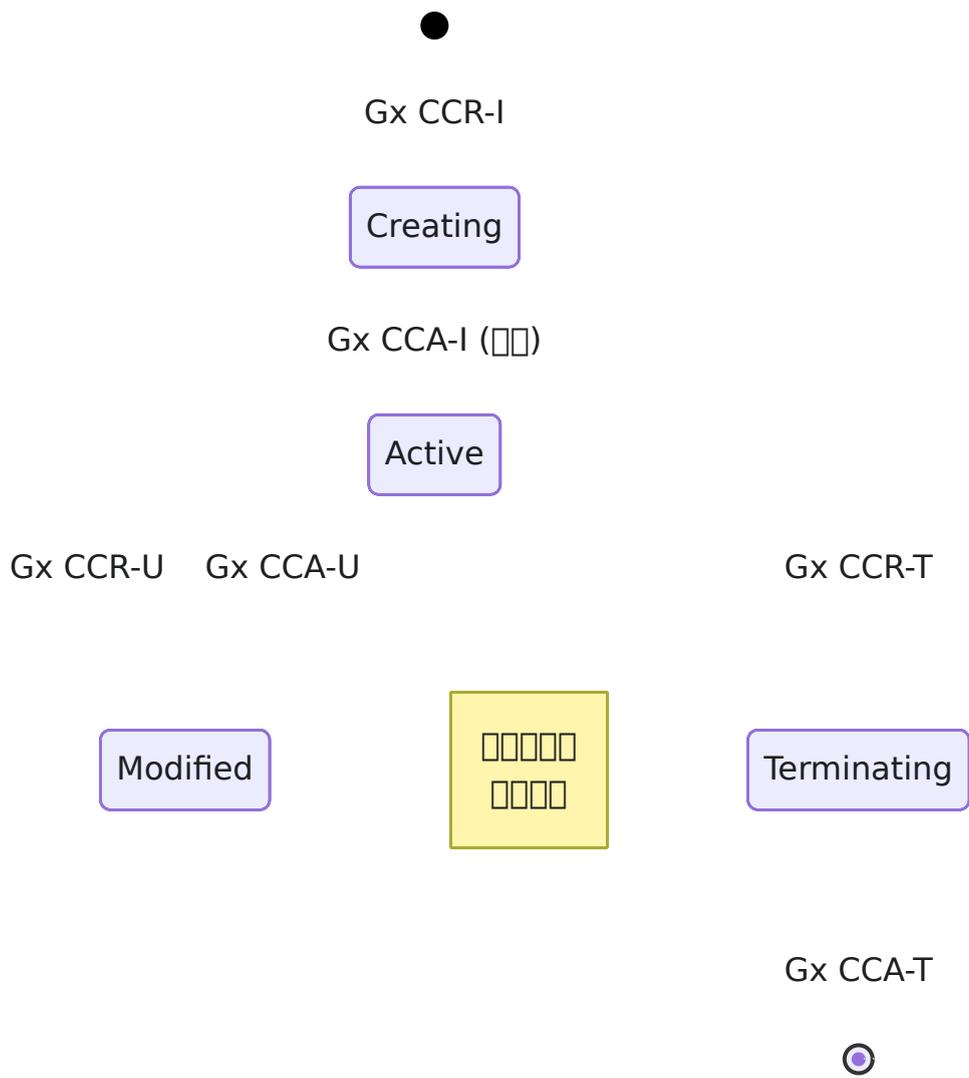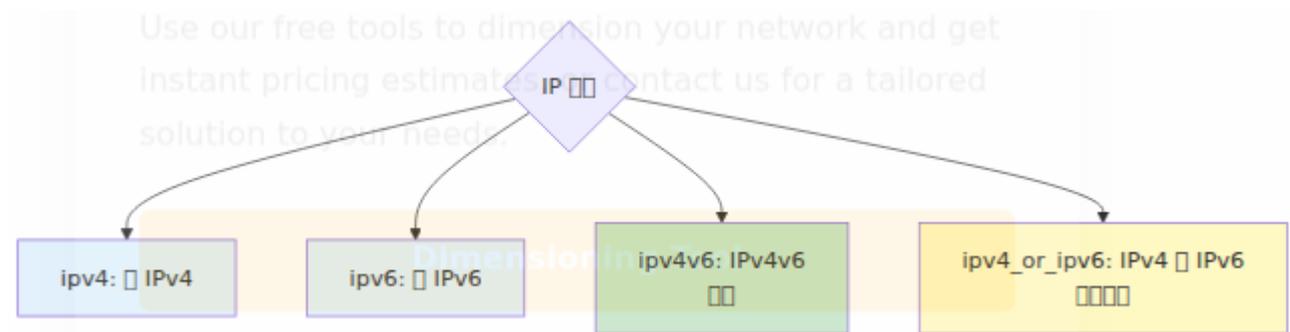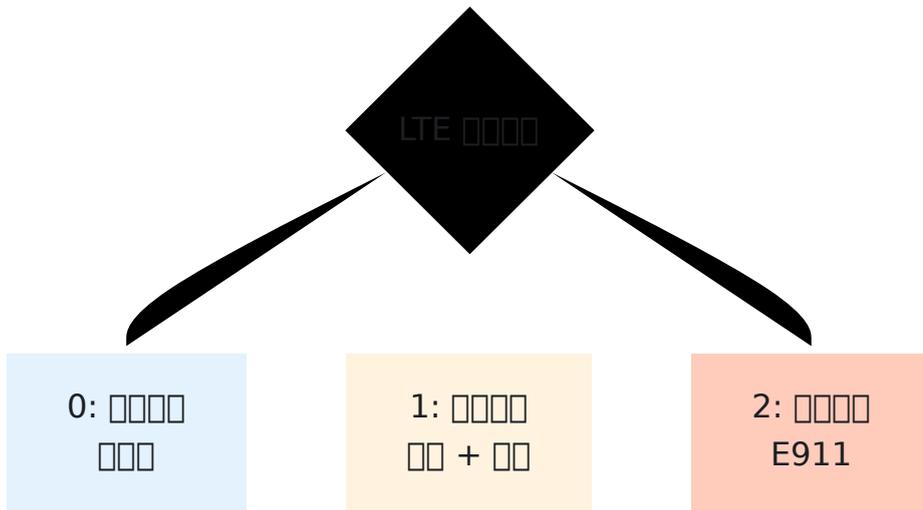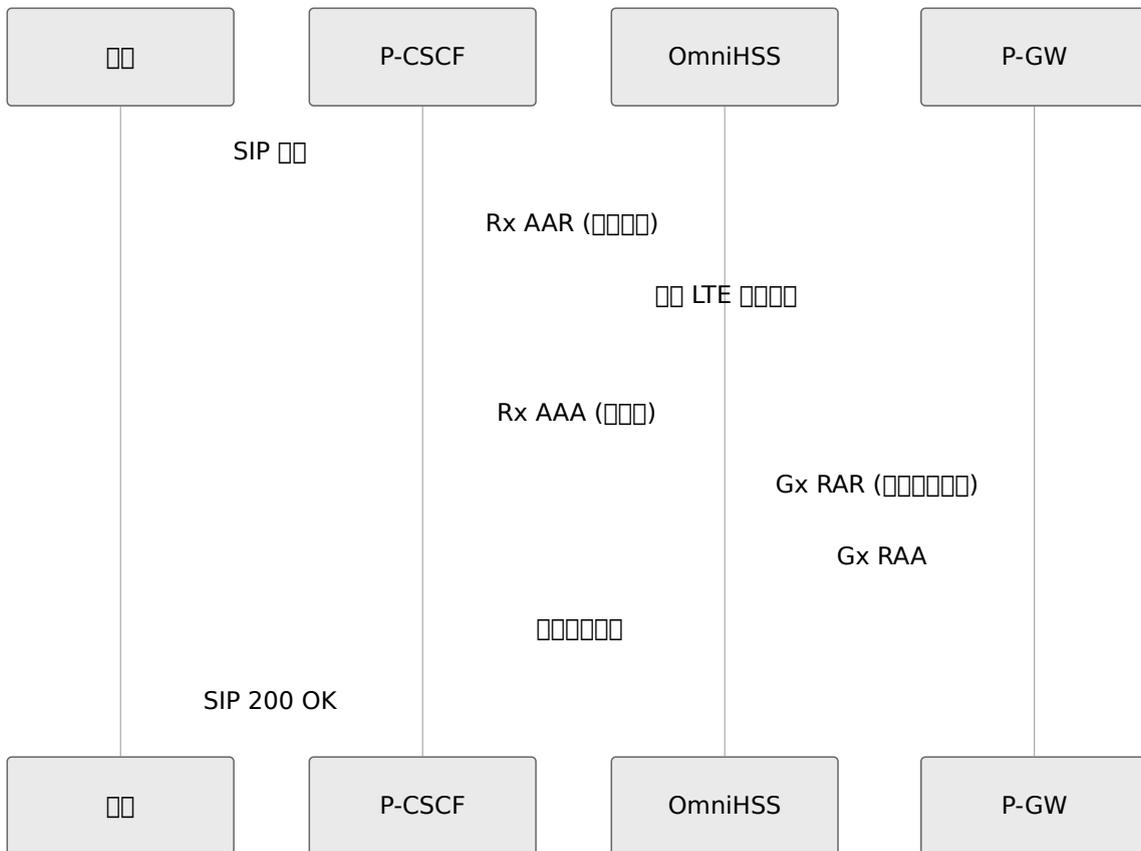
□□   □□   □□

SUBSCRIBER_STATE	APN_PROFILE	LTE_CALL

PDN □□□□□□:

●

Gx CCR-I

Creating

Gx CCA-I (□□)

Active

Gx CCR-U   Gx CCA-U   Gx CCR-T

Modified   □□□□□
                     □□□□   **Terminating**

Gx CCA-T

◉

LTE □□

LTE □□□□□□□□□□ VoLTE □□/□□□□□□

IP □□

ipv4: □ IPv4   ipv6: □ IPv6   ipv4v6: IPv4v6 □□   ipv4_or_ipv6: IPv4 □ IPv6 □□□□

□□□□:

LTE 承载类型

| 0: 默认承载 非保证 | 1: 专用承载 语音 + 视频 | 2: 紧急承载 E911 |

VoLTE 呼叫流程:

| 主叫 | P-CSCF | OmniHSS | P-GW |

SIP 邀请

Rx AAR (媒体授权)

创建 LTE 专用承载

Rx AAA (成功)

Gx RAR (承载修改请求)

Gx RAA

专用承载建立

SIP 200 OK

| 主叫 | P-CSCF | OmniHSS | P-GW |

□□□□□

□□□□□□

Use our free tools to dimension your network and get instant pricing estimates, or contact us for a tailored solution to your needs.

Dimensioning Tool

Pricing Calculator

QCI 9: □□□□□

□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□
APN □□□□ APN QoS □□□□
APN □□ QoS □□

□□□□□□□
APN □□□□

□□□
□□□□

EPC □□□□
□□□□□□

□□
SIM IMS □□□□ □□□□□□
□□□ □□□□ □□□□

□□
□□

顧客の体験

Plan Your Mobile Network

Use our free tools to dimension your network and ge
instant pricing estimates, or contact us for a tailored
solution to your needs.

Dimensioning Tool

Pricing Calculator

Get in Touch

ROAMING_RULE

●

开始创建流程

Create_Prerequisites

前提条件创建完成

前提条件：
创建 EPC 网络配置（含 APN）
创建 IMS 网络配置及相关
号码、路由、服务器配置等
创建 SIM卡配置等

Create_Subscriber

enabled=false

enabled=true

Disabled

设置 enabled=true   设置 enabled=false

Enabled   删除用户

分配设备后上线   用户主动下线   下线操作

Active   ◉

IMS 注册   IMS 注销

IMS_Registered

VoLTE 发起呼叫   呼叫结束

In_Call

会话状态管理

●

初始状态

No_Sessions

默认承载建立   承载释放完成

PDN_Active

VoLTE 呼叫发起   VoLTE 呼叫结束

PDN 会话与语音呼
叫并发激活

PDN_And_Call

多呼叫建立   多呼叫结束

Multiple_Calls

PDN 会话 + LTE 语音
并发激活

□□□□□

□□□□□□

□□□□□□

Plan Your Mobile Network

Use our free tools to dimension your network and get instant pricing estimates, or contact us for a tailored solution to your needs.

Dimensioning Tool

Pricing Calculator

Get in Touch

□□　　□□　　□□□□=□□□□□　　□□□□=□□□□□

□□□□□

S6a ULR □□

□□ IMSI □□□□

□□ EPC □□□□
+ APN □□□□

□□□□□□
□□□MME □

□□□□□□
AMBR□APN□QoS

S6a ULA □□

IMS 注册流程

Cx SAR 请求

提取 IMSI/MSISDN 用户标识

生成 IMS 用户标识
+ MSISDN

分配 S-CSCF
名称/能力

生成 IFC 初始
过滤器

更新用户数据
S-CSCF 名称

Cx SAA 响应

□□□□□□

□□□□□□

OmniHSS □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□

图5: S6a AIR - 接入请求消息中的密钥

位置更新请求序列

图6: S6a ULR - 将用户附加到 EPC 网络的信令流

处理程序：IMS 注册状态

框架: Cx SAR - 处理 IMS 注册的授权状态和回执

← 处理程序概述 | 下一个: API 请求 →

Diameter □□□□□□

□□□□□□□□□ mermaid □□□□□□□ Diameter □□□□□□□□□□□□□ OmniHSS □□□□□□□□

□□

- □□□□□□ (S6a ULA)
- □□□□□□ (S6a AIA)
- □□□□□□□ (Cx SAA)
- □□□□□□ (Gx CCA)
- □□□□□□ (Sh UDA)
- ME □□□□□□ (S13 ECA)

□□□□□□ (S6a ULA)

□□□□□□□ HSS □ LTE □□□□□□□□ MME□□□□□□□□□□□□ Diameter AVP □□□□□□□

□□□□□□

字段级详情
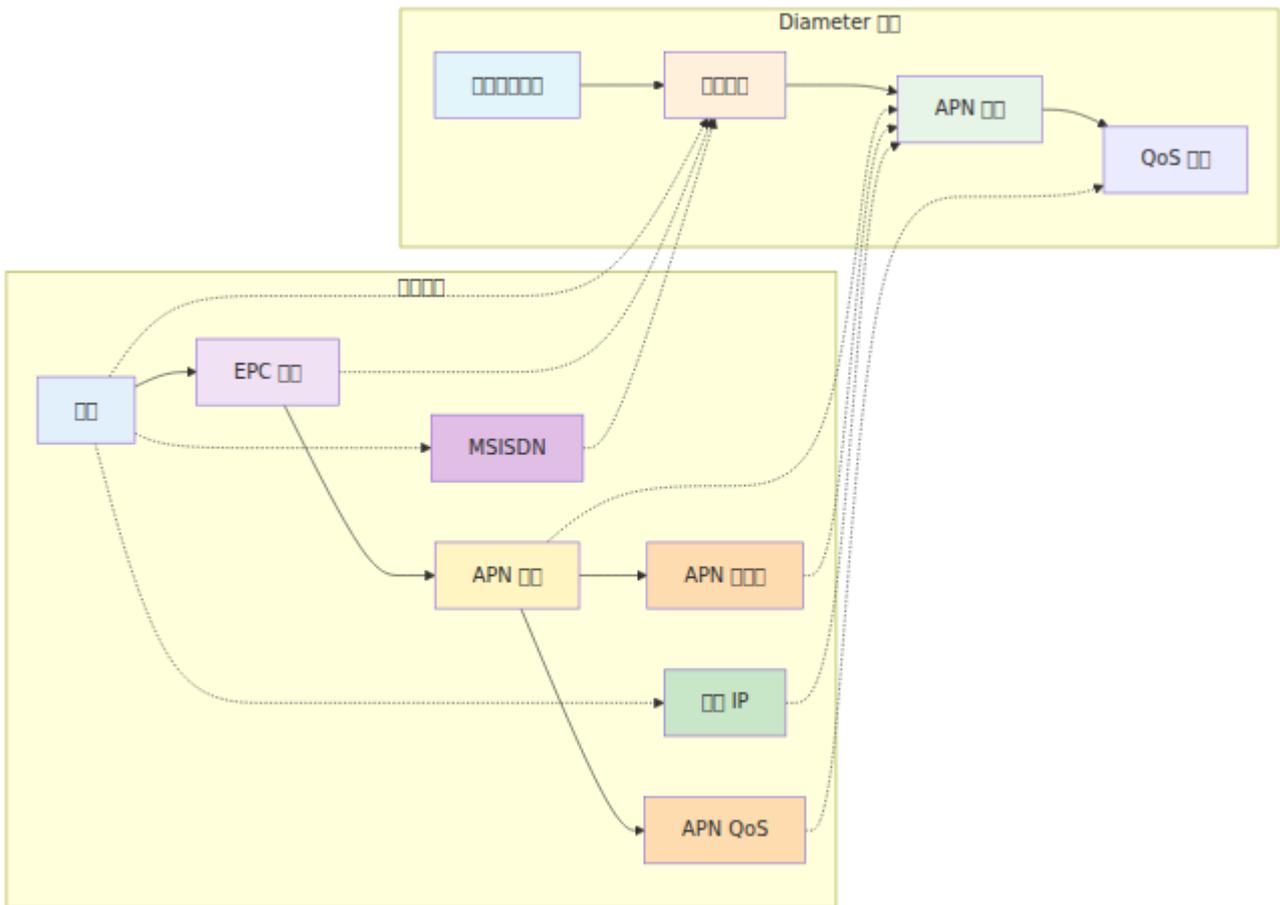
字段名称	示例	D
subscriber.enabled	`true/false`	Su St
msisdn.msisdn	`'14155551234'`	MS
epc_profile.ue_ambr_ul_kbps	`50000`	Ma Re Ba UL
epc_profile.ue_ambr_dl_kbps	`100000`	Ma Re Ba DL
epc_profile.network_access_mode	`'packet_only'`	Ne Ac Mc
apn_identifier.apn	`'internet'`	Se Se
apn_identifier.ip_version	`'ipv4v6'`	PD
apn_qos_profile.qci	`9`	Qc Id
apn_qos_profile.allocation_retention_priority	`8`	Pr Le

配置项键名	取值	D
apn_qos_profile.pre_emption_capability	`false`	Pre en Ca
apn_qos_profile.pre_emption_vulnerability	`true`	Pre en Vu
apn_qos_profile.apn_ambr_ul_kbps	`25000`	AF UL
apn_qos_profile.apn_ambr_dl_kbps	`50000`	AF DL
static_ip.ipv4_static_ip	`'100.64.1.1'`	Se Pa Ac (IF
static_ip.ipv6_static_ip	`'2606:4700::1111'`	Se Pa Ac (IF

关键说明：

1. **AMBR 单位**：配置用 kbps 存储，Diameter 侧用 bps（乘以 1000）
2. **IP 类型枚举**：`0=IPv4, 1=IPv6, 2=IPv4v6, 3=IPv4_or_IPv6`
3. **服务状态**：`enabled: true → 0 (SERVICE_GRANTED)`，`enabled: false → 1 (OPERATOR_DETERMINED_BARRING)`
4. **上下文标识**：每个 APN 有唯一的上下文编号（0, 1, 2...）
5. **静态 IP**：通过嵌套的 `static_ips` 对象关联到具体上下文

数据库存储结构

- 拜访网络必须在归属 PLMN 的 `roaming_profile.roaming_rules`
- 签约状态必须为 `subscriber.enabled == true`
- 针对 APN：已配置的最大会话数、IMS专用承载、默认 IMS APN

鉴权向量生成 (S6a AIA)

本节介绍归属网 LTE/EPC 鉴权的完整流程。

流程概览

生成步骤

1. 输入会话标识符、鉴权算法、密钥集等字段，形成 `key_set` 参数
2. **SQN** 序列号管理与滑动窗口，防止重放攻击并保证同步
3. 米尔纳日算法（3GPP TS 35.206 - 鉴权函数集）
4. **KASME** 派生，依据 TS 33.401 通过 KDF 对 CK||IK 进行绑定

注意事项

- 务必正确管理 SQN，避免失步
- Ki/OPc 密钥仅存于 HSS，不得对外泄露
- AUTN 的生成需要包含SQN、以 AMF 和鉴权管理域
- 重传机制应保证与 UE 侧的序列号保持一致

□□□□□□□ (Cx SAA)

□□□□□□□□ HSS □ IMS □□□□□□□ S-CSCF□

□□□□□

□□□□□
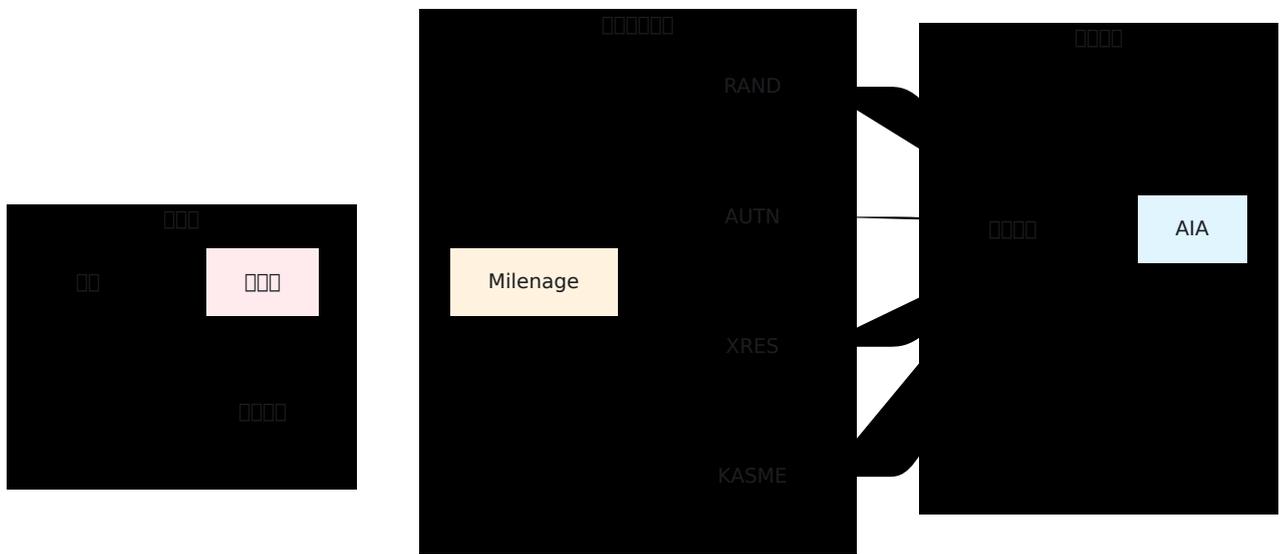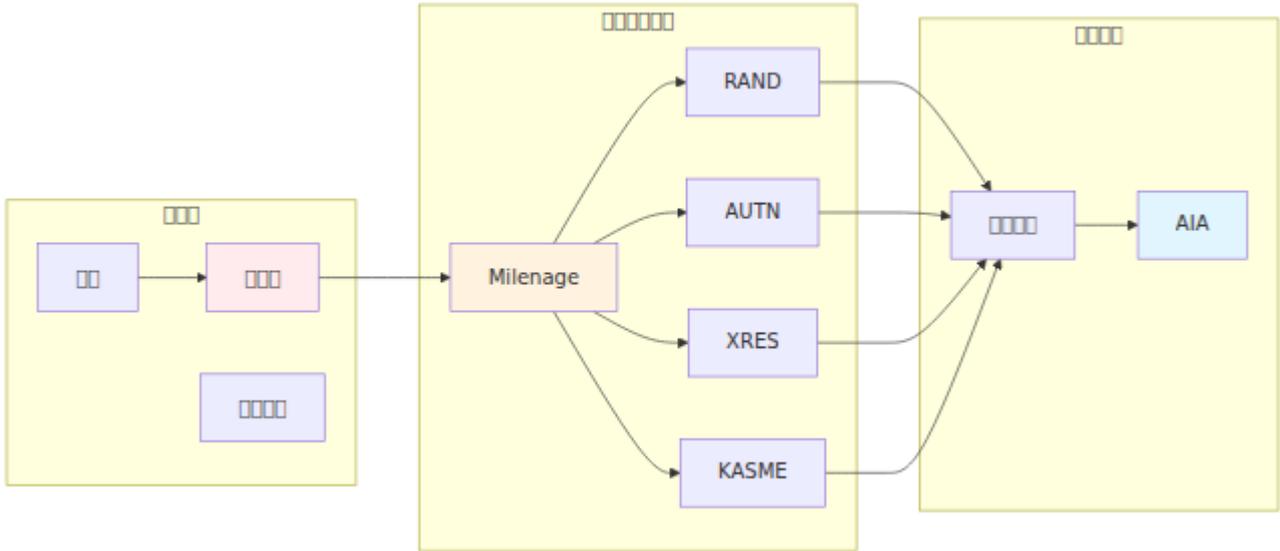
1. **IFC □□□**XML □□□□□ `ims_profile.ifc_template`
2. □□□□□□□□□□□ `{{msisdn}}`□`{{imsi}}`□`{{impu}}`
3. **S-CSCF □□□□** `subscriber_state.assigned_scscf` □□□□□□ S-CSCF
4. **IMS □□□□□□□□□**`sip:+{msisdn}@{ims_domain}` □ `tel:+{msisdn}`

IFC □□□□□

- `{{msisdn}}` - □□□□□□□ MSISDN
- `{{imsi}}` - □□ IMSI
- `{{impu}}` - IMS □□□□□□□□□ subscriber_state□
- `{{impi}}` - IMS □□□□□□□□□□ IMSI@realm□

□□□□□□ (Gx CCA)

□□□□□□□ PCRF □□□□□□□□□□□□ PGW□

□□□□□□

□□□□□□
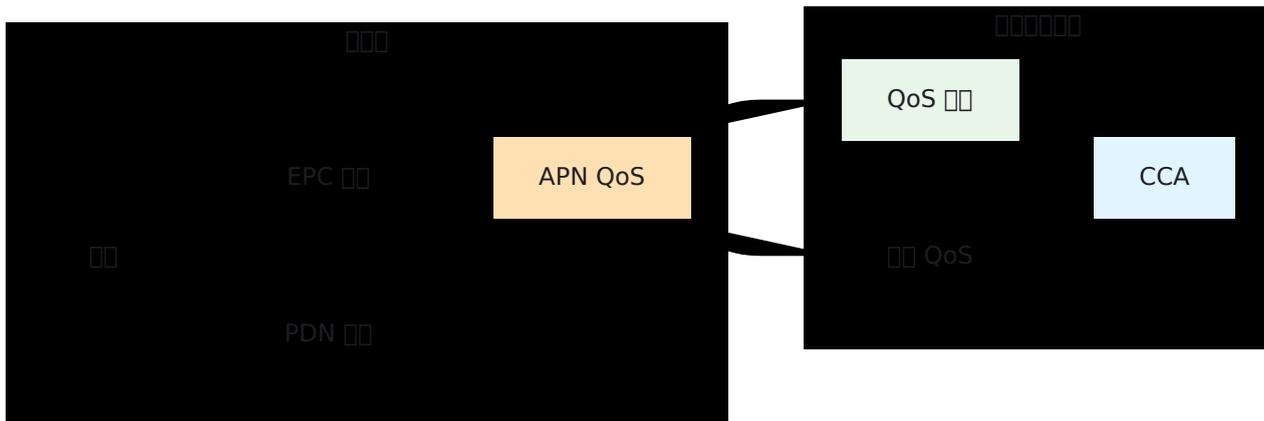
1. □□□□□□□□□□□□/□□ `pdn_session` □□
2. **QoS** □□□□□□□□ APN QoS □□□□ QCI □□□□□□
3. □□□□□□�□□□□□□□□□□□□□□□□□
4. **CC-**□□□□□□□□ INITIAL (1)□UPDATE (2)□TERMINATION (3)

□□□□□□□□

- `INITIAL_REQUEST`□□□□□□ PDN □□□□
- `UPDATE_REQUEST`□□□□□□ PDN □□
- `TERMINATION_REQUEST`□□□□ PDN □□□□

□□□□□□□ (Sh UDA)

□□□□□□□□ HSS □□ Sh □□□□□ AS□□□□□□□□

□□□□□

处理步骤：

1. 查询订阅者状态以获取 `subscriber_state.sh_repository_data` 字段中存储的 XML
2. 若文档已存在，则用请求中的数据替换仓库数据的内容
3. 若数据被修改，则向所有 IMS 分支响应
4. 以成功或故障结果响应用户数据更新请求

ME 身份检查应答 (S13 ECA)

ME 身份检查应答由 EIR 发送，用于向 MME 提供 IMEI 状态。

决策逻辑

处理步骤：

1. **IMEI 检查**：检查请求中是否存在终端信息及其设备
2. **提取 TAC**：从终端信息中提取设备的前 8 位数字
3. **状态查找**：根据 IMEI 查找设备的状态信息
4. **设备状态：**
   - `0` = 设备为白名单或未知
   - `1` = 设备为黑名单/被阻止
   - `2` = 设备为灰名单或被监控
   - `5` = 设备为灰名单且被监控

响应：

- 校验所有 IMEI 并检查是否合法
- 识别 TAC 信息并归类机型品牌
- 查询终端开户信息进行比对
- 识别终端异常使用

协议与接口

该类 Diameter 接口消息结构与 AVP：

配置回 AVP的编码和解码

命令 ID
决定消息类型

生成响应消息
主机和域标识符

源主机
来自config.diameter_host

应用程序 ID
16777251=S6a,
16777216=Cx, 等等

Any Diameter Answer

源域
来自
config.diameter_realm

会话和事务跟踪 ID
消息关联与跟踪

供应商 ID
10415=3GPP

2001:
DIAMETER_SUCCESS

响应与请求匹配的结果
结果代码 + 错误信息

5xxx: 永久失败

实验结果
3GPP 特定代码

配置示例：

```
config :diameter_ex,
  diameter_host: "hss",
  diameter_realm: "example.com",
  diameter_service_name: "OmniHSS"
```

□□□□□

□□□□□□

□□□□

□□□□□□
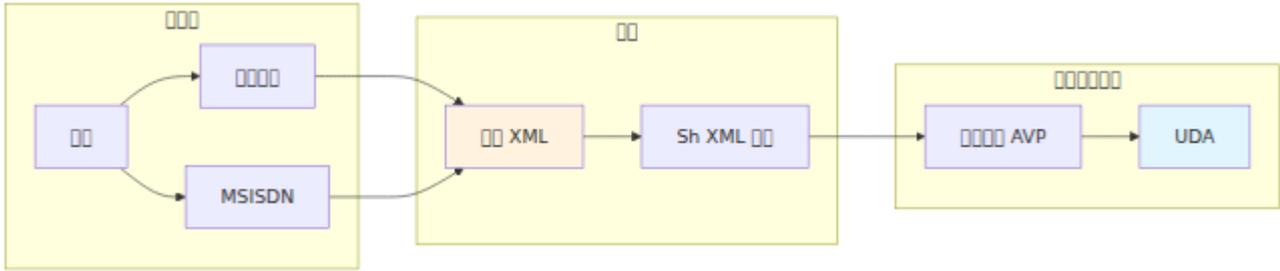
□□□□□□□ Diameter □□□□□□□□□

- **S6a** - LTE/MME □□□□□□□□□□□□□
- **Cx** - IMS/CSCF □□□□□ IMS □□□□□□□□□
- **Sh** - IMS/AS □□□□□□□□□□□
- **Gx** - PCRF □□□□□□□□□□□□
- **Rx** - IMS/AF □□□□□□□□□
- **S13** - EIR □□□□□ IMEI □□
- **SWx** - WiFi/IMS □□□□□□ 3GPP □□□□

□□□□

□□□□□□□�□□□□□□□□□□

- □□ - □□□□□□□□□□ IMSI
- □□□ - □□□□□□□□□□
- **EPC □□** - LTE □□□□□
- **APN □□** - □□□□□□
- **IMS □□** - □□ IFC □□□□ IMS □□□□

- □□□□ - □□□□□□□□
- □□□□ - □□□□□□□□□
- **PDN** □□ - □□□□□□□□□
- □□ **IP** - □□ IP □□□□
- **EIR** □□ - IMEI □□□□

OmniHSS □□□□□□□

□□

- □□□□
- □□□□□□
- □□□□□
- □□□□
- □□□□□□
- □□□□□□
- □□□□

□□□□

OmniHSS □□

□□□

□□□□□□

□□□□□□□□□□□□□□□□□□□

□□□□□

URL: `https://[hostname]:7443/overview`

□□□□□□□□

□□□□□□□□

□□	□□	□□
□□	□□□□□	□□□□□□□□□□□□□□
□□□	MME □□	□□□□□□□□
PDN □□	PDN □□□ > 0	□□□□□□
IMS □���	S-CSCF □□	□□□□□□□□
□□□	□□□□□ > 0	VoLTE □□□□□

□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□

1. □□□□□□□□□□□□□□
2. □□□□□□□□□□□□□□
3. □□□□□□□□□□□□□□□

4. □□□□□□□□□□□□□□□□□□□

Diameter □□□□

URL: `https://[hostname]:7443/diameter`

□□□□

□□□□□□□□

□□□□□□□□□□□□□□□

□□□□□	□□□	□□□□□□□□□□
MME	□	□□□□□□□ LTE □□
P-GW	□	□□□□□
S-CSCF	□	□ IMS □□
P-CSCF	□	□ VoLTE □□
I-CSCF	□	IMS □□□□
AS	□-□	□□□□□□□

□□□□□□

URL: `https://[hostname]:7443/application`

□□□□

类别	指标	当前基线	目标水平
容量规划	基于 Erlang 模型	经验估算为主	> 90% 准确率
资源利用	静态分配为	< 80%	> 90%
故障自愈	依赖人工干预为主导的	N/A	自动化处理

实战案例库

案例一：核心网

通过 SQL 查询分析定位以下故障

案例一

现象：核心网设备告警

- 多用户无法接入
- 部分区域中断
- IMS 注册失败告警

案例二

现象：承载建立失败

- 默认 PDN 连接失败
- 专用 VoLTE 承载异常
- 多 APN 场景下的多个 PDN 连接

案例三

现象：漫游用户接入

- 漫游场景下的MCC-MNC 编码解析与归属地识别
- 国际漫游场景下的测试网络 PLMN 001-001 识别

- □□□□□□□□□□□□□□

□□□□

□□□□□□□□□

- □□□□□□□□□□□□□□□
- □□□ MME □□□□□□□□
- □□□□□□□□□□□□□

□□□□□□□

□□□□□□□□□□□□□

- □□□□□□□□□□□□□
- □□□�□□□□□□
- □□□□□□□□□
- □□□□□□□□□□□

□□□□

□□□□

OmniHSS □□□□□□□ **stdout/stderr**□□□□□□□□□□□□□□

□□□□

□□□□□□□□□□□

Diameter □□□□□:

```
[info] Diameter peer connected: mme01.epc.example.com
[warn] Diameter peer disconnected: pgw01.epc.example.com
[error] Diameter peer connection failed: timeout
```

□□□□□:

```
[info] Database connection established
[error] Database connection lost: timeout
[error] Database query failed: deadlock detected
```

□□□□:

```
[info] Authentication successful: IMSI 001001123456789
[warn] Authentication failed: IMSI 001001123456789, invalid vector
[error] Roaming denied: IMSI 001001123456789, MCC 310 MNC 410
```

□□□□

□□□□□□□□□□□□□□□□

OmniHSS

stdout/stderr

□□□□□□
systemd, supervisord

□□□□

□□□□□

ELK □□   Splunk   □□□
                            CloudWatch, Stackdriver

□□□

口口口口口

口口口口口
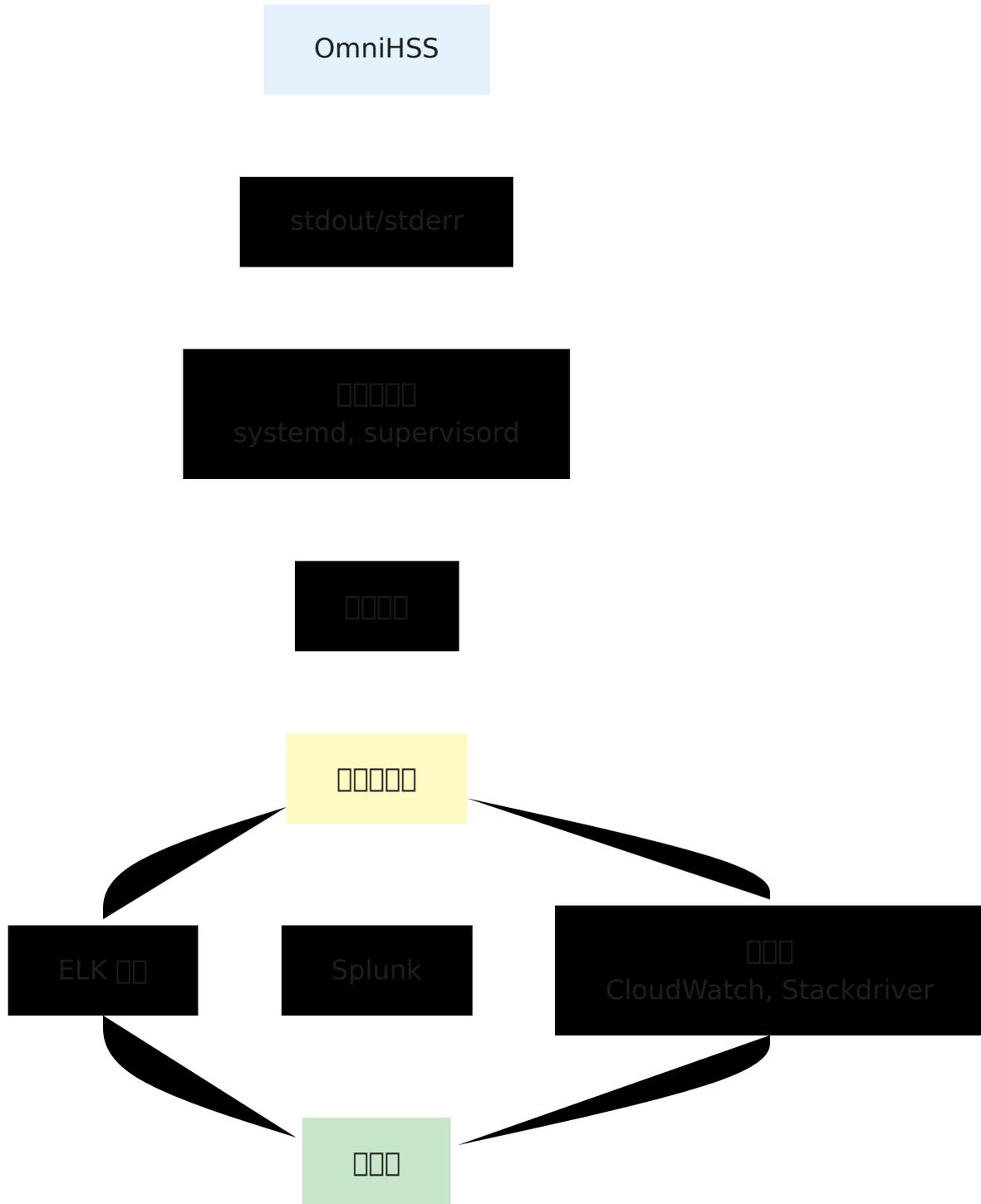
API 口口口: `GET /api/status`

```
curl -k https://hss.example.com:8443/api/status
```

口口口口:

```
{"status": "ok"}
```

HTTP 口口: 200 OK

口口口口口

Nagios/Icinga 口口

```bash
#!/bin/bash
# check_omnihss.sh

API_URL="https://hss.example.com:8443/api/status"

response=$(curl -k -s -o /dev/null -w "%{http_code}" "$API_URL" --max-time 5)

if [ "$response" = "200" ]; then
    echo "OK - OmniHSS API responding"
    exit 0
else
    echo "CRITICAL - OmniHSS API not responding (HTTP $response)"
    exit 2
fi
```

Prometheus 口口

□□□□□□□□□□ OmniHSS □□□□□□ Prometheus□□□□□□ API □□□□□□

SNMP □□

□□□□ SNMP □□□□□□□□□□□□□ SNMP □□□□□□□□□□□ API □□□□□□□□□ SNMP OID □□□□

□□□□□□□

□□ KPI

□□□ KPI □□

KPI	□□	□□	□□
□□□□□□	99.99%	< 99.95%	< 99.9%
Diameter □□□□□□□	99.9%	< 99.5%	< 99%
□□□□□	> 99%	< 99%	< 95%
Diameter □□□□	< 100ms	> 200ms	> 500ms
□□□□□□□	< 50ms	> 100ms	> 500ms
□□□	< 0.1%	> 0.5%	> 1%

□□ KPI

□□	□□	□□□□□□
□□□□	□□□□	80% □□□□□
□□ **PDN** □□	□□□□	70% □□□□□□
□□□□□	□□□ MB	80% □□□□□
□□□□□	□□□□	80% □□□□

发布计划

发布阶段

发布阶段

阶段 1 内部	阶段 2 α	阶段 3 β	阶段 4 正
内部测试 团队内部测试 发现问题	小范围外部用户 覆盖 > 95% 反馈率 > 10%	更大范围外部用户 覆盖 > 85% 稳定性	全面发布 生产环境

发布内容

初始版本 (P1)

核心功能:

- API 网关集成层
- 用户认证与授权
- 请求路由与转发
- 监控: 基础指标监控

集成 Diameter 协议支持:

- 协议转换引擎
- 监控: 协议转换成功率监控

数据存储层:

- 关系型数据库 SQL 查询层

- □□: □□□□□□□□□□□□□□□□

□□□□□□ (P2)

□□ **Diameter** □□□□□:

- □ MME □□
- □ P-GW □□
- □ S-CSCF □□
- □□: □ 15 □□□□□□□□□□□

□□□□□:

- □□ > 95%
- □□: □□□□□□□□□□□□□

□□□□□□:

- 10% □□□□□□□□
- □□: □□□□□□□□□□□□□

□□□□□□ (P3)

□□□□□□□□:

- □□□□□□□
- □□□□□□□
- □□: □ 1 □□□□□

□□□□□□:

- □□ > 85%
- □□: □□□□□□□□□□□□

□□□□□:

- □□□ > 1%
- □□: □□□□□□□□□□□□□

□□□□□□ (P4)

□□□□:

- □□ > 80% □□□
- □□□ > 80% □□□□□
- □□: □□□□□□

□□□□:

- □□□□□□□□□□□
- □□: □□□□□□□

日志的产生

```
OmniHSS
   │
   ▼
stdout/stderr
   │
   ▼
进程管理器
systemd, supervisord
   │
   ▼
日志文件
   │
   ▼
日志收集器
 ┌────────┼────────┐
 ▼        ▼        ▼
ELK 堆栈  Splunk   云服务
                  CloudWatch, Stackdriver
 └────────┼────────┘
          ▼
       聚合器
```

□□□□□□

□□□□

- □□□□□□□□ - □□□□□□
- □□ Diameter □□ - □□□□□□�□□□□
- □□□□□□ - □□□□□□□□□□□
- □□□□□□ - □□ 24 □□□□□□□□
- □□□□□□□□

□□□□

- □□□□□□ - □□□□
- □□□□□□ - □□□□
- □□□□□□□ - □□□□□□
- □□□□□ - □□□□
- □□□□□□ - □□□□□□

□□□□

- □□□□□□ - □□ 6 □□□
- □□□□□□ - □□□□□
- □□□□ - □□□□□□□
- □□□□ - □□□□□
- □□□□□□ - □□□□□□□□

OmniHSS 【MSISDN】与【IMSI】绑定

目录

- 概述
- 【MSISDN】与手机号绑定
- 【IMSI SIM】卡与手机号绑定
- 功能操作
- 功能操作
- 功能操作

概述

OmniHSS 能将你的手机号码与电话号码进行绑定。

【MSISDN】绑定

【IMSI → 手机号绑定】

在移动通信中，IMSI（国际移动用户识别码）与MSISDN（手机号码）是两个不同的标识，分别用于用户识别和通信寻址。

多IMSI SIM 功能

多？？的SIM → 多个IMSI

一个物理SIM可以存储多个IMSI，这使得设备能够在不同的网络之间切换，从而优化覆盖范围、成本或MVNO的选择等方面。

多MSISDN路由与用户模型

架构描述

HSS需要维护多个独立的MSISDN到订阅配置文件的映射。每个IMS身份（即MSISDN）可以路由到IMS中的不同订阅配置文件，从而支持多身份场景。

核心组件

- **主IMSI** - 物理存储在SIM卡上的IMSI
- **多个MSISDN** - 对应不同身份的电话号码
- **IMS身份** - 每个MSISDN对应IMS中的一个
- **订阅配置** - 控制每个身份的服务权限和EPC、IMS网络访问

路由关系

注意： 每个MSISDN可以关联独立的订阅配置，从而支持多个虚拟身份共享同一个MSISDN。

多MSISDN路由与用户模型

□□□□

1. □□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□


```
┌─────────────┐        ┌──────────────────┐      ┌──────────┐
│  □□□□       │───────▶│  □□              │      │ □□□□□   │
│  □□SIM      │        │  +1-415-555-0001 │      └──────────┘
└─────────────┘        └──────────────────┘
                       ┌──────────────────┐      ┌──────────┐
                       │  □□              │      │ □□□□□   │
                       │  +1-415-555-0002 │      └──────────┘
                       └──────────────────┘
```

2. □□□□

□□□□□□□□□□□□□□□□□□□□


```
                        ┌──────┐
                        │  □□  │
                        └──────┘
        ┌──────────────┐  ┌──────────────┐  ┌──────────────────┐
        │  □□□□        │  │  □□□□        │  │  □□□□            │
        │ +1-415-555-  │  │ +44-20-7123- │  │ +49-30-1234-5678 │
        │ 0001         │  │ 4567         │  │                  │
        └──────────────┘  └──────────────┘  └──────────────────┘
```

3. □□□□

□□□□□□□□□□□□□□□□□□□□□□


```
┌─────────────┐        ┌──────────────────┐      ┌──────────┐
│  □□□□       │───────▶│  □□              │─────▶│ □□□□□   │
│  □□SIM      │        │  +1-415-555-0001 │      └──────────┘
│             │        └──────────────────┘
│             │        ┌──────────────────┐      ┌──────────┐
│             │───────▶│  □□              │─────▶│ □□□□□   │
└─────────────┘        │  +1-415-555-0002 │      └──────────┘
                       └──────────────────┘
```

（例如 ，OmniHSS）中预先配置好每个SIM/IMSI，然后将其与适当的MSISDN。

4. 最终呼叫流程

以下是完整的呼叫流程，汇集了上述所有内容：

术语

主要MSISDN

分配给订阅者的主要号码（MSISDN）

```
# □□□□□??MSISDN
curl -k -X POST https://hss.example.com:8443/api/msisdn \
    -H "Content-Type: application/json" \
    -d '{"msisdn": {"msisdn": "14155551001"}}'

# □□□□□□MSISDN
curl -k -X POST https://hss.example.com:8443/api/msisdn \
    -H "Content-Type: application/json" \
    -d '{"msisdn": {"msisdn": "14155551002"}}'
```
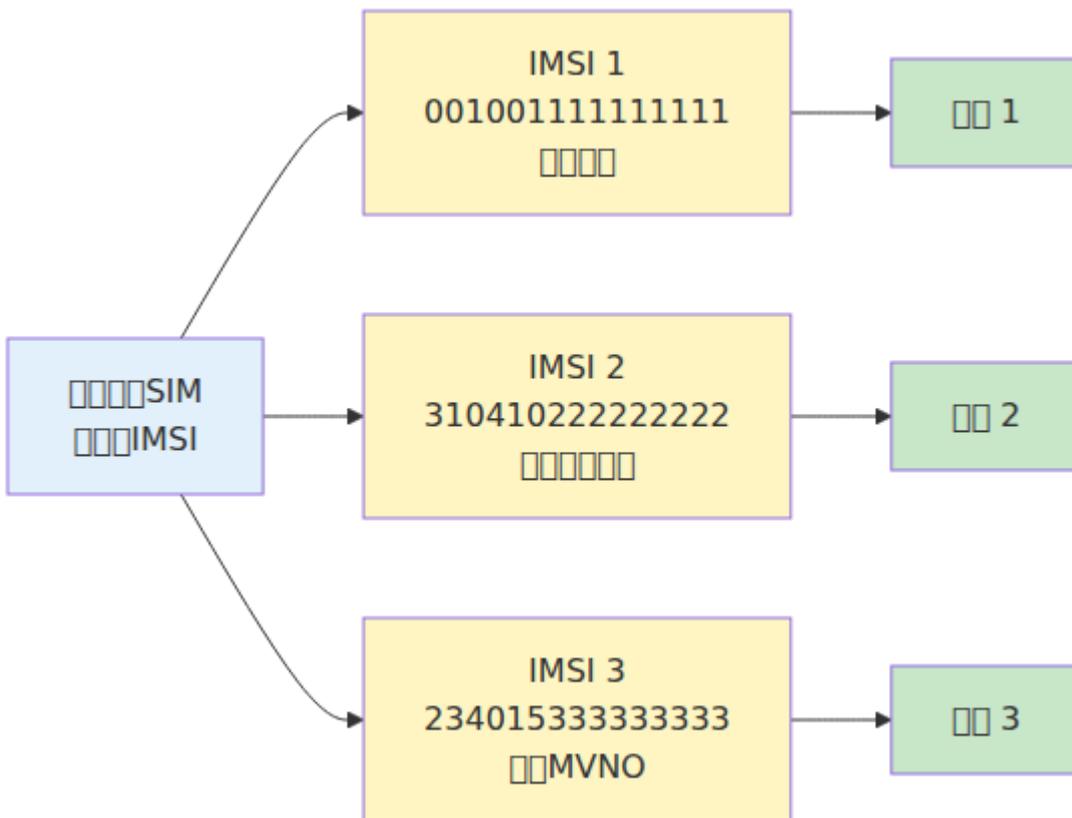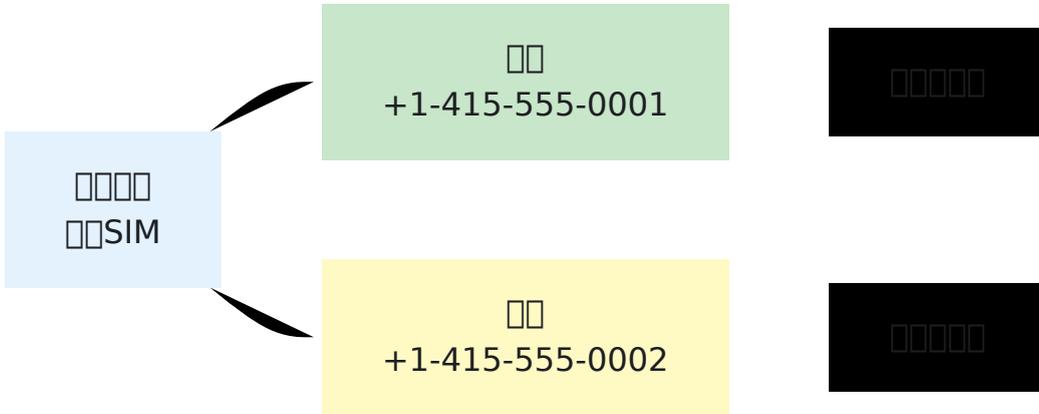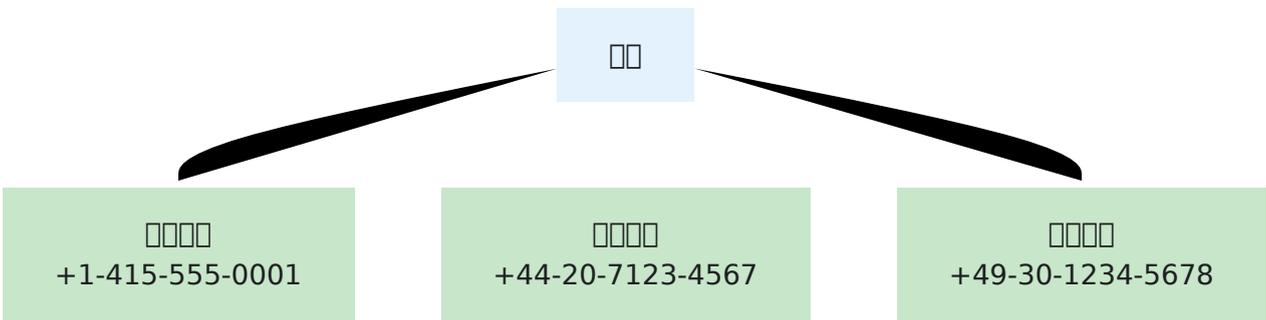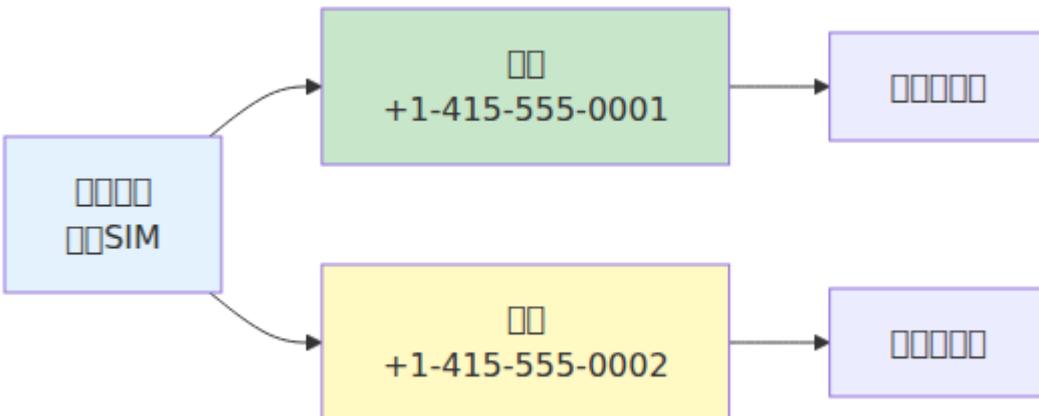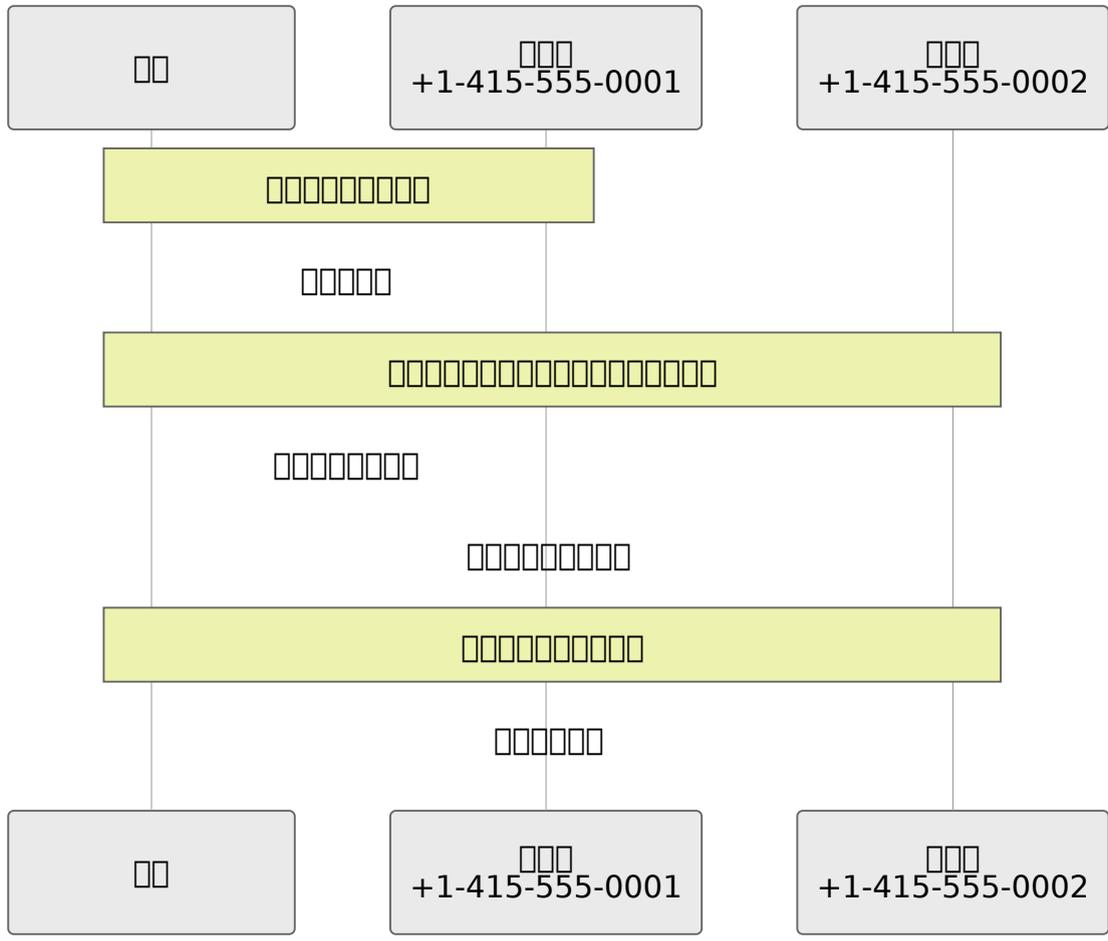
□MSISDN□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□

1. □□□□□□□□□□□□IMSI□□□□ID
2. □□□□□□□□□□□□□□□□MSISDN ID
3. □□□□□□□□□□□□□□subscriber_id□□□□□□msisdn_id

□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□

□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□MSISDN□

- □□□□□□□□□□□□□□
- □□□□□□msisdn□□□□□
- □□□□□□□□□□□□□□□□□□□□□□
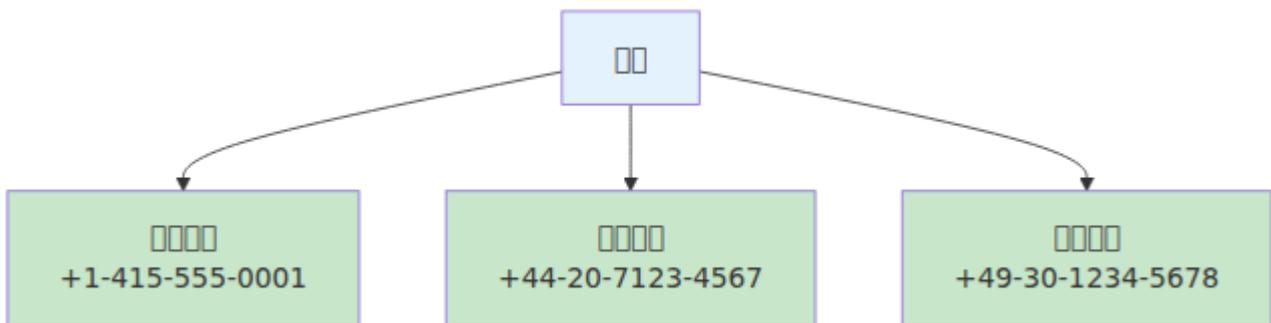
口口口口口口ID（IMSI）口口口口口口口MSISDN口口口口

IMS口口

IMS口口

口口口口口口口IMS口口口口口口口**MSISDN**口口口口口口口口S-CSCF口IMS口口口口口口口口

口口	I-CSCF	OmniHSS	S-CSCF

口口 (IMSI)

Cx UAR (口口口口)

Cx UAA (口口口口)

口口 (口口)

Cx SAR (口口口口)

口口口口
+ 口口MSISDN

口口IFC口口
口口口口MSISDN

Cx SAA
(IFC口口口口口口)

200 OK

口口	I-CSCF	OmniHSS	S-CSCF

IFC口口口口

IMS IFC口口口口口口口 `{{msisdns}}` 口口口口口口口MSISDN口

口口IFC口口口
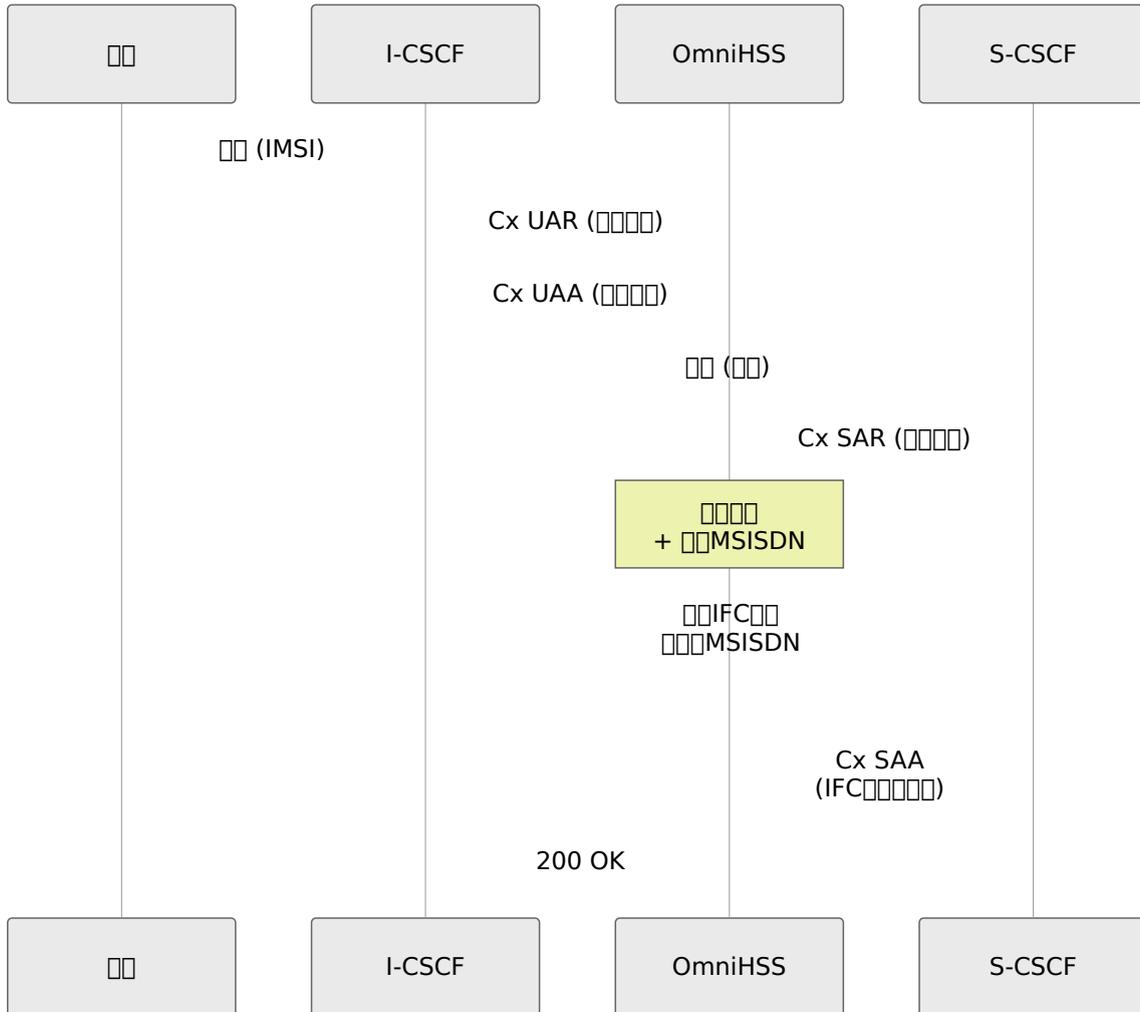
```xml
<ServiceProfile>
  <PublicIdentity>
    <Identity>sip:
{{imsi}}@ims.mnc{{mnc}}.mcc{{mcc}}.3gppnetwork.org</Identity>
  </PublicIdentity>
  <!-- 下面是MSISDN身份 -->
  <PublicIdentity>
    <Identity>sip:+14155551001@ims.example.com</Identity>
  </PublicIdentity>
  <PublicIdentity>
    <Identity>tel:+14155551001</Identity>
  </PublicIdentity>
  <PublicIdentity>
    <Identity>sip:+14155551002@ims.example.com</Identity>
  </PublicIdentity>
  <PublicIdentity>
    <Identity>tel:+14155551002</Identity>
  </PublicIdentity>
  <!-- ... -->
</ServiceProfile>
```

模板变量：

- `{{msisdns}}` - 用户的一个或多个MSISDN列表。

图表说明

一个MSISDN号码对应多种IMS公共身份。

MSISDN: 14155551001

SIP URI
sip:+14155551001@ims.example.com

TEL URI
tel:+14155551001

图表后说明：

此图展示了一个号码在IMS系统中的多种表示形式（SIP URI、

信令流程说明

这个流程展示了一个用户如何通过电话号码发起呼叫。
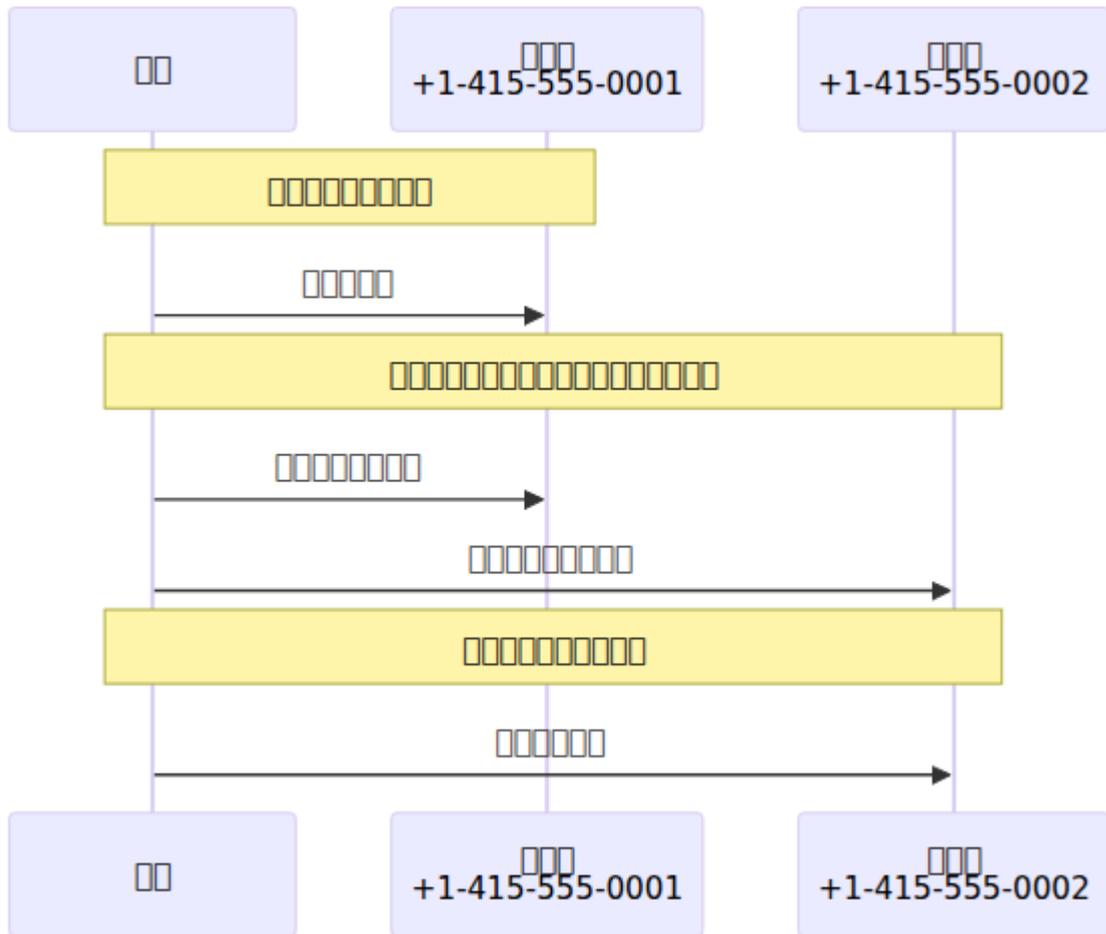
SIP INVITE请求头

```
INVITE sip:+15105551234@ims.example.com SIP/2.0
From: "+14155551002" <sip:+14155551002@ims.example.com>;tag=123
To: <sip:+15105551234@ims.example.com>
P-Asserted-Identity: <sip:+14155551002@ims.example.com>
```

`From`和`P-Asserted-Identity`头部包含了主叫用户的电话号码。

设备识别与MSISDN

当使用MSISDN注册到IMS网络时

时需要

- S-CSCF□□□□□□□□□□□
- □□□□□□□□□□□□

□□□□□□□□

1. **□□□□□□□MSISDN□□□□**

   - □□□□□□□□□□□□□□□□□IMSI□MSISDN
   - □□□□□□□□□□□□□□□

2. **□□IMS□□□□□□□□□**

   - □□□□□□□□ `{{msisdns}}` □□
   - □□□□□□□□□□□XML

3. **□□HSS□□□□**

   - □□IMS□□□□Cx SAR□□□□
   - □□□□MSISDN□□□□□□□□□□

4. **□□IMS□□□□**

   - □□□□□□□□□□□□
   - □□□S-CSCF□□□□□□□□□□□□□□

□□□□□□MSISDN□□□□□□

□□□□

- □□□□□□□□□
- □□□□"□□□□"□"□□□□"

□□□□□

1. **MSISDN□□□□**

   - □□□□□□□□□MSISDN□□□□□□□□□□□□
   - □□□□□□ □□□□□□□□□□□□□□□□□

2. **MSISDN□□□□**

- ○ □□□□□□□□MSISDN□□□□□□□
- ○ □□□□□ □□□□API□□□□□□□□MSISDN□□□

□□□□□□□□□□□□□□□□□□

□□□

- □□□□□□□□
- □□□□□□□□□□□□□□□□

□□□□□□□

1. **□□IMS□□□□□□□□□□**

   - ○ □□S-CSCF□□□□□□□□
   - ○ □□□□SIP URI□□□□

2. **□□IMS□□□□□**

   - ○ □□IFC□□□□□□□□□□□□□□
   - ○ □□□□□□□□□□□□□□□

3. □□□□□□□

```
# □SIP□□□□□□
sip:+14155551001@ims.example.com  # □□□□
sip:+14155551002@ims.example.com  # □□□□□
```

□□□API□□MSISDN□□□□□□□□□

□□□

- API□□`/api/subscriber/msisdn/:msisdn`□□□□□□□

□□□

□□□□□□□□MSISDN□□

□□□□

□□□□

1. □□□□□□MSISDN
2. □□□□
3. □MSISDN□□□□□
4. □□□□□□□□□

MSISDN□□

- □□□□□□□□□□□□□custom_attributes□
- □□□□□□□□□□□□□
- □□□□□□□□□□□□�□□□□□□□□□□□

IMS□□

- □□IFC□□□□□□□□□□□□□
- □□□□□□□□□□□□
- □□□□□□□□□□□□

□□

□□□□□□□MSISDN□□

分配一个1个MSISDN

分配一个MSISDN

删除一个号码

是否存在号码

否

是

删除一个号码

激活号码

结束

多IMSI SIM卡的技术原理

基本概念

多IMSI SIM卡是一种特殊类型的通信卡，它能够存储多个IMSI（国际移动用户识别码）。每个IMSI代表一个独立的用户身份，这使得用户可以在不同的网络之间切换。

传统的 单一身份信息的IMSI，这意味着一张普通的SIM卡只能对应一个IMSI。当HSS收到请求时，会根据用户提供的IMSI进

OmniHSS□□

□OmniHSS□□□IMSI SIM□□□□IMSI□□□□□□□□□□□□□□□□□□□SIM□□□

```
                    SIM□
                   ICCID:
              8991101200003204510
```

□□ 1	□□ 2	□□ 3
IMSI: 001001111111111	IMSI: 310410222222222	IMSI: 234015333333333

| EPC□□□□□□□ | EPC□□□□□□□□□ | EPC□□□□□□□□□□□ |

□□□□
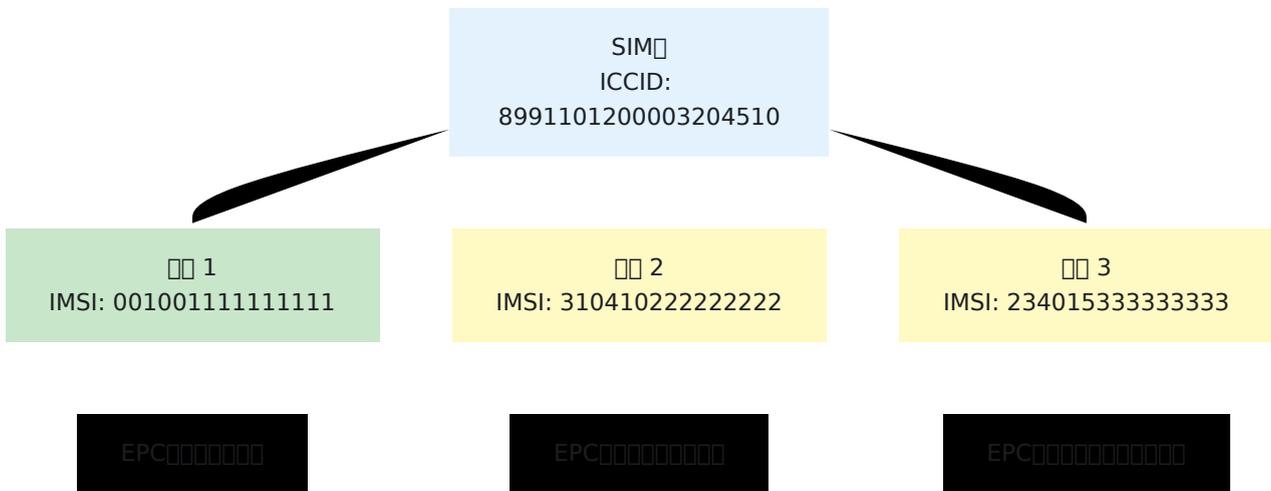
1. □□□□□□

- □□IMSI□001-001□□□□□□□□□□
- □□□□IMSI□310-410□□□□□□□□□□
- □□□□IMSI□234-015□□□□□□□□□□
- □□□□□□□□□IMSI

2. MVNO□□

- □□IMSI□MVNO□□□□□□□□□
- □□IMSI□□□□□□□□□□□
- □□MVNO□□□□□□□□□□□□□□

3. IoT/M2M□□□□

- IMSI 1□□□□□□□
- IMSI 2□□□□□□□□□□
- IMSI 3□□□□/□□□□□□
- □□□□□□□□□

4. □□□□

- □□□IMSI□□□□□□□□□□□
- □□□□□□□□□□□□□
- □□□□□□□□□□□□□□□

□IMSI□□

□□□□□□

- □□IMSI□□□□□□Ki□OPC□□□□□
- □□IMSI□□□□□□□□□□
- □□□□□□□□□□□□□

□□□□□□□□

- □□□□EPC□□□□□□□□□APN□
- □□□□IMS□□□□□□□□□◆◆□□
- □□IMSI□□□□□□□□□

□□□□□□

- □□IMSI□□□□□□SIM□□□□sim_id□
- □□□□□□□□□□□□□ICCID
- □□SIM□□□□□□□□

□□□□

- □□□□SIM□□□□□□□□IMSI
- □□□□□□□□□□□□□□
- HSS□□□□□□□□□□□IMSI

批量

```bash
# 1. 创建SIM卡（不包含IMSI）
SIM_ID=$(curl -k -X POST https://hss.example.com:8443/api/sim \
  -d '{"sim": {"iccid": "89911012000003204510", "is_esim": false}}' \
  | jq -r '.data.id')

# 2. 为IMSI 1创建密钥集（认证参数）
KEYSET1=$(curl -k -X POST https://hss.example.com:8443/api/key_set \
  -d '{"key_set": {"ki": "0123456789ABCDEF...", "opc":
"FEDCBA9876..."}}' \
  | jq -r '.data.id')

# 3. 创建用户 1（首个IMSI）
curl -k -X POST https://hss.example.com:8443/api/subscriber \
  -d "{\"subscriber\": {
    \"imsi\": \"001001111111111\",
    \"sim_id\": $SIM_ID,
    \"key_set_id\": $KEYSET1,
    \"epc_profile_id\": 1
  }}"

# 4. 为IMSI 2创建密钥集（不同的认证参数）
KEYSET2=$(curl -k -X POST https://hss.example.com:8443/api/key_set \
  -d '{"key_set": {"ki": "1111111111111111...", "opc":
"2222222222..."}}' \
  | jq -r '.data.id')

# 5. 创建用户 2（次个IMSI）
curl -k -X POST https://hss.example.com:8443/api/subscriber \
  -d "{\"subscriber\": {
    \"imsi\": \"310410222222222\",
    \"sim_id\": $SIM_ID,
    \"key_set_id\": $KEYSET2,
    \"epc_profile_id\": 2
  }}"

# 6. 此SIM卡现有两IMSI配置文件...
```

□□□□□□

□□IMSI□□□□□□

HSS□□□□□□□□□□□□IMSI SIM——□□□□□□□□□□□□□□□IMSI□

IMSI□□□□□□□□

□□IMSI SIM□□□□IMSI□□□□□□□IMSI□□□□□□□□□□□□□□IMSI□OmniHSS□□□□HSS□□□**□□□□□□□□□CLR□**□□□□□□□□□�□□□□□□□□□□□□□□□IMSI□

□□□□IMSI□□

□□□□□ □□SIM□□□□□□□□□□□IMSI□□□□□□□□□□□□□□□□□

- □□□□□□MME□□□□□□**IMSI X**
- □□HSS□□□□□□**IMSI Y**□□IMSI X□□□□SIM□□□□□□□□□□□□□□□□

- HSS**会根据注册逻辑判断已存在的IMSI X**

被相同IMSI卡上的设备抢占,执行清除流程

IMSI抢占流程


```
Device          MME          OmniHSS

设备当前使用IMSI X
0010011111111111

                IMSI X已经注册
                并指向MME-001

设备重启后使用同一
SIM卡下的IMSI Y

      附着请求 (IMSI Y: 310410222222222)

                    S6a AIR (IMSI Y)

                    S6a AIA (IMSI Y鉴权向量)

        鉴权请求

        鉴权响应

                    S6a ULR (更新IMSI Y位置)

                            数据库查询同一卡上的
                            其他已注册sim_id

        SELECT * FROM subscriber
WHERE sim_id = (SELECT sim_id FROM subscriber WHERE imsi='310410...')
                AND imsi != '310410...'
            AND subscriber_state is REGISTERED

                            找到旧IMSI X的注册��
                            为MME-001）

            S6a CLR (清除IMSI X的注册)
                发送给旧MME-001

            清除IMSI X
            上下文

                    S6a CLA (清除确认成功)

                            将IMSI X的注册信息
                            更新为IMSI Y

                    S6a ULA (IMSI Y位置更新成功)

        附着完成（IMSI Y成为唯一活跃设备）

    IMSI Y激活成功

            IMSI X被抢占
            IMSI Y成功激活

Device          MME          OmniHSS
```

□□□□□□□

□□□□□□

- □□□□□□□□SIM□□□□□□
- □□□□□□□□□□□□
- □□□□□□□□□□□□

□□□□□□

- □□□□□□□□IMSI□□□□□□□□□□
- IMSI□□□□□□□□□□□□
- □□□□CDR□□□□□□□□□□

□□□□□

- □IMSI□MME□□□□□□
- PDP□□□□□□□□□□
- □□□□□□□□□

IMSI□□□□□□

□□/SIM□□□□□□IMSI□□□□□

1. □□□□□

   - □□IMSI□□□□□□
   - □□□□□□□□□IMSI

2. □□□□

   - □□□□□□□□
   - SIM□□□□□□IMSI

3. □□□□

   - SIM□□□□□□□□□□□□□□□□□□□□□IMSI□
   - □□MCC/MNC□□□□□□

4. □□□□

- □□□□□□□□□□□IMSI
  - □□□□IMSI□□□□□□□□

IMS□□□□

□□□□□□□□□□□□□□□IMS□□□□

□□□□

□□□□□□□□
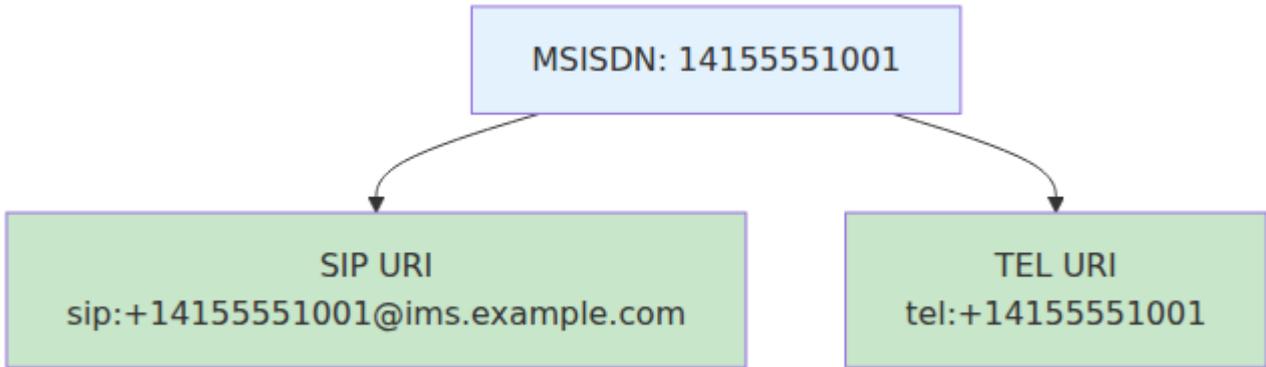
1. □□□□□□□ □IMSI□□□□□□IMSI□□HSS□□□□□"□□"□□□□□□□□□□

2. □□SIM□□□□□□□□□□ □IMSI SIM□□□□□□□□□□□□□□ `sim_id` □□□□□□□□□□"□□"□□□□

3. □□□□□□ `subscriber_state` □□□□□□IMSI□□□□MME/SGSN□□IMSI□□□□□□□□□□□□

4. □□□□□□ □□□□□□□□□□□

   - □□□□□□□□□□□□IMSI
   - □□□□□□□□□□□□□□IMSI
   - □□□□□SIM□□□□□IMSI□□□□□□□

□□□□

□IMSI + □MSISDN

□□□□□□□□□□□□□□□SIM□□□□IMSI□□□□IMSI□□□□□MSISDN□

```
               ┌──────────────┐
               │  □IMSI SIM   │
               └──────────────┘
            ╱                     ╲
┌────────────────────┐      ┌────────────────────┐
│      IMSI 1:       │      │      IMSI 2:       │
│  0010011111111111  │      │  310410222222222   │
└────────────────────┘      └────────────────────┘
         │                           │
     ┌───────┐                   ┌───────┐
     │ □□ 1  │                   │ □□ 2  │
     └───────┘                   └───────┘
      ╱      ╲                    ╱      ╲
┌──────────┐ ┌──────────┐  ┌──────────┐ ┌──────────┐
│MSISDN 1A:│ │MSISDN 1B:│  │MSISDN 2A:│ │MSISDN 2B:│
│ +1-415-  │ │ +1-415-  │  │ +1-212-  │ │ +1-212-  │
│ 555-1001 │ │ 555-1002 │  │ 555-2001 │ │ 555-2002 │
└──────────┘ └──────────┘  └──────────┘ └──────────┘
```

□□□□□
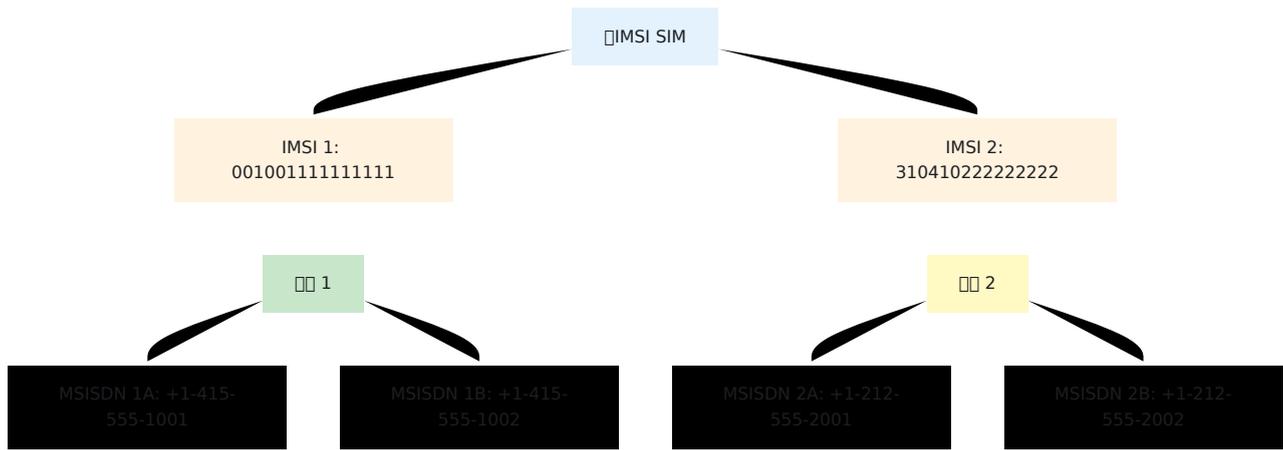
- □□□□□□**IMSI 1**□□

  - □□□□□□+1-415-555-1001
  - □□□□□□+1-415-555-1002

- □□□□□□□□**IMSI 2**□□

  - □□□□□□+1-212-555-2001
  - □□□□□□+1-212-555-2002

□□□□□□□□□□□□IMSI 1□□□MSISDN□□□□□□□□□□□□□□□□□IMSI 2□□□□□□□□MSISDN□

□□□□

□□□□MSISDN□□□

□□□□□□□□MSISDN□

> □□API□□□□GET /api/subscriber/imsi/:imsi

□□□□□□□□□□MSISDN□

□□□□□IMSI

□□□□□□□□□IMSI□□□□

1. □□□IMSI□□□□□□□□□□□□
2. □□□IMSI□key_set□□□□□□□□
3. □□□□□□□□EPC□□□□□
4. □□□□□□□□□□□□

□□□□□□IMSI□

- □□□□□/SIM□□□□□□□□□HSS
- HSS□□□□□□□□□IMSI
- □□□□IMSI□□□□□

□□□□□MSISDN

□□□□□□□□□□

1. □□MSISDN□□□□□□□□□□
2. □□IMS□□□□□□□□□□□ `{{msisdns}}` □□
3. □□IMS□□□□□□□□□□□□□
4. □□S-CSCF□□□□□□□□□□□

□□□□□□□□□□□□□

- □□□□□□□□□□□□□□□
- □□□□□□□□□□□□HSS
- HSS□□□□□□□□□□□□□

□□□□

□MSISDN□□□

✓ □□SIM□□□□□□ ✓ □□□□□□□□□□ ✓ □□□□□□□ ✓ □□□□□□□ ✓ □□□□□□□□□□□□□ ✓ □IMSI□□□□□

□IMSI SIM□□

✓ □□□□□□ ✓ □□□□□□ ✓ □□□□□□□ ✓ □□□□□□ ✓ □□□□ ✓ □□□□□□□□□

□□□□

✓ □□□□□ ✓ □□□□□□��□□□□ ✓ □□□□□□□□□□ ✓ □□□□□□□ ✓ □□□□□□□

← □□□□□□

PCRF（策略与计费规则功能）

概述

HSS 是用户，而 PCRF则是规则。它存储着网络该如何对待每一个用户的策略。PCRF 是在 LTE 核心网中负责动态策略控制与QoS管理的关键网元，决定着网络。

核心功能

- **Gx 接口**：PGW/PCEF，下发策略与计费/限速规则，实现动态带宽控制。
- **Rx 接口**：IMS（IP 多媒体子系统）交互，保障语音 QoS
- 支持动态策略更新，可主动发起RAR，实时调整用户规则。
- **VoLTE** 场景下，为语音承载分配专用 QoS 与保障带宽。
- 策略下发可包含流模板（TFT），用于区分不同数据流。
- **REST API**，便于与上层业务系统对接。

接口

Diameter 接口

接口	应用 ID	对端	用途
Gx	16,777,238	PGW (PCEF)	PDN 连接建立、QoS 与计费规则下发
Rx	16,777,236	P-CSCF (AF)	IMS 会话媒体授权请求

典型工作流程

PCRF 是承载 PDN 的��持 VoLTE 语音承载的核心。
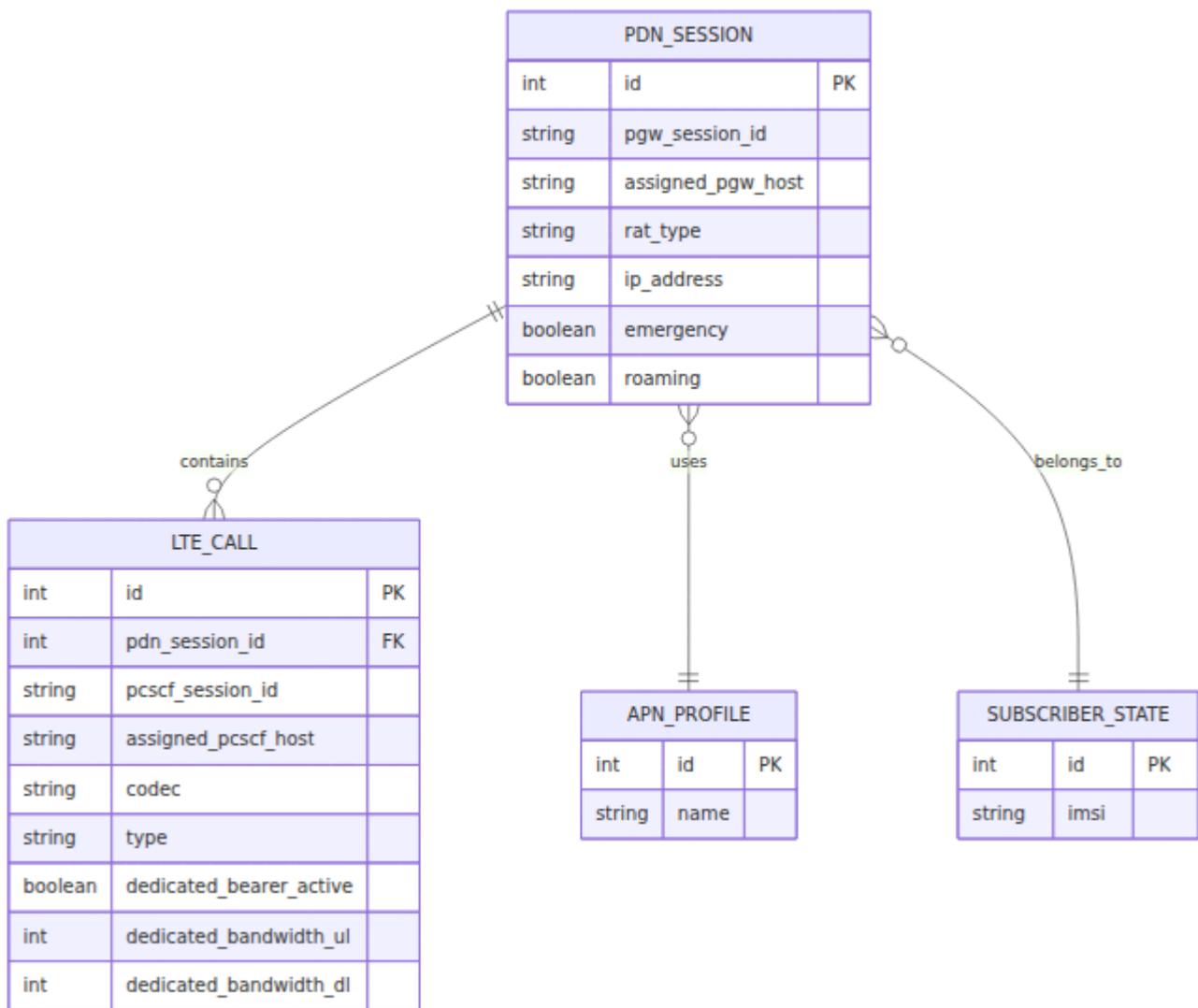
Gx □□

□□□□□

1. □□□□□□ - □□□□CCR-I□

□□□□PGW □□□□□□□□□□ PDN □□

□□ AVP□

- Session-Id
- Origin-Host, Origin-Realm
- Subscription-Id□□□□ IMSI□
- Called-Station-Id□APN □□□

- IP-CAN-Type（IP 接入网络类型）
- RAT-Type（无线接入类型）
- Framed-IP-Address（UE IP 地址）

PCRF □□□

1. □□ IMSI □□□□□
2. □□ APN □□□ QoS □□
3. □□□□□□□□
4. □ APN □□□□ QoS □□

□□ AVP：

- Result-Code: 2001（DIAMETER_SUCCESS）
- QoS-Information（APN □□□□□□□）
- Default-EPS-Bearer-QoS（QCI、ARP□□□□□）
- Bearer-Control-Mode

2. □□□□□□ - □□□（CCR-U）

□□□□PGW □□□□□□□□□□□□RAT □□□□

PCRF □□□

1. □□□□ ID □□□□□□
2. □□□□□□□RAT □□□□□□□
3. □□□□□□□□□□□

□□□□Result-Code 2001□□□□□□□□□□□

3. □□□□□□ - □□□（CCR-T）

□□□□PGW □□ PDN □□

PCRF □□□

1. □□□□ ID □□□□
2. □□□□□□□□□□□
3. □□□□

□：□Result-Code 2001

4. □□□□□□□（RAR）

方向：PCRF → PGW／HSS □□□

触发：

- IMS □□□□（Rx AAR □□ Gx RAR）
- IMS □□□□（Rx STR □□ Gx RAR）
- □□ REST API □□□□□□

RAR AVP：

- Session-Id（PGW □□ ID）
- Auth-Application-Id: 16,777,238
- Re-Auth-Request-Type（0 = □□□□）
- Charging-Rule-Install/Remove
- QoS-Information（□□□□□□□□）

PGW □□□□□□□□□□□/□□/□□□□□□

□□□□□□□□□□□□

PCRF □□□□□□□□□□□□TFT□□□□□□□□□□□□□

- □□□□□□ - □□□□□□□□□□□□□□□□□
- □□□□ - □□□□□□□□□□□□□□□□□
- □□□□□□□ - □□□□□□□□□□□□□□□ QoS

□□□□□□□：

- □□□□□□□□ Gx RAR □□□□
- □□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□5 □□□□□（□/□□ IP□□/□□□□□）TFT □□

□□□□：

- □□□ - □□□□□□（Spotify、WhatsApp、Facebook）□□□□□□□□□□□□□□□□

- 🔲🔲🔲🔲 - 🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
- 🔲🔲🔲🔲 - 🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
- 🔲🔲🔲🔲🔲🔲 - 🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
- 🔲🔲🔲🔲 - 🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
- 🔲🔲 **SLA** - 🔲🔲🔲🔲🔲🔲🔲🔲🔲 QoS

QoS 🔲🔲🔲🔲

🔲🔲🔲🔲 **QoS🔲🔲🔲** APN 🔲🔲🔲🔲

```
{
  "QoS-Class-Identifier": 9,           // QCI🔲9 = 🔲🔲🔲🔲🔲
  "APN-Aggregate-Max-Bitrate-UL": 50000,  // kbps
  "APN-Aggregate-Max-Bitrate-DL": 100000, // kbps
  "Allocation-Retention-Priority": {
    "Priority-Level": 8,
    "Pre-emption-Capability": 1,      // 🔲🔲🔲🔲
    "Pre-emption-Vulnerability": 1    // 🔲🔲🔲🔲🔲
  }
}
```

🔲🔲🔲🔲 **QoS🔲🔲🔲** VoLTE🔲🔲

```
{
  "QoS-Class-Identifier": 1,           // QCI 1 = 🔲🔲🔲🔲
  "Max-Requested-Bandwidth-UL": 128000,   // bps
  "Max-Requested-Bandwidth-DL": 128000,   // bps
  "Guaranteed-Bitrate-UL": 128000,
  "Guaranteed-Bitrate-DL": 128000
}
```

Rx 🔲🔲

🔲🔲🔲🔲🔲

1. AA 🔲🔲🔲AAR🔲/ AA 🔲🔲🔲AAA🔲

方向：P-CSCF 告知 IMS 会话的媒体需求（VoLTE 语音流）。

关键 AVP：

- Session-Id（P-CSCF 新建会话）
- Subscription-Id（IMSI 或 SIP URI）
- Media-Component-Description
    - Media-Type（音频、视频）
    - Max-Requested-Bandwidth-UL/DL
    - Codec-Data
    - Flow-Description（5 元组，定义媒体流）
- AF-Application-Identifier

PCRF 处理：

1. 根据 IMSI 或 SIP URI 识别用户
2. 关联已有 IMS 承载
3. 将媒体需求转换为授权的服务质量参数
4. 生成动态规则
5. 发送 **Gx RAR** 给 PGW（下一步动作）
6. 等待 Gx RAA 确认
7. 返回 Rx AAA 给上一跳。

关键 AVP：

- Result-Code: 2001（成功）或 5063（资源不可用）

2. 会话终止请求（STR）/ 会话终止应答��（STA）

方向：P-CSCF 告知 IMS 会话已经终止。

PCRF 处理：

1. 根据 P-CSCF 会话 ID 定位会话。
2. 发送 **Gx RAR** 给 PGW（删除规则）
3. 释放相关资源。
4. 返回 STA 确认

成功：Result-Code 2001

成功场景

场景 1：PDN 首次接入

PGW (PCEF)   HSS (PCRF)   Database

Gx CCR-I
(IMSI, APN, RAT-Type, IP)

查询签约数据

返回签约数据

查询 APN 配置

APN QoS 参数

创建 pdn_session

会话已创建

下发 QoS 策略
(QCI, AMBR, ARP)

Gx CCA-I
(QoS-Information,
Default-EPS-Bearer-QoS)

安装 QoS 规则并建立承载

PGW (PCEF)   HSS (PCRF)   Database

图 2：VoLTE 呼叫建立（Rx AAR → Gx RAR）

| PGW (PCEF) | HSS (PCRF) | Database |

Gx CCR-I
(IMSI, APN, RAT-Type, IP)

查询订阅者

订阅者信息

查询 APN 配置

APN QoS 策略

创建 pdn_session

会话已创建

计算 QoS 参数
(QCI, AMBR, ARP)

Gx CCA-I
(QoS-Information,
Default-EPS-Bearer-QoS)

发送 QoS 参数给承载建立

场景 3：VoLTE 通话结束（Rx STR → Gx RAR）

```
P-CSCF            HSS            Database          PGW
(IMS AF)         (PCRF)                           (PCEF)

    │ Rx STR          │                │               │
    │ (Session-Id)    │                │               │
    │                 │                │               │
    │                 │ 查询           │               │
    │                 │ P-CSCF 会话 ID 对应 lte_call    │
    │                 │                │               │
    │                 │ lte_call 详情  │               │
    │                 │                │               │
    │                 │ Gx RAR         │               │
    │                 │ (Charging-Rule-Remove)         │
    │                 │                │    ┌──────────────┐
    │                 │                │    │ 删除专用承载 │
    │                 │                │    └──────────────┘
    │                 │ Gx RAA         │               │
    │                 │ (Result-Code: 2001)            │
    │                 │                │               │
    │                 │ 更新 lte_call 状态             │
    │                 │                │               │
    │                 │ 已更新         │               │
    │ Rx STA          │                │               │
    │ (Result-Code: 2001)              │               │

P-CSCF            HSS            Database          PGW
(IMS AF)         (PCRF)                           (PCEF)
```

场景 4：PDN 会话更新

PGW
(PCEF)

HSS
(PCRF)

Database

Gx CCR-U
(Session-Id, 新的 RAT-Type,
其他信息)

查询
PGW 会话 ID 获取 pdn_session

pdn_session 信息

更新会话
(RAT 类型和其他属性变化)

更新成功

评估策略变化
(根据新的数据RAT 类型等)

Gx CCA-U
(Result-Code: 2001,
更新后的策略)

PGW
(PCEF)

HSS
(PCRF)

Database

场景 5：PDN 连接释放

P-CSCF (IMS AF) | **HSS (PCRF)** | **Database** | **PGW (PCEF)**

Rx STR
(Session-Id)

查询
P-CSCF 会话 ID 对应 lte_call

lte_call 信息

Gx RAR
(Charging-Rule-Remove)

删除计费规则

Gx RAA
(Result-Code: 2001)

删除 lte_call 信息

已删除

Rx STA
(Result-Code: 2001)

场景 6：发起 REST API 触发的重鉴权


```
 Administrator      HSS REST API      PCRF Core        Database        PGW
                                                                      (PCEF)

   POST /api/operation/pcrf_re_auth
   {imsi: "...", apn: "ims"}

                              查询 IMSI 对应的会话

                                返回会话信息

                              查询 APN 的策略□□□

                                 返回策略

                 发起 Gx RAR
                 (Session-Id, 策略)
                                                 Gx RAR
                                                 (Re-Auth-Request-Type)
                                                                   ┌──────────┐
                                                                   │ 策略更新/应用 │
                                                                   └──────────┘
                                                 Gx RAA
                                                 (Result-Code: 2001)

                 RAR 确认

   HTTP 200
   {data: "Gx Re-Auth Request sent,
   Result-Code: 2001"}
```

REST API

PCRF 触发重鉴权

接口：`POST /api/operation/pcrf_re_auth`

作用：主动触发 Gx 会话的重鉴权（重授权）。

适用场景：当运营商需要实时更新用户的策略或服务质量（例如修改 APN QoS 参数、触发会话重建、应用新的计费规则 - 控制策略）时。

请求参数：

```
{
   "imsi": "999999876543210",
   "apn": "ims"
}
```

成功响应（HTTP 200）：

```
{
   "data": "Gx Re-Auth Request for 999999876543210 sent to
pgw.epc.mnc999.mcc999.3gppnetwork.org, Result-Code: 2001"
}
```

失败响应（HTTP 400）：

```
{
   "error": "Unable to send Re-Auth Request for 999999876543210 on
APN ims, no active PDN Session found"
}
```

策略管理 API

PCRF 提供一组用于管理基于 APN 的策略和 QoS 配置的接口。这些接口是 REST API 的核心部分。

策略��机制允许运营商按 APN 配置 QoS 参数（如默认承载的 QCI、比特率限制等）。当某个 APN 对应的 PDN 会话在 PGW 发起 Gx 会话请求时，RAR（重授权请求）机制会据此向数据平面推送相应的策略规则。

工作流程

策略系统的核心组件如下：

```
APN 配置文件 → APN QoS 策略 → APN 规则
      ↓                ↓                ↓
   "internet"     QCI, AMBR, ARP    过滤条件与动作
```

1. 创建 APN 配置文件

定义 APN 名称和 IP 地址池等。

□□□`POST /api/apn/identifier`

□□□□

```
{
  "apn_identifier": {
    "apn": "internet",
    "ip_version": "ipv4v6"
  }
}
```

IP □□□□□

- `"ipv4"` - □ IPv4
- `"ipv6"` - □ IPv6
- `"ipv4v6"` - □□□□IPv4 □ IPv6□
- `"ipv4_or_ipv6"` - □□□□□□IPv4 □ IPv6□

□□□□HTTP 201□□

```
{
  "data": {
    "id": 1,
    "apn": "internet",
    "ip_version": "ipv4v6"
  }
}
```

□□□

- `apn`□□□□□1-254 □□□□□□□
- `ip_version`□□□□□□□□□□□□□□□□□

□□ **APN** □□□□□`GET /api/apn/identifier`

2. □□ APN QoS □□

□□ QoS □□□□□□□QCI□□□□□□□□

请求 `POST /api/apn/qos_profile`

请求体

```
{
  "apn_qos_profile": {
    "name": "Best Effort Internet",
    "qci": 9,
    "allocation_retention_priority": 8,
    "apn_ambr_dl_kbps": 100000,
    "apn_ambr_ul_kbps": 50000,
    "pre_emption_capability": false,
    "pre_emption_vulnerability": true
  }
}
```

QoS 参数表

参数	类型	范围	说明
name	string	1-254 字符	□□□□□□□□
qci	integer	1-254	QoS 等级标识 （1-4 = GBR，5-9 = 非 GBR）
allocation_retention_priority	integer	1-15	ARP 优先级，1 = □□□□□□
apn_ambr_dl_kbps	integer	1- 4,294,967,293	APN 聚合最大 下行速率 （kbps）
apn_ambr_ul_kbps	integer	1- 4,294,967,293	APN 聚合最大 上行速率 （kbps）
pre_emption_capability	boolean	true/false	□□□□□□□□□□ □
pre_emption_vulnerability	boolean	true/false	□□□□□□□□□□ □□

常见 QCI 值：

- 1 - 会话语音（VoLTE） - GBR，100ms 最大延迟
- 2 - 会话视频 - GBR，150ms 最大延迟
- 5 - IMS 信令 - 非 GBR，100ms 最大延迟
- 9 - 默认承载（互联网） - 非 GBR，300ms 最大延迟

响应（HTTP 201）：

```
{
  "data": {
    "id": 1,
    "name": "Best Effort Internet",
    "qci": 9,
    "allocation_retention_priority": 8,
    "apn_ambr_dl_kbps": 100000,
    "apn_ambr_ul_kbps": 50000,
    "pre_emption_capability": false,
    "pre_emption_vulnerability": true
  }
}
```

□□ **QoS** □□□□□`GET /api/apn/qos_profile`

3. □□ **APN** □□

□ APN □□□□ QoS □□□□□□

□□□`POST /api/apn/profile`

□□□□

```
{
  "apn_profile": {
    "name": "Internet APN",
    "apn_identifier_id": 1,
    "apn_qos_profile_id": 1
  }
}
```

□□□

- `name`□□□□□□□□□□□□□□□□
- `apn_identifier_id`□□□□ □□ APN □□□ □ ID
- `apn_qos_profile_id`□□□□ □□ APN QoS □□ □ ID

□□□□HTTP 201□□□

```
{
  "data": {
    "id": 1,
    "name": "Internet APN",
    "apn_identifier_id": 1,
    "apn_qos_profile_id": 1
  }
}
```

说明：

- `apn_identifier_id` 与 `apn_qos_profile_id` 为关联外键引用
- 一个 APN 可绑定一个 QoS 策略和一个网络标识符

查询 **APN** 列表：`GET /api/apn/profile`

请求示例（典型）：

创建 **1：语音 IMS APN 配置（VoLTE）**

```
# 1. □□ APN □□□
curl -X POST https://hss.example.com:8443/api/apn/identifier \
  -H "Content-Type: application/json" \
  -d '{
    "apn_identifier": {
      "apn": "ims",
      "ip_version": "ipv4v6"
    }
  }'
# □□□{"data": {"id": 2, ...}}

# 2. □□ QoS □□□IMS □□□□□
curl -X POST https://hss.example.com:8443/api/apn/qos_profile \
  -H "Content-Type: application/json" \
  -d '{
    "apn_qos_profile": {
      "name": "IMS Signaling QoS",
      "qci": 5,
      "allocation_retention_priority": 2,
      "apn_ambr_dl_kbps": 5000,
      "apn_ambr_ul_kbps": 5000,
      "pre_emption_capability": true,
      "pre_emption_vulnerability": false
    }
  }'
# □□□{"data": {"id": 2, ...}}

# 3. □□ APN □□
curl -X POST https://hss.example.com:8443/api/apn/profile \
  -H "Content-Type: application/json" \
  -d '{
    "apn_profile": {
      "name": "IMS APN",
      "apn_identifier_id": 2,
      "apn_qos_profile_id": 2
    }
  }'
# □□□{"data": {"id": 2, ...}}
```

□□ 2□□□□□□□□

□□□□APN □□□□ EPC □□□□□□□□□□□□□ API □□ □□□ APN □□□□□□□□

更新操作示例

更新 QoS 配置

```
PATCH /api/apn/qos_profile/{id}
PUT /api/apn/qos_profile/{id}
```

示例 - 动态调整带宽限制

```
# 更新 QoS 配置 ID 1 的带宽限制
curl -X PATCH https://hss.example.com:8443/api/apn/qos_profile/1 \
  -H "Content-Type: application/json" \
  -d '{
    "apn_qos_profile": {
      "apn_ambr_dl_kbps": 150000,
      "apn_ambr_ul_kbps": 75000
    }
  }'
```

运营影响和注意事项

1. QoS 更新会触发策略重评估
2. 现有会话可能需要根据 QoS 变化重建 APN 级别的 PDN 连接
3. 某些实现会向受影响会话的 PGW 发送 Gx RAR
4. PGW 随后应用 QoS 更新而无需会话中断
5. 某些实现可能需要 API 触发的信令来强制执行更新后的带宽限制

本示例演示调整 100 个使用此配置文件（比如 QoS 配置 ID 1 的 "internet" APN）绑定的 100 个活动订阅者的带宽。API 调用后，所有受影响会话将获得 150 Mbps 下行 / 75 Mbps 上行。

运营商通常利用 APN QoS 更新来实施套餐升级、促销带宽提升或应对网络拥塞时的动态策略调整，而无需重新配置每个订阅者。

删除操作

```
DELETE /api/apn/identifier/{id}
DELETE /api/apn/qos_profile/{id}
DELETE /api/apn/profile/{id}
```

□□□□□

- □□□□□ APN □□□□□ APN □□□□ QoS □□
- □□□□□□□□□□□□□□ APN □□

□□□□

□□□□□□□□ 100 Mbps / □□ 50 Mbps□□

```json
{
  "apn_qos_profile": {
    "name": "High Speed Internet",
    "qci": 9,
    "allocation_retention_priority": 8,
    "apn_ambr_dl_kbps": 100000,
    "apn_ambr_ul_kbps": 50000,
    "pre_emption_capability": false,
    "pre_emption_vulnerability": true
  }
}
```

□□□□□□□□ 500 Mbps / □□ 100 Mbps□□

```json
{
  "apn_qos_profile": {
    "name": "Premium Internet",
    "qci": 8,
    "allocation_retention_priority": 5,
    "apn_ambr_dl_kbps": 500000,
    "apn_ambr_ul_kbps": 100000,
    "pre_emption_capability": true,
    "pre_emption_vulnerability": false
  }
}
```

□□□/M2M□□□□□□□

```
{
  "apn_qos_profile": {
    "name": "IoT M2M",
    "qci": 9,
    "allocation_retention_priority": 10,
    "apn_ambr_dl_kbps": 1024,
    "apn_ambr_ul_kbps": 512,
    "pre_emption_capability": false,
    "pre_emption_vulnerability": true
  }
}
```

示例：为紧急服务创建：

```
{
  "apn_qos_profile": {
    "name": "Emergency APN",
    "qci": 5,
    "allocation_retention_priority": 1,
    "apn_ambr_dl_kbps": 10000,
    "apn_ambr_ul_kbps": 10000,
    "pre_emption_capability": true,
    "pre_emption_vulnerability": false
  }
}
```

配置

Diameter 接口配置

Gx 接口在`config/runtime.exs`中：

```
%{
  application_name: :gx,
  application_dictionary: :diameter_gen_3gpp_gx,
  vendor_specific_application_ids: [
    %{vendor_id: 10415, auth_application_id: 16_777_238}
  ]
}
```

Rx 应用（`config/runtime.exs`）：

```
%{
  application_name: :rx,
  application_dictionary: :diameter_gen_3gpp_rx,
  vendor_specific_application_ids: [
    %{vendor_id: 10415, auth_application_id: 16_777_236}
  ]
}
```

QoS 管理

QoS 参数的来源：

- 默认承载：来自用户的 APN 配置

  - `apn_qos_profile.qci`：QoS 类别标识符
  - `apn_qos_profile.apn_ambr_ul_kbps`：上行聚合最大比特率
  - `apn_qos_profile.apn_ambr_dl_kbps`：下行聚合最大比特率
  - `apn_qos_profile.priority_level`：分配和保留优先级

- 专用承载：来自 Rx AAR 消息，依据以下内容：

  - QCI（1，语音承载）
  - 带宽需求，来自 Max-Requested-Bandwidth AVP
  - 流描述，来自 Flow-Description AVP
```

# □□□□

| □□□□ | □□ | □□ | □□ |
|---|---|---|---|
| 2001 | □□ | DIAMETER_SUCCESS | □□□□□□ |
| 5001 | □□□ | □□□□□ | IMSI □□□□□□□□□□ |
| 5002 | □□□ | □□□□□ | PDN □□□□□□□□□□/□□ |
| 5063 | □□□ | □□□□□ | IMS □□□□□□□ |

# □□□□

## □□□□

PCRF □□□

- □□ **PDN** □□ - □□ APN□□□□□□□□□
- **VoLTE** □□ - □□ IMS □□□□□□□□□□□□□□
- **QoS** □□ - □□ APN □□□□□□□
- □□□□ - □□□□□□□□□□□□

## □□□□□□

PCRF □□□□□□□□□□□□□

- □□ Gx □□□□□□□□□□/□□
- □□□□□□□**TFT**□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ TFT □□□□□□□

# □□□□

- Diameter □□ - □□□□□□□□
- API □□ - □□□□ API □□
- □□ - □□ HSS □□
- □□□□ - □□□□□ Diameter AVP □□□

# OmniHSS □□□□□□□

← □□□□□□□

## □□

OmniHSS □□ □□□□ □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□
□□□□□□□□□



## □□□□□□□



## EPC □□□□

EPC□□□□□□□□□□□□□□ LTE □□□□□□□□□□□□□□

## □□□□

| □□ | □□ | □□□ |
|---|---|---|
| `ue_ambr_dl_kbps` | □□□ □□□ | 10,000 - 1,000,000 Kbps |
| `ue_ambr_ul_kbps` | □□□ □□□ | 5,000 - 500,000 Kbps |
| `network_access_mode` | □□□ □ | "packet_only" □ "packet_and_circuit" |
| `tracking_area_update_interval_seconds` | TAU □□□ | 54 □□□□□ |

## □□ EPC □□□□□

```
curl -k -X POST https://hss.example.com:8443/api/epc/profile \
  -H "Content-Type: application/json" \
  -d '{
    "apn_profiles": [],
    "name": "Premium 100Mbps",
    "network_access_mode": "packet_only",
    "tracking_area_update_interval_seconds": 600,
    "ue_ambr_dl_kbps": 100000,
    "ue_ambr_ul_kbps": 50000
  }'
```

□□ EPC □□□□□□□

□□□□□:

- □□: 10 Mbps (10,000 Kbps)
- □□: 5 Mbps (5,000 Kbps)

□□:

- □□: 50 Mbps (50,000 Kbps)
- □□: 25 Mbps (25,000 Kbps)

□□:

- □□: 100 Mbps (100,000 Kbps)
- □□: 50 Mbps (50,000 Kbps)

□□□:

- □□: 1 Gbps (1,000,000 Kbps)
- □□: 500 Mbps (500,000 Kbps)

IMS □□□□

IMS □□□□□□□□□□□□□□□□□□□□□ IFC□□□□□□□□□□□□□□□□□

IFC □□

IFC □□□□□□ S-CSCF □□□□□□□□□ XML □□□□

□□□□:

- `{{imsi}}` - □□□ IMSI
- `{{msisdns}}` - □□□□□□□
- `{{mcc}}` - □□□□
- `{{mnc}}` - □□□□□

创建 IMS 用户配置

```
curl -k -X POST https://hss.example.com:8443/api/ims/profile \
  -H "Content-Type: application/json" \
  -d '{
    "ims_profile": {
      "name": "Standard VoLTE",
      "ifc_template": "<InitialFilterCriteria>...
</InitialFilterCriteria>"
    }
  }'
```

IFC 模板示例

```xml
<ServiceProfile>
  <PublicIdentity>
    <Identity>sip:
{{imsi}}@ims.mnc{{mnc}}.mcc{{mcc}}.3gppnetwork.org</Identity>
  </PublicIdentity>
  <InitialFilterCriteria>
    <Priority>0</Priority>
    <TriggerPoint>
      <ConditionTypeCNF>0</ConditionTypeCNF>
      <SPT>
        <ConditionNegated>0</ConditionNegated>
        <Group>0</Group>
        <Method>INVITE</Method>
      </SPT>
    </TriggerPoint>
    <ApplicationServer>
      <ServerName>sip:as.ims.example.com</ServerName>
      <DefaultHandling>0</DefaultHandling>
    </ApplicationServer>
  </InitialFilterCriteria>
</ServiceProfile>
```

APN □□□□

APN□□□□□□□□□□□□□□□□□□□□□□□□□□□□

APN □□

```
          ┌──────────┐
          │ □□□□□□□ │
          └──────────┘
      ╱       │        │        ╲
     ╱        │        │         ╲
┌────────┐ ┌────────┐ ┌────────┐ ┌──────────┐
│EPC □□□□│ │IMS □□□□│ │APN □□□□│ │□□□□□□□    │
│□□□□    │ │□□□□    │ │□□□□    │ │□□□□       │
└────────┘ └────────┘ └────────┘ └──────────┘
```

APN □□□

□□ APN □□□ IP □□□□□□

□□ APN:

- `internet` - □□□□□□□□□
- `ims` - IMS/VoLTE □□
- `mms` - □□□□□□
- `vzwadmin` - □□□□□□

IP □□□□:

- `"ipv4"`: □ IPv4
- `"ipv6"`: □ IPv6
- `"ipv4v6"`: IPv4v6□□□□□
- `"ipv4_or_ipv6"`: IPv4 □ IPv6□□□□□□□

APN QoS □□□□

□□□□□□□□□□□

QCI与QoS 映射关系表:

QCI	类型	业务	优先级
1	GBR	语音通话	最高
2	GBR	视频通话	高
4	GBR	视频流	中
5	非 GBR	IMS 信令	高
9	非 GBR	默认承载数据	最低

创建基础 APN 配置

```
# 1. 创建 APN 标识符
APN_ID=$(curl -k -X POST
https://hss.example.com:8443/api/apn/identifier \
  -H "Content-Type: application/json" \
  -d '{"apn": "internet", "ip_version": "ipv4v6"}' \
  | jq -r '.response.id')

# 2. 创建 APN QoS 配置文件
QOS_ID=$(curl -k -X POST
https://hss.example.com:8443/api/apn/qos_profile \
  -H "Content-Type: application/json" \
  -d '{
    "name": "Best Effort",
    "allocation_retention_priority": 8,
    "apn_ambr_dl_kbps": 50000,
    "apn_ambr_ul_kbps": 25000,
    "pre_emption_capability": false,
    "pre_emption_vulnerability": true,
    "qci": 9
  }' | jq -r '.response.id')

# 3. 创建 APN 配置文件
curl -k -X POST https://hss.example.com:8443/api/apn/profile \
  -H "Content-Type: application/json" \
  -d "{
    \"apn_identifier_id\": $APN_ID,
    \"apn_qos_profile_id\": $QOS_ID,
    \"name\": \"Internet APN\"
  }"
```

将 APN 关联到 EPC 配置文件

APN 使用 `join_epc_profile_to_apn_profile` 与 EPC 配置文件关联。
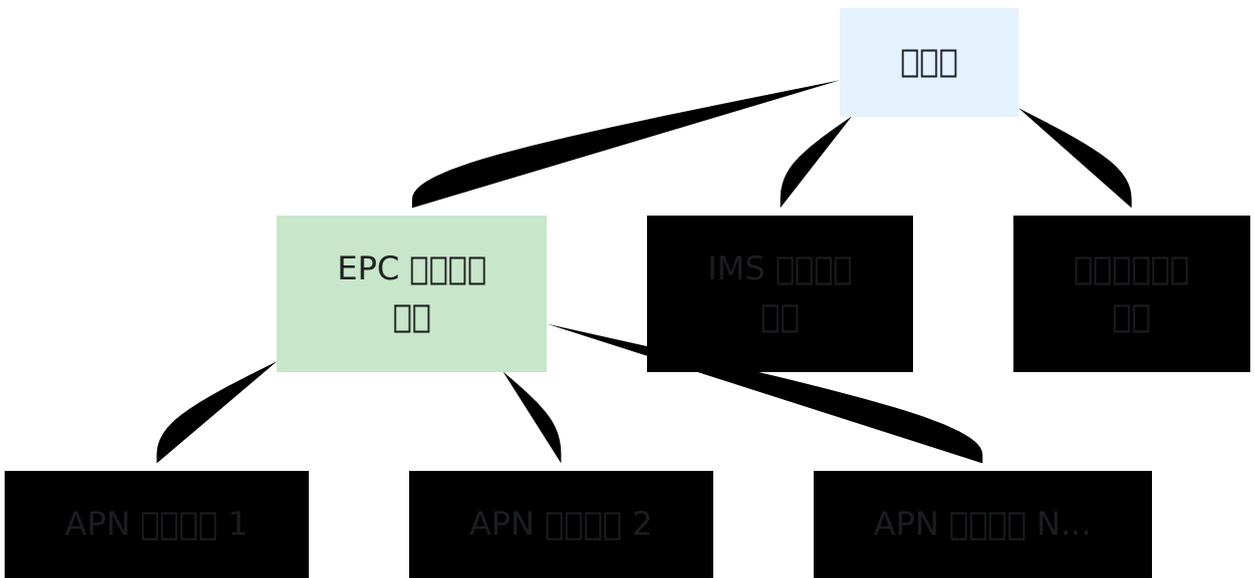
该请求提供所要关联的 APN 配置文件 ID 以及 EPC 配置文件 ID，将该 APN 配置文件与一个或多个 EPC 配置关联。

承载管理模块

这里是 承载管理模块 的简单功能介绍

示例网络拓扑

电信级承载的典型应用

□□□□□□□□□□□□□

```
# □□□□□□□□□□ EPC □ IMS □□□□
curl -k -X POST https://hss.example.com:8443/api/subscriber \
  -H "Content-Type: application/json" \
  -d '{
    "subscriber": {
      "imsi": "001001123456789",
      "key_set_id": 1,
      "epc_profile_id": 1,
      "ims_profile_id": 1,
      "roaming_profile_id": 1
    }
  }'

# □□□□□□□□□□
curl -k -X PUT https://hss.example.com:8443/api/subscriber/1 \
  -H "Content-Type: application/json" \
  -d '{
    "subscriber": {
      "epc_profile_id": 2
    }
  }'
```

□□□□□□□□□□□□□

□□□□

1. □□□□□□□□ - □□□□□□□□□□□�□□□□□□□
2. □□□□□ - □□□□□□□□□□□□□□
3. □□□□ - □□□□□□□
4. □□□□ - □□□□□□□□□□□□□□□□□

□□□□□□□□□

```
[□□□□]-[□□]-[□□]

□□:
- "Basic-10Mbps-Internet"
- "Premium-100Mbps-VoLTE"
- "Enterprise-1Gbps-MultiAPN"
```

□□□□□□□

□□□□□□□□□□□□□


```
┌──────────┐    ┌──────────┐         ┌──────────┐
│    □□    │    │OmniHSS API│         │   □□□    │
└────┬─────┘    └────┬─────┘         └────┬─────┘
     │               │                    │
     │        □□□□□□□□□□□□□               │
     │               │                    │
     │              □□                    │
     │               │              ┌─────┴──────┐
     │               │              │ □□□□□□/TAU □│
     │               │              └─────┬──────┘
     │               │                    │
     │               │          □□□□□□□□□  │
     │               │                    │
     │               │    □□□□□□□□□AMBR□APNs□
     │               │              ┌─────┴──────┐
     │               │              │ □□□□□□□□□  │
     │               │              └─────┬──────┘
┌────┴─────┐    ┌────┴─────┐         ┌────┴─────┐
│    □□    │    │OmniHSS API│         │   □□□    │
└──────────┘    └──────────┘         └──────────┘
```

□□: □□□□□□□□□□□□□□□□

- □□□□□□□□□TAU□

- □□

- IMS □□□□□□ IMS □□□□□□□□□□

□□□□□□□□□□□□

□□□□□□□□□□□□:
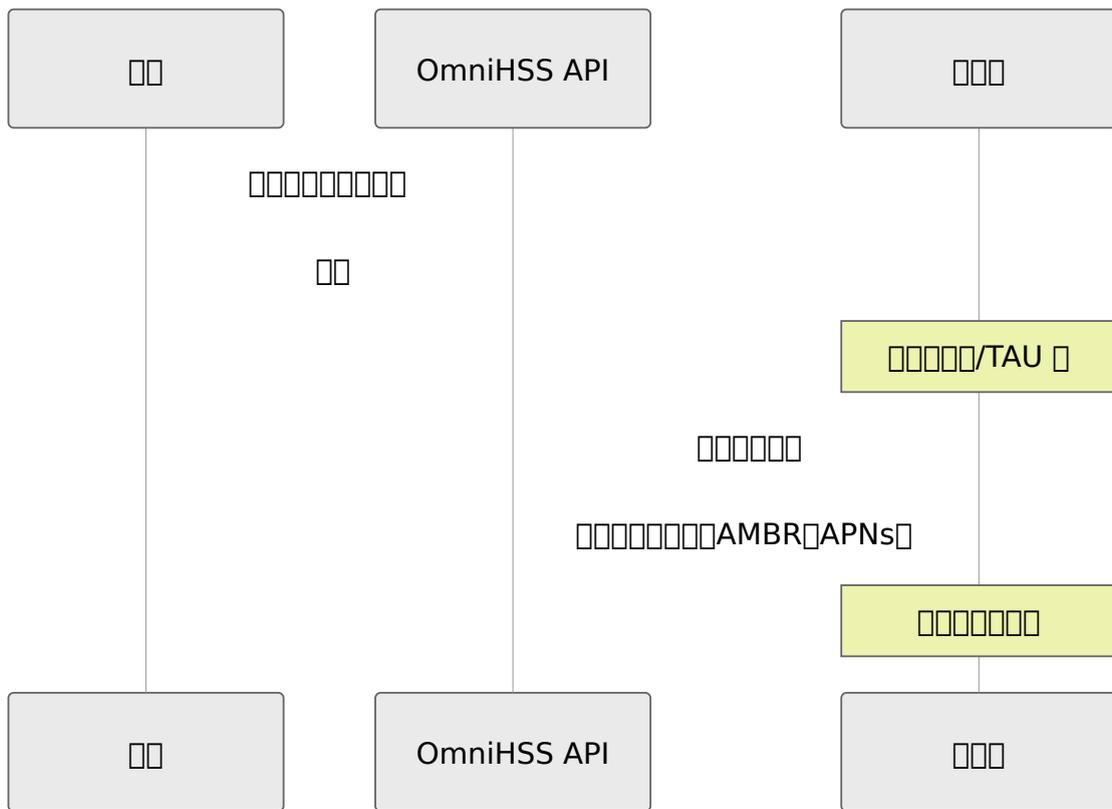
1. □□□□□ EPC □□□□ AMBR □
2. □□ APN QoS □□□□ AMBR □
3. □□ MME/P-GW □□□□□□ QoS
4. □□□□□□

IMS □□□□:

1. □□□□□□□ IMS □□□□
2. □□ IFC □□ XML □□□□
3. □□ S-CSCF □□□��□ IFC □□□□
4. □□ S-CSCF □□□□

APN □□□:

1. □□ APN □□□□□□□□□ EPC □□□□
2. □□ APN □□□□□□□□□□□
3. □□ UE □ PDN □□□□

OmniHSS 接口文档

概述

本文档详细描述了 OmniHSS 支持的 Diameter 接口、关键信令流程以及相关的参数定义和错误代码。

S6a 接口 (LTE/EPC)

鉴权信息请求 (AIR/AIA)

MME 请求用户鉴权向量。

MME — OmniHSS (HSS) — Database

S6a AIR
(IMSI, □□ PLMN, □□□□□)

□□□□ + key_set + roaming_profile

□□□□

□□□□□□□□

alt [□□□□□]

S6a AIA
□□: □□□□□

[□□□□□]

□□□□□□ (Milenage)

S6a AIA
□□: □□
(□□□□)
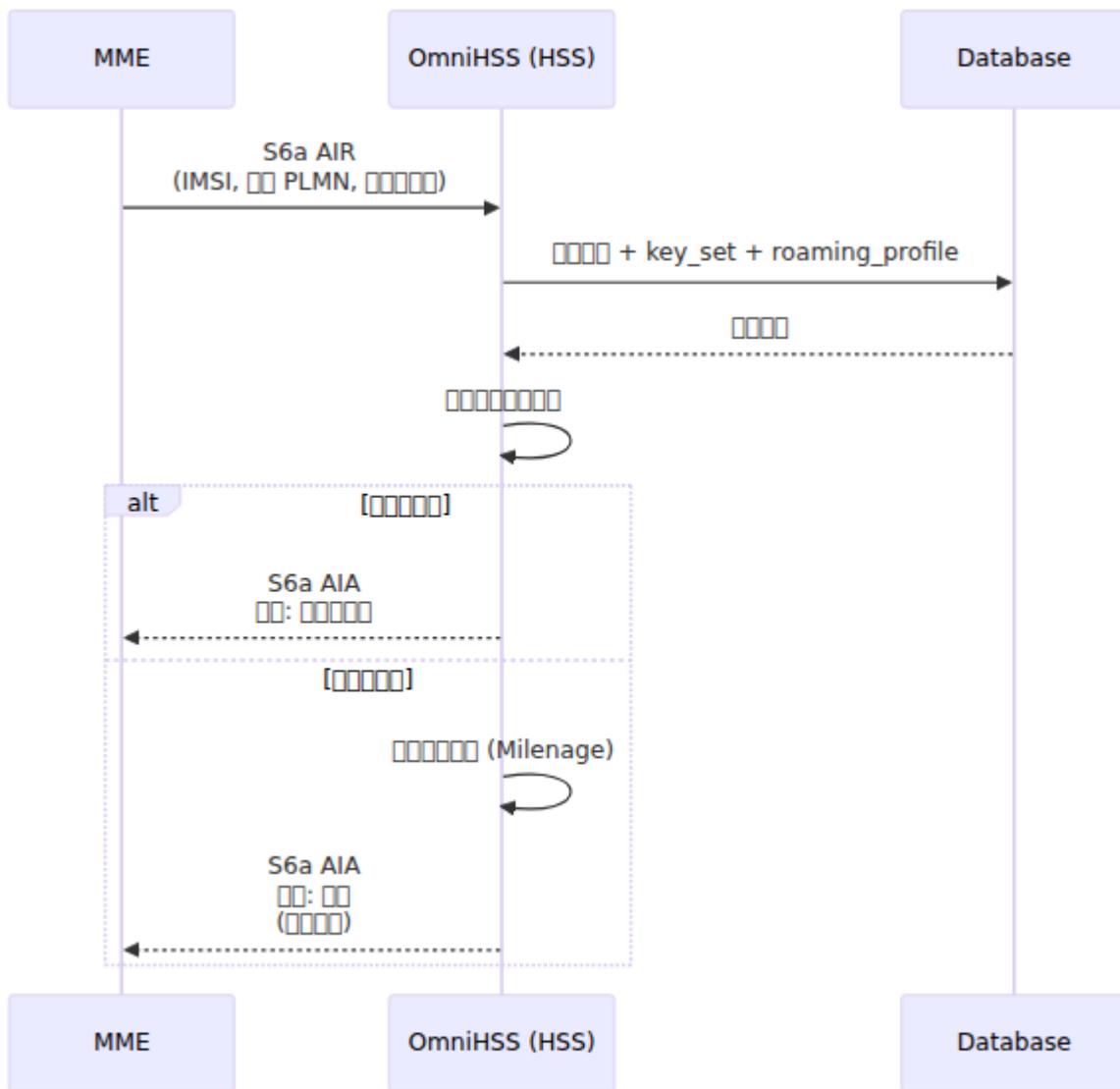
□□ AVP:

- □□: □□□ (IMSI), □□-PLMN-Id, □□□□□□□
- □□: □□□□ (RAND, AUTN, XRES, KASME)

□□□□□□ (ULR/ULA)

MME □□ HSS □□□□□□□□□□□□□□�□□
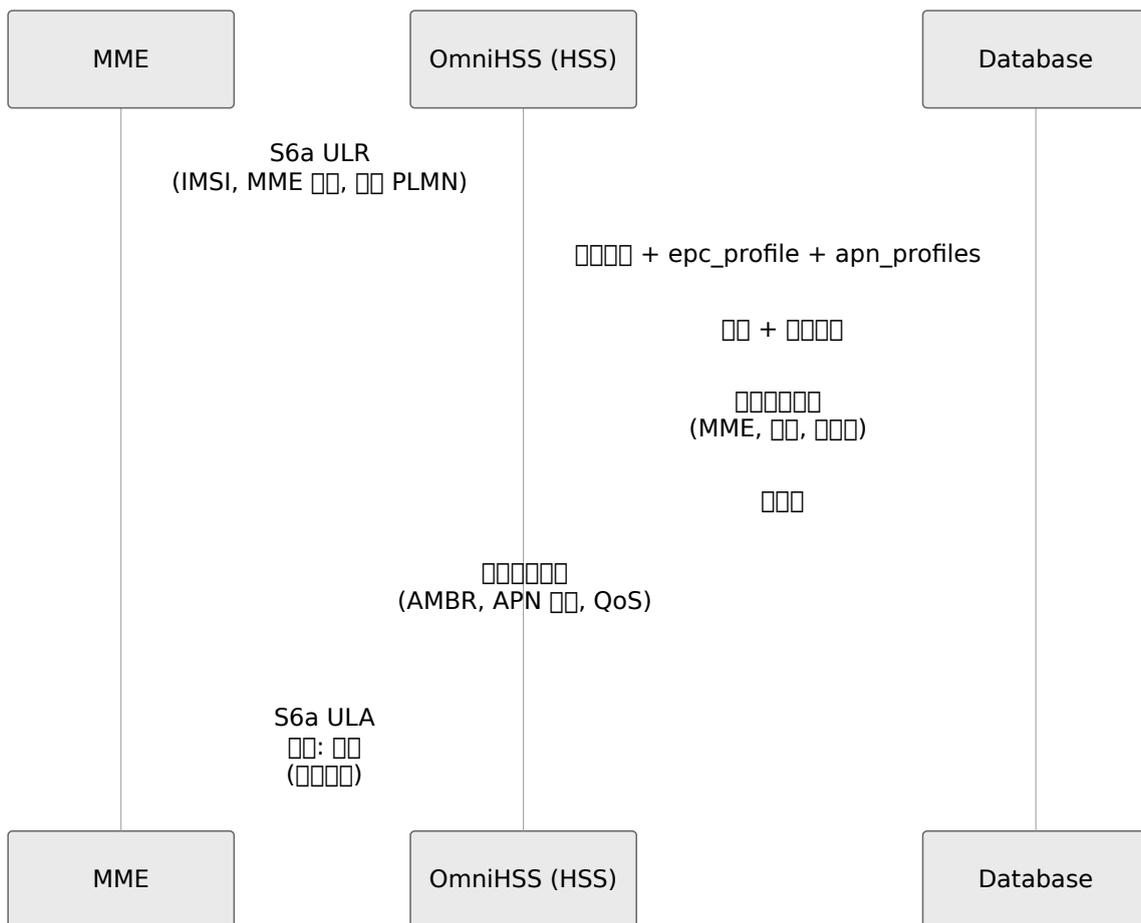
返回 AVP:

- 请求: 用户名 (IMSI), RAT 类型, ULR 标志, 访问-PLMN-Id, UE-SRVCC 能力
- 响应: 订阅数据 (AMBR, APN 配置, 特征位掩码)

清除 UE 流程 (PUR/PUA)

MME 通知 HSS 删除用户订阅上下文：

MME | OmniHSS (HSS) | Database

S6a ULR
(IMSI, MME □□, □□ PLMN)

□□□□ + epc_profile + apn_profiles

□□ + □□□□

□□□□□□
(MME, □□, □□□)

□□□

□□□□□□
(AMBR, APN □□, QoS)

S6a ULA
□□: □□
(□□□□)

MME | OmniHSS (HSS) | Database

□□□□ (NOR/NOA)

MME □□ HSS □□□□□

MME   OmniHSS (HSS)

S6a NOR
(IMSI, □□□□)

□□□□

S6a NOA
□□: □□
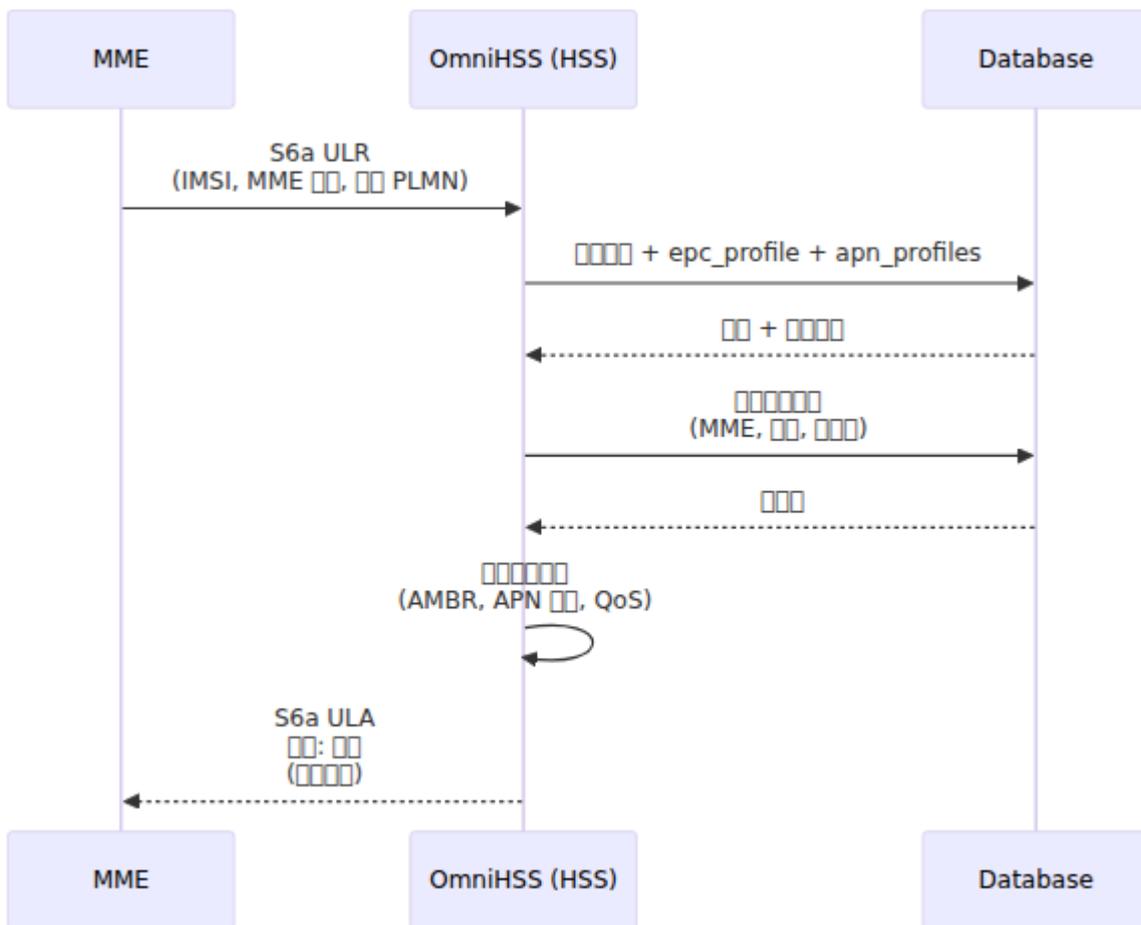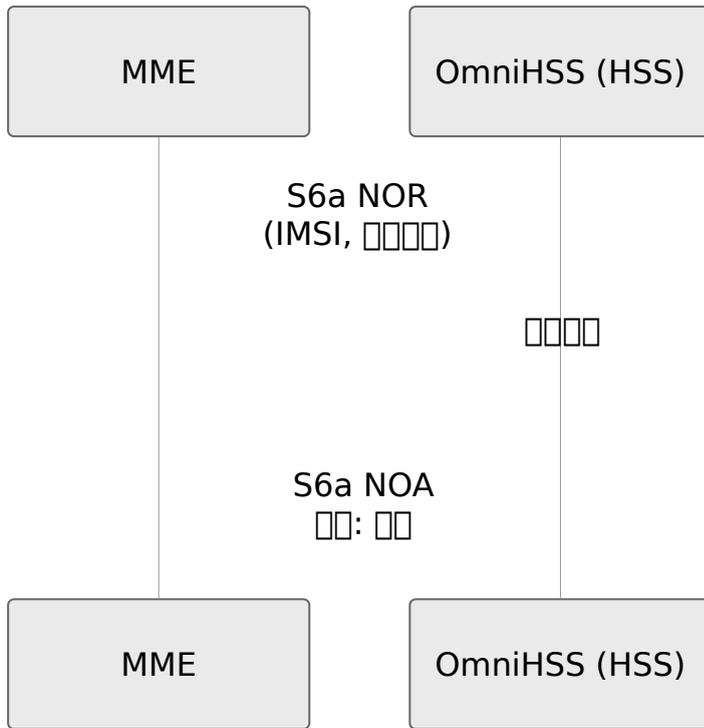
MME   OmniHSS (HSS)

□□□□□□ (CLR/CLA)

HSS □□□□□□□□□ MME □□□□□□□□OmniHSS □□□□□□□□ CLR □□□□

□□ CLR (MME □□)

□□□□□ MME □□□□□□□□□□OmniHSS □□□□□ MME □□ CLR □□□□□□□□

| Previous MME | New MME | OmniHSS (HSS) | | Database |

| | 注册到新的 MME | |

S6a ULR
(IMSI, 新 MME 身份)

查询用户数据

last_seen_mme: "old-mme.example.com"

[旧的 MME 身份与新的不同]

触发 CLR 流程

S6a CLR
(IMSI, 取消类型: 0)
(CLR 标志: 0)

删除用户上下文

S6a CLA
结果: 成功

更新用户数据
(last_seen_mme: new MME)

已更新

S6a ULA
结果: 成功
(用户数据)

| Previous MME | New MME | OmniHSS (HSS) | | Database |

关键 AVP (位置 CLR):

- 用户名: 用户的 IMSI
- 目标主机: 旧的 MME 主机名
- 源主机: 新的 MME 名
- 取消类型: 0 (MME 更新流程)
- CLR 标志: 0
- 会话标识: 唯一的会话标识符

手动的 CLR (API 流程)

运营商可�� 过管理 API 触发 CLR，无论何时需要强制将用户从特定的接入节点分离。

关键 AVP (派生的 CLR):

- 用户名: 触发的 IMSI
- 源主机名: 当前服务的 MME 主机名
- 源域名: 当前服务的 MME 域
- 取消类型: `:subscription_withdrawal` (按 3GPP TS 29.272 标准定义)
- CLR 标志:
  - s6a_indicator: 1 (表明这是 S6a 接口)
  - reattach_required: 1 (UE 需要重新附着到网络)

合规性

OmniHSS 的实现严格遵循了 3GPP TS 29.272。

类型	值	描述	备注
MME 位置更新	0	来自 MME 节点	触发 MME 发起的 ULR
SGSN 位置更新	1	SGSN 更新	3G/2G 网络更新
管理更新	2	手动更新	通过 API 手动更新
自动更新 IWF	3	系统自动更新	内部触发的更新
应用程序更新	4	应用层	应用发起更新

CLR 标志

CLR-Flags AVP 用于在取消位置请求中设置标志。

类型	值	描述
S6a/S6d 指示器	0	1 = 使用 S6a 接口
重新附着要求	1	1 = UE 需要重新附着网络

设置 CLR-Flags 示例:

```
clr_flags: %{
  s6a_indicator: 1,        # 使用 S6a 接口
  reattach_required: 1     # 需要重新附着
}
```

按 IMSI 查询

OmniHSS 使用国际移动用户识别码 (IMSI) 向 MME 发送取消位置请求。MSISDN，系统可以根据 IMSI 查询并触发 CLR 或其他相关操作。

方法 1: 通过 MSISDN 查询到 IMSI

```
用户 A:
  - IMSI: 999000123456789
  - MSISDNs: ["+1234567890", "+9876543210"]
  - last_seen_mme: "mme01.operator.com"
```

当订阅者移动到新的 MME 时：

- 系统发送 **CLR** 到 "mme01.operator.com"，IMSI 为 999000123456789
- 这两 MSISDN 会一起转移，因为它们属于同一 SIM。
- 会话内 AVP 包含 IMSI，而不是 MSISDNs

阶段 2: 不同订阅者的 **IMSIs**，相同的 **MSISDN**

OmniHSS 不会要求 每个 **MSISDN** 唯一。如果 MSISDN 在企业共享线路场景中被重复使用（计费/显示原因）

```
用户 A:
  - IMSI: 999000111111111
  - MSISDN: "+1234567890"
  - last_seen_mme: "mme01.operator.com"

用户 B（不同人）:
  - IMSI: 999000222222222
  - MSISDN: "+1234567890"  # 相同 MSISDN，不同 SIM/IMSI
  - last_seen_mme: "mme02.operator.com"
```

当用户 B 移动时

- 发出的 **CLR**，其中 IMSI = 该订阅者
- 用户 A 仍注册 mme01 上的
- 用户 B 在 mme02 上的
- 两个条目是独立的，因为它们有不同的主键

阶段 3: 为什么 **CLR** 使用 **MSISDN** 标识

总结：

- 系统发送 **CLR** 时并不 last_seen_mme
- 因为 **MSISDNs** 不是 IMSI 唯一标识订阅者的主键

- IMSI 正確的 MME 對應關係

更新流程

1. **IMSI 查找:** CLR 包含用戶的 舊 IMSI。系統依據 MSISDN、`subscriber_state` 或舊身分 (IMSI) 查找 `last_seen_mme`。

2. **會話識別:** 系統嘗試找出先前的 MME 對應此次 CLR 會話。此關聯用於確保正�連續性。

3. **僅針對活躍的 MME 發送的 CLR:** 如果 `last_seen_mme` 為 `nil`，則系統不會處理此 ULR 所產生的 CLR。

4. **更新訂閱資料:** 處理 CLR 時，ULR 必須包含有效的 `Subscription-Data` AVP，以更新 MME 中的訂閱資料。

5. **應答:** CLR 在成功處理後會產生回覆給 MME 的 ULA，接著傳送給 MME 的 CLA。

6. **CLA 處理:** OmniHSS 收到 CLA 後若偵測到任何錯誤將回傳 398 並帶 `:discard`，以確保不會將錯誤資料寫入 HSS 資料庫

Cx 介面 (IMS)

用戶授權請求 (UAR/UAA)

I-CSCF 向用戶發起註冊時會先查詢

```
┌─────────┐      ┌─────────┐      ┌──────────────┐
│  Phone  │      │ I-CSCF  │      │ OmniHSS (HSS) │
└─────────┘      └─────────┘      └──────────────┘
     │                │                  │
     │  SIP REGISTER   │                  │
     │                │                  │
     │                │   Cx UAR         │
     │                │  (□□□□, □□□□)     │
     │                │                  │
     │                │                  │  □□□□
     │                │                  │
     │   ████         │   [□□□□□□□]       │
     │                │                  │
     │                │   Cx UAA         │
     │                │   □□: □□          │
     │                │  (□□□□□□)         │
     │                │                  │
     │          □□ S-CSCF                 │
     │                │                  │
     │                │   [□□□□□□□□]       │
     │                │                  │
     │                │   Cx UAA         │
     │                │   □□: □□□□        │
     │                │                  │
     │   403 Forbidden │                  │
     │                │                  │
┌─────────┐      ┌─────────┐      ┌──────────────┐
│  Phone  │      │ I-CSCF  │      │ OmniHSS (HSS) │
└─────────┘      └─────────┘      └──────────────┘
```
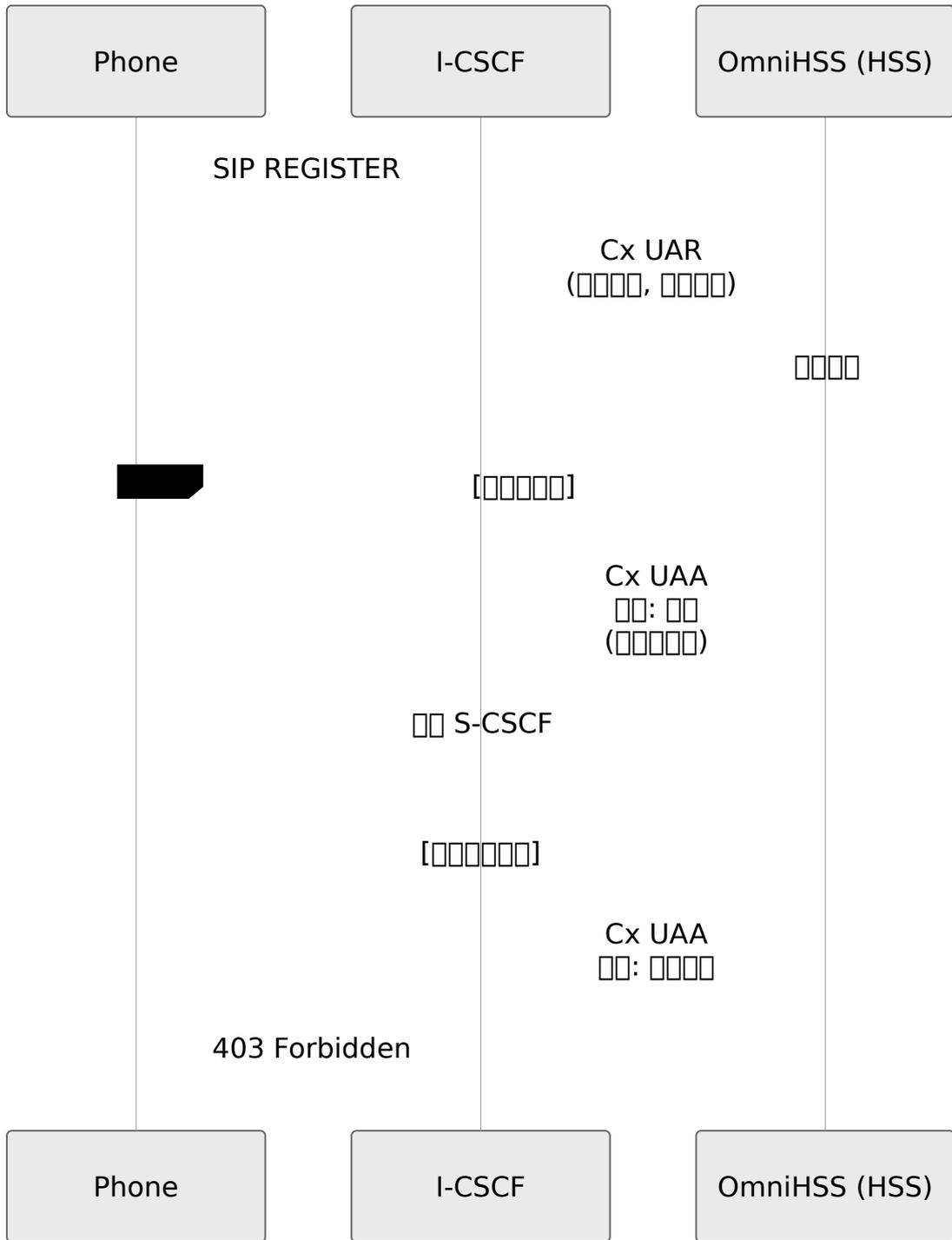
□□□□□□□□□ (SAR/SAA)
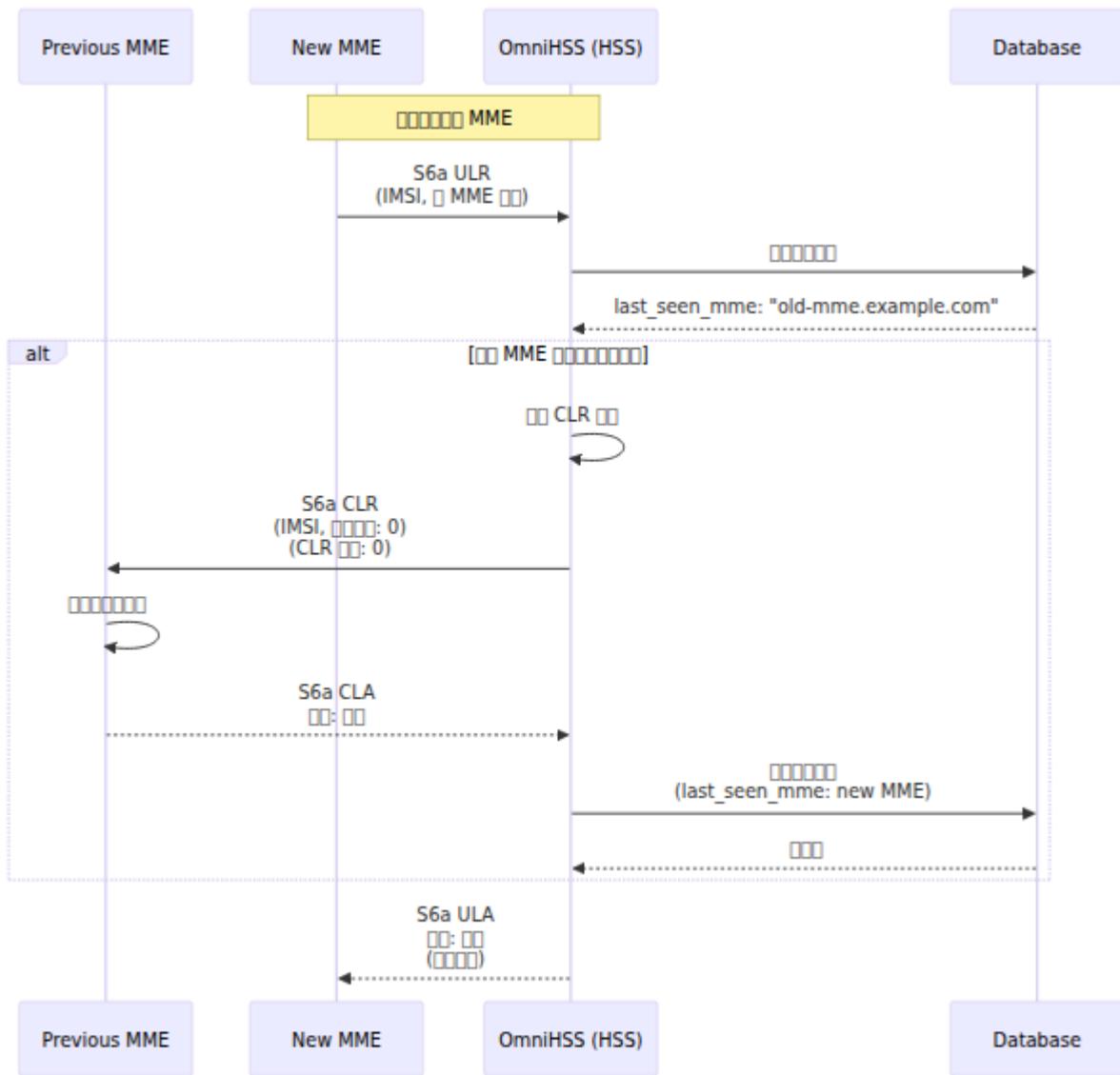
S-CSCF □□/□□□□□□□□□ IMS □□□□□□

IFC □□□□:

- `{{imsi}}` → □□ IMSI
- `{{msisdns}}` → □□□□□□□
- `{{mcc}}`, `{{mnc}}` → □□ PLMN □□

□□□□□□□ (MAR/MAA)

S-CSCF □□ IMS □□□□□□□□□

S-CSCF	OmniHSS (HSS)	Database

Cx MAR
(IMSI, □□□□)

□□□□ + key_set

□□ + □□

□□ IMS □□□□
(CK, IK □□ Milenage)

Cx MAA
□□: □□
(SIP-Authenticate, CK, IK)

S-CSCF	OmniHSS (HSS)	Database

□□□□□□ (LIR/LIA)

I-CSCF □□□□ S-CSCF □□□□□□□□

```
   ┌─────────┐      ┌──────────────┐      ┌──────────┐
   │ I-CSCF  │      │ OmniHSS (HSS)│      │ Database │
   └─────────┘      └──────────────┘      └──────────┘
        │                  │                    │
        │   Cx LIR         │                    │
        │   (□□□□)         │                    │
        │                  │                    │
        │                  │   □□□□□□□          │
        │                  │  (□□□□□□ S-CSCF)   │
        │                  │                    │
        │                  │   S-CSCF □□        │
        │                  │                    │
     ███████   [□□□□□□]    │                    │
        │                  │                    │
        │   Cx LIA         │                    │
        │   □□: □□         │                    │
        │  (S-CSCF □□)      │                    │
        │                  │                    │
   □□□□□□□ S-CSCF           │                    │
        │                  │                    │
        │   [□□□□□□]        │                    │
        │                  │                    │
        │   Cx LIA         │                    │
        │   □□: □□□□□       │                    │
        │                  │                    │
   ┌─────────┐      ┌──────────────┐      ┌──────────┐
   │ I-CSCF  │      │ OmniHSS (HSS)│      │ Database │
   └─────────┘      └──────────────┘      └──────────┘
```

Sh □□ (IMS □□□□□□□)

□□□□□□ (UDR/UDA)

□□□□□□□□□□□□□□□□□□□□□

```
     Phone              I-CSCF            OmniHSS (HSS)

        │   SIP REGISTER    │                    │
        │──────────────────▶│                    │
        │                   │    Cx UAR          │
        │                   │ (用户身份, 网络名称)   │
        │                   │───────────────────▶│
        │                   │                    │
        │                   │                 查找用户 ↻
        │                   │                    │
   ┌ alt ─────────────────────[用户已注册]──────────────────────┐
   ┊    │                   │                    │             ┊
   ┊    │                   │    Cx UAA          │             ┊
   ┊    │                   │   结果: 成功         │             ┊
   ┊    │                   │  (已分配服务器)       │             ┊
   ┊    │                   │◀╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌│             ┊
   ┊    │                   │                    │             ┊
   ┊    │              选择 S-CSCF ↻              │             ┊
   ┊    │                   │                    │             ┊
   ├╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌[用户未注册]╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌┤
   ┊    │                   │                    │             ┊
   ┊    │                   │    Cx UAA          │             ┊
   ┊    │                   │   结果: 用户未知       │             ┊
   ┊    │                   │◀╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌│             ┊
   ┊    │   403 Forbidden   │                    │             ┊
   ┊    │◀╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌│                    │             ┊
   └╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌╌┘

     Phone              I-CSCF            OmniHSS (HSS)
```

用户配置文件更新 (PUR/PUA)

本节将在后续版本中进行介绍。

```
┌──────────────────────┐    ┌──────────────────────┐    ┌──────────────────────┐
│  Application Server   │    │    OmniHSS (HSS)     │    │      Database        │
└──────────────────────┘    └──────────────────────┘    └──────────────────────┘
           │                           │                           │
           │   Sh PUR                  │                           │
           │  (□□□□, □□□□)              │                           │
           │                           │                           │
           │                      □□□□□□□□□                         │
           │                   (sh_repository_data)                │
           │                                                       │
           │                          □□□                          │
           │   Sh PUA                  │                           │
           │   □□: □□                  │                           │
           │                           │                           │
┌──────────────────────┐    ┌──────────────────────┐    ┌──────────────────────┐
│  Application Server   │    │    OmniHSS (HSS)     │    │      Database        │
└──────────────────────┘    └──────────────────────┘    └──────────────────────┘
```

□□□□□□□ (SNR/SNA)

□□□□□□□□□□□□□□□□□□

Application Server   OmniHSS (HSS)

Sh SNR
(□□□□, □□□□)

□□□□

Sh SNA
□□: □□

□□□□□□…

Sh PNR
(□□□□□□□□□)

Sh PNA
□□: □□

Application Server   OmniHSS (HSS)

Gx □□ (□□□□)

OmniHSS □□ Gx □□□□ PCRF□□□□□□□□□□□□□□

□□□ **PCRF** □□ □□□□□□□□□□□□□□ **QoS** □□□

□□□□□□□ - □□ (CCR-I/CCA-I)

P-GW □□□ PDN □□□□□□□□□□□□□
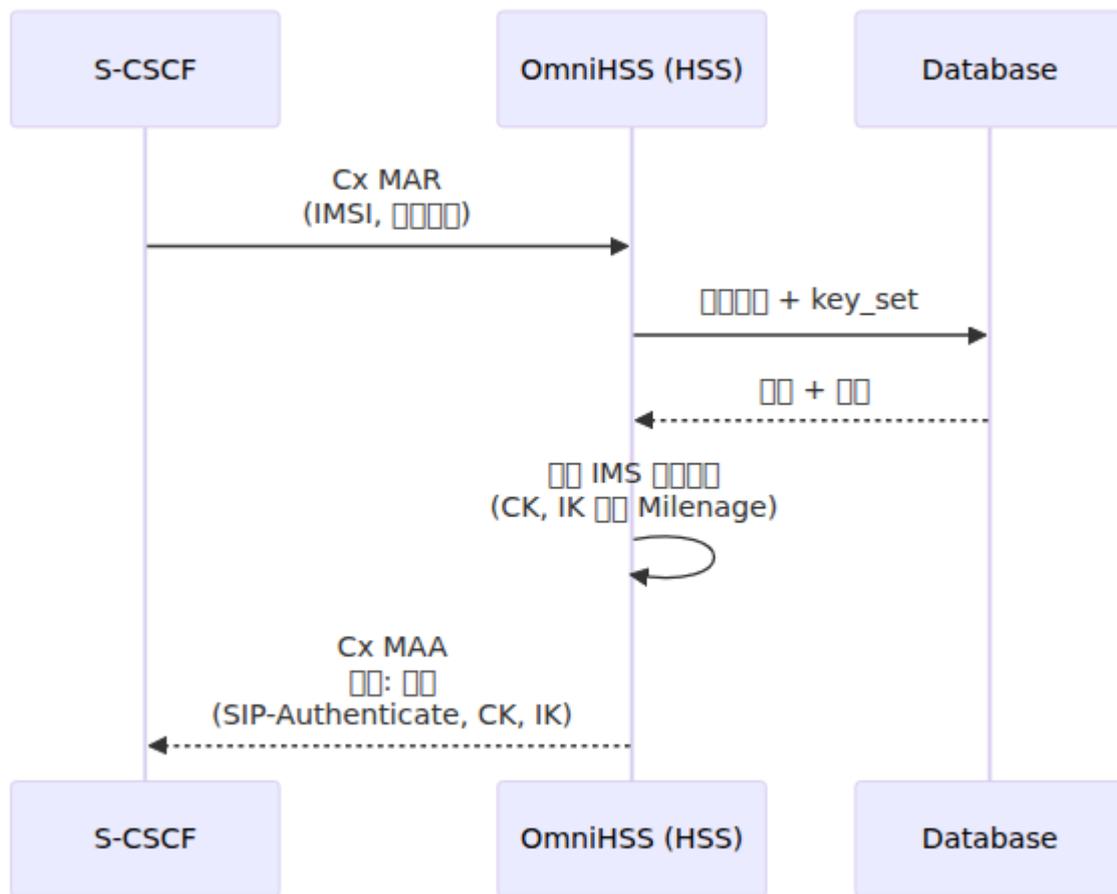
□□ AVP:

- □□: □□ ID (IMSI), □□□ ID (APN), RAT □□, IP-CAN □□
- □□: QoS □□ (QCI, ARP, AMBR), □□□□□□

□□□□□□ - □□ (CCR-U/CCA-U)

P-GW □□□□□□□

P-GW	OmniHSS (PCRF)	Database

Gx CCR-U
(Session-Id, □□□□)

□□ pdn_session
(□□□□□□, RAT □□□□)

□□□

Gx CCA-U
□□: □□
(□□□□□□□□□□□)

P-GW	OmniHSS (PCRF)	Database

□□□□□□ - □□ (CCR-T/CCA-T)

P-GW □□ PDN □□□□□□□

重新认证请求 (RAR/RAA)

OmniHSS (PCRF) 向 P-GW 发起重新认证

OmniHSS (PCRF)		P-GW

| □□□□□□ |
| (□□□VoLTE □□□□) |

Gx RAR
(□□ ID, □□□ QoS □□)

□□□□□
(□□□□□□□□�□)
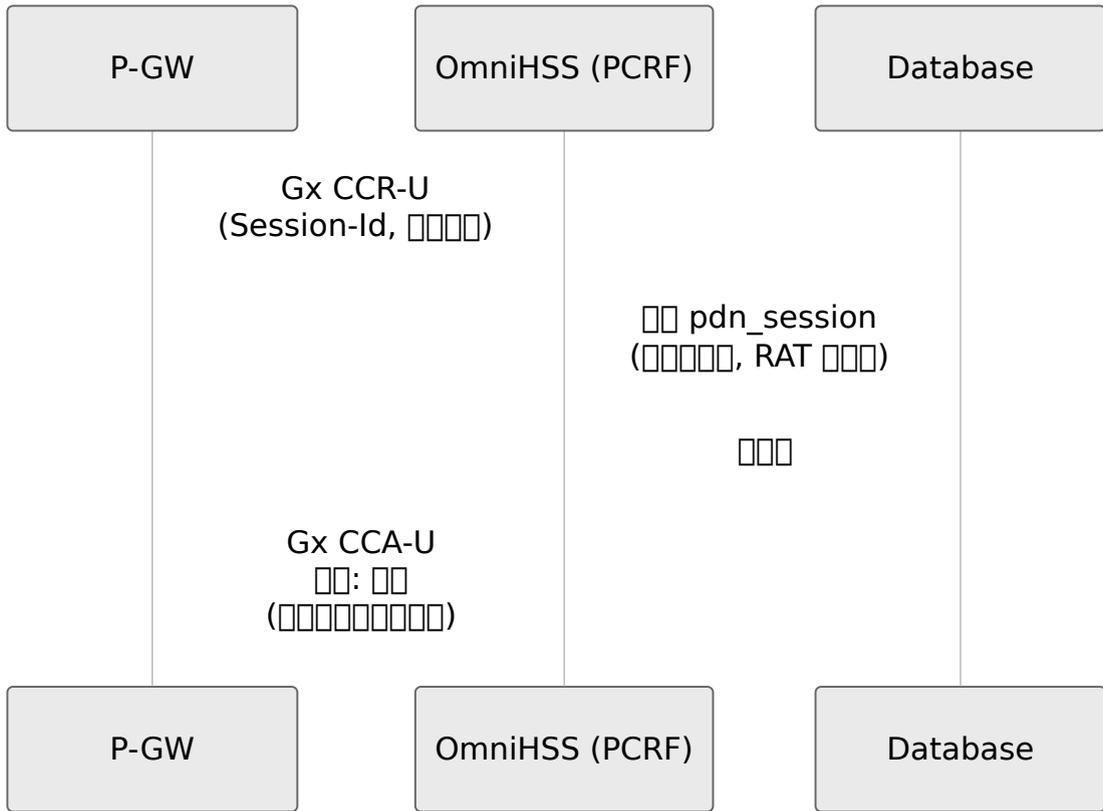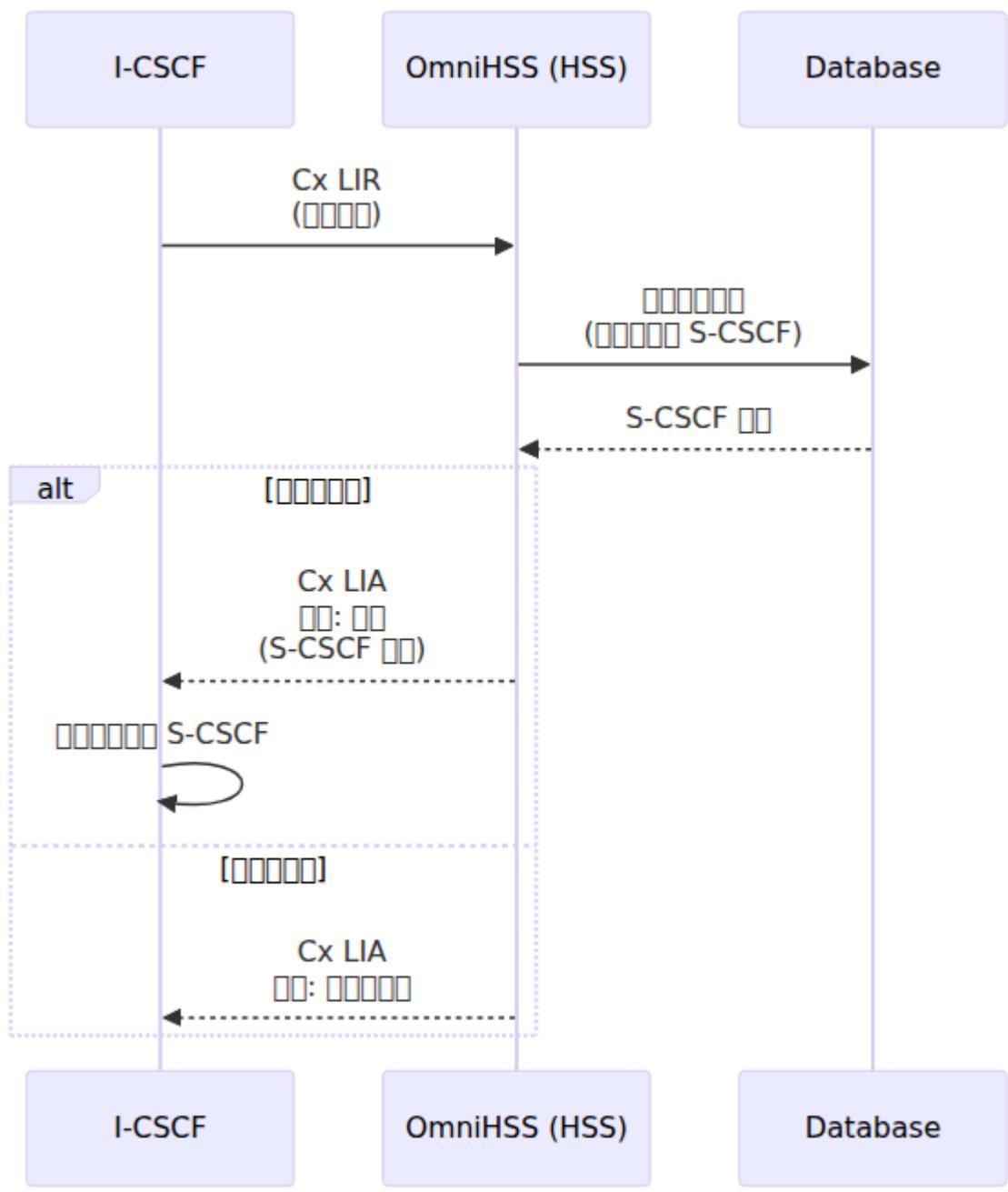
Gx RAA
□□: □□

OmniHSS (PCRF)		P-GW
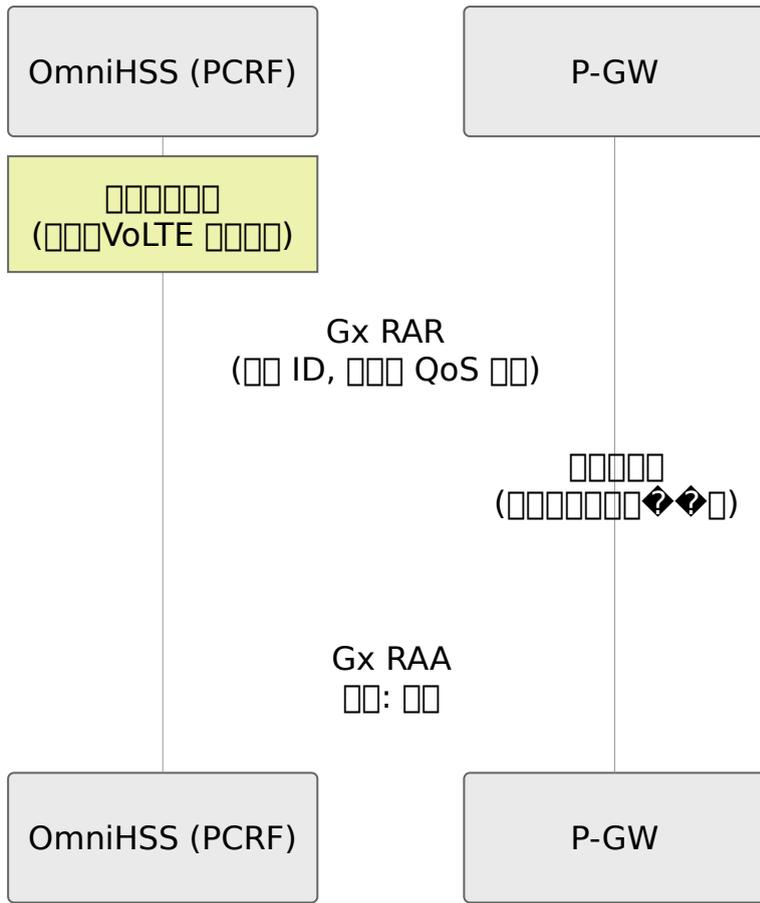
Rx □□ (IMS □□□□)

OmniHSS □□ Rx □□□□ PCRF □□□ IMS □□□□□□

□□□ **PCRF** □□ □□□□□□ **VoLTE** □□□□□□□□□□□

AA □□ (AAR/AAA)

P-CSCF □□ IMS □□□□□□□□□

□□□□:
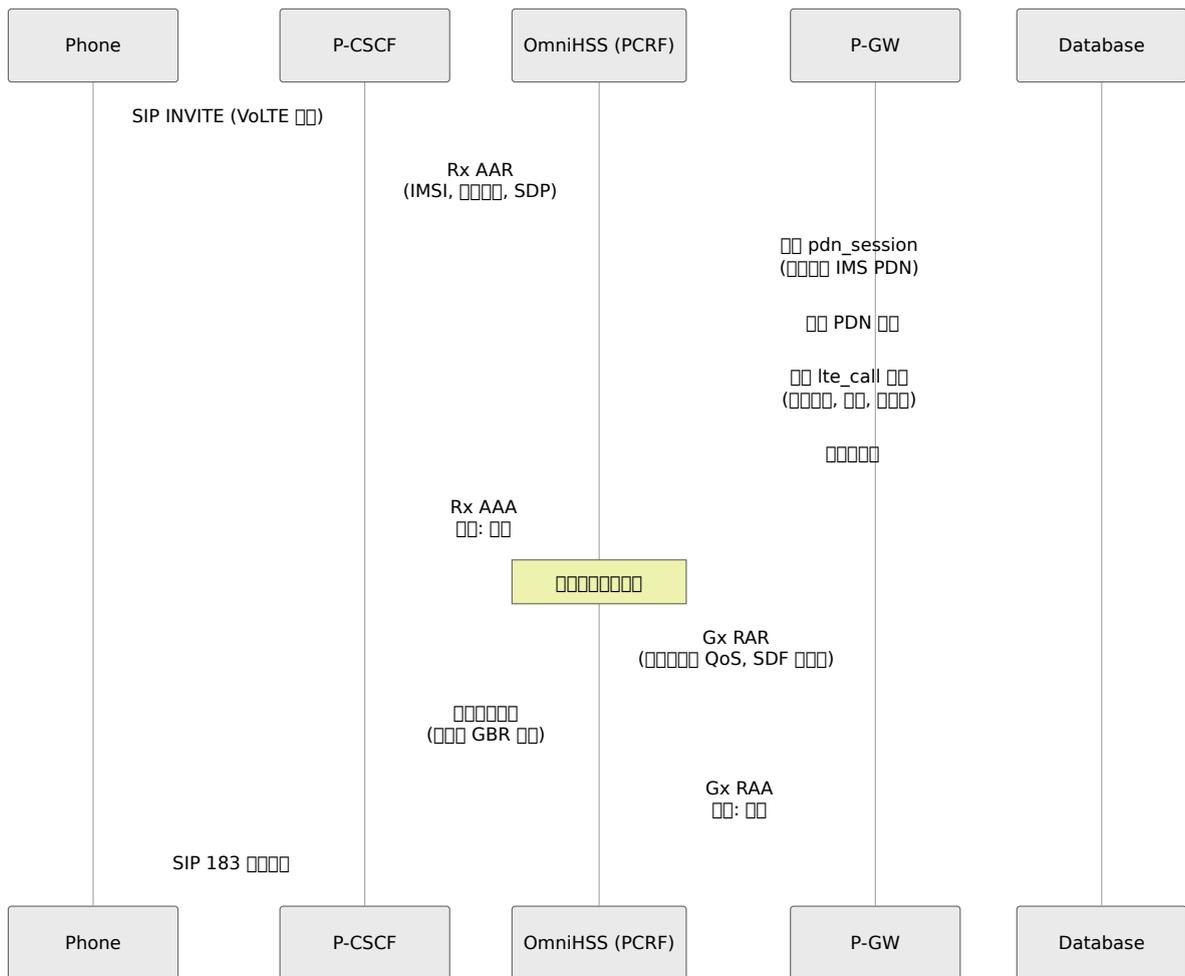
- □□ SDP □□□□□□□□□□□
- □□□□□□ (□□/□□)
- □□□□□□ SDF □□□
- □□ Gx RAR □□□□□□

□□□□□□ (STR/STA)

P-CSCF □□□ IMS □□□□□□

S13 接口 (EIR)

OmniHSS 支持 S13 接口充当 EIR（设备身份寄存器）。

该功能 **EIR 接口** 是基于设备识别检查功能，通过 **IMEI** 验证是否允许设备接入。

ME 标识检查请求 (ECR/ECA)

用于 EIR 接口，由 MME 发起检查设备身份。

```
┌─────────────────┐      ┌─────────────────┐      ┌─────────────────┐
│  MME/EIR 实体   │      │  OmniHSS (EIR)  │      │    Database     │
└─────────────────┘      └─────────────────┘      └─────────────────┘
         │                        │                        │
         │       S13 ECR          │                        │
         │       (IMEI)           │                        │
         │                        │                        │
         │                        │     检查 eir_rules     │
         │                        │ (按优先级顺序匹配 IMEI) │
         │                        │                        │
         │                        │        匹配规则        │
         │                        │                        │
         │                        │       返回匹配结果       │
         │                        │                        │
         │  ▆▆▆    [匹配规则: 黑名单]                        │
         │                        │                        │
         │       S13 ECA          │                        │
         │       结果: 禁止使用    │                        │
         │                        │                        │
         │       [匹配规则: 灰名单]  │                        │
         │                        │                        │
         │       S13 ECA          │                        │
         │       结果: 仅监控/允许使用 │                     │
         │                        │                        │
         │       [匹配规则: 白名单]  │                        │
         │                        │                        │
         │       S13 ECA          │                        │
         │       结果: 允许使用/白名单 │                     │
         │                        │                        │
         │       [无匹配规则]       │                        │
         │                        │                        │
         │  按照 unknown_equipment_behaviour 处理            │
         │                        │                        │
         │       S13 ECA          │                        │
         │       结果: (取决于配置)  │                        │
         │                        │                        │
└─────────────────┘      └─────────────────┘      └─────────────────┘
│  MME/EIR 实体   │      │  OmniHSS (EIR)  │      │    Database     │
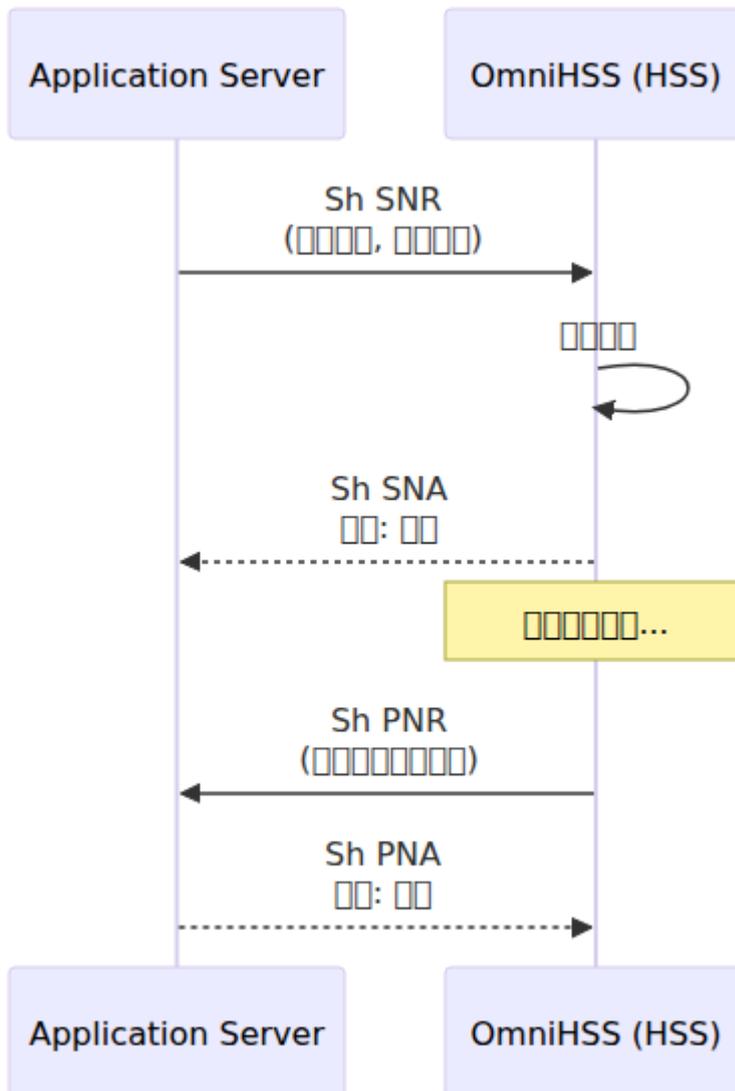└─────────────────┘      └─────────────────┘      └─────────────────┘
```

□□□□□:

- □□□□ (0) - □□□□□□□□□□
- □□□□□□□□□ (1) - □□□□□□
- □□□□□□□□□ (2) - □□□□□□□□

□□□□□□: VoLTE □□

□□□ VoLTE □□□□□□□□□□□□□

□□□□□□:

Application Server → **OmniHSS (HSS)**
Sh SNR
(□□□□, □□□□)

□□□□ (OmniHSS self-loop)

Sh SNA
□□: □□

□□□□□□...

Sh PNR
(□□□□□□□□□)

Sh PNA
□□: □□

□□□□□□□□

□□□□ (S6a AIR)

□□:

1. □□□□□□□ (Ki, OPC, AMF)
2. SQN □□ (□□□□□□)
3. □□□□□□□□□□□

□□□□□□ (S6a ULR)

□□:

1. EPC □□□□□□□□□□□ APN
2. □□□□□□□□□
3. MME □□□□□□□

IMS □□□□ (Cx SAR)

□□:

1. IMS □□□□□□□□□□□
2. IFC □□□□ XML
3. S-CSCF □□□□□
4. □□□□□□□□□□□□ MSISDN

PDN □□□□ (Gx CCR-I)

□□:

1. APN □ EPC □□□□□ APN □□□□□
2. APN QoS □□□□□□□
3. PDN □□□□□□□□□□□□□

OmniHSS □□□□

← □□□□□□

□□

OmniHSS □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ IMS □□□□□□□

漫游策略流程

漫游配置管理

概述

漫游规则

每条规则针对特定的 MCC/MNC 组合定义行为。

字段：

- `name` - 规则名称
- `mcc` - 移动国家代码（3 位数字）
- `mnc` - 移动网络代码��（2-3 位数字）
- `data_action` - "allow" 或 "deny"
- `ims_action` - "allow" 或 "deny"

默认行为

当没有规则匹配时的后备行为。

字段：

- `name` - □□□□
- `data_action_if_no_rules_match` - "allow" □ "deny"
- `ims_action_if_no_rules_match` - "allow" □ "deny"

□□□□

□□□□□□

```
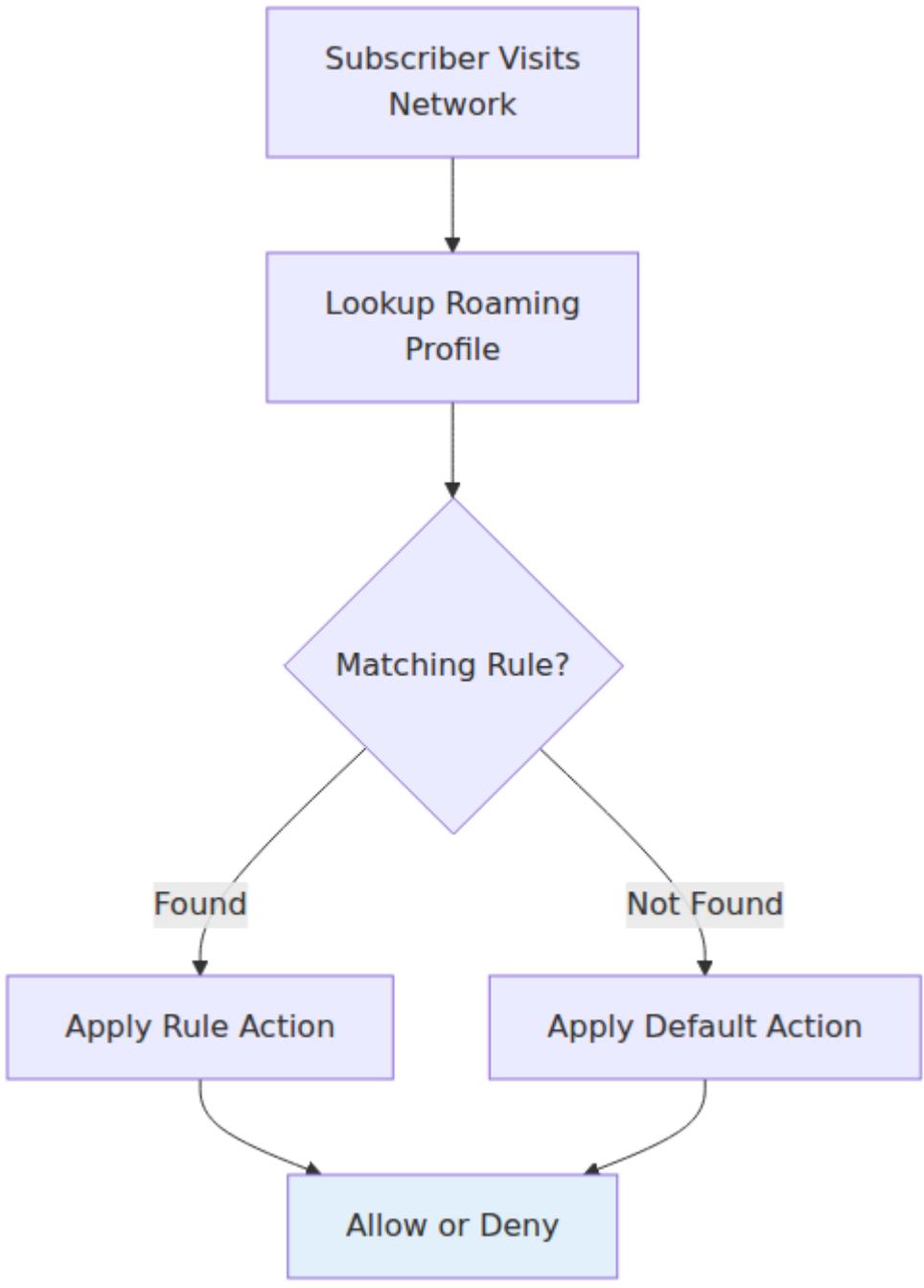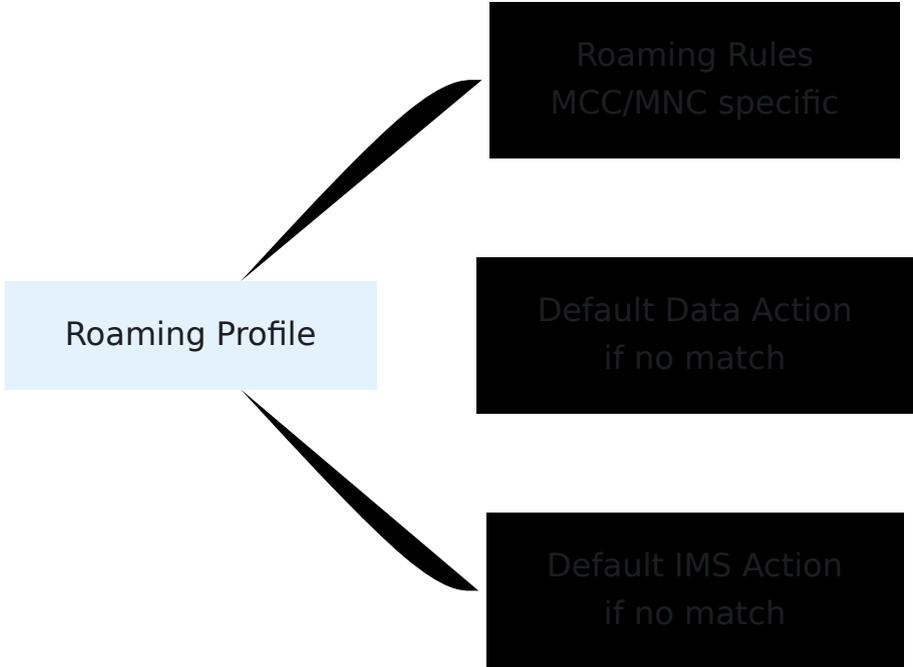# □□□□□□□□□□□□
curl -k -X POST https://hss.example.com:8443/api/roaming/profile \
  -H "Content-Type: application/json" \
  -d '{
    "roaming_profile": {
      "name": "Allow All",
      "data_action_if_no_rules_match": "allow",
      "ims_action_if_no_rules_match": "allow",
      "roaming_rules": []
    }
  }'
```

□□□□□□

```
# □□□□□□□□□□□□
curl -k -X POST https://hss.example.com:8443/api/roaming/profile \
  -H "Content-Type: application/json" \
  -d '{
    "roaming_profile": {
      "name": "No Roaming",
      "data_action_if_no_rules_match": "deny",
      "ims_action_if_no_rules_match": "deny",
      "roaming_rules": []
    }
  }'
```

□□□□□□□□□□□□□

```bash
# □□ AT&T □□
RULE1=$(curl -k -X POST
https://hss.example.com:8443/api/roaming/rule \
  -H "Content-Type: application/json" \
  -d '{
    "roaming_rule": {
      "name": "Allow AT&T",
      "mcc": "310",
      "mnc": "410",
      "data_action": "allow",
      "ims_action": "allow"
    }
  }' | jq -r '.response.id')

# □□ Verizon □□
RULE2=$(curl -k -X POST
https://hss.example.com:8443/api/roaming/rule \
  -H "Content-Type: application/json" \
  -d '{
    "roaming_rule": {
      "name": "Allow Verizon",
      "mcc": "311",
      "mnc": "480",
      "data_action": "allow",
      "ims_action": "allow"
    }
  }' | jq -r '.response.id')

# □□□□□□□□□□□□□□□□
curl -k -X POST https://hss.example.com:8443/api/roaming/profile \
  -H "Content-Type: application/json" \
  -d "{
    \"roaming_profile\": {
      \"name\": \"US Carriers Only\",
      \"data_action_if_no_rules_match\": \"deny\",
      \"ims_action_if_no_rules_match\": \"deny\",
      \"roaming_rules\": [$RULE1, $RULE2]
    }
  }"
```

配置国际漫游策略

```
# 允许数据漫游但阻止 IMS 注册
curl -k -X POST https://hss.example.com:8443/api/roaming/rule \
  -H "Content-Type: application/json" \
  -d '{
    "roaming_rule": {
      "name": "Data Only - T-Mobile",
      "mcc": "310",
      "mnc": "260",
      "data_action": "allow",
      "ims_action": "deny"
    }
  }'
```

漫遊規則配置範例腳本

```
# 先建立一條昂貴網路封鎖規則
RULE=$(curl -k -X POST
https://hss.example.com:8443/api/roaming/rule \
  -H "Content-Type: application/json" \
  -d '{
    "roaming_rule": {
      "name": "Block Expensive Network",
      "mcc": "206",
      "mnc": "01",
      "data_action": "deny",
      "ims_action": "deny"
    }
  }' | jq -r '.response.id')

# 再建立漫遊設定檔套用
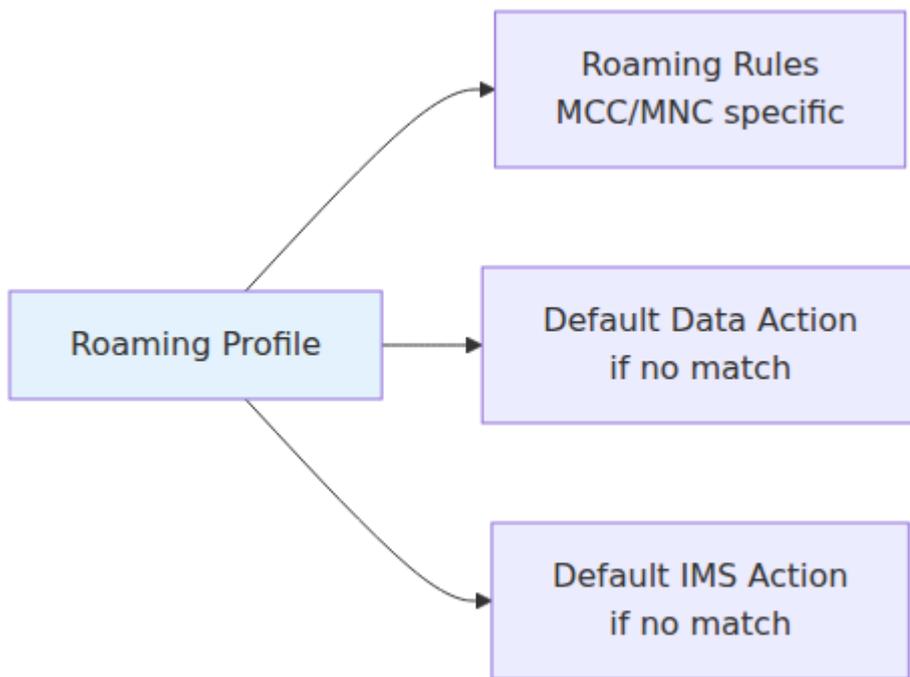curl -k -X POST https://hss.example.com:8443/api/roaming/profile \
  -H "Content-Type: application/json" \
  -d "{
    \"roaming_profile\": {
      \"name\": \"Block Expensive Networks\",
      \"data_action_if_no_rules_match\": \"allow\",
      \"ims_action_if_no_rules_match\": \"allow\",
      \"roaming_rules\": [$RULE]
    }
  }"
```

常見問題排解

問題 1：無法連線

若無法連線至伺服器，請檢查網路設定。

说明：

- 可以设置多条规则
- 默认动作可以根据 MCC 来设置（310、311、312、313、314、315、316）
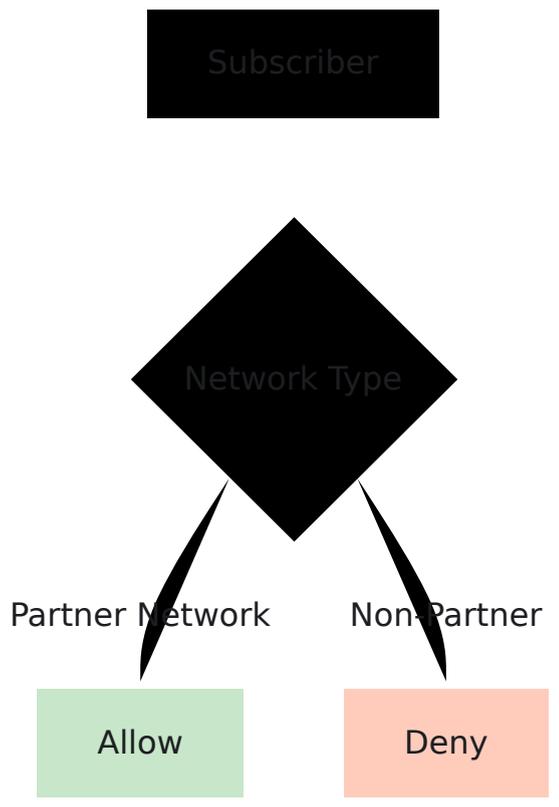
步骤 2：创建漫游配置文件

现在我们有了规则集，让我们创建配置文件

```
        ┌─────────────┐
        │  Subscriber │
        └─────────────┘

           ◆ Network Type ◆

  Partner Network      Non-Partner

  ┌─────────┐         ┌─────────┐
  │  Allow  │         │  Deny   │
  └─────────┘         └─────────┘
```

说明：

- 此为简化逻辑。
- 实际需结合运营商协议与漫游区域 MCC/MNC。

场景 3：国际漫游下的数据与语音

假设用户身处海外，运营商仅支持 WiFi 通话的部分功能。

```
          ┌──────────────────┐
          │ Subscriber Abroad│
          └──────────────────┘

  ┌──────────────┐      ┌──────────────┐
  │ Data Request │      │ Voice Request│
  └──────────────┘      └──────────────┘

  ┌──────────────┐      ┌──────────────┐
  │  Allow Data  │      │   Deny IMS   │
  └──────────────┘      └──────────────┘
```

说明：

- 行为：`data_action: "allow"`，`ims_action: "deny"`

场景 4：多网络切片场景

用户签约了多个网络切片，需要分别控制。

解决方案 ：可以通过配置不同的 MME/会话管理策略，在 OmniHSS 中实现精细化的切片管理。

MCC/MNC 参考

常用国家代码（MCC）
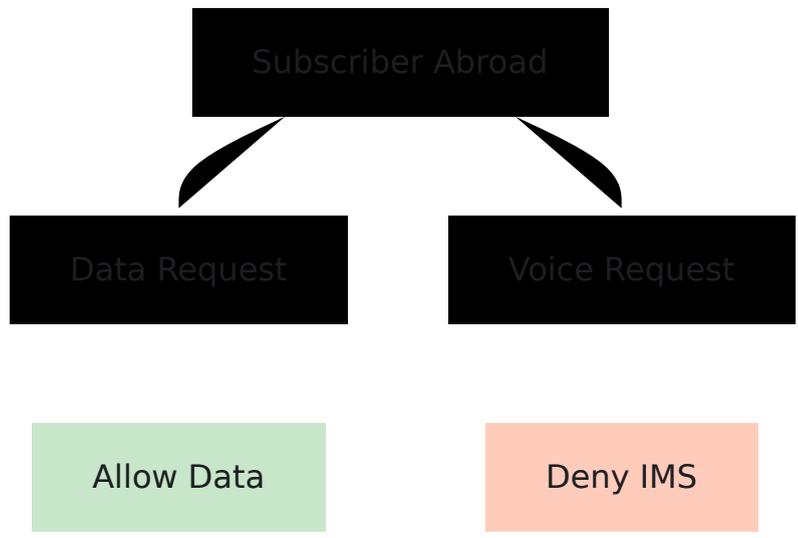
MCC	国家	运营
310-316	美国	AT&T, Verizon, T-Mobile 等
302	加拿大	Rogers, Bell, Telus
234-235	英国	Vodafone, O2, EE
262	德国	Deutsche Telekom, Vodafone
208	法国	Orange, SFR, Bouygues
222	意大利	TIM, Vodafone, Wind
214	西班牙	Movistar, Vodafone

美国主要运营商（MCC 310-316）

MCC	MNC	运营商
310	410	AT&T
311	480	Verizon
310	260	T-Mobile
310	120	Sprint
313	380	待定（新分配）
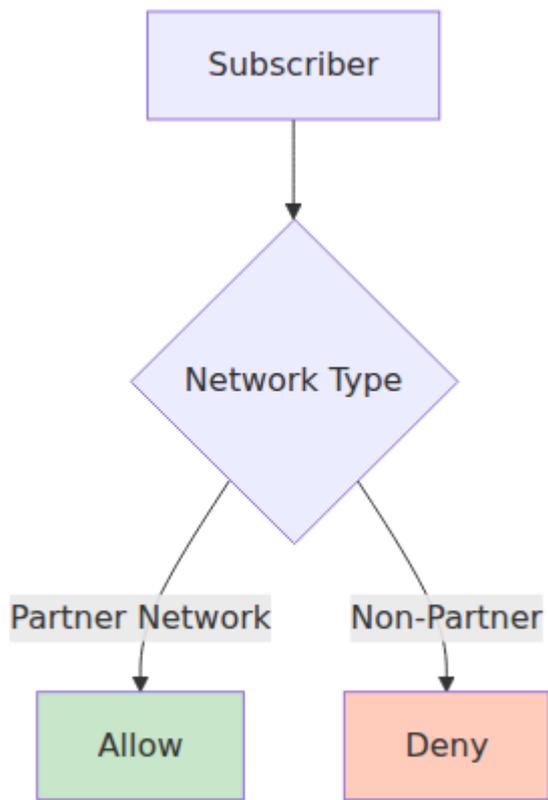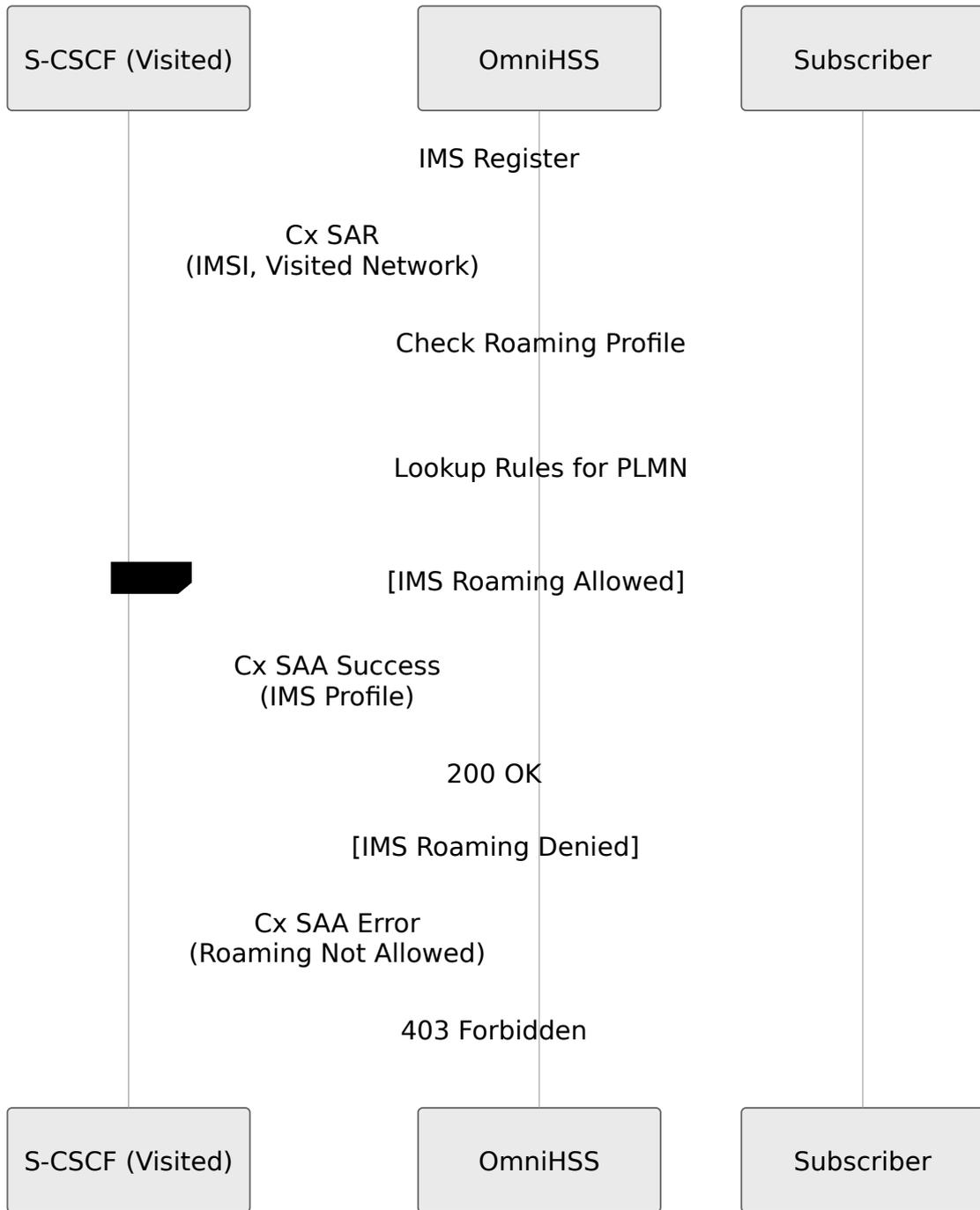
完整列表见 官方的 ITU-T E.212 和 MCC/MNC 数据库

第七部分

S6a 认证与鉴权

在核心网中的位置与作用

Cx 接口（IMS）

用于控制用户是否能够接入 IMS 网络

```
┌─────────────────┐      ┌─────────────────┐      ┌─────────────────┐
│ S-CSCF (Visited)│      │    OmniHSS      │      │   Subscriber    │
└─────────────────┘      └─────────────────┘      └─────────────────┘
        │                        │                        │
        │                   IMS Register                  │
        │                        │                        │
        │         Cx SAR                                  │
        │   (IMSI, Visited Network)                       │
        │                        │                        │
        │                Check Roaming Profile            │
        │                        │                        │
        │                Lookup Rules for PLMN            │
        │                        │                        │
        █▌                  [IMS Roaming Allowed]         │
        │                        │                        │
        │      Cx SAA Success                             │
        │        (IMS Profile)                            │
        │                        │                        │
        │                   200 OK                        │
        │                        │                        │
        │                 [IMS Roaming Denied]            │
        │                        │                        │
        │      Cx SAA Error                               │
        │   (Roaming Not Allowed)                         │
        │                        │                        │
        │               403 Forbidden                     │
        │                        │                        │
┌─────────────────┐      ┌─────────────────┐      ┌─────────────────┐
│ S-CSCF (Visited)│      │    OmniHSS      │      │   Subscriber    │
└─────────────────┘      └─────────────────┘      └─────────────────┘
```

□□□□□□□

□□□□□□□□□□□□□□□□

□□□□□□□□□□□

- □□□□□□□□□□□□□□□□□□□□□□□□

- □□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□

- □□□□□□□□□□□□□□□□ MCC/MNC □□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□ data_action □

□□□□□□□□□□□□□ IMS

□□□□ **IMS** □□□

- □□□□□□□□□□□□□□
- □□ data_action □ ims_action □
- □□□□□□□□□ IMS □□□□□□

□□□□□□□

□□□□□□□□□

```
[info] Roaming check: IMSI 001001123456789, Visited PLMN 310-410
[info] Roaming rule matched: "Allow AT&T"
[info] Data action: allow, IMS action: allow
```

□□□□

□□□□

1. □□□□□ - □□□□□□□□□□□□□□
2. □□□□ - □□□□□□□□□□□□□□□
3. □□□□ - □□□□□□□□□□□□
4. □□□□ - □□□□□□□□□□□□□□

□□□□

1. □□□□□□□ - "Allow-ATT-Data-Only" □□□ "Rule1"
2. □□ **MCC/MNC** - □□□□□□□□□□□□□
3. □□□□□□ - □□□□□□□□ IMS
4. □□□□□□ - □□□□□□□□□□□□□

□□□□

1. □□□□ - □□□□□/□□□□□□□
2. □□□□ - □□□□□□□□□□□□□□
3. □□ - □□□□□□□□□□□□□
4. □□□□ - □□□□□□□□□□

OmniHSS 系统操作手册

目录

OmniCharge OmniRAN Downloads 文A 〔□□□〕▼ Omnitouch Website

□□□□□□

□□□□□

□□□□□□□□□□□□□
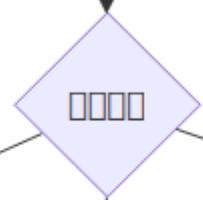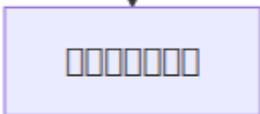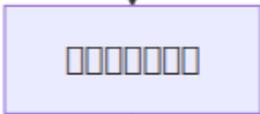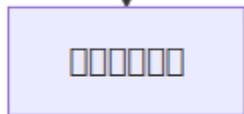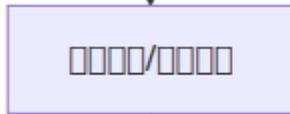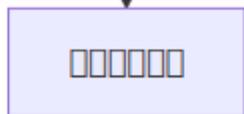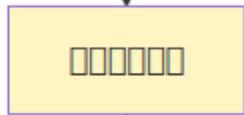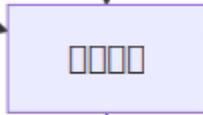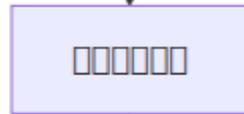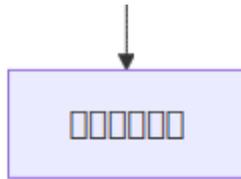
1. □□□□□□□□□□□□□

   - IMSI
   - MSISDN□□□□□□
   - □□□□□□
   - □□□□□□□□

2. □□□□

   - □□□□□□
   - □□□□□□□□□
   - □□□□□□□□□

3. □□□□

   - □□□□□□□□□
   - □□□□□□□□□
   - □□□□□□/□□□□□□□

4. □□□□

   - □□ □□□□ □□□□□
   - □□ Diameter □□□□□
   - □□□□□□□

鉴权不匹配

症状

- 手机无法注册到网络
- "鉴权拒绝"错误
- 间歇性连接问题

诊断与解决方案

步骤 1：验证密钥材料

检查：

- 数据库中的鉴权密钥正确无误
- 密钥格式正确（十六进制与二进制转换）

诊断步骤：

1. 查询订阅者的 key_set_id：

   ```
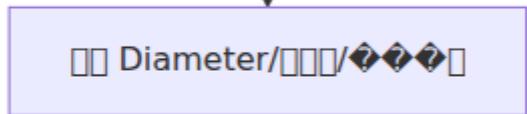   curl -k https://hss.example.com:8443/api/subscriber/imsi/[IMSI]
   ```

2. 检查此密钥集关联的鉴权密钥：

   ```
   curl -k https://hss.example.com:8443/api/key_set/[KEY_SET_ID]
   ```

3. 将 Ki 和 OPC 值与 SIM 卡供应数据进行对比

关键检查项：

- 确保密钥采用 **十六进制** 格式存储
- 核实密钥匹配与SIM 卡上的密钥匹配

步骤 2：SQN 不同步

症状：

- □□□□□□□□□□□□□□□□□□□
- □□□□"SQN □□□□"
- □□□□□□

□□□□□

1. □□□□□□□□□□□□□ SQN □
2. □□□□□□□□ SQN □□□□□
3. □□□□□□□□□□ SQN □

□□□□□

- □□□□□□ AUTS □□SQN □□□□□□□
- □□□□□□□□□□□□□□□ SQN □ 0□□□□□□□□□□□□

□□□□ □□ SQN □□□□□□□□□□□□□□□□□□□□□□

□□ 3□□□□□□□

□□□

- □□□□□□□□□□
- □□□□□□□□□□□

□□□□□

1. □□□□□□□□□□□

   ```
   curl -k https://hss.example.com:8443/api/subscriber/imsi/[IMSI]
   ```

2. □□ `enabled` □□□ `true`

□□�□�□□

- □□□□□□

```
curl -k -X PUT https://hss.example.com:8443/api/subscriber/[ID]
\
   -H "Content-Type: application/json" \
   -d '{"subscriber": {"enabled": true}}'
```

步骤 4：检查 EPC 配置档案

前提：

- 至少一个已配置的配置档案
- 管理员"查看器 EPC 配置档案"

操作步骤：

1. 找到分配给的 `epc_profile_id` 订户
2. 查询 EPC 配置档案的详细信息

   ```
   curl -k
   https://hss.example.com:8443/api/epc/profile/[PROFILE_ID]
   ```
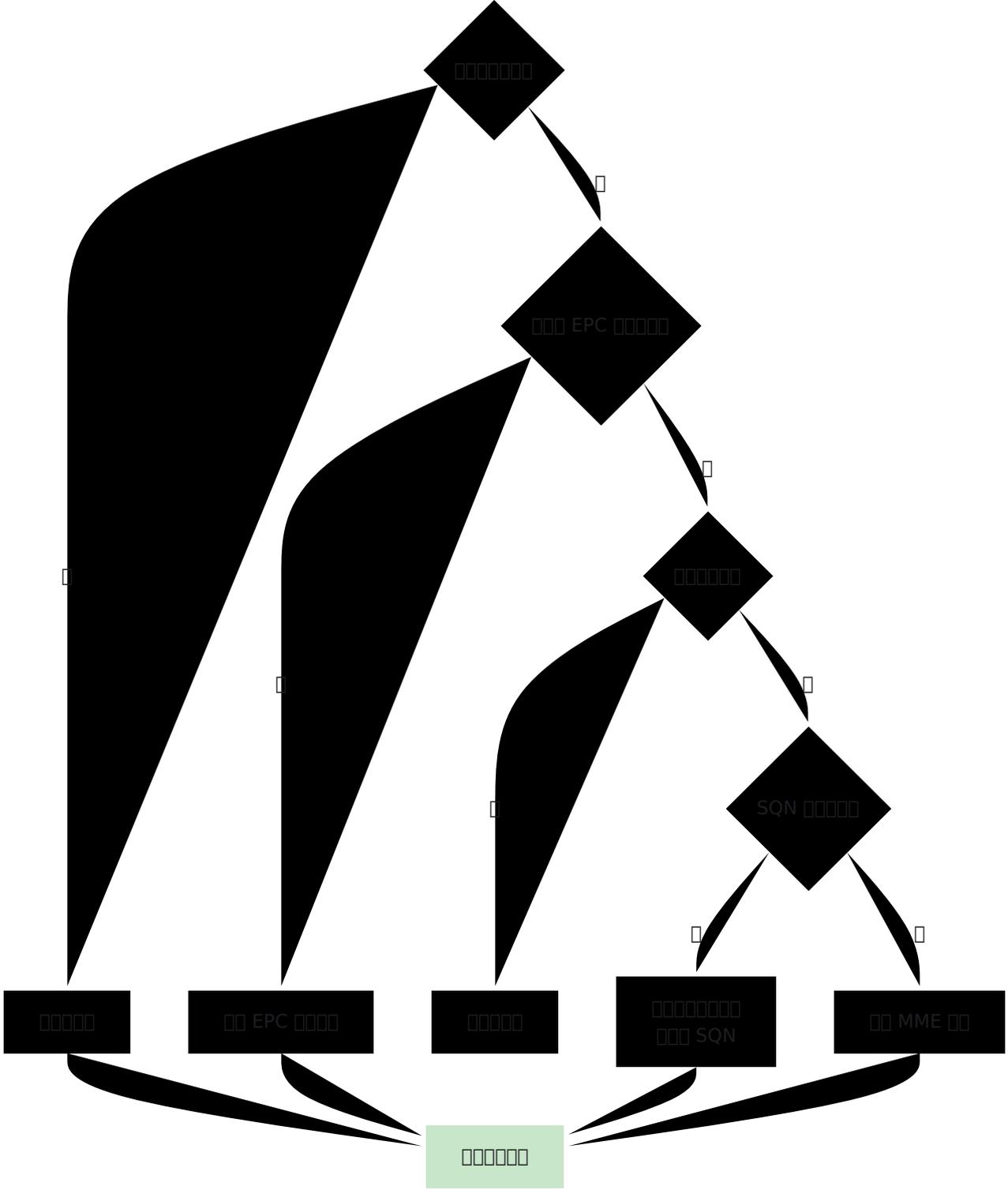
相关信息：

- 有关管理 EPC 配置档案 的更多信息，请

用户接入认证流程图

用户发起接入请求

认证是否通过

是否在 EPC 网络范围

响应是否超时

SQN 是否在范围

拒绝接入　　接入 EPC 核心网　　重新认证　　重新同步并更新SQN　　触发 MME 重选

认证流程结束

否　是　否　是　否　是　否　是

Diameter 连接问题

描述

- 与 **对端节点** 之间的 Diameter 连接无法建立。
- "连接超时或拒绝"错误。
- 服务不可用或中断。

诊断与排查步骤

步骤 1：检查网络

检查项

- 网络连接是否正常
- 路由是否正确
- Ping 测试是否通过

排查方法：

1. 确认与 OmniHSS 之间的网络连通性正常

   ```
   ping [PEER_IP]
   ```

2. 检查 Diameter 端口是否可达：

   ```
   telnet [PEER_IP] 3868
   ```

3. 如果无法连接，检查 Diameter 端口（默认 3868）

验证方法：

- 网络连接正常
- 路由配置是否正确
- 防火墙是否放行相关端口

步骤 2：Diameter 连接建立

知识点

- 传输层连接建立
- CER/CEA 消息交换
- 能力协商过程

实践任务：

1. 配置 runtime.exs 中的 Diameter 参数：

    - 设置唯一的 origin_host（建议使用服务器主机名）

    - 设置 origin_realm 参数

    - 配置监听的 IP 地址和端口

2. 观察 CER/CEA 消息的�￼�换

3. 验证连接建立后，检查与 OmniHSS 的 origin_host

检查点：

- 成功建立与 Diameter 节点 连接 runtime.exs

- 正确识别并连接到 OmniHSS

- 能力交换成功且参数正确

步骤 3：安全连接（TLS Diameter）

知识点

- 与 TLS 传输层安全协议集成
- 证书管理机制
- "双向认证"与"单向认证"区别

实践任务：

1. 准备 `priv/cert/` 目录结构和相关证书文件

2. 配置服务器证书

```
openssl x509 -in priv/cert/diameter.crt -noout -dates
```

3. □□□□□□□□□

4. □□□□□□□□□□□□□□ TLS

□□□□□

- □□□□□□□
- □□□□□□□□
- □□□□□□□□□ OmniHSS

□□ **4**□□□□□□□□□□□□

□□□

- □□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□
- "□□□□□□□□"□□

□□□□□

1. □□ □□□□ Diameter □□ □□□□□□□□
2. □□□□□□□□□□□□□□□□□□S6a□Cx□Sh □□
3. □□□□□□ CER/CEA □□

□□□□□

- □□□□□□□□□□□□□□□ Diameter □□□□
- �□□□□□□□□□□□□□□□□□
  - MME □□□□ S6a□16777251□
  - S-CSCF □□□□ Cx□16777216□
  - P-GW □□□□ Gx□16777238□

Diameter □□□□□□□□

□□□□□□□□

□

□□□ EPC □□□□□□

□

□□□□□□□

□

SQN □□□□□□

□

□□□□□□□

□

□□□□□□□

□□□□□□□

□□ EPC □□□□

□□□□□□□

□□□□□□□□□□
□□□ SQN

□□ MME □□

□□□□□□□

□□□□□

□□
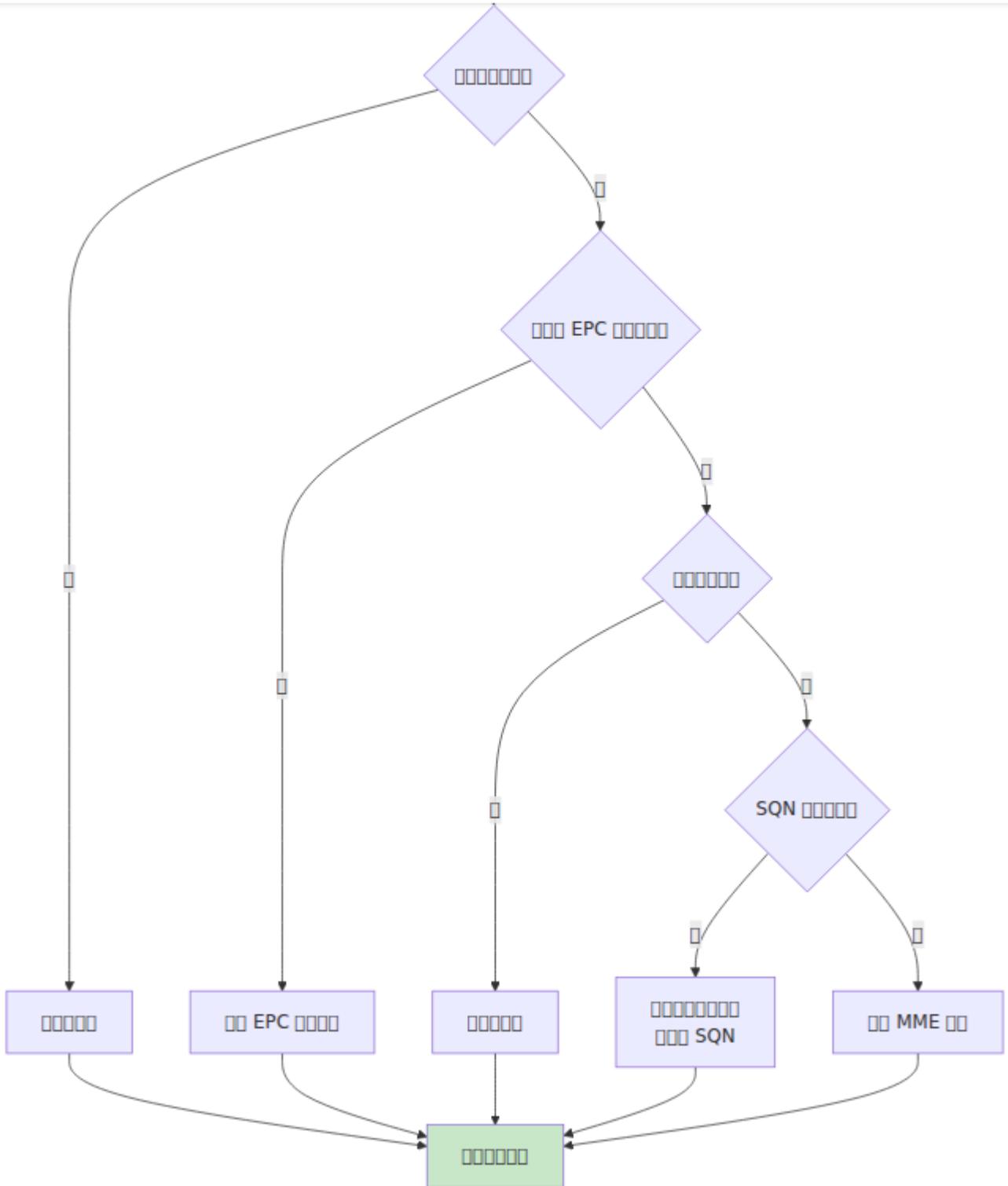
- API □□ 500 □□
- □□□□□□□□
- "□□□□□□□"□□
- □□□□□□

□□□□□□□□□

□□ **1**□□□□□□□□□

□□□

- □□ API □□□□
- □□□□□□□□
- "□□□□��□"□□

□□□□□

1. □□□□□□□□

   ```
   # □□□□ PostgreSQL
   psql -h [DB_HOST] -U [DB_USER] -d [DB_NAME]

   # □□□□ MySQL
   mysql -h [DB_HOST] -u [DB_USER] -p [DB_NAME]
   ```

2. □□□□□□□□□□□□□□□□

3. □□□□□□□□□□□□□

□□□□□

- □□□□□□□
- □□□□□□□□□

- □□□□□□□□□□□□□□□□

步骤 2：□□□□□□□□

前提：

- "□□□□□□□"□□
- OmniHSS □□□□□□□□□

操作步骤：

1. □□ runtime.exs □□□□□□□
2. □□□□□□□□□□□□□□
3. □□□□□□□□□

预期结果：

- □ runtime.exs □□□ □□□□□
- □□□□□□□□□□□□
- □□□□□□□□ OmniHSS

步骤 3：□□□□□

前提：

- □□□ 500 □□
- "□□□□□□□"□□
- □□□□□□□□□

操作步骤：

1. □□□□□□□□□□□□□
2. □□ runtime.exs □□□□□□□□
3. □□□□□□□□□□□□□

预期结果：

- □ runtime.exs □□□□□□□
- □□□□□□□□□□□□□□

- □□□□□□□□□□□□□□□□

第 4 章练习

问答题

- API □□□□□
- □□□□□□□
- □□□□ CPU □

计算与应用

1. □□□□□□□□□□□□
2. □□□□□□□□
3. □□□□□□□
4. □□□□□□□□□□

设计与开发

- □□□□□
- □□□□□□
- □□□□□□□□
- □□□□□□□□□□

□□□□□□□□□

□□□□□

□□□□□□

□ □

□□□□□□□□□□ □□□□□

□ □

□□□□ □□□□

□ □

□ □

□□ runtime.exs □□□□□ □□□□□ □□□□

□□□□□□□ □□/□□ □□ OmniHSS

□□□□

EPC 核心�络

功能

- 移动性管理：支持 LTE 接入
- MME 实体管理
- 会话与 PDN 连接

典型呼叫流程示例

步骤 1：附着请求

发起方

- 用户设备通过无线接入网发送附着请求
- "附着请求"消息
- 携带用户身份与能力信息等参数

处理过程：

1. 解析用户标识 roaming_profile_id
2. 检查用户签约信息状态
3. 验证网络标识 MCC/MNC
4. 返回附着接受或拒绝消息

返回信息：

- 网络标识符 MCC/MNC 以及 跟踪区域
- 分配临时标识与承载上下文信息
- 附着完成状态码 指示结果

步骤 2：默认 APN 处理

发起方

- 用户请求与 PDN 建立连接
- MME 获取"默认 APN"信息

- □□□□□□□□□□□□

□□□□：

1. □□ EPC □□□□□□□□□□ APN □□□□□
2. □□ APN □□□□□□□□□□□□□□
3. □□ APN □□□□□□□

□□□□：

- □ APN □□□□ □□□□□□□□ EPC □□□□□
- □□ APN □□□□□□□□□□
- □□ APN QoS □□□□□□□□□

□□ 3：MME □□□

□□□：

- □□□□□□□□□□□
- □ MME □□□□□
- Diameter □□□□□□

□□□□：

1. □□ □□□□ Diameter □□
2. □□ MME □□□□□□□"□□□"
3. □□ MME □□□□□ S6a □□□□

□□□□：

- □□□□ Diameter □□□□
- □□ MME □□
- □□ MME □□□

□□ 4：□□□□□□□

□□□：

- □□□□□□□□□□□□□□□□□□□

- □□□□□□□
- □□□□□□□□□

□□□□□

1. □□□□□□□□□□□□
2. □□□□□□□□ MME □□
3. □□□□□□□□□□

□□□□□

- □□□□□□□□□□□□□□
- □□□□□□□□□□□□ MME
- □□□□□□□□□□□□

EPC 经营项目策划书

EPC 附着流程

MME 接入控制

APN 策略检查

IMS 故障排除

概述

- 故障排除无法使用 VoLTE
- 检查用户的"IMS 订阅状态"
- 解释常见故障排除步骤

常见问题及解决方案

问题 1：用户无法注册 IMS

症状：

- 用户手机无法注册 IMS
- 用户无法拨打电话

排查步骤：

1. 检查用户订阅是否 `ims_enabled` 启用
2. 确认用户配置文件是否有 `ims_profile_id`

解决方案：

- 启用用户订阅 IMS
- 配置 IMS 配置文件

问题 2：S-CSCF 分配失败

症状：

- 加入 IMS 网络连接
- 订购与 IMS 相关的 Diameter 服务

处理流程：

1. 连接 归属网络 Diameter 服务
2. 配置 S-CSCF 服务器及地址
3. 确保 S-CSCF 可通过 Cx 接口访问

预期结果：

- 通过 Diameter 服务器 连 S-CSCF
- 配置 S-CSCF 地址

实验 3：服务组合与 IFC 配置

目标：

- 了解如何组��服务及服务链
- 掌握如何用 IFC 配置服务链

处理流程：

1. 确定用户连接 IMS 网络连接
2. 配置 IFC 服务链规则组
3. 生成 IFC XML 文件

预期结果：

- 按规则配置 IFC 并上传到 IMS 网络连接
- 验证 IFC 服务链规则组 及其服务规则

实验 4：IMS 用户的连接

目标：

- IMS 用户连接相关概念
- 用户的注册
- 确定如何将用户 IMS 连接

□□□□□

1. □□□□□□□□ IMS □□
2. □□□□□□□□□□□□ `ims_action`

□□□□□

- □□ □□□□ □□□ IMS
- □□□□□□□□□□□□ IMS □□

IMS 􀀀􀀀􀀀􀀀􀀀􀀀􀀀

VoLTE 呼叫流程

概述

- IMS 注册与认证流程概览
- 呼叫建立
- 语音承载建立
- 媒体协商与"资源预留"过程

呼叫流程详细步骤

步骤 1：P-CSCF 发现

说明：

- 用户设备发起注册请求
- 网络返回响应

详细步骤：

1. 发送 接入网络 Diameter 请求
2. 接收 P-CSCF 地址分配信息
3. 建立 P-CSCF 与核心网 Rx 接口连接（OmniHSS PCRF 节点）

交互内容：

- 发送 Diameter 请求消息 给 P-CSCF
- 接收 P-CSCF 返回的来自 OmniHSS 节点的 Rx

步骤 2：注册请求发起

说明：

- 用户设备发起注册请求
- AAR/AAA 消息交互
- Rx 接口交互

交互内容：

1. □□□□□□ Rx Diameter □□
2. □□□□□□ AAR（AA-□□）
3. □□ AAA（AA-□□）□□□

□□□□□

- □□ P-CSCF □□□□ AAR □□□□□□
- □□ OmniHSS Rx □□□□□□
- □□□□□□□□□□□□ IMS □□

□□ 3：QoS/□□□□

□□□

- □□□□□□□□□
- □□□□
- □□□□

□□□□□

1. □□□□ APN □ APN QoS □□□□
2. □□ QCI □□□□□□□□□□□ QCI 1）
3. □□ P-GW □□□□□□□ Gx（PCRF □□□

□□□□□

- □□ APN QoS □□□□ □□□□□ IMS APN
- □□□□□□□□□ QCI 1
- □□□□□□ Diameter □□□□ □ P-GW

VoLTE 呼叫建立故障排查流程

VoLTE 呼叫建立

IMS 注册成功

P-CSCF 发现成功

AAR/AAA 交互成功

承载建立成功

否 → 检查 IMS 注册

是 → 检查 Diameter 与 P-CSCF

否 → 检查 Rx 接口

是 → 检查 Rx 接口

否 → 检查 S-CSCF 配置

否 → 检查承载建立请求

是 → 检查 QoS/策略

否 → 检查 QCI/APN 配置

呼叫成功

漫游管理

概述

- 本模块负责用户跨网络的漫游管理
- 支持多种漫游场景的识别
- 提供漫游策略的配置和执行能力

漫游场景处理流程

场景 1：用户首次接入漫游网

前提：

- 用户已完成鉴权
- 用户签约支持漫游

处理步骤：

1. 系统查询用户 `roaming_profile_id`
2. 如果查询结果为 null

异常处理：

- 向 漫游管理模块 上报异常信息

场景 2：用户跨区切换

前提：

- 用户已在漫游网建立连接
- 目标网络支持漫游

处理步骤：

1. 系统从目标网络 MME 获取当前网络的 MCC/MNC
2. 比较源网络与目标网络信息
3. 判断是否需要更新本地 MCC/MNC

4. □□□□□□□□□□□□□

□□□□□□

- □□ □□□□ □□□□□□□□□

```
curl -k -X POST https://hss.example.com:8443/api/roaming/rule \
  -H "Content-Type: application/json" \
  -d '{
    "roaming_rule": {
      "name": "□□□□□□",
      "mcc": "310",
      "mnc": "410",
      "data_action": "allow",
      "ims_action": "allow"
    }
  }'
```

□□ 3□□□□□□ IMS □□

□□□

- □□□□□□
- □□/IMS □□□□
- □□□□□□□

□□□□□

1. □□□□□□□□□□□□
2. □□ `data_action` □ `ims_action` □
3. □□□□□□□□□□□□□□

□□□□□

- □□□□□□□□□ IMS□
  - □□ `ims_action: "allow"`
- □□□□□□□□ `ims_action_if_no_rules_match` □ `"allow"`

□□□□□□□□□□ □□□□□

EIR □□

□□

- □□□□□□□□
- □□□□□□□□□
- EIR □□□□

□□□□□□□□□□□□

□□ 1□IMEI □□□□□□□□□

□□□□

- □□□□□□□□□/□□
- □□□□□□□□

□□□□□□

1. □□ EIR □□
2. □□□□□□□□□□□□
3. □□□□□□□□□□□□□□ IMEI
4. □□□□□□□□/□□

□□□□□□

- □□□□□□□□□□□□□□□ EIR □□
- □□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□

□□ 2□MME □□□□ S13 □□

□□□□

- EIR □□□□□□□
- □□□□□□��□□□□□□□□□□□□

□□□□□□

1. 配置 MME 地址，为本端 S13 接口
2. 配置 MME Diameter 对接本端参数
3. 配置 S13 接口的属性
4. 增加 MME 实例

功能验证

- 查询 MME 本端 S13 接口 EIR 属性
- 查询 Diameter 对接本端参数 S13 应用为"16777252"
- 查询已经添加的 MME 实例

步骤 3：新增接口组

前提条件

- 已经对接地址及相关接口属性

操作步骤

1. 新建一个 EIR 实例
2. 为该实例添加组组信息
3. 保存完成配置

功能验证

- 查询新增的接口组信息，使用 `.*` 可以查询到 IMEI
- 查询新增接口组的对接地址信息
- 查询新增接口组的对接属性信息

黑名单

概述

- API 接口查询
- Diameter 接口查询
- CPU 占用率统计查询

- □□□□□□□□□□

□□□□□□□□□□□

□□ **1**□□□□□□□□

□□□

- □□□□□□
- □□□□ CPU □
- □□□□

□□□□□

1. □□□□□□□□□□□□□□□
2. □□□□□□
3. □□□□□□□□
4. □□□□□□□

□□□□□

- □□□□□□
- □□□□□□□
- □□□□□□□
- □□□□□□□
- □□□ □□�□□□□

□□ **2**□□□□□□□□

□□□

- □□□□□□□
- □□□□□□□□□□
- □□□□□□□□□

□□□□□

1. □□□□□□□□□

2. □□□□□

3. □□□□□□□□

4. □□□□□□□□

□□□□□

- □□□□□□
- □□□□□□□□□□
- □□□□□□□□
- □□□□□□□□□

□□ 3□Diameter □□□□□

□□□

- Diameter □□□□
- □□□□□□□□□□
- □□□□□□□□□□

□□□□□

1. □□ □□□□ Diameter □□
2. □□□□□□
3. □□□□□□□□□□□
4. □□□□□□□□□□□

□□□□□

- □□□□□□□□□
- □□□□□□□□□□
- □□□□□□□□□
- □□□□□□□ Diameter □□

□□ 4□□□□□

□□□

- OmniHSS □□□□□□

- □□□□□□
- □□□□□□□

□□□□□

1. □□□□□□□□□ OmniHSS □□□□□□□
2. □□□□□□□
3. □□□□□□□□□□□
4. □□ Erlang VM □□

□□□□□

- □□ OmniHSS □□□□□□□
- □□□□□□□□□□□□□□□
- □ runtime.exs □□□□ Erlang VM □□□□
- □□□□□□□□□□□□□

□□□□□□□□

□□

- □□□□□□□□□□□□□□□□□
- □□□□□□□
- □□□□□□□
- □□□□□□□

□□□□□□□□□

□□ 1□MME □□/□□

□□□

- □□□□□□□ MME □□ MME □□□□
- □□□□ MME □□□□□□□
- □□□□

□□□□□

1. □□□□□□□□□□ MME
2. □□ MME □□□□□
3. □□ MME □□□□□□□

□□□□□

- □□□□□□□□□□□□□□□□
- □□□□□□□□□□□
- MME □□□□□□□□ Cancel-Location

段 2：□□□□□□□□

□□□

- □□□□□□□□□□□□
- PDN □□□□□□□□□□□
- □□□□□

□□□□□

1. □□□□□□ last_seen □□□
2. □□□□□□□□□□□□□□□□□
3. □□□□□□□□□□□

□□□□□

- □□□□□□□□□□□□□□
- □□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□

段 3：□□□□□□

□□□

- □□□□□□□□□
- □□□□
- □□□□□

□□□□�□

1. □□□□□□□□□□□□□□
2. □□□□□□□
3. □□□□□□□□

□□□□□

- □□□□□□□□□□□□
- □□□□□□□□□□□□
- □□□□□□□□□□□□

API □□

□□

- API □□□□
- API □□□□
- □□□□/□□□□
- 500 □□

□□□□□□□□□

□□ **1**□□□□□□□

□□□

- 400 □ 422 □□
- □□□□□□
- □□□□□

□□□□□

1. □□□□□□□□□□□□□□□□
2. □□ API □□□□

3. □□□□□□□□□□□□
4. □□□□□□□

□□□□□：

- □□□□□□□□□□□ API □□
- □□□□□□□□□□□□
- □□□□□□□□□□□□□□□□ ID □□

□□ 2：□□□□□

□□□：

- □□□□□□□□
- □□□□"key_set_id □□□□"
- □□□□□□□□□□

□□□□：

1. □□□□□□□□□
2. □□□□□□□□□□□□□□
   - key_set_id → □□□□
   - epc_profile_id → EPC □□□□
   - ims_profile_id → IMS □□□□

□□□□：

- □□□□□□□□□□□
- □□□□□□□□ ID
- □□ □□□□□□□□□

□□ 3：□□□□□□

□□□：

- 500 □□□
- □□ API □□□□
- □□□□□□□□

ご不明な点

- 詳細 お問い合わせ

管理画面の概要

主要なセクション

1. 概要画面

    - URL: `https://[hostname]:7443/overview`
    - システム全体の状態を一目で確認できます

2. **Diameter** 画面

    - URL: `https://[hostname]:7443/diameter`
    - 接続状況をリアルタイムで監視

3. アプリケーション

    - URL: `https://[hostname]:7443/application`
    - 各種アプリケーションの設定と管理を行います

API アクセス

ステータス取得

```
curl -k https://hss.example.com:8443/api/status
```

レスポンス例

```
# 按 IMSI
curl -k
https://hss.example.com:8443/api/subscriber/imsi/001001123456789

# 按 MSISDN
curl -k
https://hss.example.com:8443/api/subscriber/msisdn/14155551234

# 按 ID
curl -k https://hss.example.com:8443/api/subscriber/1
```

获取用户列表

```
curl -k https://hss.example.com:8443/api/subscriber
```

获取用户配置文件

```
# EPC 配置文件
curl -k https://hss.example.com:8443/api/epc/profile/1

# IMS 配置文件
curl -k https://hss.example.com:8443/api/ims/profile/1

# 漫游配置文件
curl -k https://hss.example.com:8443/api/roaming/profile/1
```

网络连接检查

检查 Diameter 连接可达性

```
telnet [PEER_IP] 3868
```

检查 TLS 证书链

```
openssl s_client -connect [hostname]:8443 -showcerts
```

核对订户数量：

```
# PostgreSQL
psql -h [DB_HOST] -U [DB_USER] -d [DB_NAME] -c "SELECT COUNT(*)
FROM subscriber;"

# MySQL
mysql -h [DB_HOST] -u [DB_USER] -p -e "SELECT COUNT(*) FROM
subscriber;" [DB_NAME]
```

日志分析

搜索特定 IMSI 的日志：

```
grep "001001123456789" /var/log/omnihss/omnihss.log
```

搜索认证失败的日志：

```
grep "authentication.*fail" /var/log/omnihss/omnihss.log
```

查找 Diameter 对等体的日志：

```
grep "Diameter peer" /var/log/omnihss/omnihss.log
```

查找数据库错误：

```
grep -i "database.*error" /var/log/omnihss/omnihss.log
```

□□□□

□□□□

□□□□□□□□□□□□□□□□□□□□/□□□□□□□

1. □□□□□□□
2. □□□□□□□□□□□□□
3. □□□□□□□□□□
4. □□□□□□□□□□□□
5. □□□□□□□□□□□
6. □□□□□□□□□

□□□□□

□□□□□□□

1. □□□□ - □□□□□□□□□
2. □□□□□ - □□□□□□□□□
3. □□ - □□□□□□□□□□
4. □□ - runtime.exs □□□□□□□□□□□□
5. □□ - OmniHSS □□□□□□□□□□□□□
6. □□ - □□□□□□□□□□□□□
7. □□□□ - □□□□□□□ IMSI

□□□□□

□□□□□□□□□□□

- □□□□□□
- □□□□□□□□
- □□□□□
- □□□□

□□□□□□□□□□□□□□□□□

- □□□□□□□□□□□
- □□□□□□□□
- □□□□
- □□□□

□□□□□□□

□□□□□□

□□□□	□□	□□□□
"□□□□□□□□□□"	□□□□□□□□□	□□ □□□□□
"SQN □□□□"	SQN □□□	□□□□□□
"□□□□□□"	□□ IMSI	□□ IMSI□□□□□□
"□□□□□□"	enabled=false	□□□□□

Diameter □□

□□□□	□□	□□□□
"Diameter □□□□□□□"	□□□□	□□□□□□
"CER/CEA □□□□"	□□□□□	□□ Diameter □□
"□□□□□□□"	□□□□□□□□□□□	□□□□□□□□
"TLS □□□□"	□□□□	□□□□

□□□□□

□□□□	□□	□□□□
"□□□□□"	□□□□□	□□□□□
"□□□□□□"	□□□□	□□□□
"□□□□□□"	□□□	□□□□□
"□□□□"	□□□	□□□□

API □□

□□□□	□□	□□□□
"key_set_id □□□"	□□□□	□□□□□□□
"IMSI □□□□"	□□ IMSI	□□□□□ IMSI □□□□□□
"□□□□"	□□□□	□□□□□□□□

OmniHSS Webhook □□

□□

□□

OmniHSS 提供 **webhooks** □□□ IMS □□□□□□OmniHSS □□□□□□□□ webhook □□□□ HTTP POST □□□□□□□□□□□□□□□□□□□□□□□□

□□□ Webhooks？

Webhooks □ HTTP □□□□□□ OmniHSS □□ HSS API □□□□□□□
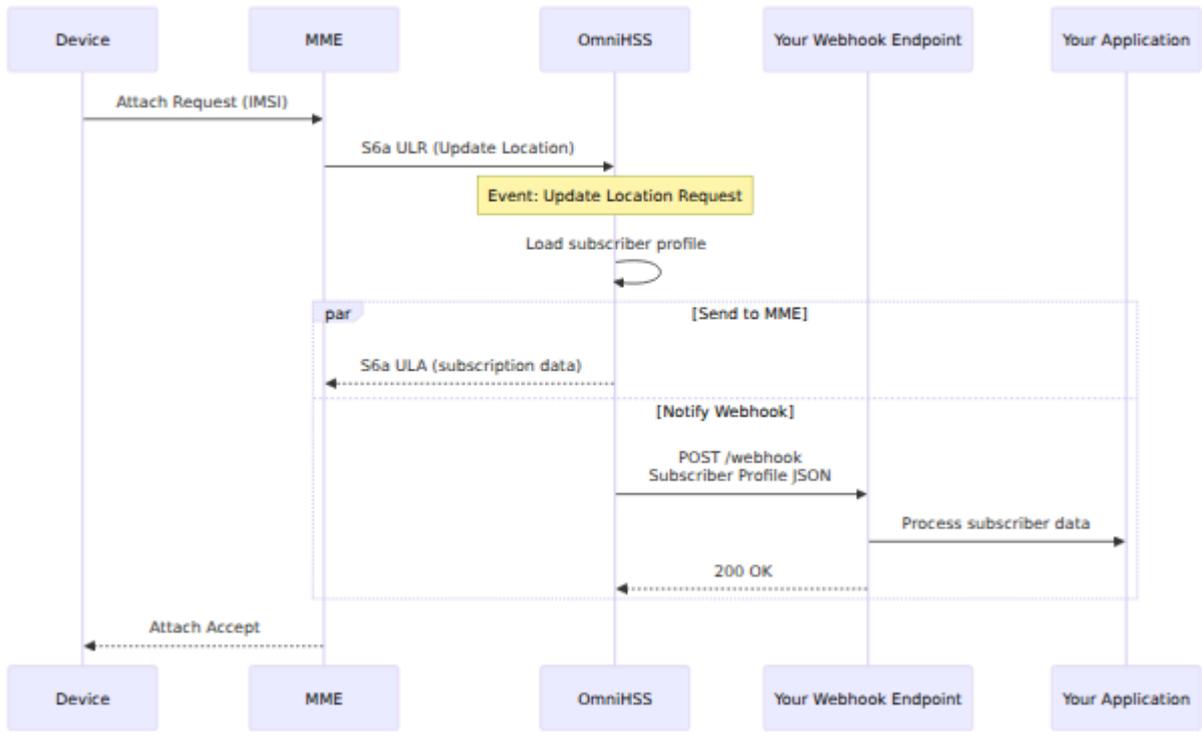
🔑🔑🔑🔑

- 🔑🔑🔑🔑 - 🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑
- 🔑🔑🔑🔑🔑🔑🔑 - 🔑🔑 webhook 🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑 `GET /api/subscriber` 🔑🔑🔑
- 🔑🔑🔑🔑🔑🔑🔑 - 🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑
- 🔑🔑🔑🔑 - 🔑🔑🔑🔑🔑🔑 API 🔑🔑🔑🔑🔑🔑🔑🔑🔑
- 🔑🔑🔑🔑🔑 - 🔑 OmniHSS 🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑

Webhook 🔑🔑🔑🔑🔑

🔑🔑🔑

1. 🔑🔑🔑🔑 - 🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑IMS 🔑🔑🔑🔑
2. **HSS 🔑🔑🔑🔑** - OmniHSS 🔑🔑🔑🔑 Diameter 🔑🔑/🔑🔑
3. **Webhook 🔑🔑🔑** - 🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑 webhook🔑HSS 🔑🔑🔑🔑🔑🔑🔑🔑 HTTP POST
4. 🔑🔑🔑🔑🔑🔑🔑 - Webhook 🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑🔑 JSON
5. 🔑🔑🔑🔑🔑🔑🔑🔑 - 🔑🔑🔑🔑��🔑🔑 HTTP 200-299 🔑🔑🔑🔑🔑

□□□□

- □□□□□□□□ - Webhooks □□□□□□□□□□□□□□□□□□□
- □□ - Webhook □□□□ 5 □□□□
- □□□ - □□□□□□□□□□□□□□□□□□□□□□ webhook
- □□□□□ - □□□□□□□□□□□□□□□□□□□

□□□ □□□□□□□□□□□□□□□□□□□□□□□□ webhook □□□□□□□□□ webhook □□□□□□□□□□□□□□□□□□□□□□

Webhook □□

OmniHSS □□□□□□□□□□ webhooks□

EPC/LTE □□

□□	□□	□□
`update_location_request`	S6a ULR	□□□□□□□□□□□□□□□□□
`authentication_information_request`	S6a AIR	□□□□□□□□□□□□□□□□□
`purge_request`	S6a PUR	MME □□□□□□□□□□□□□□□□□□□
`cancel_location_answer`	S6a CLA	MME □□□□□□□

IMS □□

□□	□□	□□
`ims_registration`	Cx SAR	□□□□□□ IMS/VoLTE □□
`ims_deregistration`	Cx SAR (de-reg)	□□□□□□ IMS
`ims_profile_request`	Sh UDR	□□□□□□□□□□□ IMS □□□□

□□□□ (PCRF)

□□	□□	□□
`policy_request`	Gx CCR	P-GW □□□□□□□□□□□□
`media_authorization`	Rx AAR	P-CSCF □□ IMS □□□□□□□

□ IMSI □□

□□	□□	□□
`imsi_switch`	ULR for different IMSI on same SIM	□□□□ IMSI SIM □□□□□□□ IMSI

Webhook □□

□□□□

□□□□□□□□OmniHSS □□□□□□ webhook URL □□ HTTP POST □□□

```
POST /your-webhook-endpoint HTTP/1.1
Host: your-server.com
Content-Type: application/json
X-OmniHSS-Event: update_location_request
X-OmniHSS-Event-ID: 550e8400-e29b-41d4-a716-446655440000
X-OmniHSS-Timestamp: 2025-01-15T14:30:00Z

{
  "event": "update_location_request",
  "event_id": "550e8400-e29b-41d4-a716-446655440000",
  "timestamp": "2025-01-15T14:30:00Z",
  "subscriber": {
    "id": 1234,
    "imsi": "001001123456789",
    "enabled": true,
    "ims_enabled": true,
    "msisdns": [
      {"id": 1, "msisdn": "14155551001"},
      {"id": 2, "msisdn": "14155551002"}
    ],
    "sim": {
      "id": 5678,
      "iccid": "8991101200003204510",
      "is_esim": false
    },
    "key_set": {
      "id": 100,
      "amf": "8000"
    },
    "epc_profile": {
      "id": 1,
      "name": "Premium 100Mbps",
      "ue_ambr_dl_kbps": 100000,
      "ue_ambr_ul_kbps": 50000
    },
    "ims_profile": {
      "id": 1,
      "name": "Standard VoLTE"
    },
    "roaming_profile": {
      "id": 1,
      "name": "International Roaming Allowed"
    },
```

```
    "subscriber_state": {
      "mme_host": "mme-01.example.com",
      "mme_realm": "epc.mnc001.mcc001.3gppnetwork.org",
      "visited_plmn": "001001",
      "last_update": "2025-01-15T14:30:00Z"
    },
    "custom_attributes": {
      "account_type": "premium",
      "billing_plan": "unlimited"
    }
  },
  "event_context": {
    "visited_plmn": "310410",
    "mme_host": "mme-roaming.example.com",
    "location_update_type": "initial_attach"
  }
}
```

字段说明

字段	类型	说明
`event`	string	事件类型，始终为 `update_location_request`
`event_id`	string	该 webhook 事件的唯一 UUID
`timestamp`	string	事件发生的 ISO 8601 时间戳
`subscriber`	object	用户对象的快照，与 `GET /api/subscriber/:id` 相同
`event_context`	object	特定于事件类型的附加信息

事件上下文字段

`event_context` 的内容取决于�� 事件类型。

对于 **update_location_request**：

```json
{
  "visited_plmn": "310410",
  "mme_host": "mme-roaming.example.com",
  "mme_realm": "epc.mnc410.mcc310.3gppnetwork.org",
  "location_update_type": "initial_attach"
}
```

对于 `imsi_switch`：

```json
{
  "previous_imsi": "001001111111111",
  "new_imsi": "310410222222222",
  "sim_id": 5678,
  "previous_mme_host": "mme-home.example.com",
  "new_mme_host": "mme-roaming.example.com"
}
```

对于 `ims_registration`：

```json
{
  "scscf_host": "scscf-01.ims.example.com",
  "public_identities": [
    "sip:001001123456789@ims.mnc001.mcc001.3gppnetwork.org",
    "sip:+14155551001@ims.example.com",
    "tel:+14155551001"
  ]
}
```

HTTP □□

□□	□□	□□
`Content-Type`	□□□ `application/json`	`application/json`
`X-OmniHSS-Event`	□□□□	`update_location_request`
`X-OmniHSS-Event-ID`	□□□□□□□	UUID
`X-OmniHSS-Timestamp`	□□□□□	ISO 8601 □□
`User-Agent`	OmniHSS □□	`OmniHSS/1.0`

□□

□□ Webhooks

Webhooks □□ OmniHSS API □□□

□□□□ Webhook

```
curl -k -X POST https://hss.example.com:8443/api/webhook \
  -H "Content-Type: application/json" \
  -d '{
    "webhook": {
      "url": "https://your-server.com/omnihss-webhook",
      "events": [
        "update_location_request",
        "ims_registration",
        "imsi_switch"
      ],
      "enabled": true,
      "description": "□□□□□ webhook"
    }
  }'
```

□□□

```
{
  "data": {
    "id": 1,
    "url": "https://your-server.com/omnihss-webhook",
    "events": [
      "update_location_request",
      "ims_registration",
      "imsi_switch"
    ],
    "enabled": true,
    "description": "□□□□□ webhook",
    "created_at": "2025-01-15T14:00:00Z"
  }
}
```

□□ Webhooks

```
curl -k https://hss.example.com:8443/api/webhook
```

□□ Webhook

```
curl -k -X PUT https://hss.example.com:8443/api/webhook/1 \
  -H "Content-Type: application/json" \
  -d '{
    "webhook": {
      "enabled": false
    }
  }'
```

删除 Webhook

```
curl -k -X DELETE https://hss.example.com:8443/api/webhook/1
```

Webhook 接收端点

实现 webhook 接收端点

1. 接受 **POST** 请求，内容为 `Content-Type: application/json`
2. 快速响应 - 在 5 秒内返回 HTTP 200-299
3. 异步处理 - 将处理逻辑放在后台
4. 使用 **HTTPS** - 保护传输中的 TLS/SSL 加密数据安全
5. 验证来源 - 检查请求来自 OmniHSS（应用安全加固）

示例 Webhook 接收端点（Node.js/Express）：

```
const express = require('express');
const app = express();

app.post('/omnihss-webhook', express.json(), (req, res) => {
  const { event, subscriber, event_context } = req.body;

  console.log(`Received event: ${event}`);
  console.log(`Subscriber IMSI: ${subscriber.imsi}`);

  // □□□□□□□□
  // ... □□□□□□□□□□ ...

  // □□□□□□□□□□□
  res.status(200).json({ received: true });

  // □□□□□□□□□□□□
  processWebhook(req.body).catch(console.error);
});

async function processWebhook(payload) {
  // □□□□□□□□□
  // □□□□□□□□□□□□□□□□
}

app.listen(3000);
```

□□

1. □□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□

Device	OmniHSS	Webhook Handler	Billing System

Attach to Network (Roaming)

update_location_request
(visited_plmn: 310410)

Check roaming rates for PLMN 310410

$0.05/MB rate

Start roaming session
Enable roaming charges

200 OK

Device	OmniHSS	Webhook Handler	Billing System

□□□

- □□□□□□□□□□□□□□
- □□□□□□□□□□□□□
- □□□□□□□□/□□□□
- □□□□□□□□□□□□□

2. □□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□ □□□□□□□□□□□□

```
// Webhook 🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲
app.post('/omnihss-webhook', async (req, res) => {
  const { event, subscriber, event_context } = req.body;

  if (event === 'update_location_request') {
    await analytics.track({
      event: 'subscriber_location_update',
      imsi: subscriber.imsi,
      visited_plmn: event_context.visited_plmn,
      timestamp: req.body.timestamp,
      profile: subscriber.epc_profile.name
    });
  }

  res.status(200).send();
});
```

🔲🔲🔲🔲🔲

- 🔲🔲 MME 🔲🔲🔲🔲🔲🔲
- 🔲🔲🔲🔲🔲🔲🔲🔲
- 🔲🔲🔲🔲🔲
- IMS 🔲🔲🔲🔲🔲

3. 🔲🔲🔲🔲🔲🔲

🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲🔲

□□□□□□□
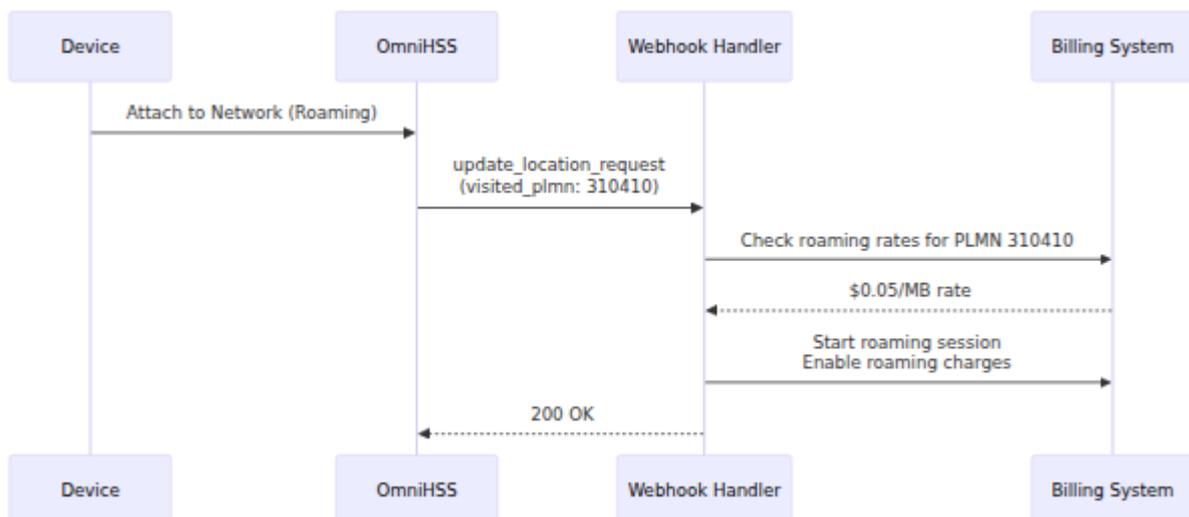
1. □□□□□□

   - □□□□□ A □□□
   - 30 □□□□□ B □□□□□□□□□□□
   - □□□□□□□□□□□□□□□□□□

2. **IMSI □□□□**

   - □□ SIM □□□□□ IMSI □□
   - □□□ SIM □□□□□□□□□ IMSI □□
   - □□□□□ SIM □□□□ IMSI□□□□□□□

3. □□□□□□□

   - □□□□□□□□□□□□□□□□□□□□□□
   - □□□□□□□□□□□□□□□□□□□

□□□□□

```python
@app.route('/omnihss-webhook', methods=['POST'])
def webhook_handler():
    data = request.json
    subscriber = data['subscriber']
    event_context = data.get('event_context', {})

    if data['event'] == 'update_location_request':
        visited_plmn = event_context.get('visited_plmn')

        # 〇〇〇〇〇〇〇〇〇
        if visited_plmn in BLOCKED_PLMNS:
            disable_subscriber(subscriber['imsi'])
            alert_security_team(subscriber, 'Roaming to blocked
PLMN')

        # 〇〇〇〇〇〇〇〇〇
        if is_impossible_travel(subscriber['imsi'], visited_plmn):
            flag_for_review(subscriber['imsi'])
            alert_fraud_team(subscriber, 'Impossible travel
detected')

    return jsonify({'status': 'ok'}), 200
```

4. 〇〇〇〇〇

〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇

〇〇〇 〇〇〇〇〇〇〇〇〇 VoLTE 〇〇〇〇〇 IMS

```
app.post('/omnihss-webhook', async (req, res) => {
  const { event, subscriber } = req.body;

  if (event === 'ims_registration' && !subscriber.ims_enabled) {
    // 激活 IMS 功能 - 按需激活 IMS
    await omnihss.updateSubscriber(subscriber.id, {
      ims_enabled: true,
      custom_attributes: {
        ...subscriber.custom_attributes,
        volte_activated_at: new Date().toISOString()
      }
    });

    // 更新 CRM
    await crm.updateCustomer(subscriber.imsi, {
      features: ['volte']
    });
  }

  res.status(200).send();
});
```

5. 国际漫游

场景：用户在国外旅行并接入网络。

流程： 欢迎短信与费率提醒流程

用户接��收

- "□□□□ [□□]□□□□□□□□□"
- "□□□□ 80% □□□□□"
- "□□□□□□□□□ VoLTE □□"
- "□□□□□□□□□□□□"
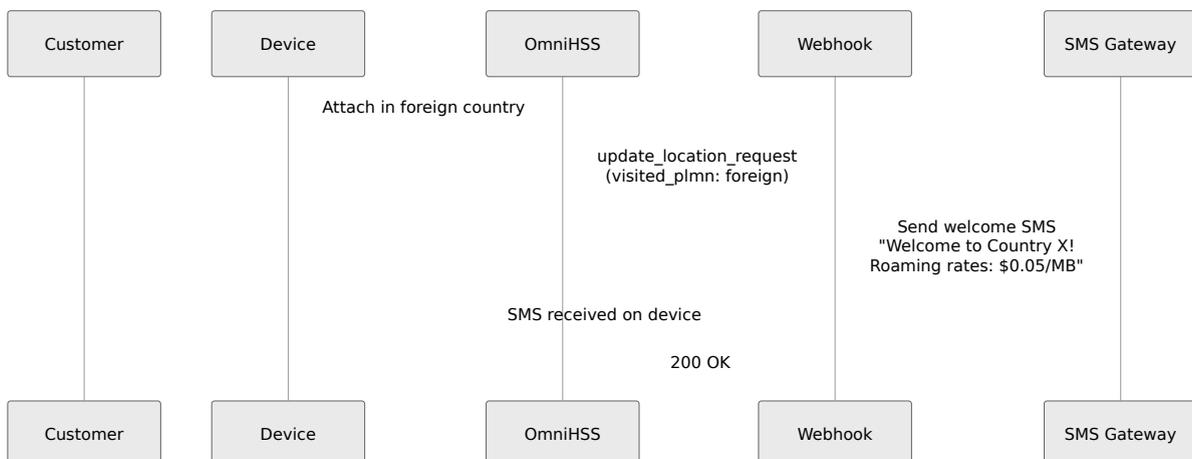
6. □ IMSI SIM □□

□□□□□□□□□ IMSI SIM □□□□□□□□□□□□ IMSI □□□□□□

```javascript
app.post('/omnihss-webhook', async (req, res) => {
  const { event, subscriber, event_context } = req.body;

  if (event === 'imsi_switch') {
    const { previous_imsi, new_imsi, sim_id } = event_context;

    // □□ IMSI □□□□□□□□□
    await db.logImsiSwitch({
      sim_id,
      from_imsi: previous_imsi,
      to_imsi: new_imsi,
      timestamp: req.body.timestamp
    });

    // □□□□□□□
    await billing.endSession(previous_imsi);
    await billing.startSession(new_imsi);

    // □□□□□□□□□□□□□□□□□□□□□□□□
    const switchCount = await db.getSwitchCount(sim_id, '24h');
    if (switchCount > 10) {
      await alertFraudTeam(`Excessive IMSI switching: SIM ${sim_id}`);
    }
  }

  res.status(200).send();
});
```

7. 实际应用示例

将 OmniHSS 与现有系统集成的示例：

集成场景：

- **CRM 系统** - 当用户位置变化时触发工作流
- 计费系统 - 根据位置信息计算服务费用
- 监控系统 - 实时跟踪用户状态
- 安全系统 - 检测异常的位置变更行为
- 分析系统 - 收集用户行为数据用于业务分析

集成接口

Webhook 订阅/通知

您可以将 webhooks 集成 OmniHSS，以接收实时通知。

```
# 注册一个新的 webhook
curl -k -X POST https://hss.example.com:8443/api/webhook \
  -H "Content-Type: application/json" \
  -d '{
    "webhook": {
      "url": "https://your-server.com/omnihss-webhook",
      "events": ["update_location_request"],
      "secret": "your-secret-key-here"
    }
  }'
```

OmniHSS 会发送带有 `X-OmniHSS-Signature` 的请求

```
X-OmniHSS-Signature:
sha256=5d7a8f9b2c1e3a4d6f7e8b9c0a1b2c3d4e5f6a7b8c9d0e1f2a3b4c5d6e7f8a
```

处理事件：

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

function verifyWebhook(req) {
  const signature = req.headers['x-omnihss-signature'];
  const secret = process.env.WEBHOOK_SECRET;
  const payload = JSON.stringify(req.body);

  const expectedSignature = 'sha256=' +
    crypto.createHmac('sha256', secret)
      .update(payload)
      .digest('hex');

  return crypto.timingSafeEqual(
    Buffer.from(signature),
    Buffer.from(expectedSignature)
  );
}

app.post('/omnihss-webhook', (req, res) => {
  if (!verifyWebhook(req)) {
    return res.status(401).json({ error: 'Invalid signature' });
  }

  // □□ webhook...
  res.status(200).send();
});
```

□□□□

1. **□□ HTTPS** - □□□ webhook □□□□□ TLS
2. □□□□ - □□ webhook □□□□□□□□
3. □□□□ - □ webhook □□□□□□□□□□
4. **IP □□□** - □□ webhook □□ OmniHSS IP □□
5. □□□□ - □□ webhook □□□□□□□□
6. □□□□ - □□□□□□□□□□ webhook □□
7. □□□□ - □□□□□□□□□ webhook □□□□□□□□□□□□□□

□□□□

Webhook □□□□ □□□□□□□□□□

- IMSI□□□□□□□□
- MSISDN□□□□□□□
- □□□□□□□□ PLMN□MME□
- □□□□□□□□□

□□□□□

- **GDPR** - □□ webhook □□□□□□□□ GDPR
- □□□□ - □□□□□□□□□□□□
- □□□□ - □□ webhook □□□□
- □□ - □□ TLS □□ webhook □□
- □□□□ - □□□□ webhook □□□□□□□

□□□□

Webhook □□□

□□□

- □□□□□ webhook □□□
- Webhook □□□□□□□□□□

□□□□□□□

1. **□□ webhook □□□□□**

   ```
   curl -k https://hss.example.com:8443/api/webhook
   # □□ "enabled": true
   ```

2. **□□ webhook □□□□□**

- 确认目标系统已配置 webhook 的 `events` 订阅。
- 如需注册通知，检查 `ims_registration` 相关的订阅是否正确开启。

3. **检查 HSS 配置**

- 确认 webhook 的端点。
- 检查端点的可达性。
- 检查相关 DNS 的解析。

4. 手动触发测试事件：

```
curl -X POST https://your-server.com/omnihss-webhook \
  -H "Content-Type: application/json" \
  -d '{"test": true}'
```

Webhook 机制

概述

- HSS 主动推送 webhook 事件。
- Webhook 异步处理，不阻塞 HSS 核心逻辑。

传递保证

1. **超时设置**
   - 默 5 秒超时（ HTTP 200）
   - 超时视为推送失败。

2. **重试策略说明**

```
// 错误 - 同步处理数据
app.post('/webhook', (req, res) => {
  processData(req.body); // 耗时 10 秒
  res.status(200).send();
});

// 对 - 先返回再处理数据
app.post('/webhook', (req, res) => {
  res.status(200).send(); // 立即返回
  processData(req.body); // 异步处理
});
```

🔁 Webhooks

幂等性

- 使用唯一事件标识
- `event_id` 可能会多次发送同一事件

重试机制

- 如果没有收到 OmniHSS 的成功响应，系统会自动重试发送
- 确保正确处理重复的 webhook

处理重复事件

记录 `event_id` 避免重复处理

```
const processedEvents = new Set();

app.post('/omnihss-webhook', (req, res) => {
  const eventId = req.body.event_id;

  if (processedEvents.has(eventId)) {
    // ☐☐☐☐☐☐
    return res.status(200).json({ status: 'duplicate' });
  }

  processedEvents.add(eventId);

  // ☐☐ webhook...
  processWebhook(req.body);

  res.status(200).json({ status: 'processed' });
});
```

Webhook ☐☐☐☐

☐☐☐

- ☐☐☐☐ HTTP 4xx ☐ 5xx
- HSS ☐☐ webhook ☐☐☐☐

☐☐☐☐☐

1. **401 Unauthorized** - ☐☐☐☐☐☐

    ○ ☐☐ webhook ☐☐☐☐☐☐☐☐☐☐
    ○ ☐☐☐☐☐☐☐☐☐

2. **400 Bad Request** - ☐☐☐☐

    ○ ☐☐ webhook ☐☐☐☐
    ○ ☐☐☐☐ Content-Type ☐☐

3. **500 Internal Server Error** - ☐☐☐☐

    ○ ☐☐☐☐☐☐☐☐

◦ ▢▢▢▢▢▢▢▢▢▢▢

▢▢▢▢▢

▢▢▢▢▢▢▢▢▢▢

```javascript
app.post('/omnihss-webhook', async (req, res) => {
  try {
    // ▢▢▢▢
    if (!verifyWebhook(req)) {
      return res.status(401).json({ error: 'Invalid signature' });
    }

    // ▢▢▢▢
    if (!req.body.event || !req.body.subscriber) {
      return res.status(400).json({ error: 'Invalid payload' });
    }

    // ▢▢ webhook
    await processWebhook(req.body);

    res.status(200).json({ status: 'ok' });

  } catch (error) {
    console.error('Webhook processing error:', error);
    // ▢▢ 200 ▢▢▢▢▢▢▢▢▢▢▢▢▢▢
    res.status(200).json({ status: 'error', message: error.message });
  }
});
```

▢▢▢▢▢▢▢

▢▢▢

- ▢▢ webhook ▢▢▢▢▢▢▢▢▢
- ▢▢▢▢▢ null ▢▢▢

▢▢▢▢▢

1. ▢▢▢▢▢▢▢▢ - ▢▢▢▢▢▢▢▢▢▢▢▢IMS▢▢▢▢

2. □□□□□□ - □□□□□□ webhook □□□□□□□□□□□□□

□□□□□

□□□□□□□□□□□

```javascript
const { subscriber } = req.body;

// □□□□□□□
const imsProfile = subscriber.ims_profile || { name: 'No IMS' };
const roamingProfile = subscriber.roaming_profile || { name: 'No Roaming' };

// □□□□□ MSISDN
const msisdns = subscriber.msisdns || [];
```

□□□□□□□

Webhook □□

□□ webhook □□□□□□□

□□□□□□□□

- Webhook □□□□□□□□□□
- Webhook □□□□□□□□□□□□□□
- □□□□□□
- □□□□□□□
- □□□□□

□□□□□□□（Prometheus/Grafana）

```
# Webhook □□□
rate(omnihss_webhook_success_total[5m]) /
rate(omnihss_webhook_attempts_total[5m])

# Webhook □□
histogram_quantile(0.95, omnihss_webhook_duration_seconds)
```

Webhook □□

□□□□□ webhook □□□□□□□□□□□□□

□□□□□

```
{
  "timestamp": "2025-01-15T14:30:00Z",
  "level": "info",
  "component": "webhook",
  "event_id": "550e8400-e29b-41d4-a716-446655440000",
  "webhook_id": 1,
  "event_type": "update_location_request",
  "subscriber_imsi": "001001123456789",
  "endpoint": "https://your-server.com/omnihss-webhook",
  "http_status": 200,
  "duration_ms": 145,
  "error": null
}
```

OmniHSS □□□□

□□

OmniHSS □□□□ 4G LTE (EPC) □ IMS (IP □□□□□□) □□□□□□□□□□□□□□□ (HSS) □□□□□□□□□□□□□□□□□□□□□□□□□OmniHSS □□

OmniHSS □□ Elixir □ Erlang VM □□

□□□□□□□□□□□□□

HSS □ LTE □ IMS □□□□□□□□□□□□□□□□□

- □□□□□□□ - □□□□□□□□□□□□□□□
- □□□□ - □□□□□□□□□□□□□
- □□□□ - □□□□□□□□□□□□□□
- □□□□ - □□□□□□□□□□□□□□□□
- □□□□ - □□□□□□□□□□□□□□□□□
- □□□□ - □□□□□□□□□□ (EIR) □□□□□□□

□□□□

□□□□

- **S6a □□** - LTE/EPC □□□□□□□□□□□□□□
- **Cx □□** - IMS □□□□□□
- **Sh □□** - IMS □□□□□□□□□□□□
- **S13 □□** - □□□□□□□ (OmniHSS □□ EIR)
- **Gx □□** - □□□□□□□□ (OmniHSS □□ PCRF)
- **Rx □□** - IMS □□□□□□□ (OmniHSS □□ PCRF)
- **□□□□** - □ PLMN □□□□ IMS □□□□□□□□□□

- 分配 MSISDN - 为用户分配电话号码等
- **RESTful API** - 通过标准化 API管理 OmniHLR 功能
- **Web 管理界面** - 用户管理的图形界面

互操作性接口

OmniHSS 支持以下网络接口：

- **MME** (移动管理实体) - LTE 网络的移动管理
- **P-GW** (PDN 网关) - 与 OmniHSS 集成以 (PCRF 功能)
- **P-CSCF** (代理呼叫会话控制功能) - IMS 接入点
- **I-CSCF** (查询 CSCF) - IMS 入口点
- **S-CSCF** (服务 CSCF) - IMS 核心节点
- **AS** (应用服务器) - IMS 应用服务器
- **OmniHLR** - 通过 API 与 OmniHSS 集成的独立 HLR

系统功能

本节介绍系统的主要功能。

核心功能

- 订户管理 - 管理订户及其 Diameter 配置
- 安全管理 - 管理安全配置和密钥
- 系统监控 - 监控系统运行状况

高级功能

- 仪表盘 - 自定义的 Web 管理界面
- 数据可视化 - 数据的图形化展示
- 自动化工作流 - 自动化的业务流程
- **API** 管理 - 管理的 API 访问权限
- **Webhooks** - 基于事件的通知机制

□□□□

- □□□□ - EPC□IMS□APN □□□□□□
- □□□□ - □□□□□□□
- □□□□ - Diameter □□□□□□□□□□
- **PCRF** - □□□□□□□□□□ (Gx/Rx □□□□QoS□VoLTE)
- **EIR** - □□□□□□□ (S13 □□□IMEI □□)
- **□ MSISDN □□ IMSI □□** - □□□□□□□□□□□□ IMSI

□□□□□□

□□□□

□□□□ (Web □□)

```
URL: https://[hostname]:7443
```

□□□□□□□□□□□ Diameter □□□□□□□□□□

API □□

```
URL: https://[hostname]:8443
```

RESTful API □□□□□□□□□□□□□

□□□□□□

- `config/runtime.exs` - □□□□□ (□□□□Diameter□□□□□□)
- `priv/cert/` - HTTPS □ Diameter □ TLS □□

□□□□

1. □□□□□□ - □□□□□□□□□□□
2. **□□□ Diameter □□□** - □□□□□□ Diameter □□

3. □□□□ - □□ API □□ `/api/subscriber/imsi/:imsi`
4. □□□□□ - □□□□□□□□□ SQL □□□

□□□□□□□

□□□□

□□□□□□□□ stdout/stderr□□□□□□□□□□□□ (systemd, supervisord □) □□□

□□□□

- **Diameter** □□□ - □□ Diameter □□□□□□□□□□□
- □□□□□□ - □□ runtime.exs □□□□□□□□
- □□□□□□ - □□□□□□□□□□□□□

□□□□

- **API** □□□□ - `GET /api/status`
- □□□□ - □□□□□□□□□□□
- □□□ - □□□ SQL □□□□□□□□□

□□□□

- □□ **TLS** - API □□□□□□□□ HTTPS
- □□□□ - `priv/cert/` □□□□□□□□□
- □□□□□ - □ runtime.exs □□□□□□□□□
- □□□□ - Diameter □□□□□□□□□
- **API** □□ - □□□□□□□□□□□□

架构概述

关于本

本指南提供有关以下内容的概述和信息：

- 的 **架构概述** 以及它与其他网元
- 安装 **系统要求** 以及准备安装系统
- 安装 **安装指南** 以及系统安装步骤
- 关于 **API 文档** 以及系统集成和参考

文档版本 1.0
版权所有 Omnitouch 网络技术