HTTP API 参考

OmniRoam API 提供 TAP 文件、RAP 文件、CDR 视图、总计、导出控制和指标。此指南列出了端点。

目录

- 访问
- TAP 文件
- TAP 验证
- RAP 文件
- CDR 视图
- CDR 上传
- 总计和监控
- TAP 导出
- 指标

访问

OmniRoam API 监听 4000 端口。该 API 提供普通 HTTP，无需 TLS。所有端点都在 `/api` 路径下。API 返回无信封的 JSON。

资源	路径
Swagger 用户界面	`GET /api/docs`
OpenAPI 架构	`GET /api/schema`

TAP 文件

方法和路径	目的	□数
`GET /api/tap_files_in`	列出入站 TAP 文件。	`tadig`（可选）
`GET /api/tap_files_in/:filename`	以 JSON 格式返回一个入站 TAP 文件。	`tadig`，`page`，`per_page`
`GET /api/tap_files_out`	列出出站 TAP 文件。	`tadig`（可选）
`GET /api/tap_files_out/:filename`	以 JSON 格式返回一个出站 TAP 文件。	`tadig`，`page`，`per_page`

文件响应包含头部和一页通话事件。`page` 值从 `1` 开始。`per_page` 值的默认值为 `100`，最大值为 `2000`。当文件未找到时，响应返回 `404`。

TAP 验证

方法和路径	目的	□数
`GET /api/tap_validate_in/:filename`	验证一个入站 TAP 文件。	`tadig`（可选）
`GET /api/tap_validate_out/:filename`	验证一个出站 TAP 文件。	`tadig`（可选）

响应报告文件是否有效、按严重性分类的发现数量以及发现列表。请参见 TAP 验证。

RAP 文件

方法和路径	目的	□数或主体
`GET /api/rap_files`	列出 RAP 文件。	`tadig` (可选)
`GET /api/rap_files/:filename`	以增强的 JSON 格式返回一个 RAP 文件。	`tadig` (可选)
`POST /api/rap_files/send_alert`	发送 RAP 文件的电子邮件警报。	主体: `filename` (必需)，`tadig`

请参见 RAP 处理。

CDR 视图

方法和路径	目的	□数
`GET /api/part_cdrs/:page`	列出来自 Redis 的正在进行的部分 CDR 键。	路径: `page` (整数)
`GET /api/cdrs_from_csv`	列出原始 CDR 文件或下载一个。	`filename`, `raw_cdr_view_max_age`
`GET /api/config_yaml`	返回 `config.yaml`。API 移除 InfluxDB 令牌。	无
`GET /api/config`	返回 `config` 部分和计数器。	无

CDR 上传

方法和路径	目的	主体
`POST /api/ui_upload`	将 CSV CDR 文件上传到 OmniCore 目录。	多部分 `file`。名称必须是以 `.csv` 结尾的裸文件名——无路径分隔符。返回 400（拒绝）、409（已持有）、413（过大）。

响应在成功时返回 `200`，在文件重复时返回 `409`，在文件无效时返回 `500`。请参见 CDR 处理。

总计和监控

方法和路径	目的	□数
`GET /api/monitoring`	返回监控总计，或下载匹配的文件。	`filename`（可选）
`GET /api/totals_monthly`	列出具有每月总计的合作伙伴。	无
`GET /api/totals_monthly/:tadig`	返回某个 TADIG 的每月总计。	`year`，`month`
`GET /api/totals_daily`	返回所有 TADIG 的每日总计。	无
`GET /api/totals_daily/:tadig`	返回某个 TADIG 的每日总计。	无

TAP 导出

��法和路径	目的	主体
`POST /api/tap_export`	为一个合作伙伴排队导出。	`partner`（必需），`records_per_file`，`max_files`
`GET /api/tap_export`	列出导出作业。	无
`GET /api/tap_export/:job_id`	返回一个导出作业。	无
`GET /api/export_schedule`	返回调度程序状态和队列状态。	无
`POST /api/export_schedule`	立即为每个合作伙伴排队导出。	无

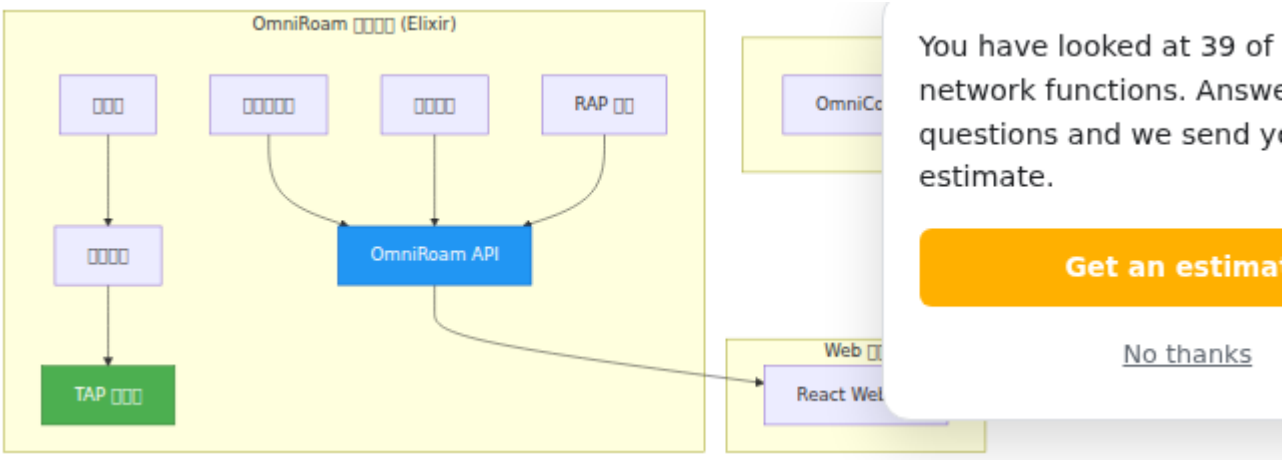
导出端点在 OmniRoam 排队作业时返回 `202`。作业队列一次运行一个导出。请参见 操作指南。

指标

方法和路径	目的
`GET /api/metrics`	返回 Prometheus 指标。

响应是 Prometheus 文本展现。抓取读取缓存快照。请参见 指标参考。

- 概述



架构

本指南描述了 OmniRoam 组件、端到端数据流以及 OmniRoam 集成的外部系统。

目录

- 组件概述
- 可部署包
- 数据流
  - 摄取和计费
  - TAP 文件生成
  - 文件传输
  - RAP 返回
- 集成点

组件概述

OmniRoam 有两个部分。一组 Python 工具用于摄取和计费 CDR。一个 Elixir 应用程序生成 TAP 文件、处理 RAP 文件，并□供 API 和指标。

摄取和计费 (Python)

这些工具作为定时作业运行。它们并不作为长期服务运行。

- **OmniCore** 导入器 - 读取 OmniCore 服务网关 CSV 文件，将部分记录关联成完整数据会话，并将会话写入 Redis。
- **ASN.1** 导入器 - 将 3GPP ASN.1 CDR 文件转换为 XML 并导入。
- **Redis** 导入器 - 清空 Redis 队列，应用年龄过滤，四舍五入使用量，并将每条记录发送到计费引擎进行计费。
- 计费配置器 - 从 `config.yaml` 加载每个合作伙伴的计费计划到计费引擎。

OmniRoam 应用程序 (Elixir)

OmniRoam 应用程序作为 systemd 服务运行。它启动一组受监督的进程。

进程	角色
文件索引	缓存解析的 TAP 和 RAP 文件元数据以快速列出。
总计缓存	缓存每个合作伙伴的每月和每日总计。
指标收集器	在后台构建 Prometheus 指标快照。
作业队列	一次运行一个 TAP 导出。
调度器	在配置的时间启动每日导出。
OmniRoam API	在 4000 端口供 HTTP API。
RAP 监控	扫描新 RAP 文件并发送警报。
序列	保留 TAP ���件序列号并写入 `counters.yaml`。
空格清理器	清理数据库中计费 CDR 字段的多余空格。

TAP 生成器读取已计费的 CDR，将每个 CDR 映射到 TAP 记录，组装批次，将批次编码为二进制 TAP 文件，并将 CDR 标记为已处理。有关详细信息，请参见 TAP 文件生成。

Web 界面

Web 界面是一个 React 单页应用程序。它由 nginx 作为静态内容供。Web 界面通过 4000 端口调用 OmniRoam API。

注册器

注册器是一个可选的出站连接器。注册器打开一个 WebSocket 连接到管理系统，并将该系统的请求转发到本地 OmniRoam API。使用注册器可以在 NAT 后面访问 API，而无需入站端口。

可部署包

OmniRoam 作为一组 Debian 包发布。`omniroam` 元包依赖于组件包并锁定它们的版本。

包	内容	运行时
`omniroam`	元包。依赖于组件包。	无。
`omniroam-api`	Elixir 应用程序发布。	4000 端口上的 systemd 服务，用户 `omniroam`。
`omniroam-generator`	Python 摄取和计费工具及配置。	无服务。由 cron 或手动运行。
`omniroam-ui`	静态 Web 界面和 nginx 站点。	80 端口上的 nginx。
`omniroam-registrar`	出站注册器连接器。	systemd 服务。

数据流

一个 CDR 经过四个阶段：摄取和计费、TAP 文件生成、文件传输和 RAP 返回。

摄取和计费

两个 CDR 源为一个管道供数据。

1. **OmniCore** 服务网关 发送 CSV CDR 文件。文件通过 rsync 或 HTTP 上传到 API。OmniRoam 将文件写入本地目录。
2. **3GPP ASN.1 CDR** 源 供 ASN.1 CDR 文件。OmniRoam 拉取文件并将其转换为 XML。
3. 导入器关联部分记录，将完整数据会话写入 Redis，然后将 Redis 的数据传输到计费引擎进行计费。

已计费的 CDR 存储在计费引擎数据库中（CDR 表）。每个 CDR 都带有一个 `hasBeenProcessed` 标记。值为 `False` 表示 CDR 仍需要一个 TAP 文件。有关摄取和计费的详细信息，请参见 CDR 处理。

TAP 文件生成

调度器或 API 请求将导出作业放入队列。作业队列一次运行一个导出。生成器读取合作伙伴的未处理 CDR，从最旧的开始，并写入一个或多个 TAP 文件。然后生成器将 CDR 标记为已处理并记录文件序列号。有关详细信息，请参见 TAP 文件生成。

文件传输

当作业队列清空时，OmniRoam 运行上传命令。上传命令将新的 TAP 文件发送到清算中心。一个单独的拉取作业检索入站 TAP 和 RAP 文件。有关传输作业的详细信息，请参见 操作指南。

RAP 返回

合作伙伴验证每个 TAP 文件。如果合作伙伴拒绝记录，合作伙伴将返回一个 RAP 文件。拉取作业将 RAP 文件放入入站目录。RAP 监控扫描该目录，读取每个新 RAP 文件并发送电子邮件警报。有关详细信息，请参见 RAP 处理。

集成点

系统	协议	目的
Redis	本地 Redis	计费前的关联部分 CDR 队列。
计费引擎	2080 端口上的 JSON-RPC	对 CDR 进行计费并返回费用。
CDR 数据库	3306 端口上的数据库	存储已计费的 CDR 并标记为已处理。
InfluxDB	HTTP API	存储使用和导出分析。
清算中心	SCP	发送 TAP 文件并拉取入站 TAP 和 RAP 文件。
邮件□供商	HTTPS	发送 RAP 警报电子邮件。
Prometheus	在 `/api/metrics` 上的 HTTP 抓取	收集 OmniRoam 指标。

有关端点的详细信息，请参见 HTTP API 参考，有关连接设置的详细信息，请参见 配置参考。

- 概述

CDR 处理

本指南描述了 OmniRoam 如何摄取通话详单记录（CDR），对其进行计费，并存储以生成 TAP 文件。

目录

- CDR 来源
- 摄取管道
- 年龄过滤器
- 使用量四舍五入
- 计费
- 存储

CDR 来源

OmniRoam 从两种来源类型摄取 CDR。

来源	格式	传输	登陆目录
OmniCore 服务网关	CSV	rsync，或通过 API 的 HTTP 上传	`/var/OmniCore_CDRs`
3GPP ASN.1 CDR 来源	ASN.1	rclone	`/var/asn1_cdrs`

服务网关还可以通过 `POST /api/ui_upload` 请求将 CSV 文件直接发送到 API。OmniRoam 将上传的文件保存到 OmniCore 目录。请参见 HTTP API 参考。

摄取管道

摄取工具作为计划任务运行。请参见 操作指南 以获取计划。

1. 关联。单个数据会话可以生成多个部分 CDR。导入程序将一个会话的部分记录合并为一个完整记录。
2. 队列。导入程序将每个完整记录写入 Redis。
3. 过滤。 Redis 导入程序排空队列，并删除超出年龄窗口的记录。导入程序还会删除使用量为零的记录。
4. 四舍五入。导入程序对使用量进行四舍五入。请参见 使用量四舍五入。
5. 计费。导入程序将每个记录发送到计费引擎进行定价。请参见 计费。
6. 存储。计费引擎将计费后的 CDR 存储在 CDR 表中。导入程序还将使用点写入 InfluxDB。

年龄过滤器

Redis 导入程序在对记录进行计费之前应用两个年龄过滤器。

过滤器	操作
超过 30 天	导入程序删除该记录。TAP 文件不能包含超过 30 天的 CDR。
少于 24 小时	导入程序延迟该记录。导入程序将在稍后的运行中处理该记录。

30 天的限制来自 GSMA 对 TAP 文件的规定。及时摄取和计费 CDR，否则 30 天的限制将使其无法结算。

使用量四舍五入

OmniRoam 将每个 CDR 的使用量四舍五入到下一个 `round_up_to` 字节的倍数。`round_up_to` 值是按合作伙伴设置的。请参见 配置参考。

OmniRoam 应用两种独立的四舍五入。

1. 按方向四舍五入。 OmniRoam 将传入量和传出量分别四舍五入到下一个倍数。
2. 计费单位。 OmniRoam 将原始传入和传出量相加，然后将该总量四舍五入到下一个倍数。这个四舍五入的总量就是计费使用量，计费引擎会对其进行定价。

OmniRoam 保留原始的未四舍五入的量以供审计。

示例。一个 CDR 有 500 字节的传入和 700 字节的传出。合作伙伴使用 `round_up_to: 1024`。

- 按方向：传入四舍五入到 1024，传出四舍五入到 1024。
- 计费单位：总量为 1200 字节，四舍五入到 2048 字节。

计费

OmniRoam 使用计费引擎对每个 CDR 进行定价。计费发生在摄取过程中，而不是在 TAP 文件生成期间。

1. 费率□供者将每个合作伙伴的费率加载到计费引擎中。费率来自合作伙伴配置中的 `unit_price` 和 `unit_bytes`。
2. 导入程序将每个 CDR 发送到计费引擎，并附上计费使用量。
3. 计费引擎返��费用。OmniRoam 将费用与 CDR 一起存储。

要更改合作伙伴的价格，请在 配置参考 中更新 `unit_price` 或 `unit_bytes`，然后将费率重新加载到计费引擎中。

存储

OmniRoam 在计费引擎数据库中存储一个计费后的 CDR。每个 CDR 都携带一个 `hasBeenProcessed` 标记。

标记值	含义
`False`	CDR 已计费但尚未在 TAP 文件中。
文件名	CDR 在命名的 TAP 文件中。

生成器读取 `hasBeenProcessed` 设置为 `False` 的 CDR，按最旧的优先。请参见 TAP 文件生成。

OmniRoam 还将使用分析写入 InfluxDB。请参见 指标参考。

- 概述

配置参考

本指南记录了 OmniRoam 配置文件中的每个参数。OmniRoam 读取两个文件：

- `config.yaml` - 主要配置。它有一个全局 `config` 部分和一个每个合作伙伴的 `partners` 部分。
- `counters.yaml` - TAP 文件序列计数器。

Python 工具从工作目录读取 `config.yaml`。Elixir 应用程序从 `OMNIROAM_CONFIG_PATH` 环境变量中的路径读取 `config.yaml`，或从 `/etc/pytap3/config.yaml` 读取。OmniRoam 从与 `config.yaml` 相同的目录读取 `counters.yaml`。

目录

- 配置结构
- 全局配置
  - TAC 配置
  - 家庭 PLMN
  - 处理器身份和时区
  - 文件路径
  - 原始 CDR 视图
  - InfluxDB 连接
  - 日志记录
  - 导出计划
  - 导出完成命令
  - Webhooks
  - 每个文件的记录数
- 合作伙伴配置
  - IMSI 前缀
  - 接入点名称
  - 费率
  - 使用四舍五入
  - 批次信息
  - 会计信息

    - 呼叫类型级别
- 序列计数器
- 已弃用的键

配置结构

```
config:
  # 全局系统配置

partners:
  # 每个合作伙伴的配置
```

全局配置

TAC 配置

`config.tac_config` 将跟踪区域代码 (TAC) 映射到服务位置。OmniRoam 使用服务位置在每个 TAP 记录中设置计费标识符和时区。每个条目都是一个命名模板，包含 TAC 列表。

```
config:
  tac_config:
    ExampleNetwork:
      tac_list: ['1101', '10000', '10100']
      servingBid: 12345
      servingLocationDescription: '示例网络'
      timezone: 'Etc/UTC'
```

□数	类型	必需	默认值	描述
`tac_list`	字符串列表	是	-	与此模板匹配的 TAC 值。OmniRoam 将 CDR 的 TAC 与此列表进行匹配。
`servingBid`	整数	否	-	写入 TAP 记录的 5 位计费标识符。省略以不写入该字段。
`servingLocationDescription`	字符串	否	-	写入 TAP 记录的��置描述。
`timezone`	字符串	是	-	IANA 时区名称。OmniRoam 使用此时区计算匹配 CDR 的 UTC 偏移。

家庭 PLMN

`config.home_plmns` 列出家庭 PLMN 前缀。OmniRoam 将 IMSI 以家庭 PLMN 前缀开头的 CDR 视为家庭用户，而不是漫游用户。摄取工具读取此键。

```
config:
  home_plmns:
    - '001010'
```

□数	类型	必需	默认值	描述
`home_plmns`	字符串列表	是	-	识别家庭用户的 IMSI 前缀（前 6 位数字）。

处理器身份和时区

```
config:
  cdr_processor_tz: America/New_York
```

□数	类型	必需	默认值	描述
`cdr_processor_tz`	字符串	是	-	默认 IANA 时区。OmniRoam 使用此时区用于文件时间戳和不匹配任何 TAC 模板的 CDR。

文件路径

```
config:
  cdr_search_paths:
    - '/var/asn1_cdrs/'
    - '/var/OmniCore_CDRs/'
  tap_output_path: '/etc/pytap3/OutputFiles'
  tap_human_readable_output_path: '/etc/pytap3/OutputFiles_Human'
  tap_in_path: '/etc/pytap3/TAP_In'
  roaming_in_filename: False
```

□□	□□	□□	□□□	
`cdr_search_paths`	□□□□□□	□	-	O □□ □□
`tap_output_path`	□□□□	□	`/etc/pytap3/OutputFiles`	□□ □□ O □ □□ □□
`tap_human_readable_output_path`	□□□□	□	-	□□ □□ □□
`tap_in_path`	□□□□	□	`/etc/pytap3/TAP_In`	□□ □□
`roaming_in_filename`	□□□□	□	`False`	□□ □□ □□ "r □□ □□

□□ CDR □□

```yaml
config:
  raw_cdr_view_max_age: 7
```

□数	类型	必需	默认值	描述
`raw_cdr_view_max_age`	整数	否	-	Web 界面显示的原始 CDR 的最大天数。请求可以通过查询参数覆盖此值。

InfluxDB 连接

`config.influx_db` 设置 InfluxDB 连接以进行使用和导出分析。所有四个值必须存在，否则 OmniRoam 将跳过指标写入。有关测量，请参见 Metrics Reference。

```yaml
config:
  influx_db:
    influxDbUrl: 'http://influxdb.example.com:8086'
    influxDbOrg: 'example-org'
    influxDbBucket: 'tap3'
    influxDbToken: '<influx-token>'
```

□数	类型	必需	默认值	描述
`influxDbUrl`	字符串	是	-	InfluxDB 基本 URL。
`influxDbOrg`	字符串	是	-	InfluxDB 组织。
`influxDbBucket`	字符串	是	-	InfluxDB 存储桶。
`influxDbToken`	字符串	是	-	InfluxDB API 令牌。API 从 `config_yaml` 响应中删除此值。

将 `influxDbToken` 设置为您自己的令牌。请勿将真实令牌□交到版本控制中。

日志记录

```yaml
config:
  log_level: DEBUG
```

□数	类型	必需	默认值	描述
`log_level`	字符串	是	-	Python 工具的日志级别。使用 `DEBUG`、`INFO`、`WARNING` 或 `ERROR` 之一。

导出计划

`config.export_schedule` 控制每日自动导出。调度程序在每次滴答时读取此部分，因此更改会立即生效，无需重启。有关调度程序行为，请参见 Operations Guide。

```yaml
config:
  export_schedule:
    enabled: false
    at: '02:00'
    partners: all
```

□数	类型	必需	默认值	描述
`enabled`	布尔值	否	`false`	每日导出的主开关。设置为 `true` 以自动运行导出。
`at`	字符串	否	-	每日运行时间，格式为 `HH:MM`，采用 UTC。
`partners`	字符串或列表	否	`all`	要导出的合作伙伴。使用 `all` 表示每个合作伙伴，或列出合作伙伴名称。

导出完成命令

`config.on_export_complete` 设置 OmniRoam 在导出作业队列清空时运行的命令。使用此命令将新 TAP 文件发送到清算中心。

```
config:
  on_export_complete: '/etc/pytap3/smart_tap_upload.sh'

  # 首选：每个待处理 TAP 文件调用一次，路径作为参数
  # 并在环境中包含 TAP_FILE / TAP_FILENAME / TAP_TADIG / TAP_SENT_DIR。
  # 退出 0，OmniRoam 将文件移动到 OutputFiles_Sent；任何其他状态
  # 将其保留为待处理，以便下次重试。脚本不进行通配符匹配、移动文件，
  # 或跟踪部分失败——因此通常只有十行。优先于
  # on_export_complete，后者保留用于尚未迁移的部署。
  on_tap_file: '/etc/pytap3/send_one.sh'

  # 入站：按计划运行站点的提取脚本，而不是通过 cron。
  # 在环境中以 TAP_IN_DIR 和 RAP_IN_DIR 调用；退出 0 表示提取完成
  # （无论是否找到任何内容）。运行在自己的锁上序列化，失败计数，
  # tap3_inbound_newest_age_seconds 会使停止响应的清算中心发出警报，而不是安静
  # 一周。未设置则不运行
  on_fetch_files: '/etc/pytap3/pull_tap.sh'
  fetch_interval_minutes: 60
```

数	类型	必需	默认值	描述
`on_export_complete`	字符串	否	-	导出完成后要运行的 shell 命令。留空以禁用。

Webhooks

`config.webhooks` 设置 Slack 传入 webhook URL。当导出完成或失败时，OmniRoam 会发布消息。

```
config:
  webhooks:
    on_complete: '<slack-webhook-url>'
    on_failure: '<slack-webhook-url>'
```

□数	类型	必需	默认值	描述
`on_complete`	字符串	否	-	成功导出的 Slack webhook URL。留空以禁用。
`on_failure`	字符串	否	-	失败导出的 Slack webhook URL。留空以禁用。

每个文件的记录数

`config.records_per_tap_file` 设置一个 TAP 文件中 CDR 的数量上限。每个合作伙伴的值和每个请求的值可以覆盖此全局值。当合作伙伴的未处理 CDR 数量超过限制时，OmniRoam 会写入多个 TAP 文件。

```
config:
  records_per_tap_file: 140000
```

□数	类型	必需	默认值	描述
`records_per_tap_file`	整数	否	-	每个 TAP 文件的最大 CDR 数量。

合作伙伴��置

`partners` 为每个漫游合作伙伴保存一个条目。条目名称是您选择的标签。摄取工具使用 `imsi_prefixes` 将 CDR 分配给合作伙伴。生成器使用合作伙伴名称选择要导出的合作伙伴。

```yaml
partners:
  ExamplePartner_Live:
    imsi_prefixes:
      - '001020'
    accessPointNameOI: mnc020.mcc001.gprs
    rates:
      unit_price: 0.000476800
      unit_bytes: 1024
    round_up_to: 1024
    batch_info:
      sender: AAAAA
      recipient: BBBBB
      specificationVersionNumber: 3
      releaseVersionNumber: 12
      accountingInfo:
        localCurrency: 'USD'
        tapCurrency: 'USD'
        roundingAction: 'Simple'
        tapDecimalPlaces: 5
    call_type_level:
      qci_1: 21
      default: 20
```

IMSI 前缀

□数	类型	必需	默认值	描述
`imsi_prefixes`	字符串列表	是	-	将 CDR 映射到此合作伙伴的 IMSI 前缀。摄取工具使用最长匹配。

接入点名称

□数	类型	必需	默认值	描述
`accessPointNameOI`	字符串	是	-	接入点名称运营商标识符。OmniRoam 将此值写入每个 TAP GPRS 记录。

费率

`rates` 设置 OmniRoam 加载到计费引擎中的此合作��伴的价格。计费引擎根据此费率对每个 CDR 进行定价。有关计费流程，请参见 CDR Processing。

□数	类型	必需	默认值	描述
`unit_price`	浮点数	是	-	每个 `unit_bytes` 使用的价格，以当地货币计。
`unit_bytes`	整数	是	-	一个费率单位中的字节数。

使用四舍五入

□数	类型	必需	默认值	描述
`round_up_to`	整数	是	-	以字节为单位的四舍五入单位。OmniRoam 将使用量四舍五入到此值的下一个倍数。有关详细信息，请参见 CDR Processing。

批次信息

`batch_info` 设置此合作伙伴每个 TAP 文件的身份。

□数	类型	必需	默认值	描述
`sender`	字符串	是	-	您的 5 字符 TADIG 代码。OmniRoam 将此值写入 `batchControlInfo.sender` 和文件名。
`recipient`	字符串	是	-	合作伙伴的 5 字符 TADIG 代码。OmniRoam 使用此值作为输出子目录、文件名和序列计数器键。
`specificationVersionNumber`	整数	是	`3`	TAP 规范版本。使用 `3` 表示 TAP 3.12。
`releaseVersionNumber`	整数	是	`12`	TAP 发布版本。使用 `12` 表示 TAP 3.12。
`fileTypeIndicator`	字符串	否	-	当存在时，OmniRoam 生成测试文件。有关详细信息，请参见 TAP File Generation。省略此键以生成实时文件。

会计信息

`accountingInfo` 设置每个 TAP 文件的货币和货币格式。

□数	类型	必需	默认值	描述
`localCurrency`	字符串	是	-	作为 ISO 4217 代码的当地货币。
`tapCurrency`	字符串	是	-	作为 ISO 4217 代码的结算货币。
`roundingAction`	字符串	是	-	货币四舍五入模式。使用 `Up`、`Down` 或 `Simple`。
`tapDecimalPlaces`	整数	是	-	TAP 货币值的小数位数。OmniRoam 将收费值缩放为 1000，因此将此值设置为 `3`。有关详细信息，请参见 TAP File Generation。

呼叫类型级别

`call_type_level` 将 CDR 的 QoS 类标识符 (QCI) 映射到 TAP 呼叫类型级别。OmniRoam 查找 CDR 的 QCI 对应的 `qci_<n>` 键。如果缺少该键，OmniRoam 将使用 `default`。

```
call_type_level:
  qci_1: 21
  qci_2: 22
  default: 20
```

□数	类型	必需	默认值	描述
`qci_1` 到 `qci_9`	整数	否	-	每个 QCI 值的 TAP 呼叫类型级别。
`default`	整数	是	-	对于没有 `qci_<n>` 键的 QCI 的 TAP 呼叫类型级别。

序列计数器

`counters.yaml` 保存 TAP 文件序列号。OmniRoam 根据 接收方 **TADIG** 为文件键入，每种文件类型有一个单独的计数器。有关计数器行为，请参见 TAP File Generation。

```yaml
BBBBB:
  CD: 11
  TD: 50
CCCCC:
  CD: 3
  TD: 2
```

键	描述
顶级键	5 字符接收方 TADIG 代码。
`CD`	实时（可计费数据）文件的下一个序列号。
`TD`	测试（测试数据）文件的下一个序列号。

注意：

- 使用 `CD: 1` 和 `TD: 1` 初始化新合作伙伴。
- OmniRoam 在写入文件后递增计数器。
- 序列号是 5 位字段。在达到 `99999` 之前重置计数器。有关详细信息，请参见 Operations Guide。
- 备份 `counters.yaml` 和 `config.yaml`，以便在恢复后序列可以正确继续。

已弃用的键

请勿使用这些键。OmniRoam 不再在生产路径中读取它们。

键	状态
`config.exporters`	已弃用`??`当前应用程序不读取此键。
`config.cdr_processor_name`	已弃用。当前应用程序不读取此键。

- 概述

指标□考

OmniRoam □供两个指标来源。OmniRoam API □供服务状态的 Prometheus 指标。数据摄取和导出工具将使用分析写入 InfluxDB。

目录

- Prometheus 指标
- 服务状态
- 示例查询
- InfluxDB 测量

Prometheus 指标

OmniRoam API 在 `GET /api/metrics` □供 Prometheus 指标。抓取读取缓存快照，因此抓取不会运行任何数据库查询。指标收集器在后台刷新快照，大约每分钟一次。

所有 OmniRoam 指标都是仪表。

指标	标签	描述
`tap3_exporter_up`	-	收集器有一个新的快照。`1` 表示是。
`tap3_service_up`	`service`	服务正在运行。`1` 表示正常，`0` 表示故障。
`tap3_cgrates_up`	-	Charging Engine API 正在响应。
`tap3_cgrates_goroutines`	-	Charging Engine 工作例程的数量。
`tap3_cdr_total`	-	数据库中 CDR 的总数。
`tap3_files_pending`	-	等待上传的文件数量。
`tap3_files_sent`	-	已发送的文件数量。
`tap3_files_total`	-	TAP 文件的总数。
`tap3_files_generated_today`	-	今天生成的文件数量。
`tap3_last_file_generated_timestamp`	-	最后一个文件生成的 Unix 时间。
`tap3_last_file_sent_timestamp`	-	最后一个文件发送的 Unix 时间。
`tap3_partner_files_pending`	`partner`	每个合作伙伴的待处理文件数量。
`tap3_partner_files_sent`	`partner`	每个合作伙伴的已发送文件数量。
`tap3_upload_status`	-	最后一次上传状态。`1` 表示成功，`0` 表示失败，`-1` 表示未知。
`tap3_hours_since_last_file`	-	自最后一个 TAP 文件以来的小时数。

指标 `tap3_cdr_total` 读取一个近似的行计数。收集器不会运行完整的表计数，因为在大型 CDR 表上进行完整计数会使导出数据库池耗尽。

服务状态

`tap3_service_up` 指标为每个依赖项报告一个系列。收集器使用系统服务管理器检查每个服务。

服务标签	组件
`omniroam-api`	OmniRoam API。
`cgrates`	Charging Engine（标签字面上发出）。
`mysql`	CDR 数据库（标签字面上发出）。
`redis-server`	数据摄取队列��?
`influxdb`	分析数据库。

示例查询

```
# 等待上传的 TAP 文件
tap3_files_pending

# 自最后一个 TAP 文件以来的小时数，超过时发出警报
tap3_hours_since_last_file > 26

# 有待处理文件的合作伙伴
tap3_partner_files_pending > 0

# 处于故障状态的依赖项
tap3_service_up == 0
```

InfluxDB 测量

数据摄取和导出工具将四个测量写入 InfluxDB。在 配置参考 中设置连接。

测量	写入时间	标签	字段
`raw_cdr`	每个 CDR 一次，评级后。	`operator`, `input_file`, `apn`, `cellId`, `imsi`, `tac`, `sGWAddress`, `pGWAddress`	`chargeableUnits` `chargedUnits`
`omnicore_cdr_part`	在 OmniCore 数据摄取期间每个部分 CDR 一次。	`accessPointNameNI`, `hpmln`, `sgw_ip`, `tac`	`dataVolumeIncom` `dataVolumeOutgc`
`file_processing`	每处理一个 OmniCore 文件一次。	`individual_filename`	`no_imsi_count`, `sgw_cdr_count`, `pgw_cdr_count`, `charging_id_mis` `roaming_sgw_cdr` `empty_record`, `processing_time`
`tap_cdr`	每导出一个 TAP 文件一次。	`operator`, `filename`	`totalcharge`, `totalconsumed`, `cdr_count`

`raw_cdr` 测量具有高基数，因为标签包括 IMSI、单元标识符和输入文件名。管理 InfluxDB 保留策略以控制存储增长。请参阅 操作指南。

- 概述

操作指南

本指南描述了 OmniRoam 的定时作业、文件传输、归档、日志位置和常规维护。

目录

- 定时作业概述
- CDR 导入
- 定时 TAP 导出
- 文件传输
- 归档
- 序列计数器维护
- 存储维护
- 日志位置

定时作业概述

OmniRoam 在每日周期内运行。下表显示了主要作业及其顺序。

作业	运行时间	目的
CDR 导入	由 cron 定时	收集、关联和评分 CDR。
TAP 导出	每天，在配置的时间	为每个合作伙伴生成 TAP 文件。
上传	每次导出后	将新的 TAP 文件发送到清算所。
入站拉取	每天，以及每周的完整运行	拉取入站的 TAP 和 RAP 文件。
RAP 监控	每 4 小时	向操作员发出新 RAP 文件的警报。
归档	每月	���数据库中转储和清除旧的 CDR。

CDR 导入

导入作业从服务网关收集 CDR，关联它们并进行评分。该作业按顺序运行 Python 工具。

1. 拉取 3GPP ASN.1 CDR 文件，并删除本地超过 60 天的文件。
2. 将 3GPP ASN.1 文件转换为 XML，并导入它们。
3. 拉取 OmniCore 服务网关的 CSV 文件。
4. 关联 OmniCore 记录，并将其写入 Redis。
5. 清空 Redis，评分每个 CDR，并存储结果。

有关管道详细信息和年龄过滤器，请参见 CDR 处理。

定时 TAP 导出

调度程序启动每日导出。调度程序在每次滴答时读取 `config.export_schedule`，因此更改会立即生效，无需重启。请参阅 配置参考。

- 当 `enabled` 为 `true`，当前 UTC 时间达到 `at`，并且今天尚未运行导出时，调度程序会运行。
- 调度程序为每个目标合作伙伴排队一个导出作业。
- 作业队列一次运行一个导出。此限制保护共享数据库池。
- 当队列清空时，OmniRoam 会运行 `on_export_complete` 中的命令。
- OmniRoam 在完成或失败时向配置的 webhook 发送 Slack 消息。

调度程序在启动时不会运行额外的导出。如果服务在每日时间之后启动，则第一次导出将在第二天运行。

要立即运行导出，请使用 API。请参阅 HTTP API 参考。

文件传输

OmniRoam 在本地存储和清算所之间移动文件。

上传

OmniRoam 在每次导出后通过 `on_export_complete` 运行上传命令。上传命令将新的 TAP 文件发送到清算所，并将每个发送的文件移动到 `OutputFiles_Sent` 目录。

□□□

存储维护

监控这些存储的增长。

存储	操作
`OutputFiles`	确认上传将每个发送的文件移动到 `OutputFiles_Sent`。不断增长的待处理计数意味着上传失败。监视 `tap3_files_pending` 和 `tap3_hours_since_last_file`。
InfluxDB	设置保留策略。`raw_cdr` 测量具有高基数。
CDR 数据库	按计划运行归档作业。
日志文件	Python 工具在 `/tmp` 中每次运行写入一个带时间戳的日志文件。删除旧的日志文件。

有关监控指标，请参见 指标参考。

日志位置

Python 工具每次运行写入一个带时间戳的日志文件。

文件模式	来源
`/tmp/import_CDR_Logger_Marben_*.log`	ASN.1 和 Redis 导入器。
`/tmp/import_CDR_Logger_OmniCore_*.log`	OmniCore 导入器。
`/tmp/tap3_export_*.log`	传统导出工具。
`/tmp/convert_cdr_logger_*.log`	3GPP ASN.1 到 XML 转换器。
`/tmp/rate_creator_Logger_*.log`	费率□供者。

OmniRoam 应用程序通过系统服务管理器记录日志。上传脚本将其状态写入一个日志文件，指标收集器会读取该文件。

- □□

RAP 处理

本指南描述了合作伙伴返回的已退回账户程序 (RAP) 文件，以及 OmniRoam 如何向操作员发出警报。

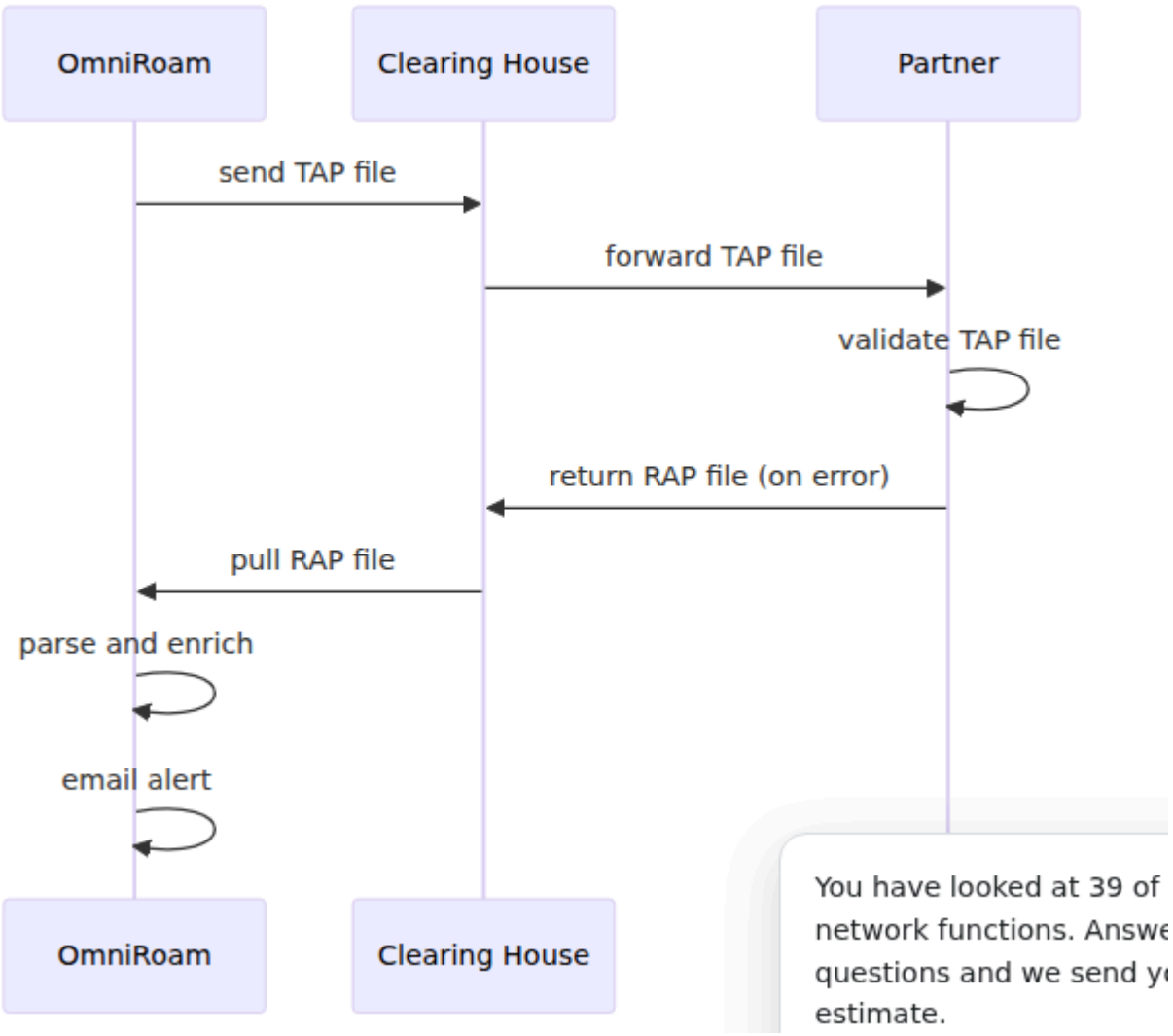
目录

- 什么是 RAP 文件
- RAP 文件命名
- 返回类型和严重性
- 错误代码
- RAP 监控和警报
- 接收和确认 RAP 文件
- 重新□交已退回的 TAP 文件
- 读取 RAP 文件

什么是 RAP 文件

合作伙伴验证 OmniRoam 发送的每个 TAP 文件。如果合作伙伴发现错误，则返回 RAP 文件。RAP 文件报告特定 TAP 文件中的错误。RAP 格式遵循 GSMA TD.32。

RAP 文件是信使。错误代码描述原始 TAP 文件中的故障。OmniRoam 读取 RAP 文件，为每个错误添加描述，并向操作员发出警报。



RAP 文件命名

RAP 文件名的固定长度为 17 个字符。该名称与 TAP 文件名相似，但使用 RAP 前缀。

前缀	意义
RC	针对实时（可收费数据）TAP 文件的返回。
RT	针对测试（测试数据）TAP 文件的返回。

名称的其余部分包含 5 个字符的发送者、5 个字符的接收者和 5 位数字的序列号。OmniRoam 仅读取以 RC 或 RT 开头的文件作为 RAP 文件。

返回类型和严重性

RAP 文件包含一个或多个返回。每个返回都有一个类型和一个严重性。

返回类型	严重性	影响
致命返回	严重	合作伙伴拒绝整个 TAP 文件。
严重返回	警告	合作伙伴拒绝单个记录，但接受文件的其余部分。
缺失返回	警告	合作伙伴报告文件序列中的间隙。
停止返回	信息	合作伙伴报告在 7 天或更长时间内未收到 TAP 文件。

首先处理致命返回。致命返回意味着合作伙伴丢弃了整个文件，您必须纠正故障并重新发送。

��误代码

OmniRoam 为 RAP 文件中的每个错误代码添加描述。这些代码来自 GSMA TD.57。路径标识符来自 TD.57 的原始 TAP 文件，以及来自 TD.32 的 RAP 文件。

下表显示了常见错误代码。有关类别，请参见 TAP 验证。

代码	描述
20	缺少必需的数据项。
50	ASN.1 结构中的 RAP 编码错误。
107	无效结构。
200	数据与漫游协议不一致。
250 到 257	重复呼叫事件。
260	呼叫年龄错误。
263	无效的时间戳格式。

RAP 监控和警报

RAP 监视器是 OmniRoam 应用程序中的后台进程。监视器监视入站目录，并向操作员发出新 RAP 文件的警报。

属性	值
扫描间隔	每 4 小时。
目录	入站 TAP 目录和 RAP 目录。RAP 文件与入站 TAP 文件一起到达。
警报方式	通过配置的电子邮件□供商发送电子邮件。
去重	OmniRoam 记录每个警报文件。OmniRoam 不会对同一文件发出两次警报。

电子邮件警报命名返回类型、错误计数和文件名。警报还包括批次信息和错误代码。

在应用程序运行时配置中设置警报接收者和电子邮件□供商凭据��请勿将凭据存储在 `config.yaml` 中。

要发送已知文件的测试警报，请使用 `POST /api/rap_files/send_alert`。请参见 HTTP API 参考。

接收和确认 RAP 文件

当 `config.inbound_check` 设置并启用 `exporter` 角色时，OmniRoam 会接收入站 RAP 目录中的每个 RAP 文件和确认，一旦它在 `inbound_check.settle_seconds` 内没有更改。该目录每 `inbound_check.interval_seconds` 扫描一次。

文件	**OmniRoam** 的操作
`RC` 或 `RT` 指向 `inbound_check.home_tadig`	保留 RAP 返回的内容：每个致命、缺失和停止返回，以及每个 TAP 文件的严重返回，包括记录计数、值和错误代码。将 TD.32 要求的 RAP 确认写入 `rap_output_path` 中合作伙伴的文件夹，格式为 `AC` 或 `AT`，后跟 RAP 的序列号。
`AC` 或 `AT` 对于我们发送的 RAP	标记该 RAP 已确认。

无法确认的 RAP，例如因为未设置 `rap_output_path`，将在下次扫描时重试。

端点	目的
`GET /api/rap_received`	合作伙伴发送给我们的每个 RAP 文件，最新的在前：其返回、我们的确认以及任何再次发送的 TAP 文件。
`GET /api/rap_received/:filename`	合作伙伴发送给我们的一个 RAP 文件。
`GET /api/rap_sent`	我们发送的每个 RAP 文件，以及到达时合作伙伴的确认。

重新提交已退回的 TAP 文件

合作伙伴以致命返回退回的 TAP 文件在相同文件序列号下再次发送。TD.57 要求它携带返回该文件的 RAP 文件序列号。

`POST /api/tap_resubmit` 携带 `rap_filename`、`tap_sequence` 和 `commit` 从我们发送的副本中重建文件，位于 `tap_output_path` 或其 `_Sent` 目录。OmniRoam 添加 RAP 文件序列号和新的文件可用时间戳。如果没有 `commit`，响应将是重建文件的预览。如果有 `commit`，OmniRoam 将其写入合作伙伴的 `tap_output_path`，并进行传输。重新□交需要 `exporter` 角色。

在重新□交之前修复致命返回的原因。OmniRoam 将再次发送相同的记录。

读取 RAP 文件

OmniRoam API 和网页界面读取 RAP 文件。对 `GET /api/rap_files/:filename` 的请求返回解析和丰富的 RAP 内容，格式为 JSON。

OmniRoam 从 RAP 文件中□取以下字段：

字段	描述
`sender`	发送 RAP 文件的网络的 TADIG 代码。
`recipient`	接收 RAP 文件的网络的 TADIG 代码。
`roamingPartner`	漫游合作伙伴的 TADIG 代码。
`rapFileSequenceNumber`	RAP 文件序列号。
`rapFileCreationTimeStamp`	RAP 文��创建时间。
`tapCurrency`	原始 TAP 文件的结算货币。
`tapDecimalPlaces`	原始 TAP 文件的小数位数。
`returnDetailsCount`	RAP 文件中的返回数量。
`totalSevereReturnValue`	严重返回的总值。

- 概述

TAP 文件生成

本指南描述了 OmniRoam 如何构建 TAP 文件，如何命名它们，以及如何管理序列计数器。

目录

- 概述
- 生成运行的工作原理
- 文件命名
- 测试文件和实时文件
- 序列计数器
- TAP 文件结构
- 费用值
- NOTIFY 文件
- 触发导出

概述

TAP 文件是一个二进制文件，用于向一个合作伙伴报告评级漫游使用情况。OmniRoam 将每个 TAP 文件编码为 ASN.1 格式，采用 BER 格式，符合 TAP 3.12 语法。该格式遵循 GSMA TD.57。

生成运行的工作原理

1. 调度程序或 API 请求为一个合作伙伴排队导出作业。

2. 作业队列一次运行一个作业。此限制保护共享数据库连接池。
3. 生成器读取该合作伙伴的未处理 CDR，按最旧的优先。
4. 如果 CDR 的数量超过每个文件的记录限制，生成器将 CDR 拆分为多个文件。
5. 生成器保留一个序列号，编码文件并以原子方式写入。
6. 生成器在数据库中标记每个 CDR 的文件名。

生成器在名为接收方 TADIG 的子目录下写入每个 TAP 文件，位于 `tap_output_path` 内。生成器还将可读的 JSON 副本写入 `tap_human_readable_output_path`。

文件命名

TAP 文件名的固定长度为 17 个字符。

```
CD AAAAA BBBBB 00011
```

部分	长度	描述
前缀	2	`CD` 表示实时文件，或 `TD` 表示测试文件。
发送方	5	您的 TADIG 代码，来自 `batch_info.sender`。
接收方	5	合作伙伴的 TADIG 代码，来自 `batch_info.recipient`。
序列	5	以零填充的序列号。

名称 `CDAAAAABBBBB00011` 表示一个实时文件，从 `AAAAA` 到 `BBBBB`，序列 `11`。

测试文件和实时文件

`batch_info` 中的 `fileTypeIndicator` 键选择文件类型。

- 如果 `batch_info` 不包含 `fileTypeIndicator`，OmniRoam 生成一个 实时 文件。名称使用 `CD` 前缀和 `CD` 序列计数器。

- 如果 `batch_info` 包含值为 `T` 的 `fileTypeIndicator`，OmniRoam 生成一个 测试 文件。名称使用 `TD` 前缀和 `TD` 序列计数器。

在为新合作伙伴入驻时，将 `fileTypeIndicator` 设置为 `T`。删除该键以将合作伙伴移至实时结算。仅使用值 `T`。不同的值可能会生成一个 `TD` 名称，带有 `CD` 序列号，从而打破计数器对齐。

序列计数器

OmniRoam 跟踪每个接收方 TADIG 和每种文件类型的序列号。OmniRoam 将计数器存储在 `counters.yaml` 中。请参阅 配置参考。

- **CD** 计数实时（可收费数据）文件。
- **TD** 计数测试（测试数据）文件。

每个文件的序列号增加一。对于重传，数字不会增加。TAP 和 RAP 序列是独立的。

OmniRoam 通过单个进程保���序列号，并以原子���式写入 `counters.yaml`。此方法防止两个导出使用相同的数字。写入失败会保持之前的计数器文件不变。

序列号是一个 5 位数字字段，范围为 `00001` 到 `99999`。OmniRoam 不会自动重置计数器。请参阅 操作指南 以获取重置程序。

TAP 文件结构

TAP 文件是一个 `transferBatch` 结构。该批次包含控制信息和呼叫事件记录的列表。OmniRoam 仅生成 GPRS（数据）记录。

部分	内容
`batchControlInfo`	发送方、接收方、文件序列号、文件时间戳和版本号。
`accountingInfo`	本地货币、结算货币、小数位数和汇率信息。
`networkInfo`	UTC 偏移表和记录实体列表（数据包网关和服务网关）。
`callEventDetails`	GPRS 呼叫记录列表。
`auditControlInfo`	最早和最新的呼叫时间戳、总费用和记录计数。

每个 GPRS 呼叫记录包含以下字段。

字段	描述
可收费订阅者	IMSI，以及存在时的 MSISDN。
PDP 地址	订阅者 IP 地址。OmniRoam 优先使用 IPv6。
接入点名称	APN 的网络标识符和运营商标识符。
呼叫事件开始时间戳	收费开始时间，带有服务位置的 UTC 偏移。
总呼叫事件持续时间	会话持续时间（以秒为单位）。
收费 ID	会话收费标识符。
位置信息	记录实体、位置区域、小区标识符和可选的服务计费标识符。
数据量	进出字节计数。
收费信息	收费、可收费单位、收费单位和呼叫类型级别。

费用值

TAP 文件将金额存储为整数。读取器将整数除以 `10^tapDecimalPlaces` 以获取小数值。

OmniRoam 将每个费用乘以 1000 以生成整数。此比例与 `tapDecimalPlaces` 的 `3` 匹配。为每个合作伙伴将 `tapDecimalPlaces` 设置为 `3`。不同的值会生成一个费用整数，而不匹配声明的小数位数。

`roundingAction` 设置控制 OmniRoam 如何对缩放后的费用进行四舍五入：

- `Up` 向上四舍五入。
- `Down` 向下四舍五入。
- `Simple` 四舍五入到最接近的整数。

NOTIFY 文件

即使没有使用，合作伙伴也期望收到文件。当合作伙伴没有未处理的 CDR 时，OmniRoam 会写入一个 NOTIFY 文件。NOTIFY 文件使用与 TAP 文件相同的名称格式，但它包含一个 `notification` 记录，而不是一批呼叫事件。

触发导出

您可以通过三种方式启动导出。

方法	操作
调度程序	OmniRoam ���配置的时间自动运行导出。请参阅 操作指南。
API	对于一个合作伙伴发送 `POST /api/tap_export`，或对每个合作伙伴发送 `POST /api/export_schedule`。请参阅 HTTP API 参考。
命令行	使用合作伙伴名称运行遗留导出工具。仅将此方法作为后备。

- 概述

□□□□

□□□

可能的原因：

- `on_export_complete`中的上传命令为空或错误。
- 上传以错误的用户身份运行，因此清算中心拒绝了密钥。
- 清算中心无法访问。

解决方案：

1. 确认`on_export_complete`命令已设置。请参见配置参考。
2. 确认上传以传输用户身份运行，而不是root。
3. 检查上传日志和指标`tap3_upload_status`。

错误的序列号

症状：合作伙伴报告序列间隙或重复序列号。

可能的原因：

- `counters.yaml`文件是从旧备份恢复的。
- 某个计数器达到了`99999`。
- `fileTypeIndicator`值设置为`T`以外的值。

解决方案：

1. 检查`counters.yaml`中接收方TADIG的`CD`和`TD`计数器。
2. 如果计数器接近`99999`，请按照操作指南中的重置程序进行操作。
3. 对于测试文件，将`fileTypeIndicator`设置为`T`，或者对于实时文件删除该键。请参见TAP文件生成。

合作伙伴返回**RAP**文件

症状：RAP���视器发送电子邮件警报。一个RAP文件在入站目录中。

可能的原因：

- TAP文件未通过合作伙伴的验证。

解决方案：

1. □□□□

没有指标或分析

症状： InfluxDB分析为空。缺少Prometheus指标。

可能的原因：

- `influx_db`部分不完整。
- 依赖服务已关闭。

解决方案：

1. 确认所有四个`influx_db`值已设置。如果缺少任何值，OmniRoam将跳过写入。请参见配置参考。
2. 检查指标`tap3_service_up`是否有服务已关闭。请参见指标参考。

- 概述

TAP 验证

OmniRoam 根据 GSMA TD.57 和 GSMA TD.32 中可检查的规则验证 TAP 文件。您可以通过 API 验证传入的 TAP 文件或传出的 TAP 文件。

目录

- 如何验证文件
- 验证检查
- 错误严重性
- 错误代码类别
- 检查来自合作伙伴的传入文件
- 生成 RAP 文件
- 监视传入目录
- 验证的限制

如何验证文件

使用 API 验证 TAP 文件。

端点	目的
`GET /api/tap_validate_in/:filename`	验证传入的 TAP 文件。
`GET /api/tap_validate_out/:filename`	验证传出的 TAP 文件。

响应报告文件是否有效，并按严重性计数发现。每个发现包含错误代码、严重性、位置和消息。请参见 HTTP API 参考。

验证检查

OmniRoam 对解码的 TAP 文件运行四组检查。

组	检查
结构	五个强制部分存在且不为空：`batchControlInfo`、`accountingInfo`、`networkInfo`、`auditControlInfo` 和 `callEventDetails`。
批次控制	`sender` 和 `recipient` 为 5 个字符。`fileSequenceNumber` 为 5 个字符。`specificationVersionNumber` 为 `3`。
会计	`localCurrency` 和 `tapCurrency` 为 3 个字母的 ISO 4217 代码。`tapDecimalPlaces` 为非负整数。
审计算术	记录数等于呼叫事件的数量。总费用、总税和总折扣等于记录的总和。

错误严重性

OmniRoam 以严重性报告每个发现。只有在没有致命发现时，文件才有效。

严重性	影响
致命	整个文件被拒绝。
严重	单个记录被拒绝。文件的其余部分被接受。
警告	记录是可计费的。该发现是信息性的。

错误代码类别

TD.57 将错误代码分组为数字范围。

范围	类别
10 到 19	格式或语法错误。
20 到 29	值超出范围。
30 到 49	无效的组结构。
50 到 59	一般 ASN.1 或编码错误。
100 到 199	跨字段不一致。
200 到 249	特定错误，例如重复、IOT 错误或旧数据。
9000 到 9999	两个合作伙伴之间达成的双边专有错误。

检查来自��作伙伴的传入文件

`POST /api/tap_inbound_check` 运行接收方在对合作伙伴的 TAP 文件进行计费之前所做的每个检查。请求命名文件，可以是已经在 `TAP_In` 中的文件或上传的副本，并携带运营商对关系的了解：其自身的 TADIG、合作伙伴的 TADIG、其自身的 IMSI 前缀、协议的费率和商业漫游开始日期。它还可以携带 BA.08 时间表和 BA.11 汇率。

响应给出文件的裁决、每个检查的状态，以及每个发现的 TD.57 错误代码和严重性。

裁决	意义
`accept`	没有致命或严重发现。文件是可计费的。
`aged_records`	除了超过呼叫年龄限制的记录外，文件是好的。这些记录被返回；其余部分是可计费的。
`reject_records`	一些记录有严重发现。这些记录被返回；其余部分是可计费的。
`reject_file`	文件有致命发现，整个文件被返回。
`ignore_copy`	文件是已接收文件的相同副本。TD.57 规定忽略它。

检查	规则	**TD.57** 代码和严重性
`structure`	上述结构、批次控制、会计和审计检查。	如上
`parties`	发送方是协议的合作伙伴，接收方是我们。	100，致命
`file_name`	文件��的类型、发送��、接收方和序列与批次控制匹配。	100，致命
`sequence`	序列号为 5 位数字，且未曾用于不同内容。缺口命名缺失的文件。我们返回的致命返回可能在相同序列下再次出现，并且必须携带我们 RAP 的 RAP 文件序列号。任何其他文件上的 RAP 文件序列号是警告。	10, 20, 201 致命；37, 40, 200 警告
`timestamps`	截止和可用时间戳有效且有序，并且文件在到达后不超过一小时内可用。	100, 104 致命；101 警告
`transmission`	文件在 BA.08 文件传输时间表内到达。	262 警告；263 致命
`exchange_rates`	货币转换表完整且没有重复，每个记录使用的代码都已定义，每个费率与 BA.11 匹配。	30 到 34, 200, 201 致命；101 严重
`call_window`	最早和最晚的呼叫时间戳与记录匹配。	100，警告
`imsi`	每个记录的 IMSI 是我们的之一。	20, 30, 200，严重
`records`	每个记录的 UTC 偏移代码和记录实体在文件中定义，且记录不在截止后开始。	100，严重；截止是警告
`commercial_start`	没有可计费记录在商业漫游开始日期之前。	200，严重
`charges`	每个费用与该服务和日期的协议费率匹配，在 0.02 TAP 货币单位内（TD.57 第 5.8 节）。收��单位与四舍五入匹配。	200, 100，严重

检查	规则	**TD.57** 代码和严重性
`duplicates`	没有记录重复已从合作伙伴接收的记录，或同一文件中的其他记录。	250, 251, 255, 严重
`call_age`	没有记录超过呼叫年龄限制，默认 30 天。较旧的记录不必被接受，因此被拒绝，除非检查是以 `accept_aged` 运行，这样将其保留为老龄记录。	261 严重; 260 警告（当接受时）

请求未□供的参考数据的检查报告为 `not_run`，并给出原因。它从不计为通过。

每个费用和收费单位发现显示预期和实际值（以货币或字节为单位）、差异以及预期值的计算方式。当合作伙伴的收费单位都是大于协议四舍五入的较大步长的整数倍时，发现会命名该步长。

当协议有费率时，报告的 `money` 块以 TAP 货币单位汇总文件：

字段	意义
`declared`	合作伙伴收费的总额。
`at_agreement`	按协议费率定价的相同记录。
`payable`	未被拒绝记录的收费。
`rejected`	被拒绝记录的收费。
`rejected_at_agreement`	被拒绝记录在协议费率下的费用。
`overcharge`	`declared` 减去 `at_agreement`。

生成 **RAP** 文件

`POST /api/rap_generate` 采用相同请求，运行相同检查，并构建返回合作伙伴 TAP 文件的 TD.32 RAP 1.5 文件。

发现	**RAP** 内容
一个致命发现	整个文件的一个致命返回。
严重发现	每个被拒绝记录的一个严重返回，包含原始记录和每个发现的错误上下文。
一个序列缺口	一个单独的 RAP 文件，缺失返回缺口。

在没有 `commit` 的情况下，请求是预览。OmniRoam 不保留 RAP 序列号，也不写入文件。使用 `commit`，OmniRoam 为合作伙伴保留下一个 RAP 序列号，并将文件写入 `rap_output_path`。可计费数据在 `RC` 文件中返回，测试数据在 `RT` 文件中返回。OmniRoam 不会将 RAP 文件发送给合作伙伴。

OmniRoam 保留每个写入的 RAP 文件。已经有 RAP 的 TAP 文件或缺失范围将返回该 RAP，而不是新的序列号：TD.32 规定重传保持其编号。再次□交将相同文件写回 `rap_output_path`（如果不再存在），因此将再次发送。每个响应文件都有 `already_sent`，对于已经写入的文件，有 `written_at`。

`counters.yaml` 为每个合作伙伴保留一个 RAP 序列，因此 RAP 文件仅作为 `config.inbound_check.home_tadig` 发送。请求以任何其他网络发送的请求将被拒绝。

设置	位置	目�?
`config.rap_output_path`	`config.yaml`	□交的 RAP 文件写入的目录，为每个合作伙伴创建一个文件夹。
`<partner TADIG>: RC` 和 `RT`	`counters.yaml`	合作伙伴的下一个 RAP 序列号。每个从 `1` 开始。

RAP 生成需要 `exporter` 角色。

监视传入目录

OmniRoam 可以监视合作伙伴 TAP 文件到达的目录，例如 FTP 投放，并检查每个新文件一次。当文件的大小和修改时间在 `settle_seconds` 内没有变化时，文件被检查，因此正在进行的传输将被忽略。OmniRoam 保留每个报告，仅在文件发生变化时再次检查。

设置	目的
`config.inbound_check.watch_path`	要监视的目录。
`config.inbound_check.home_tadig`	我们的 TADIG，每个文件必须命名的接收方。
`config.inbound_check.interval_seconds`	扫描目录的频率。
`config.inbound_check.settle_seconds`	文件在检查之前必须保持不变的时间。
`config.inbound_check.max_call_age_days`	超过此天数的记录将被拒绝，除非作为老龄记录被接受。通[`30` 。
`config.inbound_check.ba08`	可选。BA.08 文件传输时间表，`standard_transmission_hours` `exceptional_transmission_hou`

我们的 IMSI 前缀为 `config.home_plmns`。每��发送方的 TADIG 和费率来自漫游经理协议，漫游同步保存。来自没有协议的发送方的文件将失败 parties 检查，其费用不被检查。

端点	目的
`GET /api/tap_inbound_checks`	检查的文件，最新的在前，包含发送方、裁决和计数。使用 `sender` 进行过滤。
`GET /api/tap_inbound_checks/:filename`	单个文件的完整报告。
`GET /api/tap_inbound_rap/:filename`	再次检查文件并预览其 RAP。没有任何内容被保留或写入。添加 `accept_aged=true` 以保留超过呼叫年龄限制的记录。
`PUT /api/tap_inbound_rap/:filename`	再次检查文件，保留 RAP 序列，并将 RAP 写入 `rap_output_path`。需要 `exporter` 角色。

两个 RAP 端点也返回 RAP 文件本身，因此可以下载并发送给合作伙伴。

端点	目的
`GET /api/tap_inbound_stop_return/:partner`	预览停止返回，适用于 TAP 文件已停止的合作伙伴。其最后的序列号是监视路径中接收到的最后一个文件，或在没有文件时为 `99999`。添加 `test=true` 以获取合作伙伴的测试数据。
`PUT /api/tap_inbound_stop_return/:partner`	写入停止返回。相同最后序列的停止返回不会重新编号。需要 `exporter` 角色。
`GET /api/rap_sent`	OmniRoam 写入的每个 RAP 文件，最新的在前：合作伙伴、序列、返回类型和返回的 TAP 文件。
`GET /api/rap_sent/:filename`	OmniRoam 写入的单个 RAP 文件，文件本身在 `file_b64` 中。

指标是从存储的报告中计数的，因此它们在重启后仍然存在。

指标	标签
`tap3_inbound_check_files_total`	`sender`, `verdict`
`tap3_inbound_check_findings_total`	`sender`, `check`, `severity`, `code`
`tap3_inbound_check_records_total`	`sender`
`tap3_inbound_check_records_rejected_total`	`sender`
`tap3_inbound_check_last_timestamp`	无

拒绝文件和记录、费用不匹配、不属于我们的 IMSI、重复项和序列缺口的警报规则在 `packaging/prometheus/omniroam-tap-inbound-alerts.yml` 中。

验证的限制

验证端点检查仅需要文件内容的规则。传入检查添加需要协议、已接收文件以及 BA.08 和 BA.11 参考数据的规则。这些规则不被检查。

- TADIG 代码列表。
- 完整的 ISO 4217 货币和 E.164 号码查找。
- 超出协议数据的呼叫类型和时间段、语音和 SMS 费率。
- 当合作伙伴的 TAP 文件已停止时。停止返回由手动写入 `/api/tap_inbound_stop_return`。

使用合作伙伴确认或 RAP 文件确认结算结果。请参见 RAP 处理。

- 概述