OmniSCEF 置参考

文档描述了 OmniSCEF 的每个配置参数。配置分为三个应用范围:

范围	目的
`:diameter_ex`	**T6a** Diameter 服务: 身份、监听器、应用程序和对等体。
`:api_ex`	**T8** REST API 端点: TLS、绑定地址和路由表。
`:omni_scef`	OmniSCEF 行为: 身份解析、组成员资格和监控。

OmniSCEF 在启动时评估 Diameter 服务参数，因此环境可以驱动它们（请参见 环境变量）。T8 端点和路由表在构建时固定。

目录

- Diameter 服务 (T6a)
- T8 API 端点
- 身份与组设置
- 监控设置
- 环境变量

Diameter 服务 (T6a)

OmniSCEF 作为 Diameter 服务器终止 T6a。MME（或 Diameter 路由代理）连接入站; OmniSCEF 在配置的端口上同时监听 TCP 和 SCTP。

```
config :diameter_ex,
  diameter: %{
    # 服务身份
    service_name: :omni_scef,
    host: "scef01",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    product_name: "OmniSCEF",

    # 监听器（TCP 和 SCTP 都在此地址/端口上启动）
    listen_ip: "0.0.0.0",
    listen_port: 3868,

    # 行为
    request_timeout: 5000,
    peer_selection_algorithm: :random,
    allow_undefined_peers_to_connect: true,
    log_unauthorized_peer_connection_attempts: true,

    # Diameter 能力
    vendor_id: 10415,
    supported_vendor_ids: [10415],
    auth_application_ids: [],
    acct_application_ids: [],

    # 内部处理程序（固定 - 不要更改）
    control_module: OmniScef.Control.Diameter,
    processor_module: DiameterEx.Processor,

    # T6a 应用注册
    applications: [
      %{
        application_name: :t6a,
        application_dictionary: :diameter_gen_3gpp_t6a,
        vendor_specific_application_ids: [
          %{vendor_id: 10415, auth_application_id: 16_777_346,
acct_application_id: nil}
        ]
      }
    ],

    # MME / DRA / HSS 对等体
    peers: [
      %{
```

```
        host: "mme01.epc.mnc001.mcc001.3gppnetwork.org",
        realm: "epc.mnc001.mcc001.3gppnetwork.org",
        ip: "10.7.25.185",
        port: 3868,
        tls: false,
        transport: :diameter_sctp,
        initiate_connection: false
      }
    ]
  }
```

服务身份

参数	类型	必需	默认	[
`service_name`	Atom	是	`:omni_scef`	Diame 实例的内部[
`host`	String	是	`scef01`	OmniS Diame 主机标签。[的完整 Or Host 是 `host.` 必须可被对[知。
`realm`	String	是	`epc.mnc001.mcc001.3gppnetwork.org`	OmniS Diame Realm Origin· 和路由。
`product_name`	String	否	`OmniSCEF`	在能力交换 （CER/C 广告的产品[

监听器

参数	类型	必需	默认	描述
`listen_ip`	String	否	`0.0.0.0`	用于绑定入站 Diameter 连接的 IP 地址。使用 `0.0.0.0` 代表所有接口。
`listen_port`	Integer	否	3868	用于入站 Diameter 连接的端口。TCP 和 SCTP 监听器都在此端口上启动。3868 是根据 RFC 6733 的标准 Diameter 端口。

□□

□□	□□	□□	□□	
`request_timeout`	Integer	□	5000	□ C □ T C □ □ □
`peer_selection_algorithm`	Atom	□	`:random`	□ □ □ □ □ □

`allow_undefined_peers_to_connect`	Boolean	□	`true`	□
				□
				`p`
				□
				□
				□
				□
				`t`
				□

`log_unauthorized_peer_connection_attempts`	Boolean	□	`true`	□ □ □

Diameter 能力

参数	类型	必需	默认	描述
`vendor_id`	Integer	否	10415	在能力中广告的 Vendor-Id。10415 是 3GPP。
`supported_vendor_ids`	Integer 列表	否	`[10415]`	在能力中广告的 Supported-Vendor-Ids。
`auth_application_ids`	Integer 列表	否	`[]`	基础（非供应商特定）认证应用程序 ID。为空，因为 T6a 注册为供应商特定应用程序（见 `applications`）。
`acct_application_ids`	Integer 列表	否	`[]`	基础计费应用程序 ID。OmniSCEF 不使用。

内部处理程序

这些将 Diameter 堆栈连接到 OmniSCEF 的逻辑。它们是固定的，不应更改。

参数	类型	必需	默认	描述
`control_module`	Module	是	`OmniScef.Control.Diameter`	接收入站请求并将其分派到 NIDD 和监控处理程序的模块。
`processor_module`	Module	是	`DiameterEx.Processor`	调用控制模块的 Diameter 堆栈处理器。

T6a 应用

`applications` 列表注册 T6a Diameter 应用。对于 OmniSCEF 这是固定的。

应用条目参数:

参数	类型	必需	默认
`application_name`	Atom	是	`:t6a`
`application_dictionary`	Atom	是	`:diameter_gen_3gpp_t6a`
`vendor_specific_application_ids`	Map 列表	是	见下文

供应商特定应用 **ID** 参数:

参数	类型	必需	默认	描述
`vendor_id`	Integer	是	10415	3GPP 供应商 ID。
`auth_application_id`	Integer	是	16777346	T6a Application-Id。
`acct_application_id`	Integer	否	`nil`	T6a 不使用。

对等体

`peers` 列表声明已知的 Diameter 对等体（MME、DRA 和 HSS）。`allow_undefined_peers_to_connect` 参数默认为 `true`，通常由 MME 发起连接。在 OmniSCEF 不拨出任何连接的部署中，您可以将 `peers` 留空（`[]`）。当 OmniSCEF 必须发起连接时，在此列出对等体，或限制可以连接的对等体。

对等体参数：

参数	类型	必需	默认	描述
`host`	String	是	-	对等体的 Diameter 身份（FQDN）。必须与对等体配的 Origin-Host 完全匹配。
`realm`	String	是	-	对等体的 Diameter Realm。
`ip`	String	是	-	对等体用于 TCP/SCTP 连接的 IP 地址。
`port`	Integer	否	3868	对等体的 Diameter 端口
`tls`	Boolean	否	`false`	与此对等体的连接是否使用 TLS。
`transport`	Atom	否	`:diameter_sctp`	此对等体的传输: `:diameter_sct` 或 `:diameter_tcp`
`initiate_connection`	Boolean	否	`false`	当为 `true` 时, OmniSCEF 在启动时主 连接此对等体。当为 `false` 时, OmniSCEF 等待对 ���连接入站。对于拨入 SCEF 的 MME, 请将此 置为 `false`。

T8 API 端点

T8 RESTful API 通过 HTTPS 提供。所有路由都挂载在 `/api` 前缀下，因此有效的 apiRoot 是 `https://<host>:<port>/api`。

```
config :api_ex,
  api: %{
    port: 8443,
    listen_ip: "0.0.0.0",
    product_name: "OmniSCEF",
    title: "T8 API - OmniSCEF",
    hostname: "localhost",
    enable_tls: true,
    tls_cert_path: "priv/cert/omnitouch.crt",
    tls_key_path: "priv/cert/omnitouch.pem",
    routes: [ ... ]   # □□□ T8 □□□ - □□□
  }
```

端点参数

参数	类型	必需	默认	描述
`port`	Integer	否	8443	T8 API 的 HTTPS 端口。
`listen_ip`	String	否	`0.0.0.0`	用于绑定 T8 AP 的 IP 地址。使用 `0.0.0.0` 代[所有接口。
`product_name`	String	否	`OmniSCEF`	在 API/OpenAI 元数据中使用的产品 称。
`title`	String	否	`T8 API - OmniSCEF`	在生成的 OpenAPI 文[和 Swagger UI 中显示的标题。
`hostname`	String	否	`localhost`	生成端点 URL [使用的主机名。
`enable_tls`	Boolean	否	`true`	当为 `true` 时[端点使用下面的证书 供 HTTPS。
`tls_cert_path`	String	是（如果启用 TLS）	`priv/cert/omnitouch.crt`	PEM 编码的 TLS 证书路径。
`tls_key_path`	String	是（如果启用 TLS）	`priv/cert/omnitouch.pem`	PEM 编码的 TLS 私钥路径。

参数	类型	必需	默认	描述
`routes`	列表	是	固定	T8 路由表。这是 API 表面的固定定义，无法由操作员调整。请参见 路由表[

路由表

`routes` 条目定义 T8 API 表面。它们对于 OmniSCEF 是固定的，并生成以下端点（均在 `/api` 下）。每个条目将 URL 模板映射到处理程序及其提供的操作集。

路由条目参数：

参数	类型	描述
`path`	String	URL 模板。像 `:scsAsId` 这样的段是路径变量。
`module`	Module	提供路由的处理程序。
`parameters`	String 或 `nil`	成员资源路径变量的名称（例如 `configurationId`），或集合仅路由的 `nil`。
`actions`	Atom 列表	提供的 HTTP 操作：`:index`（获取集合）、`:create`（POST）、`:show`（获取成员）、`:update`（PUT/PATCH 成员）、`:delete`（删除成员）。

生成的端点：

□□ & □□	□□
`GET /api/status`	□□□□□/□ □□
`GET /api/3gpp-nidd/v1/{scsAsId}/configurations`	□□ NIDD ��□□
`POST /api/3gpp-nidd/v1/{scsAsId}/configurations`	□□ NIDD □□□
`GET/PUT/PATCH/DELETE /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}`	□□/□/□/□□ NIDD □□□
`GET/POST /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}/downlink-data-deliveries`	□□/□ □□□□ □□
`GET/DELETE /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}/downlink-data-deliveries/{downlinkDataDeliveryId}`	□□□ □/□□ □□□ □□
`GET/POST /api/3gpp-monitoring-event/v1/{scsAsId}/subscriptions`	□□/□ □□□□ □□
`GET/PUT/DELETE /api/3gpp-monitoring-event/v1/{scsAsId}/subscriptions/{subscriptionId}`	□□/□ □/□□ □□□ □□

生成的 OpenAPI 文档可通过 `GET /api/schema` 获取，交互式 Swagger UI 可通过 `GET /api/docs` 访问。

> 可选的共享密钥认证。 T8 端点支持对入站请求的可选共享密钥检查。除非在 `config :api_ex, :api` 下添加 `auth` 密钥，否则它是禁用的。当未设置时（默认），不应用共享密钥检查，端点应通过网络控制（例如，私有互连或反向代理）进行保护。

身份与组设置

OmniSCEF 将面向 AS 的身份（外部标识符或 MSISDN）解析为网络 IMSI，并将组标识符解析为其成员 IMSI��在第一次发布中，这些是静态运营商映射。

```
config :omni_scef,
  nidd: %{
    identity_map: %{
      "device01@iot.example.org" => "001010000000001",
      "61400000001" => "001010000000001"
    },
    group_map: %{
      "fleet-a@iot.example.org" => ["001010000000001",
"001010000000002"]
    },
    default_max_packet_size: 255
  }
```

参数	类型	必需	默认	描述
`identity_map`	Map	否	`%{}`	将外部标识符（`user@domain`）或 MSISDN 映射到订阅者 IMSI。当 AS 创建单 UE NIDD 配置或监控订阅时使用。此处未出现的身份将被拒绝为未知 UE。
`group_map`	Map	否	`%{}`	将 `externalGroupId` 映射到成员 IMSI 列表。用于组 NIDD 交付。
`default_max_packet_size`	Integer	否	255	默认最大非 IP 负载大小（以字节为单位），适用于未指定 `maximumPacketSize` 的 NIDD 配置。NAS 传输将单个非 IP 消息限制为 255 字节。

> 在生产中，这些映射通常由 HSS/UDR 查找支持，而不是静态配置。面向 AS 的行为保持不变。

监控设置

可选。当您配置监控目标时，创建监控订阅会触发事件向下传送。OmniSCEF 向该节点发送配置息请求（CIR）。当您未设置目标时，OmniSCEF 仍然存储订阅并交付入站报告。在这种情况下，OmniSCEF 不会主动发送 CIR。

```
config :omni_scef,
  monitoring: %{
    destination: %{
      host: "hss01.epc.mnc001.mcc001.3gppnetwork.org",
      realm: "epc.mnc001.mcc001.3gppnetwork.org"
    }
  }
```

`destination` 参数:

参数	类型	必需	默认	描述
`host`	String	否	-	监控配置请求发送到的 HSS / 服务节点的 Diameter 身份（FQDN）。省略时，请求仅按领域路由。
`realm`	String	是（如果设置了 `destination`）	-	用作监控 CIR 的 Destination-Realm 的 Diameter Realm。

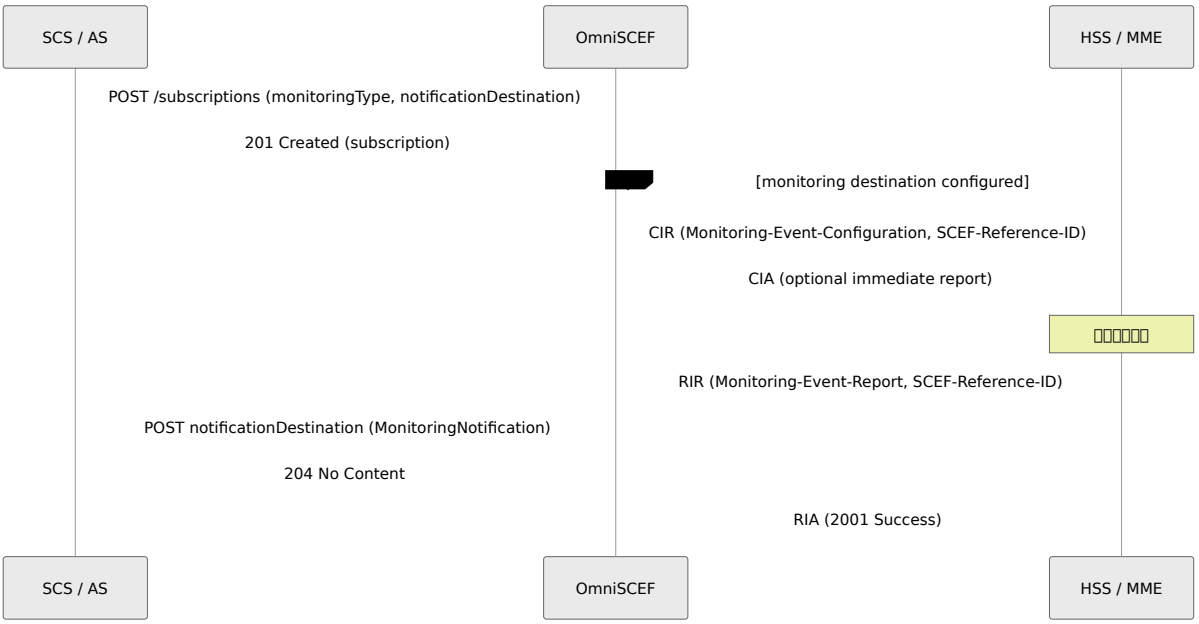
环境变量

Diameter 服务身份和监听器可以在启动时从环境中提供。这些将覆盖 Diameter 服务下显示的默认值。

变量	映射到	默认
`SCEF_DIAMETER_LISTEN_IP`	`listen_ip`	`0.0.0.0`
`SCEF_DIAMETER_LISTEN_PORT`	`listen_port`	`3868`
`SCEF_DIAMETER_HOST`	`host`	`scef01`
`SCEF_DIAMETER_REALM`	`realm`	`epc.mnc001.mcc001.3gppnetw`

□□□□

□□□



订阅和报告流程

在 **SCEF** 处触发是可选的。当您配置 监控目标 时，OmniSCEF 会发送 CIR 来触发事件。当您没有配置时，OmniSCEF 仍然会创建订阅并交付任何入站报告。在这种情况下，OmniSCEF 不会主动发送 CIR（例如，当另一路径[illegible]过 HSS 触发事件时）。

当报告的 SCEF-Reference-ID 与任何订阅不匹配时，OmniSCEF 会以 RIA 确认，但不会转发它。

T8 资源参考

MonitoringEventSubscription

通过 `POST /api/3gpp-monitoring-event/v1/{scsAsId}/subscriptions` 创建。

字段	类型	必需	描述
`notificationDestination`	字符串 (URI)	是	AS 接收监控报告的回调地址。
`monitoringType`	字符串（枚举）	是	要监控的事件。请参见 监控类型。
`externalId`	字符串	条件	设备外部标识符。`externalId`、`msisdn` 或 `externalGroupId` 之一。
`msisdn`	字符串	条件	设备 MSISDN。
`externalGroupId`	字符串	条件	组监控的组标识符。
`maximumNumberOfReports`	整数	否	在此数量的报告后停止。
`monitoringDuration`	字符串（日期时间）	否	在此时间停止监控。
`reachabilityType`	字符串（枚举）	否	对于 `UE_REACHABILITY`：`SMS` 或 `DATA`。
`maximumDetectionTime`	整数	否	对于 `LOSS_OF_CONNECTIVITY`：检测时间（秒）。
`mtcProviderId`	字符串	否	MTC 服务提供商标识符。
`self`	字符串 (URI)	只读	订阅的自身 URI。

示例订阅请求：

```
{
  "monitoringType": "UE_REACHABILITY",
  "notificationDestination":
"https://as.example.org/monitoring/callback",
  "externalId": "device01@iot.example.org",
  "reachabilityType": "DATA",
  "maximumNumberOfReports": 10
}
```

MonitoringNotification

OmniSCEF 会将 `MonitoringNotification` 通过订阅的 `notificationDestination` POST 发送给每个报告。它引用了原始订阅，并携带一个或多个解码的事件报告。

示例通知：

```
{
  "subscription": "/3gpp-monitoring-
event/v1/as01/subscriptions/XyZ...",
  "monitoringEventReports": [
    {
      "monitoringType": "UE_REACHABILITY",
      "reachabilityType": "DATA"
    }
  ]
}
```

应用服务器以 `204 No Content` 确认。

监控类型

OmniSCEF 支持完整的 TS 29.122 监控类型集。每个类型映射到直径接口上的相应 `Monitoring-Type` 值（TS 29.336 §5.3.9）。

`monitoringType` (T8)	直径值	描述
`LOSS_OF_CONNECTIVITY`	0	UE 不再可用于数据。
`UE_REACHABILITY`	1	UE 已变得可用于 SMS 或数据。
`LOCATION_REPORTING`	2	UE 的位置（或其变化）被报告。
`CHANGE_OF_IMSI_IMEI_ASSOCIATION`	3	与 IMSI 关联的 IMEI 发生变化。
`ROAMING_STATUS`	4	UE 的漫游状态（或变化）被报告。
`COMMUNICATION_FAILURE`	5	发生了通信故障事件。
`AVAILABILITY_AFTER_DDN_FAILURE`	6	UE 在下行数据通知失败后变得可用。
`NUMBER_OF_UES_IN_AN_AREA`	7	在给定区域内存在的 UE 数量。
`PDN_CONNECTIVITY_STATUS`	8	PDN 连接已建立/释放。
`DOWNLINK_DATA_DELIVERY_STATUS`	9	下行数据交付状态。
`API_SUPPORT_CAPABILITY`	10	报告 API 支持能力。

具有不支持的 `monitoringType` 的订阅将被拒绝，返回 `400 INVALID_MONITORING_TYPE`。

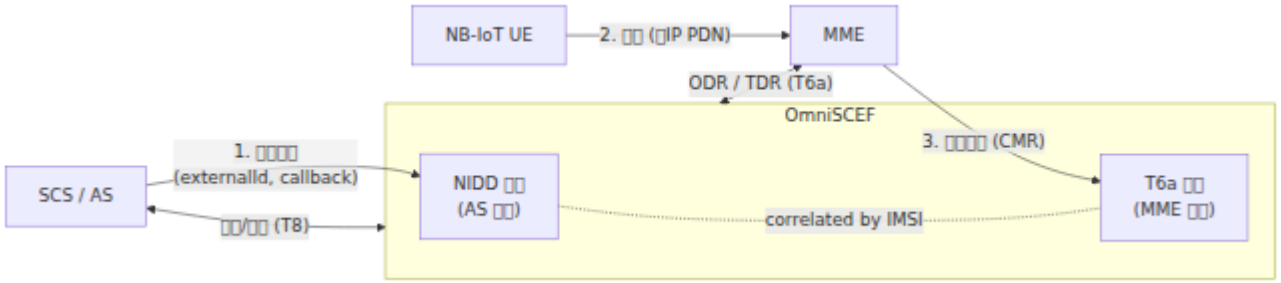
参考值

可达性类型

值	含义
`SMS`	UE 可用于 SMS。
`DATA`	UE 可用于数据。

相关直径命令

请参见 操作指南参考表 中的 CIR/CIA 和 RIR/RIA 命令及结果代码。



非IP数据传输（NIDD）

非**IP**数据传输 允许一个蜂窝物联网 / NB-IoT 设备与应用服务器交换小的 非**IP** 消息 而不需要**IP**地址。数据通过 NAS 信令平面（控制平面 CIoT EPS 优化）在用户设备（UE）和移动管理实体（MME）之间传输。MME 将其通过 **T6a** 转发给 OmniSCEF。然后 OmniSCEF 通过 **T8** 将其暴露给应用服务器。

NIDD 的架构和程序在 3GPP TS 23.682 §5.13 中定义; T6a Diameter 接口在 TS 29.128 中定义; T8 NIDD API 在 TS 29.122 中定义。

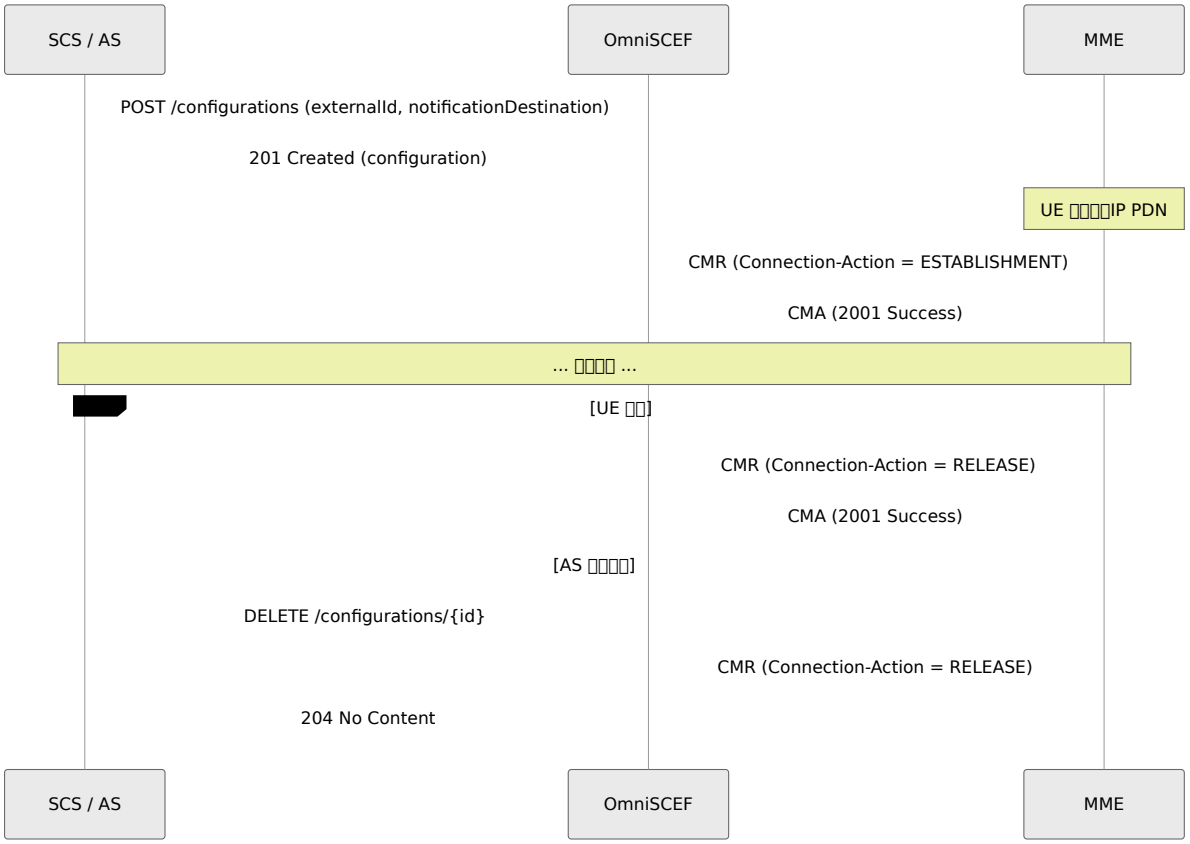
目录

- 概述
- 连接生命周期
- 移动发起数据
- 移动终止数据
- 组 NIDD
- 可靠数据服务
- MO 异常数据
- T8 资源参考
- 参考值

概述

NIDD 会话有两个相关的部分:

- **NIDD** □置 由应用服务器通过 T8 创建。它命名设备（通过外部标识符、MSISDN 或组）、通知回调和 NIDD 选项。
- **T6a** 绑定 在 MME 为该用户建立非IP连接时创建。它将 Diameter 会话和 MME 身份与用于到达 UE 的 `{IMSI, EBI}` 绑定在一起。



OmniSCEF 通过用户身份将两者关联起来，因此上行数据能够找到正确的 AS 回调，下行数据能够找到正确的实时连接。

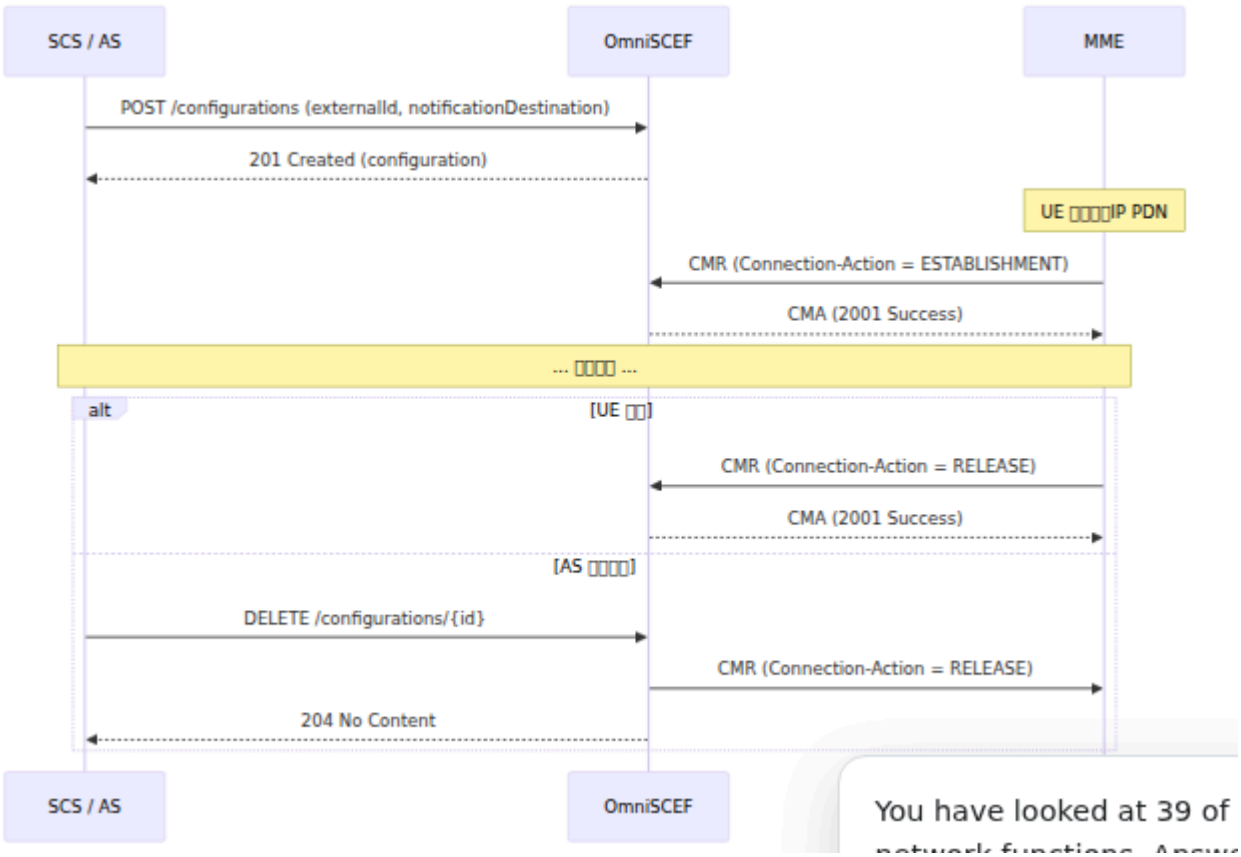
连接生命周期

MME 管理 T6a 连接，随着 UE 的附加和分离，应用服务器也可以通过删除其 NIDD 配置来结束连接。

当 MME 为没有 NIDD 配置的用户建立连接时，OmniSCEF 会用 `DIAMETER_ERROR_USER_UNKNOWN (5001)` 回应 CMR。该用户未被配置为 NIDD。

移动发起数据（上行）

UE 发送的上行非IP数据以 T6a **MO-**数据请求 **(ODR)** 的形式到达 OmniSCEF。OmniSCEF 通过向配置的回调 POST 通知将其传递给应用服务器，并用 **MO-**数据应答 **(ODA)** 回应 MME。

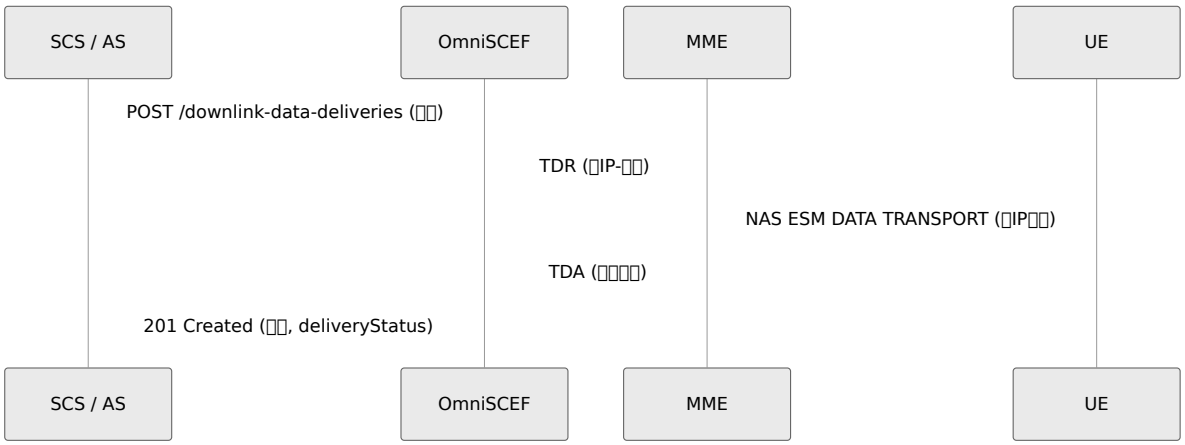


如果 OmniSCEF 无法解析配置或回调失败，它会用 `DIAMETER_UNABLE_TO_COMPLY (5012)` 回应 ODA。

示例上行通知（POST 到回调）：

```
{
  "externalId": "device01@iot.example.org",
  "data": "AQIDBA=="
}
```

字段	描述
`externalId` / `msisdn`	设备身份，按配置提供。
`data`	非IP有效负载，base64 编码。
`rdsSourcePort` / `rdsDestinationPort`	当 可靠数据服务 启用时存在。
`moExceptionDataCounter`	当上行被标记为 MO 异常数据 时存在。



移动终止数据（下行）

应用服务器通过向配置的 `downlink-data-deliveries` 集合 POST 数据来发送下行非IP数据。OmniSCEF 将其作为 T6a **MT-**数据请求 **(TDR)** 转发给 MME，并在应用服务器可以检索的单个交付资源上记录结果（来自 **MT-**数据应答，**TDA**）。

当 UE 空闲时，MME 会缓存数���并寻呼 UE，一旦它对交付承担责任，就用成功回应 TDA。OmniSCEF 在交付的 `deliveryStatus` 中反映结果（见 交付状态值）。

示例下行请求：

```
{
  "data": "SGVsbG8=",
  "maximumLatency": 3600,
  "priority": 5
}
```

字段	描述
`data`	要交付的非IP有效负载，base64 编码。必填。
`maximumLatency`	可选。最大可接受的缓存延迟，以秒为单位。
`priority`	可选。排队交付之间的相对优先级。
`pdnEstablishmentOption`	可选。当没有连接存在时的行为。见 PDN 建立选项。
`rdsSourcePort` / `rdsDestinationPort`	可选。RDS 端口。见 可靠数据服务。

示例下行响应：

```
{
  "self": "https://scef01:8443/api/3gpp-
nidd/v1/as01/configurations/AbC.../downlink-data-
deliveries/XyZ...",
  "data": "SGVsbG8=",
  "deliveryStatus": "SUCCESS"
}
```

组 NIDD

使用 `externalGroupId` 创建的 NIDD 配置（而不是单个 `externalId`/`msisdn`）表示一个 设备组 (TS 23.682 §5.13.3A)。对组配置的单个下行交付会 分发 给每个具有活动 T6a 连接的组成员。

组的成员从 组映射 中解析。交付的聚合 `deliveryStatus` 反映每个成员的 最差 结果，因此组交付仅在每个可达成员都被服务时报告 `SUCCESS`。

可靠数据服务

可靠数据服务 (RDS) 通过 端口号 在单个非IP连接上复用多个应用流，按照 TS 23.682 §4.5.14.3。通过在创建配置时设置 `reliableDataService: true` 来启用 RDS。

当 RDS 启用时：

- 下行：当交付指定 `rdsDestinationPort` 和 `rdsSourcePort` 时，OmniSCEF 会在发送 TDR 之前用包含这些端口的 RDS 头帧包装非IP有效负载。
- 上行：OmniSCEF 从 ODR 有效负载中解析 RDS 头。它在上行通知中显示 `rdsSourcePort` 和 `rdsDestinationPort`，并将内部有效负载作为 `data` 交付。

端口范围为 0–255。当 RDS 未启用或交付省略端口时，有效负载将不带帧传输。

MO 异常数据

NB-IoT 设备可以通过 RRC 建立原因将上行传输标记为 异常报告（紧急的、非模式����），在 T6a `RRC-Cause-Counter` 中计数 (TS 23.682 §5.13.4)。当 OmniSCEF 在 ODR 上看到此指示时，它会在上行通知中包含 `moExceptionDataCounter`，以便应用服务器可以优先处理。

T8 □□□□

NiddConfiguration

□□ `POST /api/3gpp-nidd/v1/{scsAsId}/configurations` □□□

字段	类型	必需	描述
`notificationDestination`	字符串 (URI)	是	应用服务器接收上行和状态通知的回调。
`externalId`	字符串	可选	设备外部标识符 (`user@domain`)。`externalId`、`msisdn` 或 `externalGroupId` 之一。
`msisdn`	字符串	可选	设备 MSISDN。
`externalGroupId`	字符串	可选	组标识符，用于 组 NIDD。
`dnn`	字符串	否	非IP连接的数据网络名称 / APN。
`pdnEstablishmentOption`	字符串 (枚举)	否	下行时没有连接存在的行为。见 PDN 建立选项。
`maximumPacketSize`	整数	否	非IP有效负载的最大大小，以字节为单位（默认为配置的 `default_max_packet_size`，255）。
`reliableDataService`	布尔值	否	启用 可靠数据服务。
`mtcProviderId`	��符串	否	MTC 服务提供者标识符。
`validityTime`	字符串 (日期时间)	否	配置过期的时间。
`status`	字符串 (枚举)	只读	生命周期状态。见 配置状态值。

字段	类型	必需	描述
`self`	字符串 (URI)	只读	资源的自身 URI。

NiddDownlinkDataTransfer

通过 `POST /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}/downlink-data-deliveries` 创建。字段在 移动终止数据 下描述。`deliveryStatus` 和 `requestedRetransmissionTime` 字段为只读，并从 TDA 设置。

通知

OmniSCEF 将以下内容 POST 到配置的 `notificationDestination`：

通知	何时
`NiddUplinkDataNotification`	上行非IP数据到达（来自 ODR）。
`NiddDownlinkDataDeliveryStatusNotification`	先前接受的下行交付的结果发生变化（异步状态）。

应用服务器用 `204 No Content` 确认每个通知。

参考值

交付状态值

值	意义
`SUCCESS`	MME 已交付数据（或通过缓存/寻呼空闲 UE 承担责任）。
`SENDING`	交付正在进行中。
`BUFFERING`	MME 为空闲 UE 缓存数据。
`FAILURE_UE_NOT_REACHABLE`	无法到达 UE。
`FAILURE_REMOTE_FAILURE`	MME / 服务节点拒绝或失败请求。
`FAILURE`	交付因其他原因失败。

PDN 建立选项

值	下行时没有连接存在的行为
`WAIT_FOR_UE`	缓存直到 UE 建立连接。
`INDICATE_ERROR`	立即拒绝交付。
`ESTABLISH_PDN_CONNECTION`	触发 UE 建立非IP连接。

□置状态值

值	意义
`ACTIVE`	配置已授权并可用。
`TERMINATED_UE_NOT_AUTHORIZED`	订阅者未被授权使用 NIDD。
`TERMINATED`	配置已过期或被移除。

OmniSCEF 故障排除

关于常见 OmniSCEF 操作问题的故障/解决指导。有关底层流程，请参见 NIDD 和 监控事件；有关参数，请参见 配置参考。

目录

- MME 无法通过 T6a 连接
- 连接被 USER_UNKNOWN (5001) 拒绝
- 下行交付失败
- 上行数据未到达应用服务器
- 组下行未到达任何设备
- 监控报告未交付
- T8 API 对有效端点返回 404
- T8 端点无法启动

MME 无法通过 T6a 连接

症状：MME 无法与 OmniSCEF 建立 Diameter 连接；没有能力交换完成。

可能原因：

- 防火墙阻止 Diameter 端口（默认 3868）。
- 传输不匹配：MME 使用 SCTP 而只有 TCP 可达，或反之亦然。
- 对等方授权受限，且 MME 不是已配置的对等方。
- Origin-Host / realm 不匹配，MME 期望的与 OmniSCEF 广告的不同。

解决方案：

1. 确认防火墙允许配置的 `listen_port` 在 TCP 和 SCTP 上。OmniSCEF 在两者上监听。请参见 Listeners。
2. 验证 MME 的配置传输是否匹配。TCP 和 SCTP 监听器始终启动。
3. 如果 `allow_undefined_peers_to_connect` 为 `false`，请将 MME 添加到 `peers` 中，并使用其确切的 Origin-Host。请参见 Peers。

4. 确认 OmniSCEF 的 `host` 和 `realm` 与 MME 配置的目标相匹配。检查 `log_unauthorized_peer_connection_attempts` 日志输出以获取被拒绝尝试的信息。

连接被 USER_UNKNOWN (5001) 拒绝

症状：MME 建立了 Diameter 连接，但 NIDD 连接请求 (CMR) 返回 `DIAMETER_ERROR_USER_UNKNOWN (5001)`。

可能原因：

- 对于该用户的 IMSI 没有 NIDD 配置。
- 应用服务器使用一个外部身份创建的配置无法解析为该用户的 IMSI。

解决方案：

1. 确认应用服务器在 UE 附加之前已为设备创建了 NIDD 配置。
2. 验证设备的 `externalId`/`msisdn` 是否在身份映射中解析为正确的 IMSI。
3. 确认身份映射中的 IMSI ��� MME 在 CMR 中呈现的 IMSI 匹配。

下行交付失败

症状：对 `downlink-data-deliveries` 的 `POST` 请求返回的交付的 `deliveryStatus` 是 `FAILURE_*` 值，或返回 `404`。

可能原因：

- 该设备没有活动的 T6a 连接（`404 NIDD_CONFIGURATION_NOT_AVAILABLE`）。UE 尚未附加，或 OmniSCEF 已释放其连接。
- UE 无法访问（`FAILURE_UE_NOT_REACHABLE`）。
- MME / 服务节点拒绝了请求（`FAILURE_REMOTE_FAILURE`）。
- `data` 字段缺失或无效的 base64（`400 DATA_MISSING`）。

解决方案：

1. 确认设备已附加且其连接已建立（之前的 CMR 被接受）。
2. 检查 `data` 字段是否存在且为 base64 编码。

3. 对于空闲的 UE，期望 MME 进行缓冲和分页；一旦 MME 负责，交付报告成功。请参见 移动终止数据。
4. 对于 `FAILURE_REMOTE_FAILURE`，检查 MME/服务节点侧的拒绝原因。

上行数据未到达应用服务器

症状：UE 发送上行数据，但应用服务器的回调未被调用。

可能原因：

- 设备没有 NIDD 配置（因此没有已知的回调）。
- `notificationDestination` URL 从 OmniSCEF 无法访问，或返回非 2xx 状态。
- 对于上行所呈现的 `{IMSI, EBI}` 没有活动绑定。

解决方案：

1. 确认设备存在 NIDD 配置并携带可达的 `notificationDestination`。
2. 验证 OmniSCEF 到回调 URL 的网络连接，包括如果是 HTTPS 的 TLS 信任。
3. 确认回调返回 2xx 状态；非 2xx 被视为交付失败。
4. 确认设备已附加（CMR 被接受）后再发送上行数据。

组下行未到达任何设备

症状：对组配置的下行请求返回 `404 NIDD_CONFIGURATION_NOT_AVAILABLE`，或到达的设备少于预期。

可能原因：

- `externalGroupId` 在组映射中不存在，或未映射到任何成员。
- 当前没有组成员具有活动的 T6a 连接。

解决方案：

1. 确认 `externalGroupId` 在 组映射中定义，并列出预期的成员 IMSI。
2. 确认至少有一个成员设备已附加。只有具有活动连接的成员才能接收 交付。聚合状态反映每个成员的最差结果。请参见 组 NIDD。

监控报告未交付

症状：创建了监控订阅，但应用服务器未收到任何 通知。

可能原因：

- 事件在网络中从未被激活（没有配置监控目标，且没有其他路径激活它）。
- 报告的 SCEF-Reference-ID 与活动订阅不匹配。这可能在重启后发生，因为订阅状态在内存中。
- 订阅的 `notificationDestination` 无法访问。

解决方案：

1. 如果 OmniSCEF 应该激活事件，请配置一个 监控目标，以便发送 CIR。
2. 在 OmniSCEF 重启后重新创建订阅。订阅和引用-ID 状态是 易失性的，无法在重启后存活。
3. 验证 OmniSCEF 到订阅回调的连接。

T8 API 对有效端点返回 404

症状：对文档化的 T8 路径的请求返回 `404`，且该端点未出现在 `/api/schema` 的 OpenAPI 文档中。

可能原因：

- 路由表已更改，但 API 层未重新编译，因此新路由未注册。
- 请求省略了所有 T8 路由挂载的 `/api` 前缀。

解决方案：

1. 确认请求路径包含 `/api` 前缀（例如：`/api/3gpp-nidd/v1/{scsAsId}/configurations`）。
2. 如果 路由表 被修改，必须重新编译 API 依赖项，以便新路由注册（`mix deps.compile api_ex --force`），然后重启服务。

T8 端点无法启动

症状：OmniSCEF 无法启动 T8 HTTPS 端点。

可能原因：

- TLS 证书或密钥文件缺失或不可读。
- 配置的 `port` 已被占用。

解决方案：

1. 确认 `tls_cert_path` 和 `tls_key_path` 指向有效且可读的 PEM 文件。请参见 T8 API 端点。
2. 确认没有其他内容绑定到 `listen_ip` 接口上的配置 `port`。
3. 在受控环境中运行而不使用 TLS，请设置 `enable_tls: false`。生产 部署必须始终在 OmniSCEF 之前或之处终止 TLS。