

# OmniSCEF Configuration Reference

This document describes every configuration parameter for OmniSCEF. Configuration is split across three application scopes:

| Scope                     | Purpose                                                                        |
|---------------------------|--------------------------------------------------------------------------------|
| <code>:diameter_ex</code> | The <b>T6a</b> Diameter service — identity, listeners, application, and peers. |
| <code>:api_ex</code>      | The <b>T8</b> REST API endpoint — TLS, bind address, and the route table.      |
| <code>:omni_scef</code>   | OmniSCEF behaviour — identity resolution, group membership, and monitoring.    |

The Diameter service parameters are evaluated at start-up so they can be driven from the environment (see [Environment Variables](#)); the T8 endpoint and route table are fixed at build time.

## Table of Contents

- [Diameter Service \(T6a\)](#)
- [T8 API Endpoint](#)
- [Identity & Group Settings](#)
- [Monitoring Settings](#)
- [Environment Variables](#)

# Diameter Service (T6a)

OmniSCEF terminates T6a as a Diameter server. The MME (or a Diameter Routing Agent) connects inbound; OmniSCEF listens on both TCP and SCTP on the configured port.

```

config :diameter_ex,
  diameter: %{
    # Service identity
    service_name: :omni_scef,
    host: "scef01",
    realm: "epc.mnc380.mcc313.3gppnetwork.org",
    product_name: "OmniSCEF",

    # Listeners (TCP and SCTP are both started on this
address/port)
    listen_ip: "0.0.0.0",
    listen_port: 3868,

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

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

    # Internal handlers (fixed – do not change)
    control_module: OmniScef.Control.Diameter,
    processor_module: DiameterEx.Processor,

    # T6a application registration
    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 peers
    peers: [

```

```
%{  
  host: "mme01.epc.mnc380.mcc313.3gppnetwork.org",  
  realm: "epc.mnc380.mcc313.3gppnetwork.org",  
  ip: "10.7.25.185",  
  port: 3868,  
  tls: false,  
  transport: :diameter_sctp,  
  initiate_connection: false  
}  
]  
}
```

# Service Identity

| Parameter    | Type   | Required | Default                           |
|--------------|--------|----------|-----------------------------------|
| service_name | Atom   | Yes      | :omni_scef                        |
| host         | String | Yes      | scef01                            |
| realm        | String | Yes      | epc.mnc380.mcc313.3gppnetwork.org |
| product_name | String | No       | OmniSCEF                          |

# Listeners

| Parameter                | Type    | Required | Default              | Description                                                                                                                                                    |
|--------------------------|---------|----------|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>listen_ip</code>   | String  | No       | <code>0.0.0.0</code> | IP address to bind for inbound Diameter connections. Use <code>0.0.0.0</code> for all interfaces.                                                              |
| <code>listen_port</code> | Integer | No       | 3868                 | Port for inbound Diameter connections. Both TCP and SCTP listeners are started on this port. 3868 is the standard Diameter port per <a href="#">RFC 6733</a> . |

# Behaviour

| Parameter                        | Type    | Required | Default |
|----------------------------------|---------|----------|---------|
| request_timeout                  | Integer | No       | 500     |
| peer_selection_algorithm         | Atom    | No       | :random |
| allow_undefined_peers_to_connect | Boolean | No       | true    |

| Parameter                                 | Type    | Required | Def  |
|-------------------------------------------|---------|----------|------|
|                                           |         |          |      |
| log_unauthorized_peer_connection_attempts | Boolean | No       | true |

## Diameter Capabilities

| Parameter                         | Type            | Required | Default              | Description                                                                                                                                                   |
|-----------------------------------|-----------------|----------|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>vendor_id</code>            | Integer         | No       | 10415                | Vendor-Id advertised in capabilities. 10415 is 3GPP.                                                                                                          |
| <code>supported_vendor_ids</code> | List of Integer | No       | <code>[10415]</code> | Supported-Vendor-Ids advertised in capabilities.                                                                                                              |
| <code>auth_application_ids</code> | List of Integer | No       | <code>[]</code>      | Base (non vendor-specific) authentication application IDs. Empty because T6a is registered as a vendor-specific application (see <code>applications</code> ). |
| <code>acct_application_ids</code> | List of Integer | No       | <code>[]</code>      | Base accounting application IDs. Not used by OmniSCEF.                                                                                                        |

## Internal Handlers

These wire the Diameter stack to OmniSCEF's logic. They are fixed and should not be changed.

| Parameter                     | Type   | Required | Default                                |                                                     |
|-------------------------------|--------|----------|----------------------------------------|-----------------------------------------------------|
| <code>control_module</code>   | Module | Yes      | <code>OmniScef.Control.Diameter</code> | T<br>t<br>r<br>i<br>r<br>a<br>c<br>t<br>i<br>o<br>n |
| <code>processor_module</code> | Module | Yes      | <code>DiameterEx.Processor</code>      | T<br>E<br>S<br>T<br>T<br>R                          |

## T6a Application

The `applications` list registers the T6a Diameter application. This is fixed for OmniSCEF.

**Application entry parameters:**

| Parameter                                    | Type        | Required | Default                       |
|----------------------------------------------|-------------|----------|-------------------------------|
| <code>application_name</code>                | Atom        | Yes      | <code>:t6a</code>             |
| <code>application_dictionary</code>          | Atom        | Yes      | <code>:diameter_gen_3c</code> |
| <code>vendor_specific_application_ids</code> | List of Map | Yes      | see below                     |

**Vendor-specific application ID parameters:**

| Parameter                        | Type    | Required | Default          | Description             |
|----------------------------------|---------|----------|------------------|-------------------------|
| <code>vendor_id</code>           | Integer | Yes      | 10415            | 3GPP vendor ID.         |
| <code>auth_application_id</code> | Integer | Yes      | 16777346         | The T6a Application-Id. |
| <code>acct_application_id</code> | Integer | No       | <code>nil</code> | Not used for T6a.       |

# Peers

The `peers` list declares known Diameter peers (the MME, a DRA, and/or the HSS). Because `allow_undefined_peers_to_connect` defaults to `true` and the MME normally initiates the connection, `peers` may be left empty (`[]`) in deployments where nothing is dialled outbound. List a peer here when OmniSCEF must **initiate** the connection, or to restrict which peers may connect.

## Peer parameters:

| Parameter                        | Type    | Required | Default                     | Description                                                                                                                                 |
|----------------------------------|---------|----------|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <code>host</code>                | String  | Yes      | -                           | Peer's Diameter Identity (IDN). Must match peer's configured Origin-Host exactly.                                                           |
| <code>realm</code>               | String  | Yes      | -                           | Peer's Diameter Realm.                                                                                                                      |
| <code>ip</code>                  | String  | Yes      | -                           | Peer's IP address for TCP/SCTP connection.                                                                                                  |
| <code>port</code>                | Integer | No       | 3868                        | Peer's Diameter port.                                                                                                                       |
| <code>tls</code>                 | Boolean | No       | <code>false</code>          | Whether connect to this peer over TLS.                                                                                                      |
| <code>transport</code>           | Atom    | No       | <code>:diameter_sctp</code> | Transport to use for this peer. Valid values are <code>:diameter_tcp</code> , <code>:diameter_sctp</code> , or <code>:diameter_tls</code> . |
| <code>initiate_connection</code> | Boolean | No       | <code>false</code>          | When <code>transport</code> is <code>:diameter_tls</code> , OmniSCE actively connects to peer at startup.                                   |

| Parameter | Type | Required | Default | Description                                                                          |
|-----------|------|----------|---------|--------------------------------------------------------------------------------------|
|           |      |          |         | When false, the T8 API will not connect to the peer that MME dial SCEF, leave false. |

## T8 API Endpoint

The T8 RESTful API is served over HTTPS. All routes are mounted under the `/api` prefix, so the effective apiRoot is `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: [ ... ] # fixed T8 route table – see below
  }
```

# Endpoint Parameters

| Parameter    | Type    | Required | Default           | Description                                                         |
|--------------|---------|----------|-------------------|---------------------------------------------------------------------|
| port         | Integer | No       | 8443              | HTTPS port for the T8 API.                                          |
| listen_ip    | String  | No       | 0.0.0.0           | IP address to bind for the T8 API. 0.0.0.0 binds to all interfaces. |
| product_name | String  | No       | OmniSCEF          | Product name used in the API/Omniscap meta-information.             |
| title        | String  | No       | T8 API - OmniSCEF | Title string used in the generated OpenAPI document and Swagger UI. |
| hostname     | String  | No       | localhost         | Hostname used to generate the endpoint URL.                         |
| enable_tls   | Boolean | No       | true              | Whether to enable TLS for the endpoint.                             |

| Parameter                  | Type   | Required     | Default                              | Description                                                                                                                           |
|----------------------------|--------|--------------|--------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|
|                            |        |              |                                      | serve HTTPS. Specify the certificate path below.                                                                                      |
| <code>tls_cert_path</code> | String | Yes (if TLS) | <code>priv/cert/omnitouch.crt</code> | Path to PEM-encoded certificate.                                                                                                      |
| <code>tls_key_path</code>  | String | Yes (if TLS) | <code>priv/cert/omnitouch.pem</code> | Path to PEM-encoded private key.                                                                                                      |
| <code>routes</code>        | List   | Yes          | fixed                                | The T8 route table. The first entry defines the API surface. If the API surface is not operational, see <a href="#">Route Table</a> . |

## Route Table

The `routes` entries define the T8 API surface. They are fixed for OmniSCEF and produce the endpoints below (all under `/api`). Each entry maps a URL template to a handler and the set of actions it serves.

### Route entry parameters:

| Parameter               | Type                       | Description                                                                                                                                                                                                 |
|-------------------------|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>path</code>       | String                     | The URL template. Segments like <code>:scsAsId</code> are path variables.                                                                                                                                   |
| <code>module</code>     | Module                     | The handler that serves the route.                                                                                                                                                                          |
| <code>parameters</code> | String or <code>nil</code> | The name of the member-resource path variable (e.g. <code>configurationId</code> ), or <code>nil</code> for collection-only routes.                                                                         |
| <code>actions</code>    | List of Atom               | The HTTP actions served: <code>:index</code> (GET collection), <code>:create</code> (POST), <code>:show</code> (GET member), <code>:update</code> (PUT/PATCH member), <code>:delete</code> (DELETE member). |

**Resulting endpoints:**

| Method & Path                                                                                                             | Purpose                                        |
|---------------------------------------------------------------------------------------------------------------------------|------------------------------------------------|
| GET /api/status                                                                                                           | Service liveness identity                      |
| GET /api/3gpp-nidd/v1/{scsAsId}/configurations                                                                            | List NIDD configurations                       |
| POST /api/3gpp-nidd/v1/{scsAsId}/configurations                                                                           | Create configuration                           |
| GET/PUT/PATCH/DELETE /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}                                         | Read / update / delete NIDD configuration      |
| GET/POST /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}/downlink-data-deliveries                            | List / send downlink data                      |
| GET/DELETE /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}/downlink-data-deliveries/{downlinkDataDeliveryId} | Read / cancel downlink data delivery           |
| GET/POST /api/3gpp-monitoring-event/v1/{scsAsId}/subscriptions                                                            | List / create monitoring subscription          |
| GET/PUT/DELETE /api/3gpp-monitoring-event/v1/{scsAsId}/subscriptions/{subscriptionId}                                     | Read / update / delete monitoring subscription |

The generated OpenAPI document is available at `GET /api/schema`, and an interactive Swagger UI at `GET /api/docs`.

**Optional shared-key authentication.** The T8 endpoint supports an optional shared-key check on inbound requests. It is disabled unless an `auth` key is added under `config :api_ex, :api`. When unset (the default), no shared-key check is applied and the endpoint should be protected by network controls (e.g. a private interconnect or reverse proxy).

## Identity & Group Settings

OmniSCEF resolves the AS-facing identity (External Identifier or MSISDN) to the network IMSI, and resolves a group identifier to its member IMSIs. In the first release these are static operator maps.

```
config :omni_scef,
  nidd: %{
    identity_map: %{
      "device01@iot.example.org" => "0010100000000001",
      "61400000001" => "0010100000000001"
    },
    group_map: %{
      "fleet-a@iot.example.org" => ["0010100000000001",
"0010100000000002"]
    },
    default_max_packet_size: 255
  }
```

| Parameter                            | Type    | Required | Default          | Description                                                                                                                                                                                          |
|--------------------------------------|---------|----------|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>identity_map</code>            | Map     | No       | <code>%{}</code> | Maps an External Identifier (user@domain) to an MSISDN to subscriber IMSI. Used when the NIDD configuration or monitoring subscription. If identity not present here is rejected as an unknown IMSI. |
| <code>group_map</code>               | Map     | No       | <code>%{}</code> | Maps an externalGroup to the list of member IMSIs. Used for group NIDD distribution.                                                                                                                 |
| <code>default_max_packet_size</code> | Integer | No       | 255              | The default maximum No payload size, octets, applied to NIDD configuration that does not specify <code>maximumPacketSize</code> . The NAS transaction caps a single message at 255 octets.           |

In production these maps are typically backed by an HSS/UDR lookup rather than static configuration. The AS-facing behaviour is unchanged.

# Monitoring Settings

Optional. When a monitoring destination is configured, creating a monitoring subscription arms the event southbound by sending a Configuration-Information-Request (CIR) to that node. When it is not set, subscriptions are still stored and inbound reports are still delivered — OmniSCEF simply does not proactively send the CIR.

```
config :omni_scef,  
  monitoring: %{\br/>    destination: %{\br/>      host: "hss01.epc.mnc380.mcc313.3gppnetwork.org",  
      realm: "epc.mnc380.mcc313.3gppnetwork.org"  
    }  
  }  
}
```

## **destination** parameters:

| Parameter    | Type   | Required                        | Default | Description                                                                                                                                               |
|--------------|--------|---------------------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>host</b>  | String | No                              | -       | Diameter Identity (FQDN) of the HSS / serving node that monitoring configuration requests are sent to. When omitted, the request is routed by realm only. |
| <b>realm</b> | String | Yes (if <b>destination</b> set) | -       | Diameter Realm used as the Destination-Realm of the monitoring CIR.                                                                                       |

# Environment Variables

The Diameter service identity and listeners can be supplied from the environment at start-up. These override the defaults shown under [Diameter Service](#).

| Variable                  | Maps to     | Default                    |
|---------------------------|-------------|----------------------------|
| 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.mnc380.mcc313.3gppnetw |

# Monitoring Events

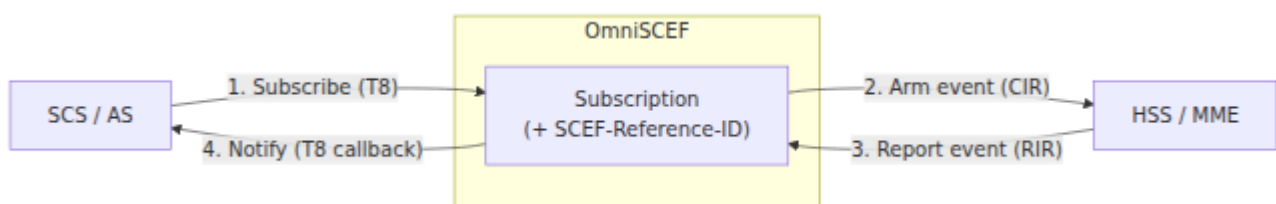
The Monitoring Event service lets an Application Server subscribe to network events about a device or group — reachability, loss of connectivity, location, roaming status, and more. The Application Server subscribes over **T8**; OmniSCEF arms the event in the network over **T6a** and delivers each report back to the subscriber's callback.

Monitoring events are defined in [3GPP TS 23.682 §5.6](#), the T8 API in [TS 29.122 §5.3](#), and the Diameter procedures in [TS 29.128](#) / [TS 29.336](#).

## Table of Contents

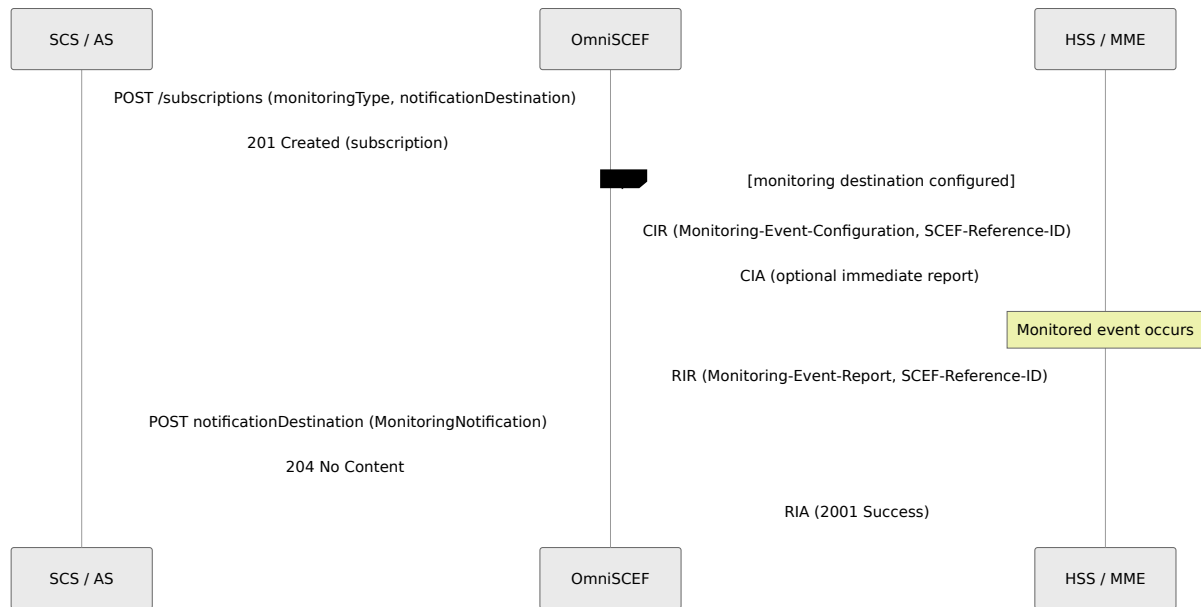
- [Overview](#)
- [Subscription and Reporting Flow](#)
- [T8 Resource Reference](#)
- [Monitoring Types](#)
- [Reference Values](#)

## Overview



Each subscription is assigned a **SCEF-Reference-ID** — a unique identifier that OmniSCEF places in the southbound configuration and that the network echoes back on every report. OmniSCEF uses it to route an inbound report to the correct subscription and its Application Server callback.

# Subscription and Reporting Flow



**Arming is optional at the SCEF.** If a **monitoring destination** is configured, OmniSCEF sends the CIR to arm the event. If it is not configured, the subscription is still created and OmniSCEF still delivers any inbound reports — it simply does not proactively send the CIR (for example, when the event is armed via the HSS by another path).

A report whose SCEF-Reference-ID does not match any subscription is acknowledged with an RIA but not forwarded.

## T8 Resource Reference

### MonitoringEventSubscription

Created by `POST /api/3gpp-monitoring-event/v1/{scsAsId}/subscriptions`.

| Field                                | Type               | Required    | Description                                                                                                          |
|--------------------------------------|--------------------|-------------|----------------------------------------------------------------------------------------------------------------------|
| <code>notificationDestination</code> | String (URI)       | Yes         | Callback the AS receives monitoring reports on.                                                                      |
| <code>monitoringType</code>          | String (enum)      | Yes         | The event to monitor — see <a href="#">Monitoring Types</a> .                                                        |
| <code>externalId</code>              | String             | Conditional | Device External Identifier. One of <code>externalId</code> , <code>msisdn</code> , or <code>externalGroupId</code> . |
| <code>msisdn</code>                  | String             | Conditional | Device MSISDN.                                                                                                       |
| <code>externalGroupId</code>         | String             | Conditional | Group identifier for group monitoring.                                                                               |
| <code>maximumNumberOfReports</code>  | Integer            | No          | Stop after this many reports.                                                                                        |
| <code>monitoringDuration</code>      | String (date-time) | No          | Stop monitoring at this time.                                                                                        |
| <code>reachabilityType</code>        | String (enum)      | No          | For <code>UE_REACHABILITY</code> : <code>SMS</code> or <code>DATA</code> .                                           |
| <code>maximumDetectionTime</code>    | Integer            | No          | For <code>LOSS_OF_CONNECTIVITY</code> : detection time in seconds.                                                   |
| <code>mtcProviderId</code>           | String             | No          | MTC service provider identifier.                                                                                     |

| Field             | Type         | Required  | Description                 |
|-------------------|--------------|-----------|-----------------------------|
| <code>self</code> | String (URI) | Read-only | The subscription's own URI. |

### Example subscription request:

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

## MonitoringNotification

OmniSCEF POSTs a `MonitoringNotification` to the subscription's `notificationDestination` for each report. It references the originating subscription and carries one or more decoded event reports.

### Example notification:

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

The Application Server acknowledges with `204 No Content`.

# Monitoring Types

OmniSCEF supports the full TS 29.122 monitoring-type set. Each maps to the corresponding `Monitoring-Type` value on the Diameter interface ([TS 29.336 §5.3.9](#)).

| <b>monitoringType (T8)</b>      | <b>Diameter value</b> | <b>Description</b>                                                  |
|---------------------------------|-----------------------|---------------------------------------------------------------------|
| LOSS_OF_CONNECTIVITY            | 0                     | The UE is no longer reachable for data.                             |
| UE_REACHABILITY                 | 1                     | The UE has become reachable for SMS or data.                        |
| LOCATION_REPORTING              | 2                     | The UE's location (or a change in it) is reported.                  |
| CHANGE_OF_IMSI_IMEI_ASSOCIATION | 3                     | The IMEI associated with the IMSI changed.                          |
| ROAMING_STATUS                  | 4                     | The UE's roaming status (or a change) is reported.                  |
| COMMUNICATION_FAILURE           | 5                     | A communication failure event occurred.                             |
| AVAILABILITY_AFTER_DDN_FAILURE  | 6                     | The UE became available after a downlink data notification failure. |
| NUMBER_OF_UES_IN_AN_AREA        | 7                     | Count of UEs present in a given area.                               |
| PDN_CONNECTIVITY_STATUS         | 8                     | PDN connection established/released.                                |
| DOWNLINK_DATA_DELIVERY_STATUS   | 9                     | Status of downlink data delivery.                                   |

| <code>monitoringType</code> (T8)    | Diameter value | Description                          |
|-------------------------------------|----------------|--------------------------------------|
| <code>API_SUPPORT_CAPABILITY</code> | 10             | Reporting of API support capability. |

A subscription with an unsupported `monitoringType` is rejected with `400 INVALID_MONITORING_TYPE`.

## Reference Values

### Reachability Type

| Value             | Meaning                       |
|-------------------|-------------------------------|
| <code>SMS</code>  | The UE is reachable for SMS.  |
| <code>DATA</code> | The UE is reachable for data. |

## Related Diameter Commands

See the [Operations Guide reference tables](#) for the CIR/CIA and RIR/RIA command and result codes.

# Non-IP Data Delivery (NIDD)

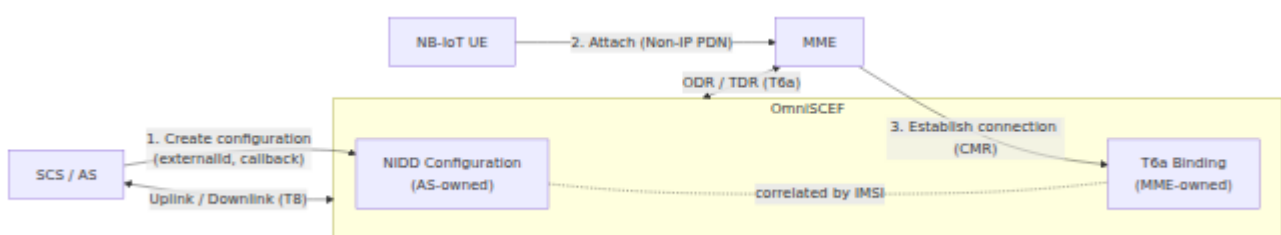
**Non-IP Data Delivery** lets a Cellular-IoT / NB-IoT device exchange small **Non-IP** messages with an Application Server **without ever being assigned an IP address**. The data rides the NAS signalling plane (the Control-Plane CloT EPS optimisation) between the UE and the MME; the MME relays it to OmniSCEF over **T6a**; and OmniSCEF exposes it to the Application Server over **T8**.

NIDD architecture and procedures are defined in [3GPP TS 23.682 §5.13](#); the T6a Diameter interface in [TS 29.128](#); and the T8 NIDD API in [TS 29.122](#).

## Table of Contents

- [Overview](#)
- [Connection Lifecycle](#)
- [Mobile-Originated Data](#)
- [Mobile-Terminated Data](#)
- [Group NIDD](#)
- [Reliable Data Service](#)
- [MO Exception Data](#)
- [T8 Resource Reference](#)
- [Reference Values](#)

## Overview



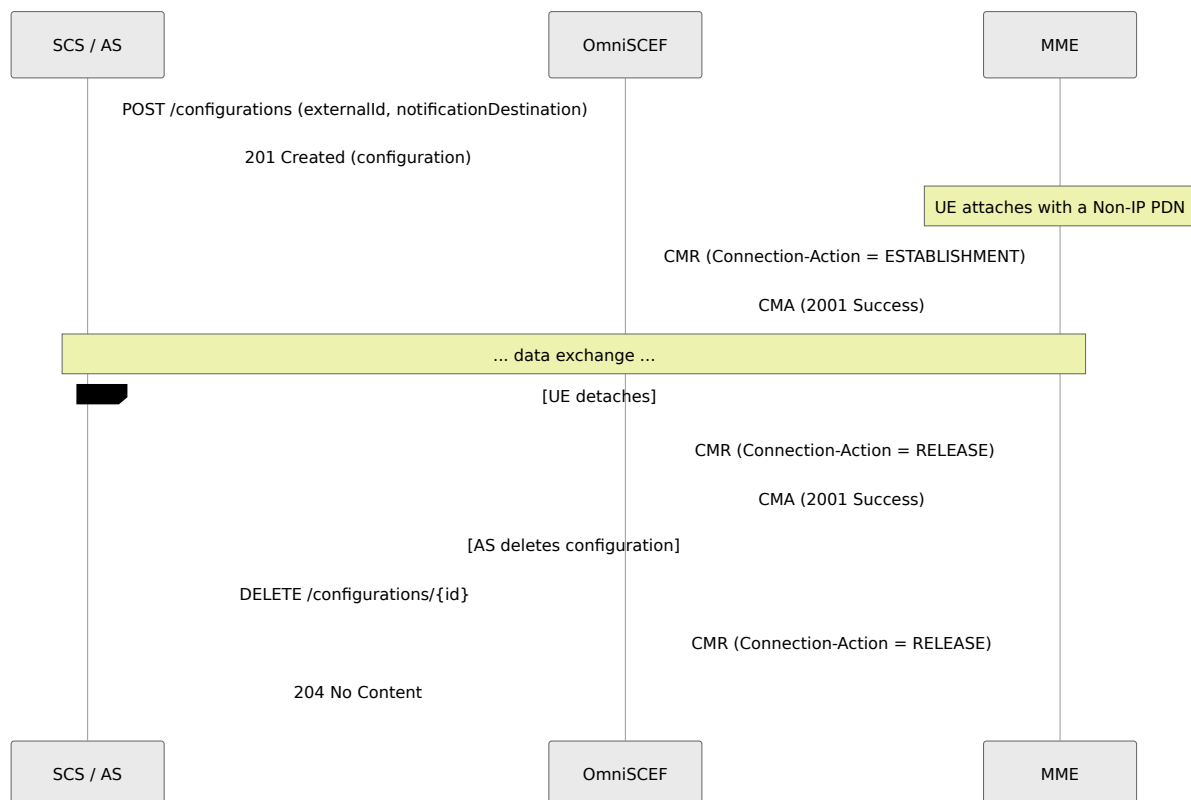
A NIDD session has two correlated halves:

- The **NIDD Configuration** is created by the Application Server over T8. It names the device (by External Identifier, MSISDN, or group), the notification callback, and NIDD options.
- The **T6a Binding** is created when the MME establishes the Non-IP connection for that subscriber. It ties the Diameter session and the MME identity to the {IMSI, EBI} used to reach the UE.

OmniSCEF correlates the two by subscriber identity, so uplink data finds the right AS callback and downlink data finds the right live connection.

## Connection Lifecycle

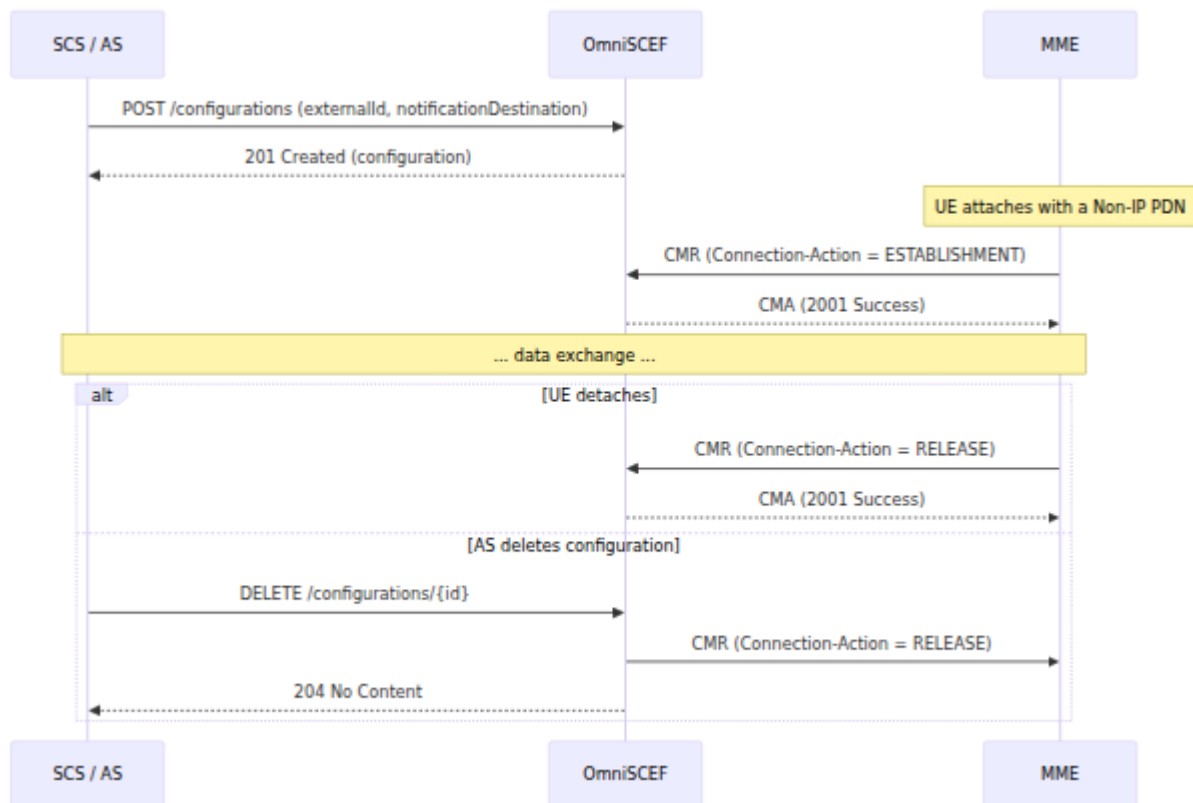
The MME manages the T6a connection as the UE attaches and detaches. The Application Server can also end a connection by deleting its NIDD configuration.



If a connection is established for a subscriber that has **no** NIDD configuration, OmniSCEF answers the CMR with `DIAMETER_ERROR_USER_UNKNOWN (5001)` — the subscriber is not provisioned for NIDD.

# Mobile-Originated Data (Uplink)

Uplink Non-IP data sent by the UE arrives at OmniSCEF as a T6a **MO-Data-Request (ODR)**. OmniSCEF delivers it to the Application Server by POSTing a notification to the configuration's callback, and answers the MME with an **MO-Data-Answer (ODA)**.



If OmniSCEF cannot resolve the configuration or the callback fails, it answers the ODA with `DIAMETER_UNABLE_TO_COMPLY (5012)`.

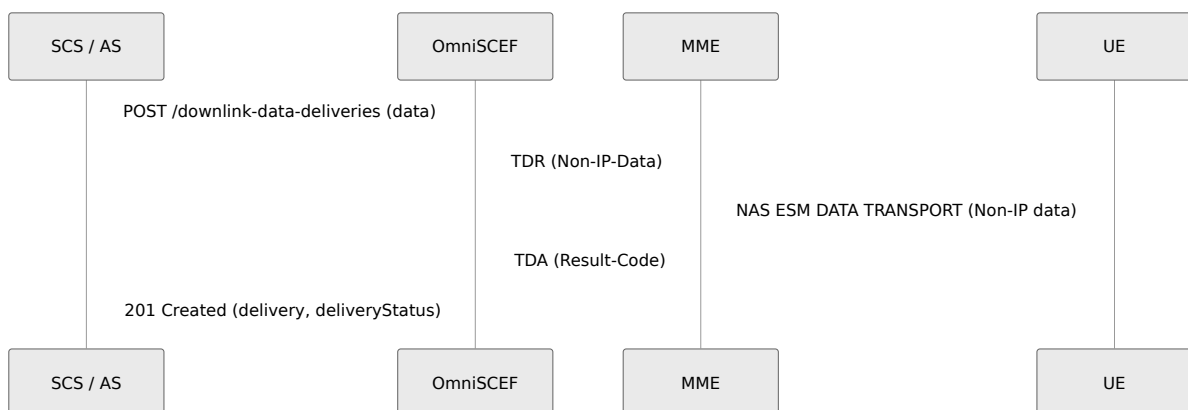
**Example uplink notification** (POSTed to the callback):

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

| Field                                                        | Description                                                       |
|--------------------------------------------------------------|-------------------------------------------------------------------|
| <code>externalId</code> / <code>msisdn</code>                | The device identity, as provisioned on the configuration.         |
| <code>data</code>                                            | The Non-IP payload, base64-encoded.                               |
| <code>rdsSourcePort</code> / <code>rdsDestinationPort</code> | Present when <b>Reliable Data Service</b> is enabled.             |
| <code>moExceptionDataCounter</code>                          | Present when the uplink was flagged as <b>MO exception data</b> . |

## Mobile-Terminated Data (Downlink)

The Application Server sends downlink Non-IP data by POSTing to the configuration's `downlink-data-deliveries` collection. OmniSCEF forwards it to the MME as a T6a **MT-Data-Request (TDR)**, and records the outcome (from the **MT-Data-Answer, TDA**) on an individual delivery resource that the AS can retrieve.



When the UE is idle, the MME buffers the data and pages the UE, answering the TDA with success once it has taken responsibility for delivery. OmniSCEF reflects the outcome in the delivery's `deliveryStatus` (see **Delivery Status Values**).

**Example downlink request:**

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

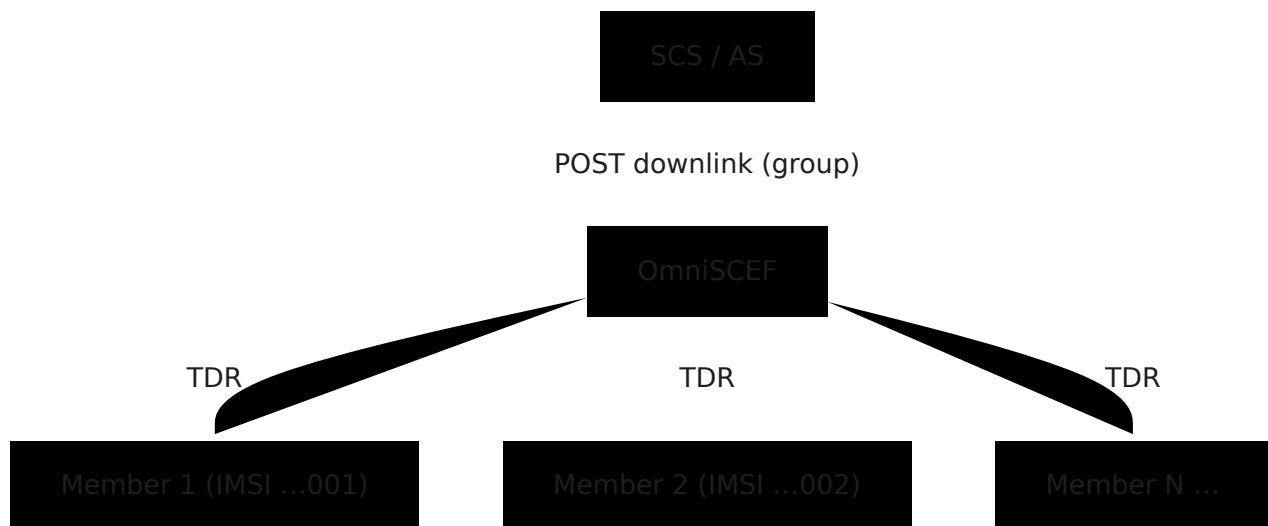
| Field                                                        | Description                                                                                     |
|--------------------------------------------------------------|-------------------------------------------------------------------------------------------------|
| <code>data</code>                                            | The Non-IP payload to deliver, base64-encoded. Required.                                        |
| <code>maximumLatency</code>                                  | Optional. Maximum acceptable buffering delay, in seconds.                                       |
| <code>priority</code>                                        | Optional. Relative priority among queued deliveries.                                            |
| <code>pdnEstablishmentOption</code>                          | Optional. Behaviour when no connection exists — see <a href="#">PDN Establishment Options</a> . |
| <code>rdsSourcePort</code> / <code>rdsDestinationPort</code> | Optional. RDS ports — see <a href="#">Reliable Data Service</a> .                               |

### Example downlink response:

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

# Group NIDD

A NIDD configuration created with an `externalGroupId` (instead of a single `externalId/msisdn`) represents a **group** of devices (TS 23.682 §5.13.3A). A single downlink delivery to a group configuration is **fanned out** to every group member that has an active T6a connection.



The group's members are resolved from the **group map**. The delivery's aggregate `deliveryStatus` reflects the **worst** per-member outcome, so a group delivery only reports `SUCCESS` when every reachable member was served.

## Reliable Data Service

Reliable Data Service (RDS) multiplexes multiple application flows over a single Non-IP connection using **port numbers**, per TS 23.682 §4.5.14.3. RDS is enabled per configuration by setting `reliableDataService: true` when the configuration is created.

When RDS is enabled:

- **Downlink** — if the delivery specifies `rdsDestinationPort` and `rdsSourcePort`, OmniSCEF frames the Non-IP payload with an RDS header carrying those ports before sending the TDR.
- **Uplink** — OmniSCEF parses the RDS header from the ODR payload and surfaces `rdsSourcePort` and `rdsDestinationPort` on the uplink

notification, delivering the inner payload as `data`.

Ports are 0–255. When RDS is not enabled, or a delivery omits the ports, the payload is carried unframed.

## MO Exception Data

NB-IoT devices can flag an uplink transmission as **exception reporting** (an urgent, out-of-pattern report) via the RRC establishment cause, counted in the T6a `RRC-Cause-Counter` ([TS 23.682 §5.13.4](#)). When OmniSCEF sees this indication on an ODR, it includes an `moExceptionDataCounter` on the uplink notification so the Application Server can prioritise handling.

## T8 Resource Reference

### NiddConfiguration

Created by `POST /api/3gpp-nidd/v1/{scsAsId}/configurations`.

| Field                                | Type          | Required    | Description                                                                                                                        |
|--------------------------------------|---------------|-------------|------------------------------------------------------------------------------------------------------------------------------------|
| <code>notificationDestination</code> | String (URI)  | Yes         | Callback the AS receive uplink and status notifications on.                                                                        |
| <code>externalId</code>              | String        | Conditional | Device External Identifier (user@domain). One of <code>externalId</code> , <code>msisdn</code> , or <code>externalGroupId</code> . |
| <code>msisdn</code>                  | String        | Conditional | Device MSISDN.                                                                                                                     |
| <code>externalGroupId</code>         | String        | Conditional | Group identifier for <a href="#">group NIDD</a> .                                                                                  |
| <code>dnn</code>                     | String        | No          | Data Network Name / Address of the Non-IP connection.                                                                              |
| <code>pdnEstablishmentOption</code>  | String (enum) | No          | Behaviour on downlink when no connection exists — see <a href="#">PDN Establishment Options</a> .                                  |
| <code>maximumPacketSize</code>       | Integer       | No          | Maximum Non-IP payload size in octets (defaults to the configured <code>default_max_packet_size</code> 255).                       |
| <code>reliableDataService</code>     | Boolean       | No          | Enable <a href="#">Reliable Data Service</a> .                                                                                     |
| <code>mtcProviderId</code>           | String        | No          | MTC service provider identifier.                                                                                                   |

| Field                     | Type               | Required  | Description                                                          |
|---------------------------|--------------------|-----------|----------------------------------------------------------------------|
| <code>validityTime</code> | String (date-time) | No        | When the configuration expires.                                      |
| <code>status</code>       | String (enum)      | Read-only | Lifecycle status — see <a href="#">Configuration Status Values</a> . |
| <code>self</code>         | String (URI)       | Read-only | The resource's own UR                                                |

## NiddDownlinkDataTransfer

Created by `POST /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}/downlink-data-deliveries`. Fields are described under [Mobile-Terminated Data](#). The `deliveryStatus` and `requestedRetransmissionTime` fields are read-only and set from the TDA.

## Notifications

OmniSCEF POSTs the following to the configuration's `notificationDestination`:

| Notification                                            | When                                                                                  |
|---------------------------------------------------------|---------------------------------------------------------------------------------------|
| <code>NiddUplinkDataNotification</code>                 | Uplink Non-IP data arrived (from an ODR).                                             |
| <code>NiddDownlinkDataDeliveryStatusNotification</code> | The outcome of a previously accepted downlink delivery changed (asynchronous status). |

The Application Server acknowledges each notification with `204 No Content`.

# Reference Values

## Delivery Status Values

| Value                                 | Meaning                                                                             |
|---------------------------------------|-------------------------------------------------------------------------------------|
| <code>SUCCESS</code>                  | The MME delivered the data (or took responsibility by buffering/paging an idle UE). |
| <code>SENDING</code>                  | The delivery is in progress.                                                        |
| <code>BUFFERING</code>                | The MME buffered the data for an idle UE.                                           |
| <code>FAILURE_UE_NOT_REACHABLE</code> | The UE could not be reached.                                                        |
| <code>FAILURE_REMOTE_FAILURE</code>   | The MME / serving node rejected or failed the request.                              |
| <code>FAILURE</code>                  | The delivery failed for another reason.                                             |

## PDN Establishment Options

| Value                                 | Behaviour when no connection exists at downlink time |
|---------------------------------------|------------------------------------------------------|
| <code>WAIT_FOR_UE</code>              | Buffer until the UE establishes the connection.      |
| <code>INDICATE_ERROR</code>           | Reject the delivery immediately.                     |
| <code>ESTABLISH_PDN_CONNECTION</code> | Trigger the UE to establish the Non-IP connection.   |

## Configuration Status Values

| Value                        | Meaning                                        |
|------------------------------|------------------------------------------------|
| ACTIVE                       | The configuration is authorised and usable.    |
| TERMINATED_UE_NOT_AUTHORIZED | The subscriber is not authorised for NIDD.     |
| TERMINATED                   | The configuration has expired or been removed. |

# OmniSCEF

# Troubleshooting

Problem/resolution guidance for common OmniSCEF operational issues. For the underlying flows, see [NIDD](#) and [Monitoring Events](#); for parameters, see the [Configuration Reference](#).

## Table of Contents

- [MME Cannot Connect Over T6a](#)
- [Connection Rejected With USER\\_UNKNOWN \(5001\)](#)
- [Downlink Delivery Fails](#)
- [Uplink Data Not Reaching the Application Server](#)
- [Group Downlink Reaches No Devices](#)
- [Monitoring Reports Not Delivered](#)
- [T8 API Returns 404 for a Valid Endpoint](#)
- [T8 Endpoint Will Not Start](#)

## MME Cannot Connect Over T6a

**Symptoms:** The MME cannot establish a Diameter connection to OmniSCEF; no capabilities exchange completes.

**Possible causes:**

- Firewall blocking the Diameter port (default 3868).
- Transport mismatch — the MME uses SCTP while only TCP is reachable, or vice versa.
- Peer authorisation is restricted and the MME is not a configured peer.
- Origin-Host / realm mismatch between what the MME expects and what OmniSCEF advertises.

### Resolution:

1. Confirm the firewall allows the configured `listen_port` on both TCP and SCTP — OmniSCEF listens on both. See [Listeners](#).
2. Verify the MME's configured transport matches. Both TCP and SCTP listeners are always started.
3. If `allow_undefined_peers_to_connect` is `false`, add the MME to `peers` with its exact Origin-Host. See [Peers](#).
4. Confirm OmniSCEF's `host` and `realm` match what the MME is provisioned to talk to. Check the `log_unauthorized_peer_connection_attempts` log output for rejected attempts.

## Connection Rejected With USER\_UNKNOWN (5001)

**Symptoms:** The MME establishes the Diameter connection, but a NIDD connection request (CMR) is answered with `DIAMETER_ERROR_USER_UNKNOWN (5001)`.

### Possible causes:

- No NIDD configuration exists for that subscriber's IMSI.
- The Application Server created a configuration using an external identity that does not resolve to the subscriber's IMSI.

### Resolution:

1. Confirm the Application Server has created a NIDD configuration for the device before the UE attaches.
2. Verify the device's `externalId/msisdn` resolves to the correct IMSI in the [identity map](#).
3. Confirm the IMSI in the identity map matches the IMSI the MME presents in the CMR.

# Downlink Delivery Fails

**Symptoms:** A `POST` to `downlink-data-deliveries` returns a delivery whose `deliveryStatus` is a `FAILURE_*` value, or a `404`.

## Possible causes:

- No active T6a connection exists for the device (`404 NIDD_CONFIGURATION_NOT_AVAILABLE`) — the UE has not attached, or its connection was released.
- The UE is unreachable (`FAILURE_UE_NOT_REACHABLE`).
- The MME / serving node rejected the request (`FAILURE_REMOTE_FAILURE`).
- The `data` field is missing or not valid base64 (`400 DATA_MISSING`).

## Resolution:

1. Confirm the device is attached and its connection is established (a prior CMR was accepted).
2. Check the `data` field is present and base64-encoded.
3. For an idle UE, expect the MME to buffer and page; the delivery reports success once the MME takes responsibility. See [Mobile-Terminated Data](#).
4. For `FAILURE_REMOTE_FAILURE`, inspect the MME/serving-node side for the rejection cause.

# Uplink Data Not Reaching the Application Server

**Symptoms:** The UE sends uplink data, but the Application Server's callback is not invoked.

## Possible causes:

- The device has no NIDD configuration (so no callback is known).
- The `notificationDestination` URL is unreachable from OmniSCEF, or returns a non-2xx status.
- No active binding exists for the `{IMSI, EBI}` presented on the uplink.

**Resolution:**

1. Confirm a NIDD configuration exists for the device and carries a reachable `notificationDestination`.
2. Verify network connectivity from OmniSCEF to the callback URL, including TLS trust if it is HTTPS.
3. Confirm the callback returns a 2xx status; a non-2xx is treated as a delivery failure.
4. Confirm the device attached (a CMR was accepted) before sending uplink data.

## Group Downlink Reaches No Devices

**Symptoms:** A downlink to a group configuration returns `404` `NIDD_CONFIGURATION_NOT_AVAILABLE`, or reaches fewer devices than expected.

**Possible causes:**

- The `externalGroupId` is not present in the group map, or maps to no members.
- No group member currently has an active T6a connection.

**Resolution:**

1. Confirm the `externalGroupId` is defined in the **group map** and lists the expected member IMSIs.
2. Confirm at least one member device is attached — only members with an active connection receive the delivery. The aggregate status reflects the worst per-member outcome. See **Group NIDD**.

## Monitoring Reports Not Delivered

**Symptoms:** A monitoring subscription is created, but the Application Server receives no notifications.

**Possible causes:**

- The event was never armed in the network (no monitoring destination configured, and no other path armed it).
- The report's SCEF-Reference-ID does not match a live subscription (for example, after a restart — subscription state is in-memory).
- The subscription's `notificationDestination` is unreachable.

**Resolution:**

1. If OmniSCEF should arm the event, configure a **monitoring destination** so the CIR is sent.
2. Recreate subscriptions after an OmniSCEF restart — subscription and reference-ID state is volatile and does not survive a restart.
3. Verify connectivity from OmniSCEF to the subscription callback.

## T8 API Returns 404 for a Valid Endpoint

**Symptoms:** A request to a documented T8 path returns `404`, and the endpoint does not appear in the OpenAPI document at `/api/schema`.

**Possible causes:**

- The route table was changed but the API layer was not recompiled, so the new route is not registered.
- The request omits the `/api` prefix that all T8 routes are mounted under.

**Resolution:**

1. Confirm the request path includes the `/api` prefix (e.g. `/api/3gpp-nidd/v1/{scsAsId}/configurations`).
2. If the **route table** was modified, the API dependency must be recompiled so the new routes register (`mix deps.compile api_ex --force`), then the service restarted.

# T8 Endpoint Will Not Start

**Symptoms:** OmniSCEF fails to start the T8 HTTPS endpoint.

**Possible causes:**

- The TLS certificate or key file is missing or unreadable.
- The configured `port` is already in use.

**Resolution:**

1. Confirm `tls_cert_path` and `tls_key_path` point to valid, readable PEM files. See [T8 API Endpoint](#).
2. Confirm nothing else is bound to the configured `port` on the `listen_ip` interface.
3. To run without TLS in a controlled environment, set `enable_tls: false` — but production deployments should always terminate TLS at or before OmniSCEF.

