OmniSCEF Configuration Reference

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

Scope	Purpose
`:diameter_ex`	The **T6a** Diameter service: identity, listeners, application, and peers.
`:api_ex`	The **T8** REST API endpoint: TLS, bind address, and the route table.
`:omni_scef`	OmniSCEF behaviour: identity resolution, group membership, and monitoring.

OmniSCEF evaluates the Diameter service parameters at start-up, so the environment can drive them (see Environment Variables). The T8 endpoint and route table are fixed at build time.

Table of Contents

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.mnc001.mcc001.3gppnetwork.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.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
      }
    ]
  }
```

Service Identity

Parameter	Type	Required	Default
`service_name`	Atom	Yes	`:omni_scef`
`host`	String	Yes	`scef01`
`realm`	String	Yes	`epc.mnc001.mcc001.3gppnetwork.org`
`product_name`	String	No	`OmniSCEF`

Listeners

Parameter	Type	Required	Default	Description
`listen_ip`	String	No	`0.0.0.0`	IP address to bind for inbound Diameter connections. Use `0.0.0.0` for all interfaces.
`listen_port`	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 RFC 6733.

Behaviour

Parameter	Type	Required	Def
`request_timeout`	Integer	No	500
`peer_selection_algorithm`	Atom	No	`:ra`
`allow_undefined_peers_to_connect`	Boolean	No	`tru`

Parameter	Type	Required	Def
`log_unauthorized_peer_connection_attempts`	Boolean	No	`tru`

Diameter Capabilities

Parameter	Type	Required	Default	Description
`vendor_id`	Integer	No	10415	Vendor-Id advertised in capabilities. 10415 is 3GPP.
`supported_vendor_ids`	List of Integer	No	`[10415]`	Supported-Vendor-Ids advertised in capabilities.
`auth_application_ids`	List of Integer	No	`[]`	Base (non vendor-specific) authentication application IDs. Empty because T6a is registered as a vendor-specific application (see `applications`).
`acct_application_ids`	List of Integer	No	`[]`	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	D
`control_module`	Module	Yes	`OmniScef.Control.Diameter`	T t r i r a c t M M h
`processor_module`	Module	Yes	`DiameterEx.Processor`	T D s p t t r

T6a Application

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

Application entry parameters:

Parameter	Type	Required	Default
`application_name`	Atom	Yes	`:t6a`
`application_dictionary`	Atom	Yes	`:diameter_gen_3`
`vendor_specific_application_ids`	List of Map	Yes	see below

Vendor-specific application ID parameters:

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

Peers

The `peers` list declares known Diameter peers (the MME, a DRA, and the HSS). The `allow_undefined_peers_to_connect` parameter defaults to `true` and the MME normally initiates the connection. You can leave `peers` empty (`[]`) in deployments where OmniSCEF dials nothing outbound. List a peer here when OmniSCEF must **initiate** the connection, or to restrict which peers can connect.

Peer parameters:

Parameter	Type	Required	Default	Descri
`host`	String	Yes	-	Peer's Di Identity (Must mat peer's configure Origin-Hc exactly.
`realm`	String	Yes	-	Peer's Di Realm.
`ip`	String	Yes	-	Peer's IP address f TCP/SCTF connecti
`port`	Integer	No	3868	Peer's Di port.
`tls`	Boolean	No	`false`	Whether connecti this peer TLS.
`transport`	Atom	No	`:diameter_sctp`	Transport this peer `:diamete` or `:diamete`
`initiate_connection`	Boolean	No	`false`	When `tr` OmniSCE actively connects peer at s

Parameter	Type	Required	Default	Descri
				When `fa` OmniSCE for the pe connect inbound. peer that MME dial SCEF, lea `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	Desc
`port`	Integer	No	8443	HTTPS for th API.
`listen_ip`	String	No	`0.0.0.0`	IP add bind f T8 AP `0.0.0` all interf
`product_name`	String	No	`OmniSCEF`	Produ name in the API/O meta
`title`	String	No	`T8 API - OmniSCEF`	Title s in the gener Open docur and Swag
`hostname`	String	No	`localhost`	Hostn used gener the er URL.
`enable_tls`	Boolean	No	`true`	When the er

Parameter	Type	Required	Default	Desc
				serve HTTP the certifi below
`tls_cert_path`	String	Yes (if TLS)	`priv/cert/omnitouch.crt`	Path t PEM- encoc certifi
`tls_key_path`	String	Yes (if TLS)	`priv/cert/omnitouch.pem`	Path t PEM- encoc privat
`routes`	List	Yes	fixed	The T table. the fi defini the A surfac is not opera tunab Route

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

Resulting endpoints:

Method & Path	Pur
`GET /api/status`	Service livenes identity
`GET /api/3gpp-nidd/v1/{scsAsId}/configurations`	List NID configu
`POST /api/3gpp-nidd/v1/{scsAsId}/configurations`	Create configu
`GET/PUT/PATCH/DELETE /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}`	Read / / delete NIDD configu
`GET/POST /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}/downlink-data-deliveries`	List / s downli
`GET/DELETE /api/3gpp-nidd/v1/{scsAsId}/configurations/{configurationId}/downlink-data-deliveries/{downlinkDataDeliveryId}`	Read s cancel downli deliver
`GET/POST /api/3gpp-monitoring-event/v1/{scsAsId}/subscriptions`	List / c monito subscr
`GET/PUT/DELETE /api/3gpp-monitoring-event/v1/{scsAsId}/subscriptions/{subscriptionId}`	Read / / delete monito subscr

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

Parameter	Type	Required	Default	Descripti
`identity_map`	Map	No	`%{}`	Maps an Exte Identifier (`user@domain` an MSISDN to subscriber IM Used when th creates a sing NIDD configur or monitoring subscription. identity not p here is rejecte an unknown U
`group_map`	Map	No	`%{}`	Maps an `externalGrou` the list of mer IMSIs. Used fo group NIDD d
`default_max_packet_size`	Integer	No	255	The default maximum No payload size, octets, applie NIDD configur that does not specify `maximumPacke` The NAS trans caps a single message at 2 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 you configure a monitoring destination, creating a monitoring subscription arms the event southbound. OmniSCEF sends a Configuration-Information-Request (CIR) to that node. When you do not set a destination, OmniSCEF still stores subscriptions and still delivers inbound reports. In that case OmniSCEF does not proactively send the CIR.

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

`destination` parameters:

Parameter	Type	Required	Default	Description
`host`	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.
`realm`	String	Yes (if `destination` 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.mnc001.mcc001.3gppnetw`

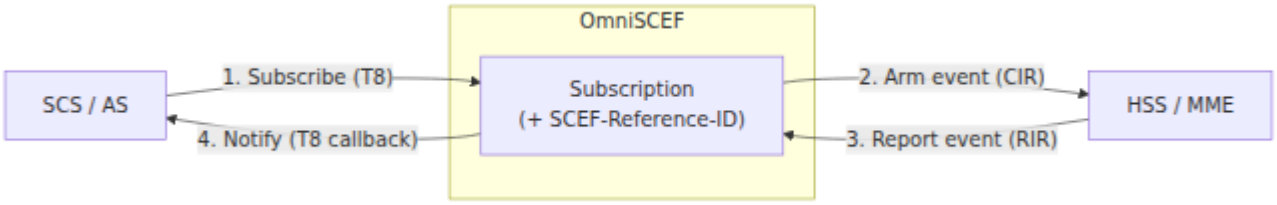
Monitoring Events

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.

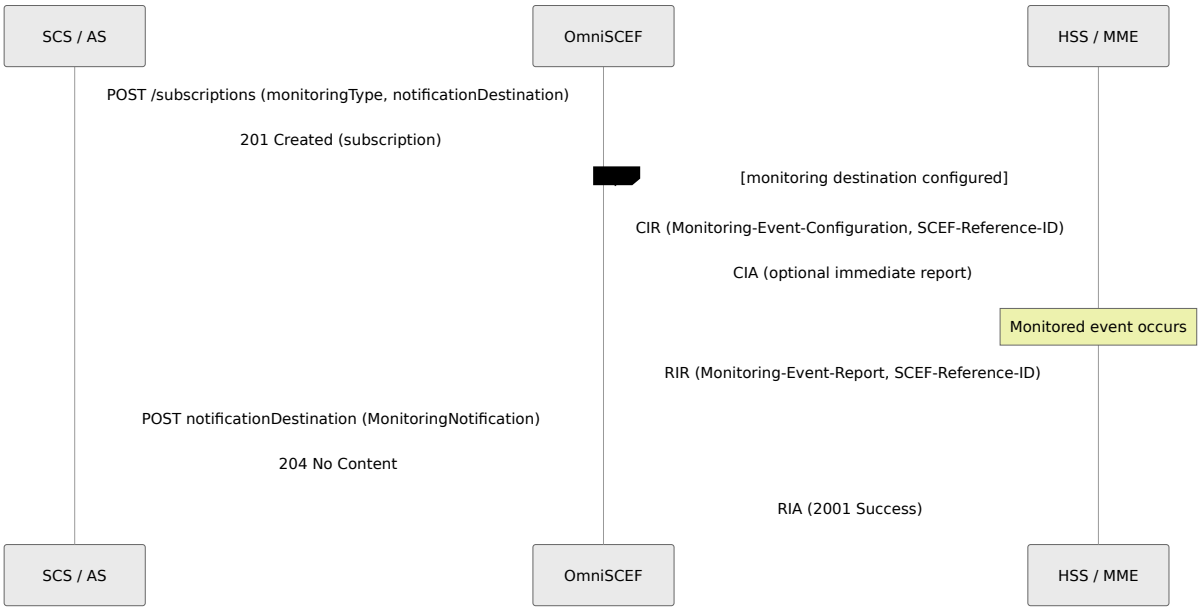
Table of Contents

Overview



OmniSCEF assigns each subscription a **SCEF-Reference-ID**. This unique identifier goes in the southbound configuration, and the network echoes it 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. When you configure a monitoring destination, OmniSCEF sends the CIR to arm the event. When you do not configure one, OmniSCEF still creates the subscription and still delivers any inbound reports. In that case OmniSCEF does not proactively send the CIR (for example, when another path arms the event through the HSS).

When a report's SCEF-Reference-ID does not match any subscription, OmniSCEF acknowledges it with an RIA but does not forward it.

T8 Resource Reference

MonitoringEventSubscription

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

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

Field	Type	Required	Description
`self`	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).

`monitoringType` (T8)	Diameter value	Description
`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.

`monitoringType` (T8)	Diameter value	Description
`API_SUPPORT_CAPABILITY`	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
`SMS`	The UE is reachable for SMS.
`DATA`	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.

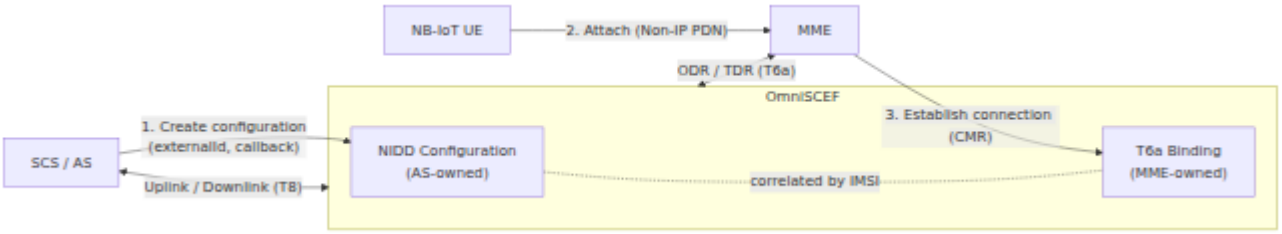
Non-IP Data Delivery (NIDD)

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



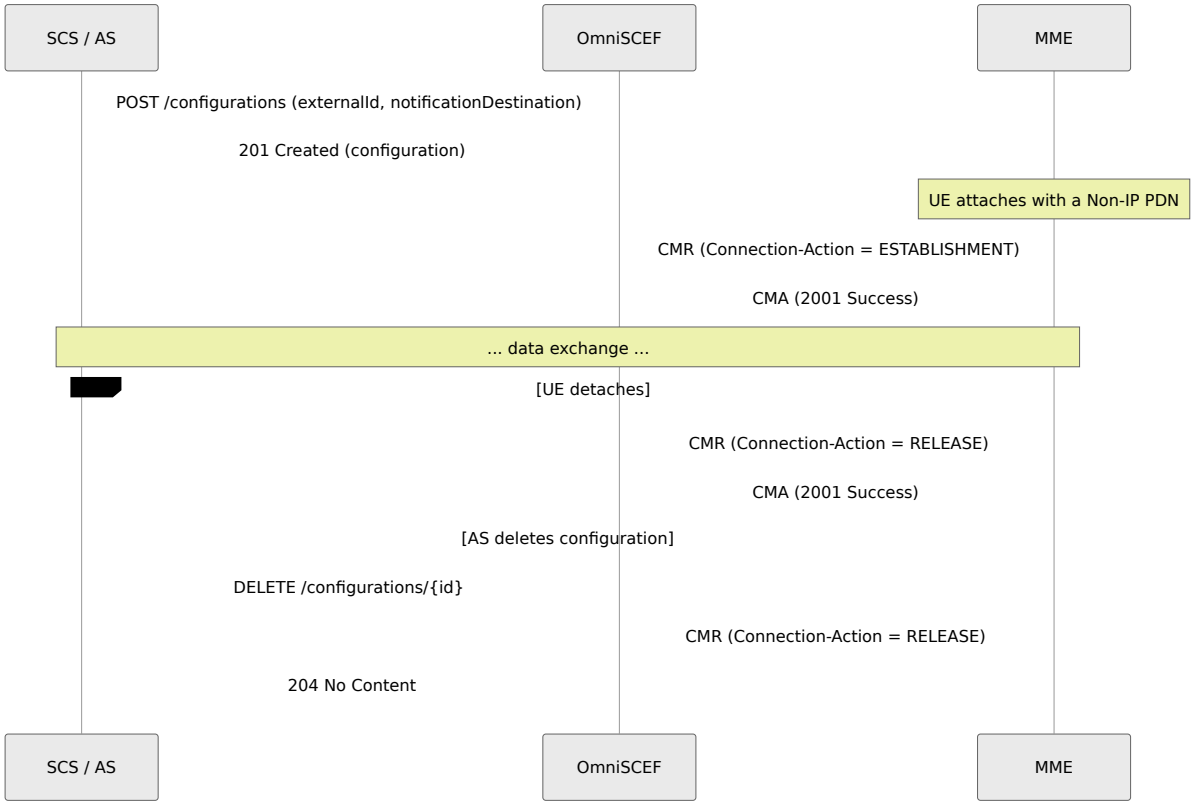
A NIDD session has two correlated halves:

- 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

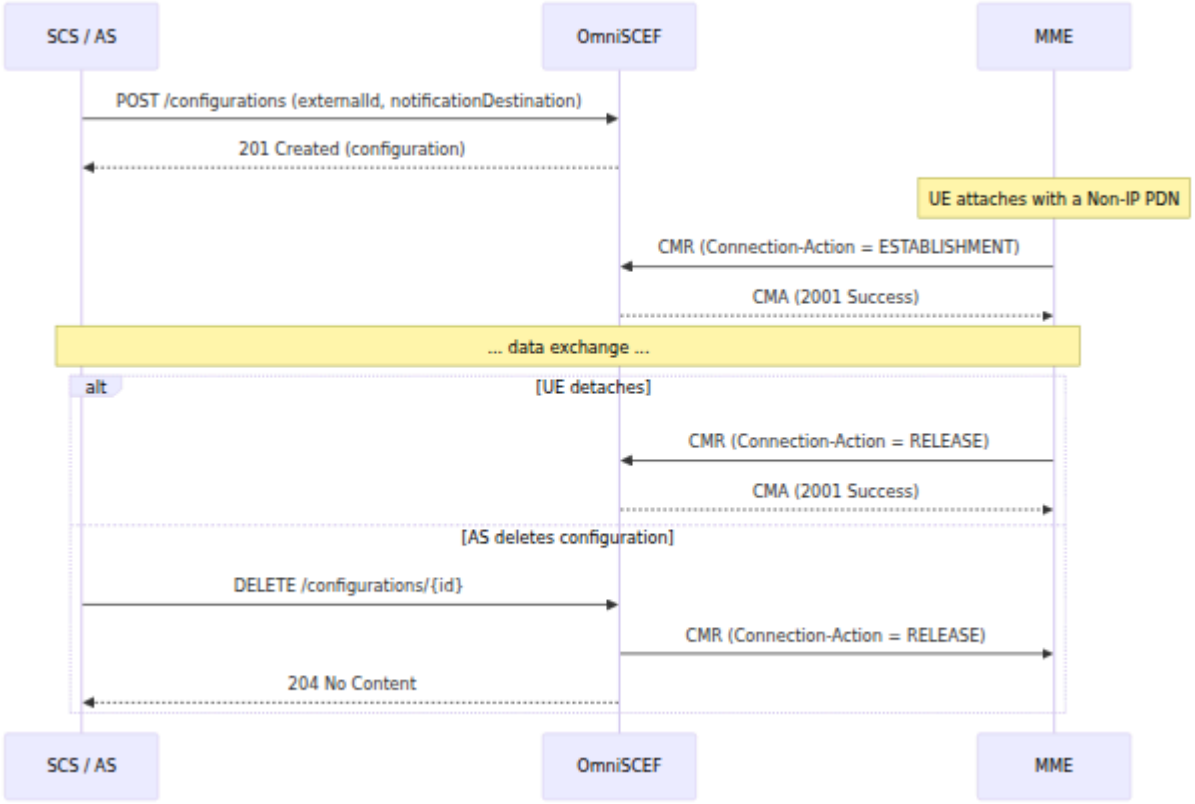
The MME manages the T6a connection as the UE attaches and detaches. The Application Server can also end a connection by deleting its NIDD configuration.



When the MME establishes a connection 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)`.

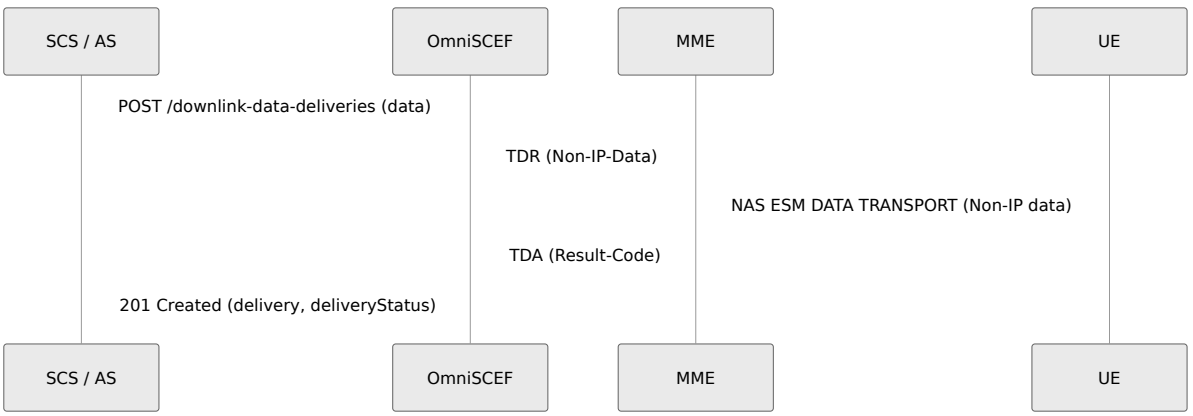
Example uplink notification (POSTed to the callback):

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

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

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
}
```

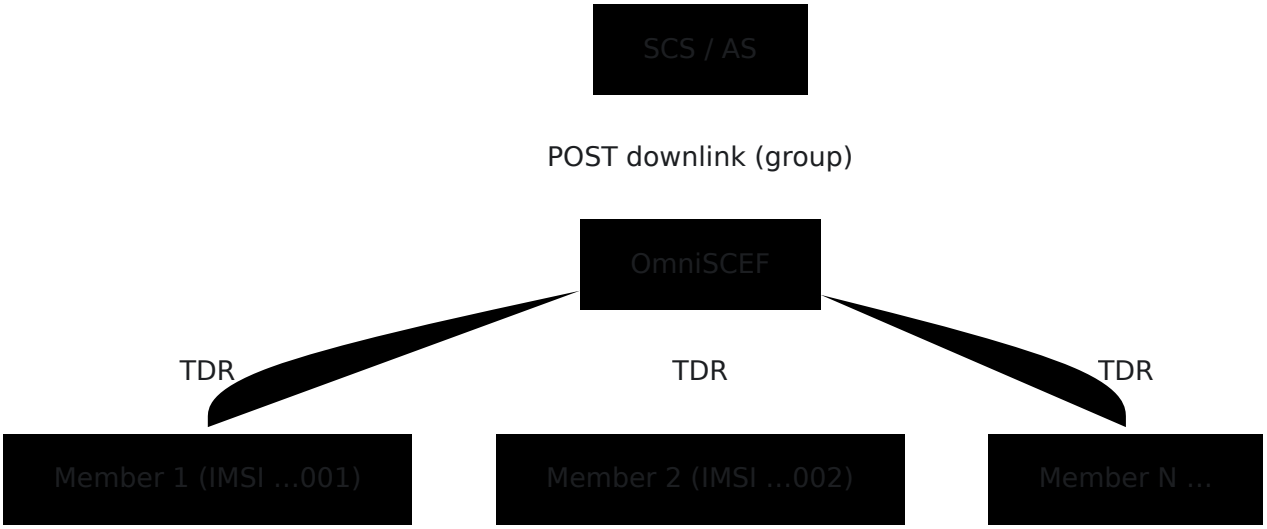
Field	Description
`data`	The Non-IP payload to deliver, base64-encoded. Required.
`maximumLatency`	Optional. Maximum acceptable buffering delay, in seconds.
`priority`	Optional. Relative priority among queued deliveries.
`pdnEstablishmentOption`	Optional. Behaviour when no connection exists. See PDN Establishment Options.
`rdsSourcePort` / `rdsDestinationPort`	Optional. RDS ports. See Reliable Data Service.

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**: when the delivery specifies `rdsDestinationPort` and `rdsSourcePort`, OmniSCEF frames the Non-IP payload with an RDS header that carries those ports before it sends the TDR.
- **Uplink**: OmniSCEF parses the RDS header from the ODR payload. It surfaces `rdsSourcePort` and `rdsDestinationPort` on the uplink

notification, and delivers 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
`notificationDestination`	String (URI)	Yes	Callback the AS receive uplink and status notifications on.
`externalId`	String	Conditional	Device External Identifi (`user@domain`). One of `externalId`, `msisdn`, o `externalGroupId`.
`msisdn`	String	Conditional	Device MSISDN.
`externalGroupId`	String	Conditional	Group identifier for gro NIDD.
`dnn`	String	No	Data Network Name / A of the Non-IP connectio
`pdnEstablishmentOption`	String (enum)	No	Behaviour on downlink when no connection ex See PDN Establishment Options.
`maximumPacketSize`	Integer	No	Maximum Non-IP paylo size in octets (defaults the configured `default_max_packet_s` 255).
`reliableDataService`	Boolean	No	Enable Reliable Data Service.
`mtcProviderId`	String	No	MTC service provider identifier.

Field	Type	Required	Description
`validityTime`	String (date-time)	No	When the configuration expires.
`status`	String (enum)	Read-only	Lifecycle status. See Configuration Status Values.
`self`	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
`NiddUplinkDataNotification`	Uplink Non-IP data arrived (from an ODR).
`NiddDownlinkDataDeliveryStatusNotification`	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
`SUCCESS`	The MME delivered the data (or took responsibility by buffering/paging an idle UE).
`SENDING`	The delivery is in progress.
`BUFFERING`	The MME buffered the data for an idle UE.
`FAILURE_UE_NOT_REACHABLE`	The UE could not be reached.
`FAILURE_REMOTE_FAILURE`	The MME / serving node rejected or failed the request.
`FAILURE`	The delivery failed for another reason.

PDN Establishment Options

Value	Behaviour when no connection exists at downlink time
`WAIT_FOR_UE`	Buffer until the UE establishes the connection.
`INDICATE_ERROR`	Reject the delivery immediately.
`ESTABLISH_PDN_CONNECTION`	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

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 the reverse.
- 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 OmniSCEF released its connection.
- 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. This can happen after a restart, because 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`. Production deployments must always terminate TLS at or before OmniSCEF.