

OmniPCF Configuration Reference

Operator configuration is read at runtime from `config/runtime.exs` into the `:omnipcf` application environment. Every key is overridable through an OS environment variable.

```
config :omnipcf,
  sbi_scheme: System.get_env("SBI_SCHEME", "http"),
  sbi_addr:   System.get_env("SBI_ADDR", "127.0.0.13"),
  sbi_port:   String.to_integer(System.get_env("SBI_PORT",
"7777")),
  nrf_uri:    System.get_env("NRF_URI", "http://127.0.0.1:7777"),
  udr_uri:    System.get_env("UDR_URI", "http://127.0.0.12:7777"),
  bsf_uri:    System.get_env("BSF_URI", "http://127.0.0.15:7777"),
  mcc:        System.get_env("MCC", "999"),
  mnc:        System.get_env("MNC", "70"),
  heartbeat_interval:
String.to_integer(System.get_env("HEARTBEAT_INTERVAL", "10000")),
  prometheus_metrics_port:
String.to_integer(System.get_env("PROMETHEUS_PORT", "9568"))

config :logger, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all
```

Parameter Reference

Parameter	Type	Required	Default
<code>sbi_scheme</code>	string	No	<code>"http"</code>
<code>sbi_certfile</code>	string	Conditional	none
<code>sbi_keyfile</code>	string	Conditional	none

Parameter	Type	Required	Default
sbi_addr	string	No	"127.0.0.13"
sbi_port	integer	No	7777
nrf_uri	string	No	"http://127.0.0.1:7
udr_uri	string	No	"http://127.0.0.12:
bsf_uri	string	No	"http://127.0.0.15:
mcc	string	No	"999"

Parameter	Type	Required	Default
mnc	string	No	"70"
heartbeat_interval	integer (ms)	No	10000
prometheus_metrics_port	integer	No	9568
policy_association_ttl_ms	integer (ms)	No	86400000

Parameter	Type	Required	Default
<code>presence_reporting_areas</code>	map	No	empty map
<code>hss_api_base_url</code>	string	No	<code>"https://127.0.0.1:</code>

The compiled default for `nrf_uri` differs between the config accessor (`http://127.0.0.10:7777`) and `runtime.exs` (`http://127.0.0.1:7777`); the value supplied via `NRF_URI` / `runtime.exs` is what a deployment actually uses. Always set `NRF_URI` explicitly.

Compiled Policy Defaults

These defaults are compiled into the PCC-rule builder and policy-authorization service, not exposed as configuration keys. They are applied only when neither the SMF-supplied subscription data nor UDR policy data provides a value. They are documented here so operators can recognize them in a decision and know when provisioning is incomplete.

Session-AMBR (required - no fallback)

The session-AMBR must be resolvable from the subscription. There is **no compiled fallback**: if neither the SMF-supplied subscribed AMBR (`subsSessionAmbr` / `subsSessAmbr`, from the UDM) nor the UDR SM policy data (`sessAmbr` / `sessionAmbr`) provides one, the `Npcf_SMPolicyControl` create is **rejected** with `422 Unprocessable Entity` (cause `ERROR_INITIAL_PARAMETERS`) and an error is logged naming the SUPI/DNN. The SMF then aborts the PDU session, so a misprovisioned subscriber fails visibly instead of silently getting an arbitrary low rate cap. Provision the session-AMBR in the UDM/HSS (per-DNN APN-AMBR) or the UDR SM policy data to resolve it.

Default (non-GBR) QoS

Applied when no subscribed default QoS is present and the DNN has no specific profile.

Field	Value
5QI	9 (non-GBR, best-effort internet)
ARP priority level	8
ARP preemption capability	NOT_PREEMPT
ARP preemption vulnerability	PREEMPTABLE

Per-DNN default QoS profiles

Used when no subscribed QoS is present, keyed by DNN network identifier. Subscription-provided QoS always takes precedence.

DNN	5QI	ARP priority	Preempt capability	Preempt vulnerability
ims	5	1	MAY_PREEMPT	NOT_PREEMPTABLE
sos	5	1	MAY_PREEMPT	NOT_PREEMPTABLE
mcs	65	1	MAY_PREEMPT	NOT_PREEMPTABLE

Default policy-control request triggers

Installed in the SmPolicyDecision when the UDR provides none. With no UDR data: `PLMN_CH`, `RES_MO_RE`, `AC_TY_CH`, `UE_IP_CH`. With UDR data but no explicit trigger list: the above plus `DEF_QOS_CH`, `SE_AMBR_CH`.

AF media-type to QoS mapping

Used by policy authorization to build dedicated flows from AF media components. Non-audio/video media types map to best-effort. Default ARP is `NOT_PREEMPT` / `PREEMPTABLE` with priority level 1 (AUDIO), 2 (VIDEO) or 9 (other).

Media type	5QI	GBR UL/DL	MBR UL/DL
AUDIO	1	64 Kbps / 64 Kbps	96 Kbps / 96 Kbps
VIDEO	2	512 Kbps / 512 Kbps	2 Mbps / 2 Mbps
other	9	n/a	n/a

PCC rule precedence

Rule source	Default precedence
Default permit-all rule (per DNN)	255
UDR-provisioned dedicated charging rule	200 (or rule-specified)
AF-derived (policy authorization) rule	100 (or <code>resPrio</code>)

Management / OAM API

Separate from the 5G SBI, OmniPCF exposes an operator-facing REST management (OAM) API served over HTTPS. All responses are JSON, wrapped in a standard `success` / `error` envelope. Every management path is mounted under the `/api` prefix, so the full request paths are as shown in the tables below. A LiveView web Control Panel (Resources, Configuration, License, and Log views) is served over HTTPS on port `7443` alongside this REST API.

Property	Value
Scheme	HTTPS (TLS)
Port	<code>8443</code>
Bind address	<code>0.0.0.0</code>
Path prefix	<code>/api</code>

Status endpoints

Method	Path	Description
GET	/api/status/api	Liveness/API status probe.
GET	/api/status/license	License status.
GET	/api/status/nrf	NRF registration status.
GET	/api/status/nf	NF service status: <code>nf_type</code> (PCF), advertised <code>npcf-smpolicycontrol</code> / <code>npcf-am-policy-control</code> services, <code>sbi_addr</code> / <code>sbi_port</code> .

Policy inspection

Method	Path	Description
GET	/api/policy_context	List active policy contexts.
GET	/api/sm_policy	List active SM policy associations.
GET	/api/sm_policy/{id}	Read a single SM policy association.
DELETE	/api/sm_policy/{id}	Remove a single SM policy association.
GET	/api/am_policy	List active AM policy associations.
DELETE	/api/am_policy/{id}	Remove a single AM policy association.
GET	/api/statistics	Aggregate statistics: active SM/AM policy counts and total installed PCC rules.

OAM operations

Method	Path	Description
GET	/api/oam/config	View current runtime configuration (<code>sbi_addr</code> , <code>sbi_port</code> , <code>nrf_uri</code> , <code>udr_uri</code> , <code>mcc</code> , <code>mnc</code> , log level, NF capacity/priority).
PATCH	/api/oam/config/{id}	Update selected runtime values. Accepted keys: <code>udr_uri</code> , <code>nrf_uri</code> , <code>nf_capacity</code> , <code>nf_priority</code> , <code>log_level</code> . Invalid or empty updates return 400.
PATCH / DELETE	/api/oam/sm_policy/{id}	Operator-initiated SM policy association update (PATCH, merges <code>pcc_rules</code> and notifies the SMF) or delete (DELETE).
PATCH / DELETE	/api/oam/am_policy/{id}	Operator-initiated AM policy association update (PATCH, accepts <code>ue_ambr</code> , <code>triggers</code> , and <code>pras</code> and notifies the AMF) or delete (DELETE).
POST	/api/oam/log_level	Change the log level at runtime. Body: <code>{"level": "<level>"}</code> .
GET	/api/oam/resources	BEAM VM resource usage.
GET	/api/oam/diagnostics	BEAM VM diagnostics.
POST	/api/oam/nrf/reregister	Force NF re-registration with the NRF.

Method	Path	Description
GET	/api/oam/peers	Configured NF peer URIs and reachability status.
GET	/api/oam/nrf_discovery	NRF discovery results for dependent NFs.

Runtime-configurable OAM values

Key	Type	Notes
udr_uri	string	Base URI of the UDR. Takes effect on the next policy-data query.
nrf_uri	string	Base URI of the NRF. Takes effect on the next registration/heartbeat cycle.
nf_capacity	integer	NF capacity advertised in the NRF profile; pushed to the NRF on update.
nf_priority	integer	NF priority advertised in the NRF profile; pushed to the NRF on update.
log_level	string	One of emergency, alert, critical, error, warning, warn, notice, info, debug.

Logging

OmniPCF logs are structured JSON (`OmniLogger.JsonFormatter`) with all metadata attached. Log lines are prefixed by functional area: `[SM Policy]`, `[AM Policy]`, `[PolicyAuth]`, `[PccRules]`, `[UDR Client]`, `[BSF Client]`, `[Policy Notification]`. SM/AM entries include the association ID and SUPI; session context (SUPI, PDU session ID, DNN, transaction ID) is attached to structured

logs for per-subscriber tracing. The log level can be changed at runtime through the `/api/oam/log_level` OAM endpoint. The `logger` block in `runtime.exs` is a formatting choice, not an operational tunable.

OmniPCF Metrics & Monitoring

Metrics are exposed in Prometheus format on `prometheus_metrics_port` (default `9568`) at `/metrics`.

Telemetry event names use dots (e.g. `omni_pcf.sm_policy.create.count`); the Prometheus exporter renders them with underscores as shown below. Confirm exact rendered names against a live `/metrics` scrape before wiring alerts.

SM Policy Metrics

Metric	Type	Labels	Description
<code>omni_pcf_sm_policy_create_count</code>	counter	<code>supi</code> , <code>dnn</code> , <code>result</code>	SM policy create operations (per SUPI/DNN, success or error).
<code>omni_pcf_sm_policy_delete_count</code>	counter	<code>supi</code> , <code>dnn</code>	SM policy delete operations.
<code>omni_pcf_sm_policy_expire_count</code>	counter	(none)	SM policy associations expired (removed on inactivity/timer).
<code>omni_pcf_sm_policy_creates_total</code>	counter	<code>result</code>	Total SM policy creates by result (<code>success</code> / <code>failure</code>).
<code>omni_pcf_active_sm_policies_count</code>	gauge	(none)	Number of active SM policy associations.

AM Policy Metrics

Metric	Type	Labels	Description
omni_pcf_am_policy_create_count	counter	supi, result	AM policy create operations.
omni_pcf_am_policy_update_count	counter	result	AM policy update operations (success or error).
omni_pcf_am_policy_delete_count	counter	supi	AM policy delete operations.
omni_pcf_am_policy_expire_count	counter	(none)	AM policy associations expired (removed on inactivity/timer).
omni_pcf_am_policy_creates_total	counter	result	Total AM policy creates by result.
omni_pcf_active_am_policies_count	gauge	(none)	Number of active AM policy associations.

Policy Authorization Metrics

These counters cover the Npcf_PolicyAuthorization (AF/VoNR) service, which turns an IMS/AF application session into dedicated guaranteed-bitrate flows.

Metric	Type	Labels
omni_pcf_policy_authorization_create_count	counter	resource
omni_pcf_policy_authorization_update_count	counter	resource
omni_pcf_policy_authorization_delete_count	counter	(none)
omni_pcf_policy_authorization_pcscf_restoration_count	counter	(none)

PCC Rule Metrics

Metric	Type	Labels	Description
<code>omni_pcf_pcc_rule_installs_total</code>	counter	(none)	Total PCC rule installations.

NRF Metrics

Metric	Type	Labels	Description
<code>omni_pcf_nrf_registration_status</code>	gauge	<code>nf_type</code>	NRF registration status (<code>1</code> = registered, <code>0</code> = not).

BEAM VM Metrics

Metric	Type	Description
<code>beam_memory_total</code>	gauge	Total BEAM memory (bytes).
<code>beam_memory_processes</code>	gauge	Memory allocated to Erlang processes.
<code>beam_memory_processes_used</code>	gauge	Memory actually used by processes.
<code>beam_memory_system</code>	gauge	System (non-process) memory.
<code>beam_memory_atom</code>	gauge	Total atom memory.
<code>beam_memory_atom_used</code>	gauge	Used atom memory.
<code>beam_memory_binary</code>	gauge	Binary memory.
<code>beam_memory_code</code>	gauge	Code memory.
<code>beam_memory_ets</code>	gauge	ETS table memory.
<code>beam_processes_count</code>	gauge	Number of Erlang processes.
<code>beam_ports_count</code>	gauge	Number of Erlang ports.
<code>beam_atom_count</code>	gauge	Number of atoms.
<code>beam_vm_uptime</code>	gauge	VM uptime (seconds).

Example PromQL

```
# SM policy creation success rate over 5 minutes
sum(rate(omni_pcf_sm_policy_creates_total{result="success"}[5m]))
  / sum(rate(omni_pcf_sm_policy_creates_total[5m]))

# SM policy create failures per DNN
sum by (dnn) (rate(omni_pcf_sm_policy_create_count{result="error"}
[5m]))

# Currently active SM + AM associations
omni_pcf_active_sm_policies_count +
omni_pcf_active_am_policies_count

# Alert if the PCF is not registered with the NRF
omni_pcf_nrf_registration_status < 1

# PCC rule install rate
rate(omni_pcf_pcc_rule_installs_total[5m])
```

OmniPCF Operations Guide

OmniPCF is the Policy Control Function (PCF) of the Omnitouch 5G Core. It is the network's policy decision point: it tells the SMF how each PDU session must be treated (PCC rules, session-AMBR, per-flow QoS) and tells the AMF the access and mobility policy for each registered UE. It also exposes the Application-Function-facing policy authorization service that turns an IMS/VoNR call into a dedicated guaranteed-bitrate flow.

OmniPCF serves three SBI services:

- **Npcf_SMPolicyControl** (TS 29.512): session-management policy associations. For each PDU session the SMF establishes, the PCF returns a Policy and Charging Control (PCC) decision: PCC rules, session rules (session-AMBR), QoS decisions and traffic-control data.
- **Npcf_AMPolicyControl** (TS 29.507): access-and-mobility policy associations. For each UE registration the AMF creates, the PCF returns the authorized UE-AMBR and policy-control triggers.
- **Npcf_PolicyAuthorization** (TS 29.514): Application Session Contexts. An Application Function (AF), typically the IMS P-CSCF, requests media authorization; the PCF derives dedicated PCC rules (e.g. a 5QI 1 voice flow for VoNR) and pushes them to the SMF that owns the UE's SM policy.

OmniPCF fetches per-subscriber policy data from the **UDR** (Nudr_DataRepository, TS 29.519) and registers the UE-address-to-PCF binding in the **BSF** (Nbsf_Management, TS 29.521) so an AF can discover which PCF serves a given UE. It registers itself and heartbeats with the **NRF**.

Each policy association is owned by a dedicated GenServer (one process per SM policy association and one per AM policy association) supervised by a policy supervisor and addressed through a registry. Association state (PCC rules, QoS decisions, session-AMBR, UE-AMBR, triggers, BSF binding) lives in the owning process. Application Session Contexts for policy authorization are held in an

ETS table. There is also a fallback in-memory Agent store used when the worker path is unavailable. This association state is runtime state held in memory; the UDR remains the authoritative source of subscriber policy data, and associations are re-established by the SMF and AMF as PDU sessions and registrations are set up.

3GPP Role and Specification References

Specification	Relevance
TS 23.501	System architecture: PCF role, PCC framework, standardized 5QI to QoS characteristics mapping
TS 23.503	Policy and charging control architecture: policy framework, PCC rules, binding
TS 29.507	Npcf_AMPolicyControl API: access and mobility policy association
TS 29.512	Npcf_SMPolicyControl API: session management policy association, SmPolicyDecision
TS 29.514	Npcf_PolicyAuthorization API: AF media authorization, application session context
TS 29.519	Nudr_DataRepository: policy data resources queried and subscribed by the PCF
TS 29.521	Nbsf_Management: PCF binding registration (BSF)
TS 29.571	Common data types (FlowInformation, S-NSSAI, Ambr)

Interfaces and SBI Endpoints

All 5G SBI endpoints are served under the base URL

`{sbi_scheme}://{sbi_addr}:{sbi_port}`.

Npcf_SMPolicyControl (TS 29.512)

Method	Path	Description	Spec
POST	<code>/npcf-smpolicycontrol/v1/sm-policies</code>	Create SM Policy Association	TS 29.512 4.2.2
GET	<code>/npcf-smpolicycontrol/v1/sm-policies/{smPolicyId}</code>	Read SM Policy Association	TS 29.512 4.2.5
POST	<code>/npcf-smpolicycontrol/v1/sm-policies/{smPolicyId}/update</code>	Update SM Policy Association	TS 29.512 4.2.3
POST	<code>/npcf-smpolicycontrol/v1/sm-policies/{smPolicyId}/delete</code>	Delete SM Policy Association	TS 29.512 4.2.4

Npcf_AMPolicyControl (TS 29.507)

Method	Path	Description	Spec
POST	/npcf-am-policy-control/v1/policies	Create AM Policy Association	TS 29.507 4.2.1
GET	/npcf-am-policy-control/v1/policies/{polAssoId}	Read AM Policy Association	TS 29.507 4.2.1
POST	/npcf-am-policy-control/v1/policies/{polAssoId}/update	Update AM Policy Association	TS 29.507 4.2.1
DELETE	/npcf-am-policy-control/v1/policies/{polAssoId}	Delete AM Policy Association	TS 29.507 4.2.4

Npcf_PolicyAuthorization (TS 29.514)

Method	Path	Description	Spec
POST	/npcf-policyauthorization/v1/app-sessions	Create Application Session Context	TS 29.514 4.2.2
GET	/npcf-policyauthorization/v1/app-sessions/{appSessionId}	Read Application Session Context	TS 29.514 4.2.5
PATCH	/npcf-policyauthorization/v1/app-sessions/{appSessionId}	Update Application Session Context	TS 29.514 4.2.3
POST	/npcf-policyauthorization/v1/app-sessions/{appSessionId}/delete	Delete Application Session Context (custom op)	TS 29.514 4.2.4
PUT	/npcf-policyauthorization/v1/app-sessions/{appSessionId}/events-subscription	Create/modify Events Subscription sub-resource	TS 29.514
DELETE	/npcf-policyauthorization/v1/app-sessions/{appSessionId}/events-subscription	Delete Events Subscription sub-resource	TS 29.514
POST	/npcf-policyauthorization/v1/app-	P-CSCF Restoration	TS 29.514

Method	Path	Description	Spec
	<code>sessions/pcscf-restoration</code>	indication	

TS 29.514 defines no standard HTTP `DELETE` on `/app-sessions/{appSessionId}`. Deletion uses the custom `POST .../delete` operation above. The `pcscf-restoration` route is matched ahead of the `{appSessionId}` wildcard routes.

PCF callback endpoint

Method	Path	Description	Spec
POST	<code>/npcf-policy-data-notify</code>	Callback the UDR posts to when subscribed policy data changes	TS 29.519

Request / Response Summary

Create SM Policy Association: mandatory request fields (`SmPolicyContextData`, enforced at the router): `supi`, `pduSessionId`, `pduSessionType`, `dnn`, `notificationUri`, `sliceInfo`; and at least one of `ipv4Address` / `ipv6AddressPrefix` for a usable decision. Returns `201 Created` with the `SmPolicyDecision` body and a `Location` header. Missing router-level mandatory IEs return `400` with cause `MANDATORY_IE_MISSING`; a service-level validation failure returns `400` with cause `ERROR_INITIAL_PARAMETERS`; an explicit policy denial returns `403` with cause `POLICY_CONTEXT_DENIED`. The subscribed session-AMBR is read from `subsSessionAmbr` (preferred), falling back to `subsSessAmbr`; the subscribed default QoS from `subsDefQos`.

Update SM Policy Association: the SMF sends `SmPolicyUpdateContextData`. The PCF applies any provided `ipv4Address`, `ipv6AddressPrefix`, `ratType`; processes `ruleReports`; evaluates reported or inferred `repPolicyCtrlReqTriggers`; re-queries the UDR when a policy-relevant trigger fired; and returns a refreshed `SmPolicyDecision`. If PCC rules changed, the SMF's `notificationUri` is notified.

Delete SM Policy Association: body is `SmPolicyDeleteData`. The worker stops and its BSF binding is deregistered. Returns `204 No Content`.

Create AM Policy Association: mandatory fields

(`PolicyAssociationRequest`): `supi`, `notificationUri`, `suppFeat`. Optional: `guami`, `gpsi`, `pei`, `accessType`, `ratType`, `allowedSnssais`, `ueAmbr`, `servingPlmn`. Returns `201 Created` with the `PolicyAssociation` (`ueAmbr`, `triggers`, `suppFeat`) and a `Location` header. A missing mandatory IE returns `400` with cause `MANDATORY_IE_MISSING`.

Update AM Policy Association: the AMF sends

`PolicyAssociationUpdateRequest`. The PCF applies changes, re-queries the UDR for UE-AMBR, and returns a `PolicyUpdate` carrying `resourceUri` (and `ueAmbr` / `triggers` when present). If policy changed, the AMF's `notificationUri` is notified.

Presence Reporting Areas (TS 29.507): the PCF includes Presence Reporting Areas in AM policy decisions. It sources them from the UDR AM-policy data (`praInfos`) and, when the UDR supplies none, falls back to the operator-configured `presence_reporting_areas` (see the [Configuration Reference](#)). When any PRA is present, the PCF injects a presence-change (`PRA_CH`) policy-control trigger so the AMF reports UE entry and exit for those areas. The `pras` body parameter is also accepted on the operator `PATCH` `/api/oam/am_policy/{id}` action.

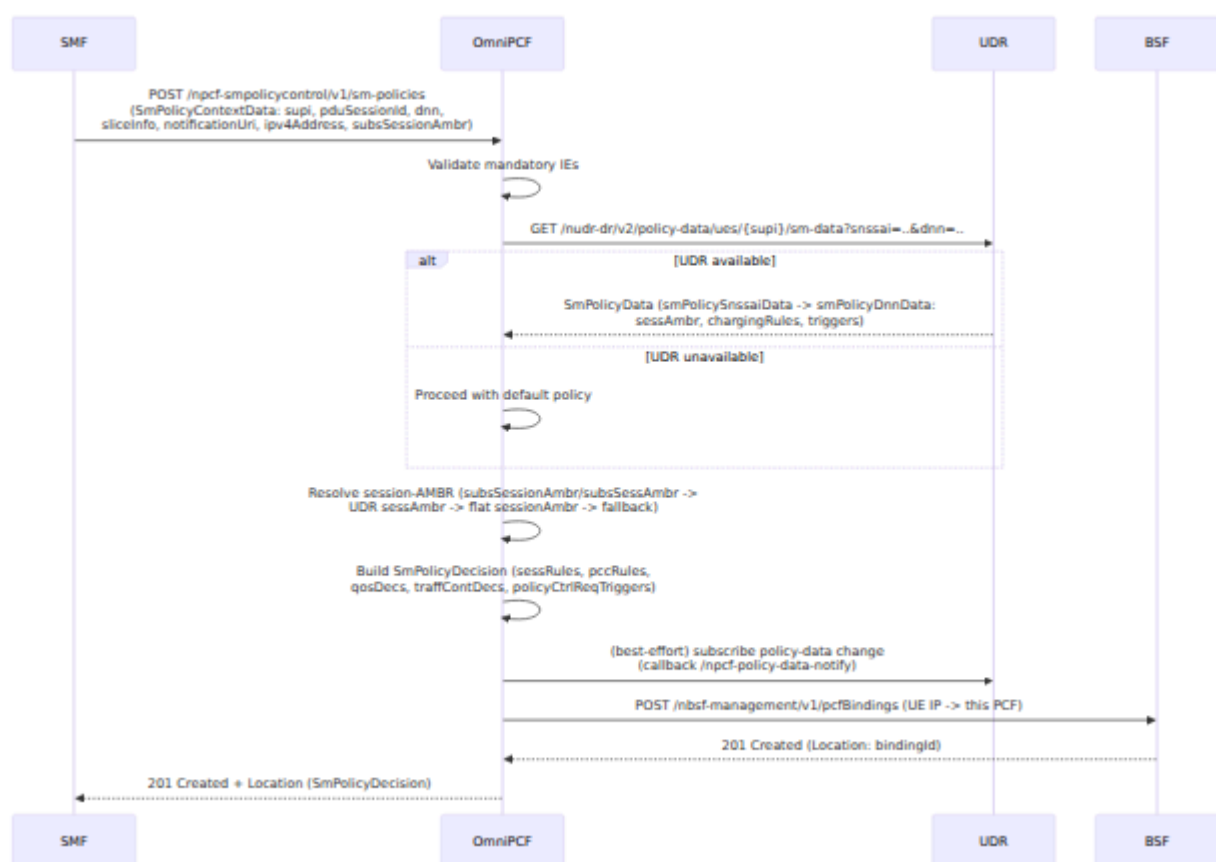
Create Application Session Context: mandatory

(`AppSessionContextReqData` under `ascReqData`): `notifUri`, `suppFeat`, and exactly one of `ueIpv4` / `ueIpv6` / `ueMac`. Media components (`medComponents`) drive PCC-rule derivation. Returns `201 Created` with the stored `AppSessionContext` and a `Location` header.

Key Procedures

SM Policy Association Creation (TS 29.512 Section 4.2.2)

When the SMF establishes a PDU session, it creates an SM policy association. The PCF validates the context, fetches SM policy data from the UDR, builds the PCC decision, spawns the association worker, subscribes (best-effort) to UDR policy-data change notifications, and registers the UE-address binding in the BSF.



The session-AMBR resolution order is: the SMF-supplied subscribed AMBR (`subsSessionAmbr`, else `subsSessAmbr`, originating from the UDM), then the nested UDR SM policy data (`smPolicySnssaiData[snssai].smPolicyDnnData[dnn].sessAmbr`), then a flat `sessionAmbr` on the UDR record, then the compiled fallback (logged). **The PCF-provided session-AMBR is authoritative: it governs the aggregate rate the SMF enforces for the PDU session.** BSF registration is best-effort and runs off the worker's init path, so a slow or absent BSF never blocks policy

creation. AF-initiated QoS will not be able to resolve this PCF until the binding exists.

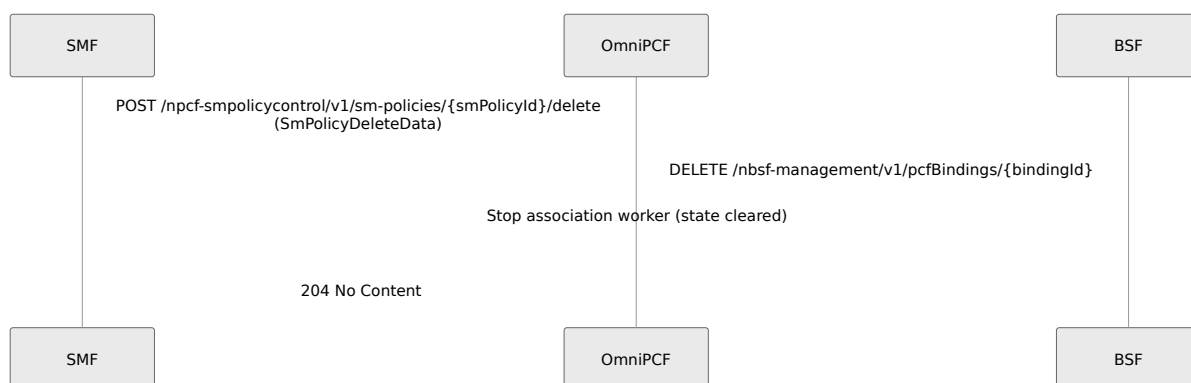
SM Policy Association Update (TS 29.512 Section 4.2.3)

Parse error on line 7: ... process ruleReports PCF->>PCF: Eval -----
-^ Expecting 'SOLID_OPEN_ARROW', 'DOTTED_OPEN_ARROW',
'SOLID_ARROW', 'BIDIRECTIONAL_SOLID_ARROW', 'DOTTED_ARROW',
'BIDIRECTIONAL_DOTTED_ARROW', 'SOLID_CROSS', 'DOTTED_CROSS',
'SOLID_POINT', 'DOTTED_POINT', got 'NEWLINE'

Try again

The UDR is re-queried on update only when a policy-relevant trigger has fired: an expensive fetch is skipped when nothing changed. The PCF decides a trigger fired either from the `repPolicyCtrlReqTriggers` list the SMF explicitly reports (if any of them is in the association's installed trigger set), or, when that list is absent, by inferring from the update body: an `ipv4Address` / `ipv6AddressPrefix` change maps to `UE_IP_CH`, a `ratType` change to `AC_TY_CH`, a `servingNetwork` change to `PLMN_CH`, and any `ruleReports` to `RES_MO_RE`. An inferred trigger only counts if the matching trigger is in the association's installed set (see the default trigger set in the [Configuration Reference](#)). If no trigger matches, the decision is rebuilt from existing state without a fresh UDR query.

SM Policy Association Deletion (TS 29.512 Section 4.2.4)



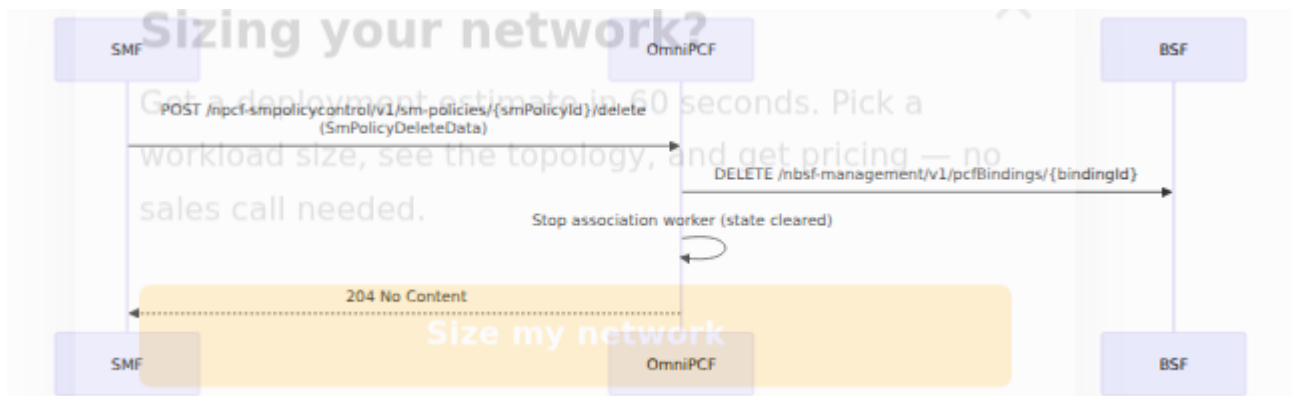
AM Policy Association Creation (TS 29.507 Section 4.2.2)

Parse error on line 14: ..._CH if AMBR differs) PCF-->>AMF: 201 -----
---^ Expecting 'SOLID_OPEN_ARROW', 'DOTTED_OPEN_ARROW',
'SOLID_ARROW', 'BIDIRECTIONAL_SOLID_ARROW', 'DOTTED_ARROW',
'BIDIRECTIONAL_DOTTED_ARROW', 'SOLID_CROSS', 'DOTTED_CROSS',
'SOLID_POINT', 'DOTTED_POINT', got 'NEWLINE'

Try again

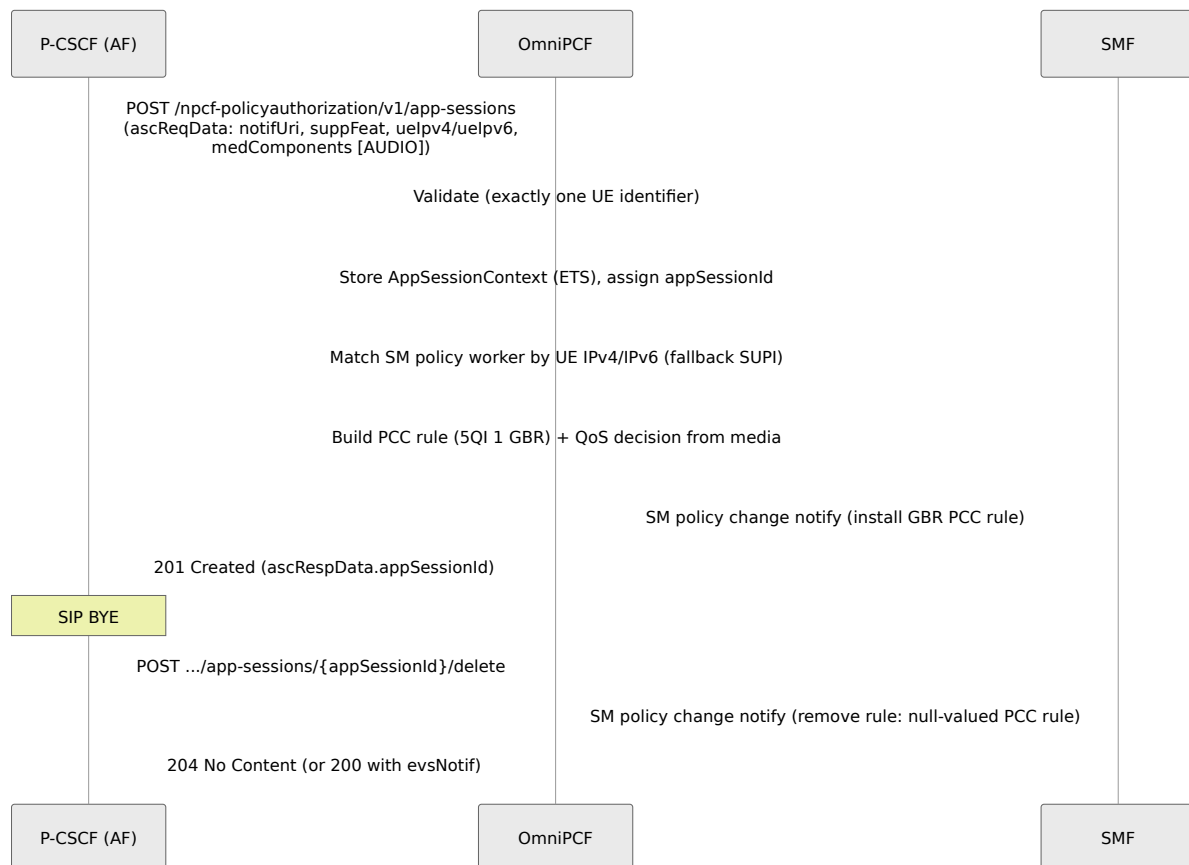
The authorized UE-AMBR is the UDR `subscribedUeAmbr` when present, otherwise the value the AMF supplied. A `UE_AMBR_CH` trigger is set when the AMF-reported and subscribed UE-AMBR differ.

AM Policy Association Update (TS 29.507 Section 4.2.3)



Policy Authorization for a VoNR Bearer (TS 29.514)

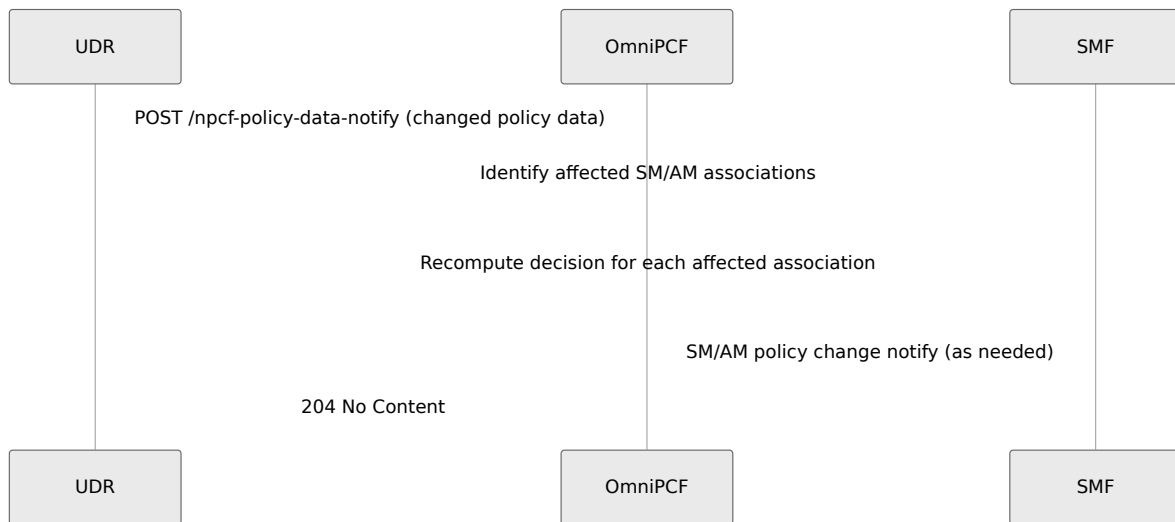
This is the PCF side of the VoNR dynamic dedicated bearer. When a call is answered, the IMS P-CSCF (acting as AF) creates an Application Session Context describing the media; the PCF derives a GBR PCC rule (5QI 1 for audio) and pushes it to the SMF that owns the UE's SM policy. On BYE, the AF deletes the session and the PCF removes the rule.



The AF addresses the UE by IP, so the PCF matches the app session to the SM policy association by UE IPv4 first, then UE IPv6, and only falls back to SUPI when no IP was supplied. An IPv6-only VoNR call carries only `ueIpv6`, so the IPv6 branch is essential on both install and removal or no dedicated flow installs/tears down. To keep one UE IP mapped to exactly one live association, SM policy creation supersedes any stale association still holding that IPv4 (supersede-on-create); when several worker states still match a reused IP, the rule is pushed to all of them: the live session installs the flow, stale ones fail harmlessly.

UDR Policy-Data Change Subscription (TS 29.519)

At SM policy creation the PCF subscribes (best-effort) to UDR policy-data change notifications for the subscriber, giving the UDR the callback `/npcf-policy-data-notify`. When subscriber policy data changes, the UDR posts to that callback and the PCF handles the change for affected associations.



Policy and PCC Concepts

SmPolicyDecision structure

For each SM policy association the PCF returns an `SmPolicyDecision` (TS 29.512 Table 5.6.2.3-1) built from subscription data, UDR policy data and local policy:

```

SmPolicyDecision
  sessRules:
    SessRule-{dnn}:
      sessRuleId:  "SessRule-{dnn}"
      authSessAmbr: {uplink, downlink}      <- subsSessionAmbr /
subsSessAmbr / UDR sessAmbr (required; else 422)
      authDefQos:  {5qi, arp}                <- subscribed / DNN
profile / 5QI 9

    pccRules:
      PccRule-{dnn}-default:                <- always present
(permit-all, precedence 255)
      flowInfos:  [permit out ip from any to {ueIp},
                    permit out ip from {ueIp} to any]
      refQosData: ["QosData-{dnn}"]
      refTcData:  ["TcData-{dnn}"]
      {udr-charging-rule}:                  <- 0..n dedicated rules
from UDR chargingRules

    qosDecs:
      QosData-{dnn}:  {5qi, arp, maxbrUl, maxbrDl}    maxbr =
session-AMBR

    traffContDecs:
      TcData-{dnn}:  {flowStatus: ENABLED}

    policyCtrlReqTriggers: [ ... ]            <- default set or UDR-
provided
    suppFeat: negotiated feature bitmap

```

Session-AMBR field naming (N7 / Npcf SM policy)

The subscribed Session-AMBR is carried by the SMF in the SM policy context and is read, in priority order, from `subsSessionAmbr` then `subsSessAmbr` (both accepted for interworking). The PCF places the resolved value into `sessRules.*.authSessAmbr` (and mirrors it as the default rule's `maxbrUl` / `maxbrDl`). UDR policy data supplies the value either nested as `smPolicySnssaiData[snssai].smPolicyDnnData[dnn].sessAmbr` or flat as `sessionAmbr`. **The PCF-provided session-AMBR governs the aggregate**

rate the SMF enforces for the PDU session: if a subscriber is capped below expectations, the value the PCF returned is the one in force. A session-AMBR is **mandatory**: if none can be resolved from any source, the create is rejected with 422 (cause ERROR_INITIAL_PARAMETERS) rather than falling back to a fabricated default that would silently cap the session (issue #34). Note the historic field-name divergence between SMF and PCF (sessionAmbr vs subsSessAmbr): OmniPCF accepts both request spellings so a mismatch does not silently drop the subscribed AMBR.

PCC rules

Every SM policy decision contains a default permit-all PCC rule for the DNN (precedence 255). Beyond that:

- **UDR-sourced dedicated rules:** static dedicated-bearer charging rules provisioned in the UDR SM policy data (chargingRules under the DNN, default precedence 200) each become a dedicated PCC rule plus QoS decision that the SMF installs as a 5G QoS flow (e.g. a 5QI 1 GBR flow for VoNR).
- **AF-derived rules:** media components from a policy-authorization app session become PCC rules named pcc-af-{component} (precedence 100 by default) with 5QI derived from media type (AUDIO→1, VIDEO→2, other→9), pushed to the SMF over the SM policy notification. On app-session delete the same rules are removed by sending them with a null value (TS 29.512 rule removal).

Provisioning a UDR dedicated charging rule

A dedicated bearer that is always present (as opposed to one an AF installs dynamically for a call) is provisioned as a chargingRules list entry inside the subscriber's UDR SM policy data, at

smPolicySnssaiData[snssai].smPolicyDnnData[dnn].chargingRules. Each list entry is one dedicated PCC rule; the PCF reads the following per-entry fields to build the rule and its QoS decision:

Field	Meaning	Default if omitted
name	PCC rule identifier (pccRuleId).	pcc-dedicated-{n}
precedence	Rule precedence (lower = higher priority; evaluated ahead of the permit-all 255 rule).	200
flowInfos	List of FlowInformation entries (packet filters) the rule applies to, passed to the SMF as-is (TS 29.571).	[] (rule matches nothing until flows are provisioned)
5qi	5QI for the flow's QoS decision.	the DNN default QoS 5QI
arp	ARP object (priorityLevel, preemptCap, preemptVuln).	passed through as provisioned (null if omitted)
gbrUl / gbrDl	Guaranteed bit rate up/down: set these for a GBR flow (e.g. "64 Kbps").	omitted (non-GBR)
mbrUl / mbrDl	Maximum bit rate up/down; falls back to the GBR value when omitted.	gbrUl / gbrDl

The rule is looked up under the DNN's network identifier (the dnn after stripping any .mnc... home-network domain). If the exact S-NSSAI or DNN key is absent from the UDR record, the PCF falls back to the first S-NSSAI entry whose sst matches and the first DNN entry, so a single generic profile still applies. Provide flowInfos: a rule with no flow information installs but matches no traffic.

5QI, GBR and ARP

QoS decisions carry the 5QI, the Allocation and Retention Priority (ARP: priority level, preemption capability, preemption vulnerability) and, for GBR flows, guaranteed and maximum bit rates (gbrUL/gbrDL/maxbrUL/maxbrDL) per TS 23.501 Table 5.7.4-1. The PCF builds GBR flows for AF media (5QI 1/2) and for UDR-provisioned dedicated bearers, and non-GBR best-effort (5QI 9) for the default rule.

Feature negotiation (suppFeat)

Each service negotiates supported features by a bitwise AND of the consumer's requested bitmap with the PCF's supported bitmap, returning the negotiated value as a lowercase hex string. Supported bitmaps: SM policy control 0x3F, policy authorization 0x3F, AM policy control 0x1F. A missing or unparseable request bitmap negotiates to 0.

OmniPCF

Troubleshooting

SM Policy Association returns 400 (MANDATORY_IE_MISSING / ERROR_INITIAL_PARAMETERS)

The `SmPolicyContextData` is missing a mandatory IE. The router requires `supi`, `pduSessionId`, `pduSessionType`, `dnn`, `notificationUri`, `sliceInfo` (missing these returns `MANDATORY_IE_MISSING`); the service additionally requires a usable UE address (`ipv4Address` / `ipv6AddressPrefix`) and returns `ERROR_INITIAL_PARAMETERS` on a service-level validation failure. Most often the UE IP is absent: the SMF populates it after UPF session establishment, so a PFCP failure before IP allocation leaves the field empty. Check the SMF-side session establishment first.

SM Policy Association returns 403 (POLICY_CONTEXT_DENIED)

Returned when policy validation explicitly denies the context. Inspect the PCF log around the request for the denial reason.

SM Policy Association returns 422 (missing session-AMBR)

The session-AMBR could not be resolved from the SMF-supplied `subsSessionAmbr` / `subsSessAmbr` (from UDM) or from the UDR (`sessAmbr` / flat `sessionAmbr`), so the create is rejected with `422 Unprocessable Entity` (cause

`ERROR_INITIAL_PARAMETERS`) and the SMF aborts the PDU session. An error `[SM Policy] No Session-AMBR available for SUPI=... DNN=...` is logged for every occurrence. There is no longer a fabricated fallback rate: previously the PCF substituted a hardcoded `1000 Kbps` default that silently became the effective 5G session cap for unprovisioned subscribers (issue #34). Fix by ensuring the UDM/HSS provisions the subscriber's session-AMBR (so the SMF forwards it as `subsSessAmbr`) or the UDR returns `sessAmbr` in the SM policy data. Note that OmniPCF accepts both `subsSessionAmbr` and `subsSessAmbr` spellings from the SMF, so a field-name mismatch is not the cause.

UDR queries fail / default policy applied

If the UDR is unreachable or returns an error, the PCF logs `[UDR Client] SM policy query failed for {supi}` (or the AM equivalent) and proceeds with default policy: a permit-all rule and DNN or 5QI 9 default QoS. Note the session-AMBR still has to come from the SMF (`subsSessAmbr`) in that case, since the UDR cannot supply `sessAmbr`; otherwise the create is rejected (see above). Verify `udr_uri`, UDR reachability, and that the UDR exposes the TS 29.519 v2 policy-data resources (`/nudr-dr/v2/policy-data/ues/{ueId}/sm-data` and `/am-data`).

AF dedicated flow (VoNR) never installs

The policy-authorization app session could not be matched to a live SM policy association. Check that: the AF supplied a UE identifier that matches the SM policy (an IPv6-only call must carry `ueIpv6`); the SM policy association for that UE IP exists and its BSF binding registered; and the log shows `[PolicyAuth] Notifying N SM policy assoc(s)`. A `No SM policy association found` warning means no live worker held that IP/SUPI. The same matching applies to rule removal on hangup: an IPv6 app-session carries only `ueIpv6`, so without a matching association the 5QI 1 flow is never torn down.

BSF binding registration fails

If the BSF is unreachable or NRF discovery of the BSF fails and the static `bsf_uri` is wrong, the log shows `[SM Policy] BSF binding registration failed`. The SM policy still succeeds, but AF-initiated QoS cannot resolve this PCF for the affected UE. Verify NRF discovery of the BSF and the `bsf_uri` fallback. A `403` on register means a binding for that UE address already exists (treated as already-bound).

AM policy associations accumulate

AM associations are created at UE registration and deleted at deregistration, and SM associations at PDU session establishment and release. The AMF and SMF are still expected to DELETE explicitly. If a consumer never sends the DELETE (for example after an AMF or SMF restart), the association does not persist indefinitely: both AM and SM associations self-expire on an idle TTL controlled by `policy_association_ttl_ms` (default 86400000 ms, that is 24 hours). The timer resets on every consumer update, and on expiry the association is torn down. This TTL is the backstop against leaked associations, not the primary lifecycle mechanism. Watch `omni_pcf_active_am_policies_count`. Associations, SM policies, and app sessions are runtime state held in memory and are re-established by the AMF and SMF as registrations and PDU sessions are set up.

Log correlation

Log lines are prefixed by area: `[SM Policy]`, `[AM Policy]`, `[PolicyAuth]`, `[PccRules]`, `[UDR Client]`, `[BSF Client]`, `[Policy Notification]`. SM/AM entries include the association ID and SUPI; session context (SUPI, PDU session ID, DNN, transaction ID) is attached to structured logs for per-subscriber tracing.

