← Overview

OmniBSF Configuration Reference

Configuration is read from the application environment key `:omnibsf`. The standard deployment mechanism is `config/runtime.exs`, which maps each value from an OS environment variable with a built-in default.

```
config :omnibsf,
  sbi_scheme: System.get_env("SBI_SCHEME", "http"),
  sbi_addr:   System.get_env("SBI_ADDR", "127.0.0.15"),
  sbi_port:   String.to_integer(System.get_env("SBI_PORT",
"7777")),
  nrf_uri:    System.get_env("NRF_URI", "http://127.0.0.1:7777"),
  mcc:        System.get_env("MCC", "999"),
  mnc:        System.get_env("MNC", "70"),
  prometheus_metrics_port:
String.to_integer(System.get_env("PROMETHEUS_PORT", "9571")),
  heartbeat_interval:
String.to_integer(System.get_env("HEARTBEAT_INTERVAL", "10000"))

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

Parameter Table

Parameter	Type	Required	Default
`sbi_scheme`	string	No	`"http"`
`sbi_addr`	string	No	`"127.0.0.15"`

Parameter	Type	Required	Default
`sbi_port`	integer	No	`7777`
`nrf_uri`	string	No	`"http://127.0.0.1:7777`
`mcc`	string	No	`"999"`
`mnc`	string	No	`"70"`

Parameter	Type	Required	Default
`prometheus_metrics_port`	integer	No	`9571`
`heartbeat_interval`	integer (ms)	No	`10000`

> Note: the `runtime.exs` default for `nrf_uri` is `http://127.0.0.1:7777`. When the environment variable is unset and the application falls back to the compiled accessor default instead, the value is `http://127.0.0.10:7777`. Set `NRF_URI` explicitly in production to avoid ambiguity.

Logging

The logger is configured to emit structured JSON via `OmniLogger.JsonFormatter` with full metadata. The active log level can be changed at runtime through the management API without a restart. See Management API for the runtime log-level endpoints.

Binding Lifetime Tuning

Binding TTL and the cleanup sweep interval are internal defaults and are not exposed as environment variables in `runtime.exs`. They may be set through the application environment if a deployment requires non-default values:

Application env key	Type	Default	Description
`:binding_ttl_seconds`	integer (s)	`86400` (24 h)	Time-to-live applied to a binding at registration and refreshed on every PATCH; after this the entry is eligible for cleanup.
`:binding_cleanup_interval_ms`	integer (ms)	`60000` (60 s)	How often the background cleaner scans for and removes expired bindings.

The TTL is stamped at registration and **refreshed on every PATCH**. A merge-patch `UpdateIndPCFBinding` (PATCH) recomputes the binding's expiry from `:binding_ttl_seconds` at patch time, so a PATCH acts as a keep-alive that extends the binding's lifetime. A registrant that periodically touches its binding (for example, on a PDU session modification) keeps that binding alive for as long as the session is active. A binding that is never patched lives at most `:binding_ttl_seconds` from its registration. Set the TTL comfortably above the interval between keep-alive touches (or above the longest expected PDU session lifetime when no keep-alive is issued); otherwise a still-active session's binding is swept away mid-session and later discovery queries return `204 No Content` until the registrant re-registers.

The cleanup interval bounds how long an expired binding lingers before removal - an entry can survive up to one full interval past its expiry. A shorter interval reclaims memory and clears stale bindings faster at the cost of more frequent full scans of the binding table. Removals are logged (`[BindingCleaner] Removed N expired binding(s)`); see Troubleshooting for stale-binding growth.

NRF Profile Capacity and Priority

The NF profile OmniBSF registers with the NRF carries a `capacity` and a `priority`. Consumers that discover the BSF through the NRF use these for selection and load distribution per TS 29.510: among candidate instances, lower `priority` is preferred, and `capacity` weights the share of traffic sent to instances of equal priority. They do not throttle OmniBSF itself - they are advertised hints for the consumer's selection logic and matter only when more than one BSF instance is registered.

Application env key	Type	Default	Description
`:nf_capacity`	integer	`100`	Relative capacity weight advertised in the NF profile (typical range 0–65535). Higher values attract a larger share of traffic among equal-priority instances.
`:nf_priority`	integer	`0`	Selection priority advertised in the NF profile. Lower values are preferred by consumers.

Both can be changed at runtime through the management API, which re-pushes the updated profile to the NRF on the next update cycle; see Management API. Runtime changes are not persisted and revert to these values on restart.

Management and Metrics Ports

In addition to the SBI listener, OmniBSF exposes:

- A Prometheus scrape endpoint on `prometheus_metrics_port` (default `9571`). See Metrics & Monitoring.
- An operator management API (OAM) for viewing statistics, inspecting/clearing bindings, adjusting NRF profile capacity/priority and log level at runtime, and forcing NRF re-registration. See Management API for the full endpoint reference.

← Overview

OmniBSF Management API

The management (OAM) API is an operator-facing REST interface, separate from the 5G SBI. It exposes status, statistics, binding inspection/removal, and runtime configuration controls. It is served over HTTPS.

Back to the Operations Guide.

Access

Property	Value
Scheme	HTTPS (TLS)
Port	`8443`
Bind address	`0.0.0.0`

All responses are JSON, wrapped in a standard envelope with a `success` / `error` status.

Status Endpoints

Method	Path	Description
`GET`	`/status/api`	Liveness probe. Returns `"ok"`.
`GET`	`/status/nrf`	NRF registration status: `nrf_status`, `nf_instance_id`, `nrf_uri`. Returns `unknown` if the registration process is unavailable.
`GET`	`/status/nf`	NF service status: `nf_type`, advertised `nbsf-management` service, `sbi_addr`/`sbi_port`, and the current active PCF binding count.

Binding Inspection

Method	Path	Description
`GET`	`/pcf_bindings`	List active PCF bindings (summary view: binding ID, UE IP, DNN, S-NSSAI, SUPI, PCF FQDN).
`GET`	`/api/bindings`	List all bindings with UE address, DNN, S-NSSAI, SUPI, GPSI, and PCF address info, plus a total count.
`GET`	`/api/bindings/{id}`	Detailed view of a single binding by ID, including framed route lists and negotiated `suppFeat`. Returns 404 if unknown.
`GET`	`/api/statistics`	Aggregate statistics: total bindings and a per-DNN breakdown.

Binding Management

Method	Path	Description
`DELETE`	`/oam/binding/{id}`	Remove a single PCF binding by ID. Returns 404 if the binding does not exist.
`POST`	`/oam/binding`	Clear all PCF bindings. Returns the number of bindings cleared. This is disruptive - active sessions lose their binding until re-registered.

Runtime Configuration

Method	Path	Description
`GET`	`/oam/config`	View current runtime configuration: `sbi_addr`, `sbi_port`, `nrf_uri`, `mcc`, `mnc`, NF `capacity`/`priority`, and log level.
`PATCH`	`/oam/config/{id}`	Update selected runtime values. Accepted keys: `nrf_uri`, `nf_capacity`, `nf_priority`, `log_level`. Invalid or empty updates return 400.
`POST`	`/oam/nrf/reregister`	Force NF re-registration with the NRF. Returns the resulting registration status and NF instance ID.
`POST`	`/oam/log_level`	Change the log level at runtime. Body: `{"level": "<level>"}`.

Configurable values

Key	Type	Notes
`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`.

Changes made through this API are applied to the running instance only and are not persisted; they revert to the configured values on restart.

← Overview

OmniBSF Metrics & Monitoring

Metrics are exposed on `prometheus_metrics_port` (default `9571`).

BSF Metrics

Metric	Type	Labels	Desc
`omni_bsf_pcf_binding_register_count`	counter	`dnn`	PCF bi registr labelle DNN
`omni_bsf_pcf_binding_discover_count`	counter	`result`	Discov reques labelle `result` / `not_`
`omni_bsf_pcf_binding_deregister_count`	counter	`binding_id`	Deregi labelle bindin
`omni_bsf_binding_registers_total`	counter	-	Aggreg registr counte
`omni_bsf_binding_discovers_total`	counter	`result`	Aggreg discov counte `result`
`omni_bsf_binding_deregisters_total`	counter	-	Aggreg deregi counte
`omni_bsf_active_bindings_count`	gauge	-	Numbe active bindin gauge by the

Metric	Type	Labels	Desc
			teleme event `[:omni :bindi :activ` emitte bindin regist deregi each c sweep expire bindin tracks curren bindin
`omni_bsf_nrf_registration_status`	gauge	`nf_type`	NRF registr status regist not)

> The `omni_bsf_pcf_binding_deregister_count` counter is labelled by `binding_id`. Because binding IDs are unique UUIDs, this produces one time-series per binding and is high-cardinality; prefer `omni_bsf_binding_deregisters_total` for aggregate rate/alerting.

> The `omni_bsf_active_bindings_count` gauge is a live gauge driven by the `[:omnibsf, :bindings, :active]` telemetry event, which the BSF emits whenever the set of active bindings changes: on binding register, on deregister, and on each BindingCleaner sweep that removes expired bindings. It therefore mirrors the current active binding count and is safe to alert on and dashboard. The same figure is also available through the OAM statistics endpoint (`GET /api/statistics`) and NF status endpoint; see Management API.

Telemetry-only Events

OmniBSF emits the following internal `:telemetry` events that are **not** currently exported as Prometheus counters. They are available to any attached telemetry handler but do not appear on the metrics scrape endpoint.

Event	Emitted	Description
`[:omnibsf, :pcf_binding, :update]`	Once per PATCH (`UpdateIndPCFBinding`)	Fired when a binding is updated by merge-patch. Carries the affected `binding_id`.
`[:omnibsf, :binding, :expired]`	Once per cleaner sweep that removes entries	Fired by the background cleaner with the count of bindings swept for having passed their TTL.

BEAM VM Metrics

Metric	Type	Description
`beam_memory_total`	gauge	Total BEAM memory in bytes
`beam_memory_processes`	gauge	Memory allocated to Erlang processes
`beam_memory_processes_used`	gauge	Memory actually used by processes
`beam_memory_system`	gauge	System memory
`beam_memory_atom`	gauge	Total atom memory
`beam_memory_atom_used`	gauge	Used atom memory
`beam_memory_binary`	gauge	Binary memory
`beam_memory_code`	gauge	Code memory
`beam_memory_ets`	gauge	ETS table memory
`beam_processes_count`	gauge	Number of Erlang processes
`beam_ports_count`	gauge	Number of Erlang ports
`beam_atom_count`	gauge	Number of atoms
`beam_vm_uptime`	gauge	VM uptime in seconds

Example PromQL

Binding registration rate (per second, per DNN):

```
sum by (dnn) (rate(omni_bsf_pcf_binding_register_count[5m]))
```

Discovery miss ratio over the last 5 minutes:

```
sum(rate(omni_bsf_binding_discovers_total{result="not_found"}
[5m]))
  /
sum(rate(omni_bsf_binding_discovers_total[5m]))
```

Alert when the BSF is not registered with the NRF:

```
omni_bsf_nrf_registration_status{nf_type="bsf"} == 0
```

> Metric names above use Prometheus dot-to-underscore normalisation. Label values for `result` are reported as `found` / `not_found`.

← Overview

OmniBSF Operations Guide

Overview

OmniBSF implements the Binding Support Function (BSF) of the 5G Core. The BSF maintains a registry of PCF bindings: for each PDU session it stores the mapping between a UE identity (IPv4 address, IPv6 prefix, MAC address, or SUPI) plus the DNN and S-NSSAI, and the PCF instance that serves that session's policy. Other network functions - an AF/NEF, an SMF, or another PCF - query the BSF to discover which PCF is responsible for a given session, so that all policy signalling for that session is directed to the same PCF instance.

OmniBSF exposes the `Nbsf_Management` service (3GPP TS 29.521) under the SBI base path `/nbsf-management/v1`. Bindings are held entirely in-memory in ETS tables and are indexed for concurrent, low-latency lookup. Bindings carry a configurable time-to-live (TTL); a background cleaner periodically removes expired entries. OmniBSF registers its NF profile with the NRF at startup and sends periodic heartbeats.

Internal Data Store

Bindings are held in six ETS tables at runtime: the primary binding table plus five dedicated lookup indices. All data is in-memory and does not survive a process restart.

ETS Table	Type	Key	Value
`bsf_bindings`	`:set`	binding_id (UUID string)	full binding map
`bsf_ipv4_index`	`:set`	UE IPv4 address (string)	binding_id
`bsf_ipv6_index`	`:set`	UE IPv6 prefix (string)	binding_id
`bsf_supi_index`	`:bag`	SUPI (string)	binding_id
`bsf_mac_index`	`:set`	UE MAC address (lowercased string)	binding_id
`bsf_dnn_snssai_index`	`:bag`	`{dnn, sst}`	binding_id

Every lookup key is served by its own index, so discovery does not scan the primary binding table. `macAddr48` lookups resolve through `bsf_mac_index` (keys are lowercased so matching is case-insensitive), and DNN+S-NSSAI lookups resolve through `bsf_dnn_snssai_index` keyed by `{dnn, sst}` (the small matched bucket is then filtered on the optional S-NSSAI `sd`).

`bsf_supi_index` is a `:bag`: a single SUPI may hold **multiple concurrent bindings** (for example, several PDU sessions on different DNNs), so one SUPI can map to several binding IDs. A discovery by `supi` returns a single `PcfBinding` per TS 29.521, so the first matching binding is returned.

3GPP Role and Specification References

Specification	Relevance
TS 23.501	System architecture - BSF network function role
TS 23.503	Policy and charging control framework - session binding context for policy association
TS 29.521	`Nbsf_Management` service API (PCF binding register / discover / update / deregister), `PcfBinding` data model, supported-feature negotiation
TS 29.500	5G SBI common framework (ProblemDetails, HTTP semantics)
TS 29.510	NF registration and heartbeat with the NRF

The BSF becomes relevant when the PCF is deployed as multiple instances (or as a PCF set). Its role is session binding: recording which PCF serves each PDU session so that subsequent policy interactions (from an AF via the NEF, or from the SMF) reach the correct PCF.

SBI Endpoints

All endpoints are served under the base URL `{sbi_scheme}://{sbi_addr}:{sbi_port}` with the base path `/nbsf-management/v1`.

Method	Path	Description	Succe
`POST`	`/pcfBindings`	Register a PCF binding for a PDU session. Returns a `Location` header pointing at the created binding.	201 Crea
`GET`	`/pcfBindings`	Discover the PCF binding matching the supplied query parameters.	200 OK `PcfBind` or 204 N Content none matches
`PATCH`	`/pcfBindings/{bindingId}`	Update an existing binding (`UpdateIndPCFBinding`). Body must be `application/merge-patch+json`.	200 OK updated `PcfBind`
`DELETE`	`/pcfBindings/{bindingId}`	Deregister a PCF binding by its ID.	204 No Content

Any other path returns `404` with a `ProblemDetails` body.

GET Query Parameters

The discovery endpoint resolves a binding using the first matching parameter, in this priority order:

Priority	Parameter	Type	Lookup	Description
1	`ipv4Addr`	string	index	UE IPv4 address (exact match)
2	`ipv6Prefix`	string	index	UE IPv6 prefix (exact match)
3	`macAddr48`	string	index	UE MAC address (case-insensitive match via `bsf_mac_index`)
4	`supi`	string	index	Subscriber Permanent Identifier (first match if several concurrent bindings share the SUPI)
5	`dnn` + `snssai`	string / JSON	index	DNN together with S-NSSAI, resolved via `bsf_dnn_snssai_index` keyed by `{dnn, sst}` (`sst` must match; `sd` matched when supplied)

When supplied, the `pcfSetId`, `pcfRegionId` and `bindLevel` query parameters further narrow the result per TS 29.521: a candidate whose stored value does not match a supplied filter is not returned. Filters absent from the query are ignored.

POST Request Body - Fields

Field	Type	Required	Description
`dnn`	string	Yes	Data Network Name
`snssai`	object	Yes	S-NSSAI (`sst` integer, optional `sd` string)
`pcfFqdn`	string	Conditional	PCF FQDN - required if `pcfIpEndPoints` absent
`pcfIpEndPoints`	array	Conditional	PCF IP endpoint list - required if `pcfFqdn` absent
`ipv4Addr`	string	No	UE IPv4 address (indexed; also drives duplicate detection)
`ipv6Prefix`	string	No	UE IPv6 prefix (indexed; also drives duplicate detection)
`macAddr48`	string	No	UE MAC address
`supi`	string	No	Subscriber Permanent Identifier (indexed)
`gpsi`	string	No	Generic Public Subscription Identifier
`ipv4FrameRouteList`	array	No	IPv4 framed routes
`ipv6FrameRouteList`	array	No	IPv6 framed routes
`pcfSetId`	string	No	PCF set identifier. Stored on the binding and used to

Field	Type	Required	Description
			narrow discovery when supplied as a query filter.
`pcfRegionId`	string	No	PCF region identifier. Stored on the binding and used to narrow discovery when supplied as a query filter.
`bindLevel`	string	No	Binding level. Stored on the binding and used to narrow discovery when supplied as a query filter.
`suppFeat`	string (hex)	No	Supported-features bitmap; negotiated via bitwise AND against feature `0x01` (BindingUpdate)

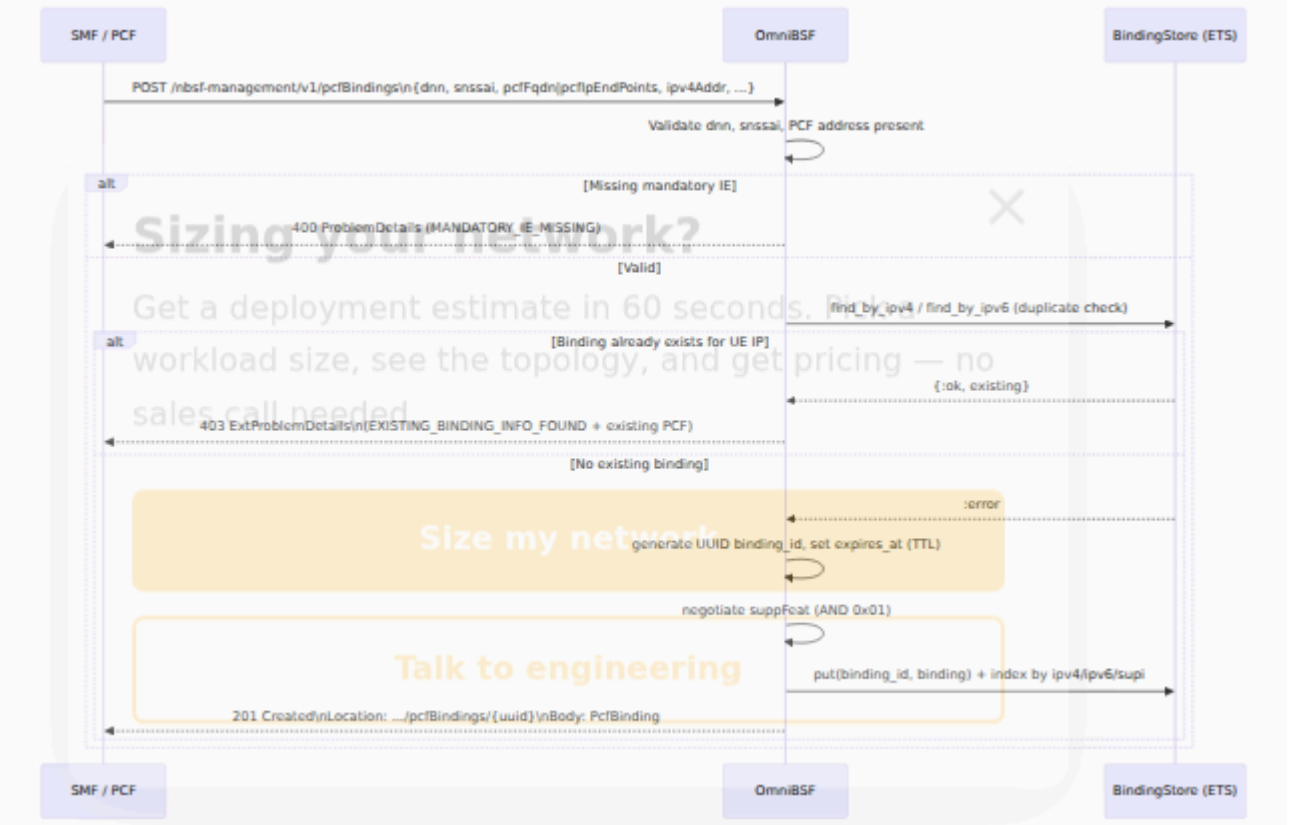
A `403` response to a duplicate registration carries an `ExtProblemDetails` body (`cause` = `EXISTING_BINDING_INFO_FOUND`) with the existing PCF's `pcfSmFqdn` / `pcfSmIpEndPoints` inlined at the top level, per TS 29.521.

Operator Management API

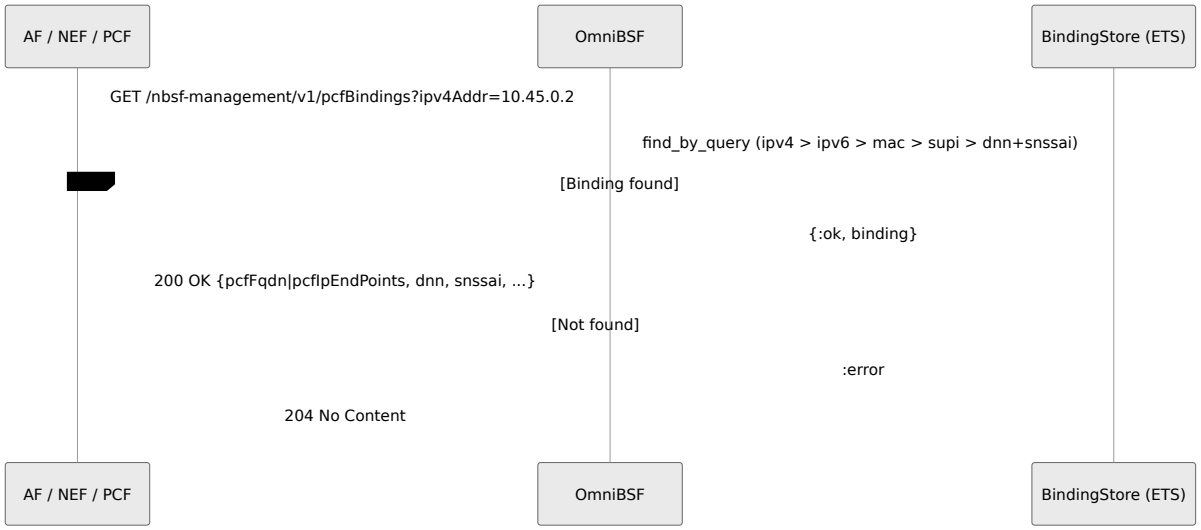
In addition to the SBI listener, OmniBSF exposes an operator management (OAM) REST interface for viewing statistics, inspecting/clearing bindings, adjusting NRF profile capacity/priority and log level at runtime, and forcing NRF re-registration. See Management API for the full endpoint reference.

Key Procedures

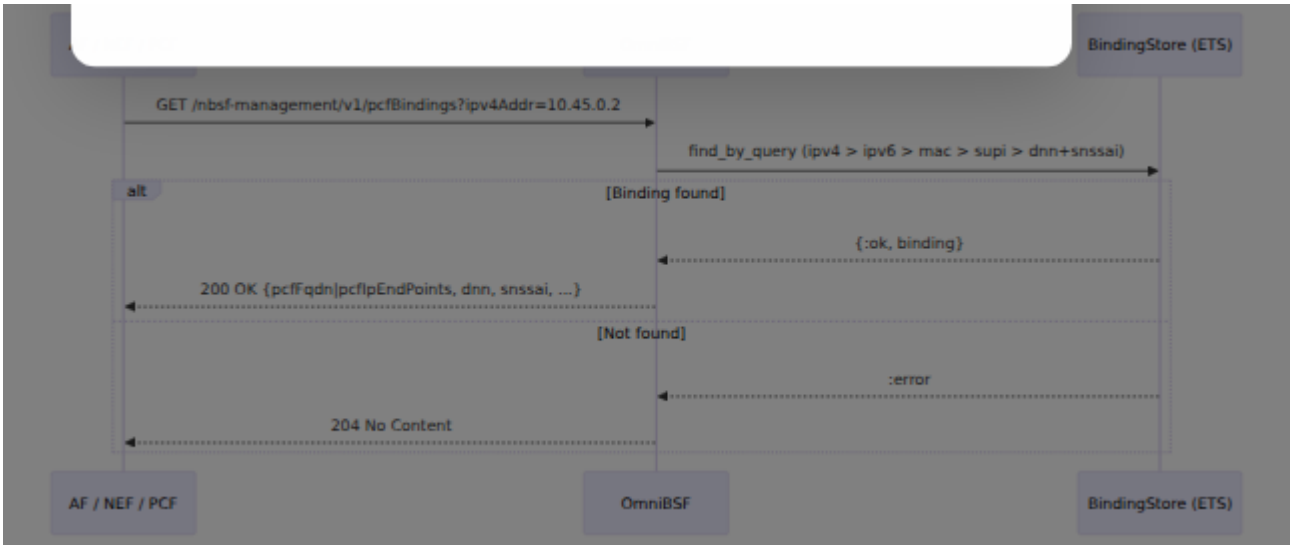
PCF Binding Registration (POST)



PCF Binding Discovery (GET)

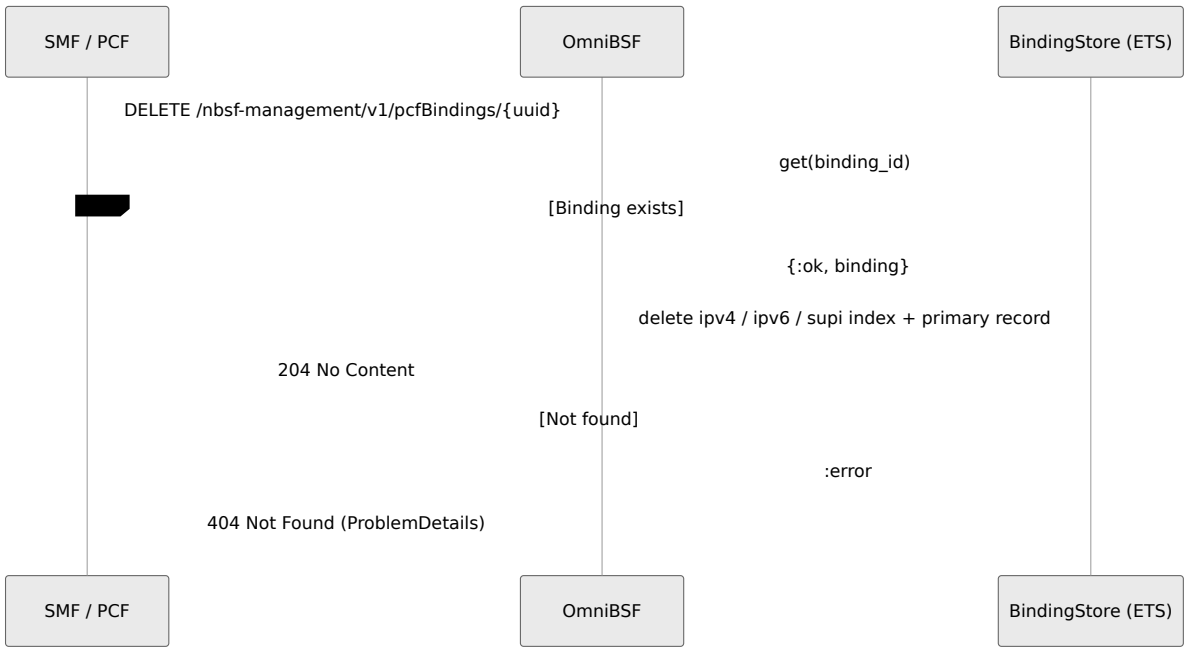


PCF Binding Update (PATCH)



Merge-patch applies only to the fields `ipv4Addr`, `ipv6Prefix`, `macAddr48`, `pcfFqdn`, and `pcfIpEndPoints`; because the IPv4/IPv6/SUPI/MAC and DNN+S-NSSAI indices key off the binding contents, the old record and its index entries are removed and the updated binding is re-inserted so the indices stay consistent. A PATCH also refreshes the binding's `expires_at` from `binding_ttl_seconds`, so it acts as a keep-alive that extends the binding's lifetime. See Binding Lifetime Tuning.

PCF Binding Deregistration (DELETE)



NRF Registration and Heartbeat

At startup OmniBSF registers a BSF NF profile with the NRF (from `nrf_uri`), advertising its SBI address/port/scheme, the configured PLMN (`mcc`/`mnc`), and the `nbsf-management` v1 service. It then sends periodic NFUpdate heartbeats every `heartbeat_interval` milliseconds. Registration can be forced at any time through the management API.

← Overview

OmniBSF Troubleshooting

204 returned when a binding is expected

1. Confirm the query key matches what was registered. `ipv4Addr`, `ipv6Prefix` and `supi` lookups are exact-string matches - formatting differences (leading zeros, prefix-length notation) cause a miss.
2. Check for `PCF binding not found for query:` in the logs to see the exact query.
3. Confirm the registrant completed the POST (look for `PCF binding registered:` in the logs).
4. DNN+S-NSSAI discovery resolves through the `{dnn, sst}` index and matches `sst` exactly; an integer-vs-string type mismatch in the original POST body will prevent a match.
5. Check whether the binding's TTL has elapsed and the cleaner removed it (`[BindingCleaner] Removed N expired binding(s)`).

403 on POST

A binding already exists for the supplied `ipv4Addr` / `ipv6Prefix`. The response is an `ExtProblemDetails` with `cause = EXISTING_BINDING_INFO_FOUND` and the incumbent PCF's `pcfSmFqdn` / `pcfSmIpEndPoints`. Reuse the returned PCF, or `DELETE` the existing binding first if a genuine rebinding is intended.

400 on POST

The body is missing a mandatory IE. The `detail` field identifies it: `No DNN`, `No S-NSSAI`, or `No PCF address information (pcfFqdn or pcfIpEndPoints required)`. The generic missing-field path reports `MANDATORY_IE_MISSING`.

404 on DELETE or PATCH

The binding ID does not match any active binding. Either it was already removed, its TTL expired, or OmniBSF restarted (all bindings are in-memory and lost on restart). Check for `PCF binding not found for deregister:` / `PCF binding not found for update:`.

415 on PATCH

The `PATCH` handler requires `Content-Type: application/merge-patch+json`. Any other content type is rejected with `415 Unsupported Media Type`.

High memory / growing binding count

Monitor the live binding count through the `omni_bsf_active_bindings_count` gauge, which is emitted on binding register, deregister, and each cleaner sweep; the OAM statistics endpoint (`GET /api/statistics`) and the NF status endpoint report the same figure. Bindings whose PDU sessions terminate without a `DELETE` remain until their TTL expires and the cleaner removes them. Because a PATCH refreshes a binding's TTL, a registrant that keeps patching a binding keeps it alive; a binding only ages out once neither a DELETE nor a PATCH has been received within `:binding_ttl_seconds`. Shorten `:binding_ttl_seconds` or `:binding_cleanup_interval_ms` if stale bindings accumulate faster than the default 24-hour TTL clears them. The management API also offers a clear-all operation.

NRF registration failing

Check `omni_bsf_nrf_registration_status{nf_type="bsf"}` and the logs. Verify that `nrf_uri` is reachable, that `mcc`/`mnc` match the PLMN configured at the NRF, and that connectivity from `sbi_addr` to the NRF is working. A re-registration can be forced through the management API.