

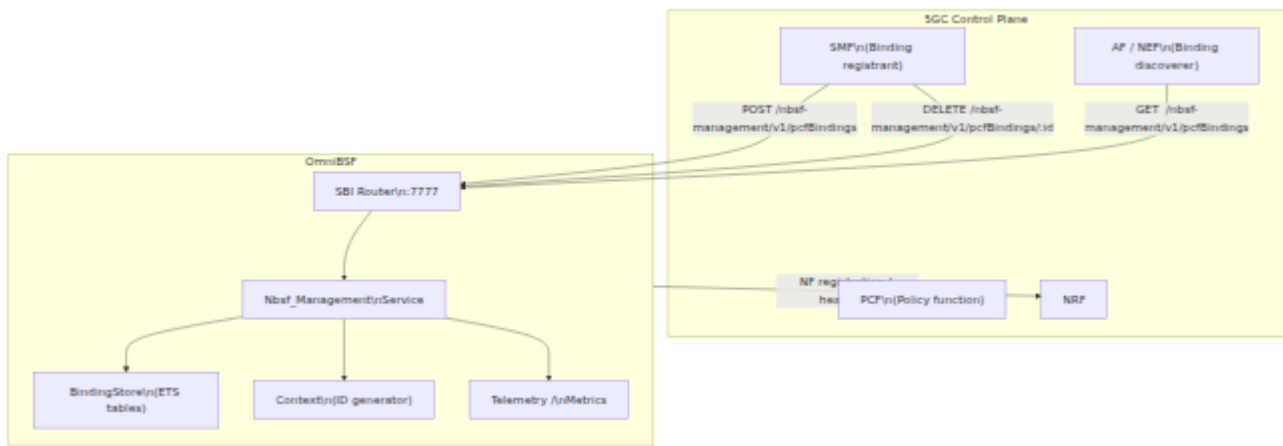
OmniBSF Operations Guide

Table of Contents

1. [Component Overview](#)
 2. [3GPP Role and Spec References](#)
 3. [SBI Endpoints](#)
 4. [Configuration Reference](#)
 5. [Key Procedures](#)
 6. [Observability](#)
 7. [Known Limitations](#)
 8. [Troubleshooting](#)
-

Component Overview

OmniBSF implements the Binding Support Function (BSF) network function defined in 3GPP TS 29.521. The BSF maintains a registry of PCF bindings, mapping each UE IP address (IPv4 or IPv6 prefix) to the PCF instance responsible for its PDU session policy. This allows NF consumers (SMF, AF, NEF) to discover the correct PCF for a given session without querying the NRF.



Internal Data Store

Bindings are held in four ETS tables at runtime. All data is in-memory and does not survive a process restart.

ETS Table	Type	Key	Value
bsf_bindings	:set	binding_id (string)	full binding map
bsf_ipv4_index	:set	IPv4 address (string)	binding_id
bsf_ipv6_index	:set	IPv6 prefix (string)	binding_id
bsf_supi_index	:set	SUPI (string)	binding_id

3GPP Role and Spec References

Item	Reference
BSF NF definition	3GPP TS 23.501 Section 6.2.15
Nbsf_Management service	3GPP TS 29.521
PCF binding registration procedure	3GPP TS 29.521 Section 4.2.2
PCF binding discovery procedure	3GPP TS 29.521 Section 4.2.3
PCF binding deregistration procedure	3GPP TS 29.521 Section 4.2.4
SBI common framework	3GPP TS 29.500
NF registration with NRF	3GPP TS 29.510
PcfBinding data model	3GPP TS 29.521 Section 6.1

The BSF is a mandatory network function when the PCF supports multiple instances. Its primary role is session binding: ensuring that all policy decisions for a PDU session are handled by the same PCF instance.

SBI Endpoints

Base path: `/nbsf-management/v1`

Method	Path	Description	Success	Error
POST	/pcfBindings	Register a PCF binding. Creates a new binding or upserts based on UE IP address. Returns a Location header with the binding ID.	201 Created	400 Bad Request (missing snssai, dnn, or PCF address)
GET	/pcfBindings	Discover a PCF binding. Accepts query parameters to look up the binding. Returns the PcfBinding object or 204 if not found.	200 OK or 204 No Content	400 Bad Request
DELETE	/pcfBindings/{bindingId}	Deregister a PCF binding by its ID.	204 No Content	404 Not Found

GET Query Parameters

Parameter	Type	Lookup Priority	Description
<code>ipv4Addr</code>	string	1 (highest)	UE IPv4 address (exact match)
<code>ipv6Prefix</code>	string	2	UE IPv6 prefix (exact match)
<code>dnn</code>	string	3 (table scan)	Data Network Name, used with <code>snssai</code>
<code>snssai</code>	string (JSON-encoded)	3 (table scan)	S-NSSAI with <code>sst</code> and optional <code>sd</code> fields

Lookup priority follows TS 29.521 and the open5gs `bsf-sm.c` reference implementation: `ipv4Addr` is checked first, then `ipv6Prefix`, then the DNN+S-NSSAI combination (which requires a full table scan).

POST Request Body — Required Fields

Field	Type	Description
snsai	object	S-NSSAI (sst integer + optional sd string). Mandatory.
dnn	string	Data Network Name. Mandatory.
pcfFqdn	string	PCF FQDN. Mandatory if pcfIpEndpoints is absent.
pcfIpEndpoints	array	PCF IP endpoint list. Mandatory if pcfFqdn is absent.
ipv4Addr	string	UE IPv4 address. Optional but required for IPv4 lookup.
ipv6Prefix	string	UE IPv6 prefix. Optional but required for IPv6 lookup.
supi	string	Subscriber Permanent Identifier. Optional.
gpsi	string	Generic Public Subscription Identifier. Optional.
ipv4FrameRouteList	array	IPv4 framed routes. Optional.
ipv6FrameRouteList	array	IPv6 framed routes. Optional.
suppFeat	string (hex)	Supported features bitmap. Negotiated via bitwise AND.

Configuration Reference

All parameters are set via the application environment (typically `config/runtime.exs`).

```
config :omnibsf,  
  sbi_scheme: "http",  
  sbi_addr:   "127.0.0.15",  
  sbi_port:   7777,  
  nrf_uri:    "http://127.0.0.10:7777",  
  mcc:        "999",  
  mnc:        "70",  
  heartbeat_interval: 10_000
```

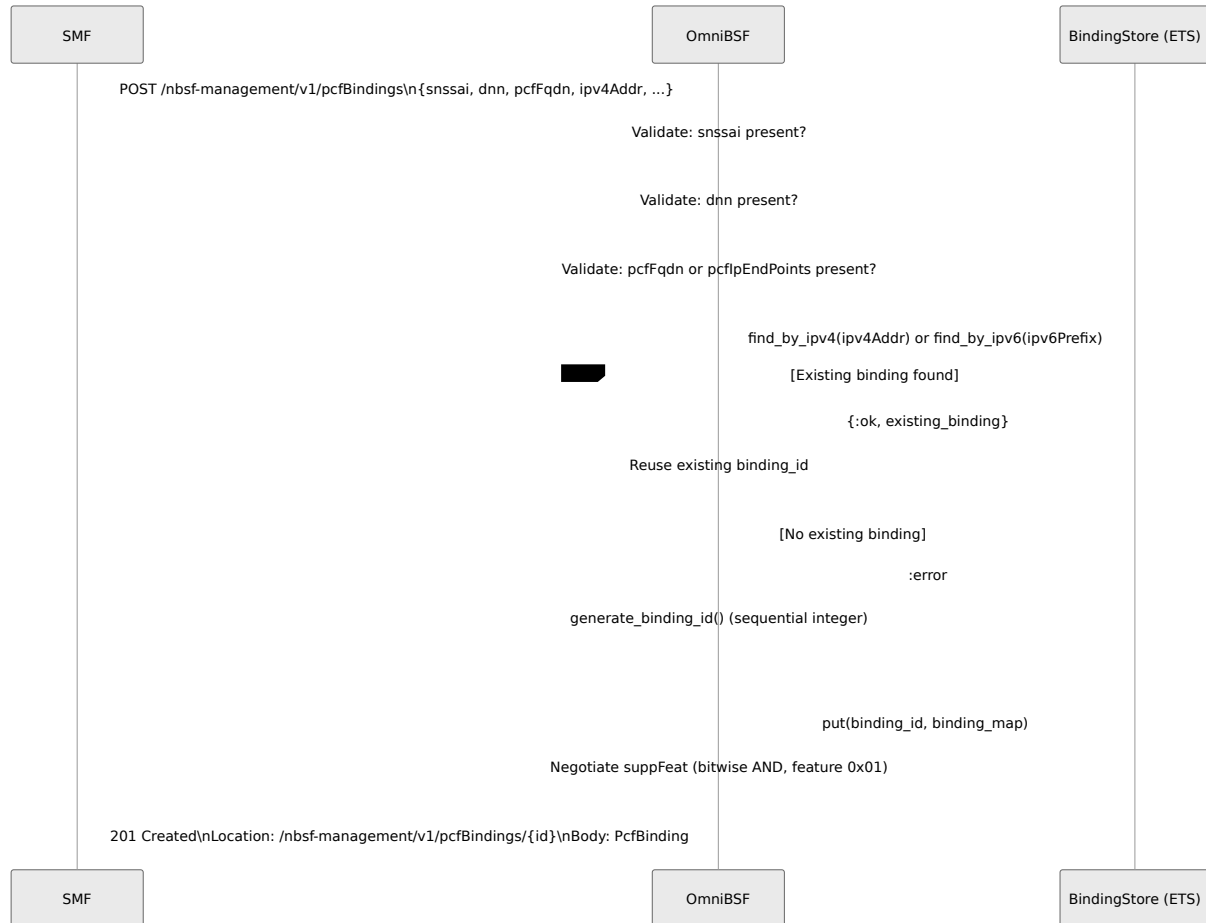
Parameter Table

Parameter	Default	Type	Description
sbi_scheme	"http"	string	Transport scheme for the SBI listener. Set to "https" when TLS termination is handled by OmniBSF directly.
sbi_addr	"127.0.0.15"	string	IP address the SBI HTTP server binds to.
sbi_port	7777	integer	TCP port the SBI HTTP server listens on.
nrf_uri	"http://127.0.0.10:7777"	string	Base URI of the NRF used for NF registration and heartbeat.
mcc	"999"	string	Mobile Country Code. Used in the NF profile

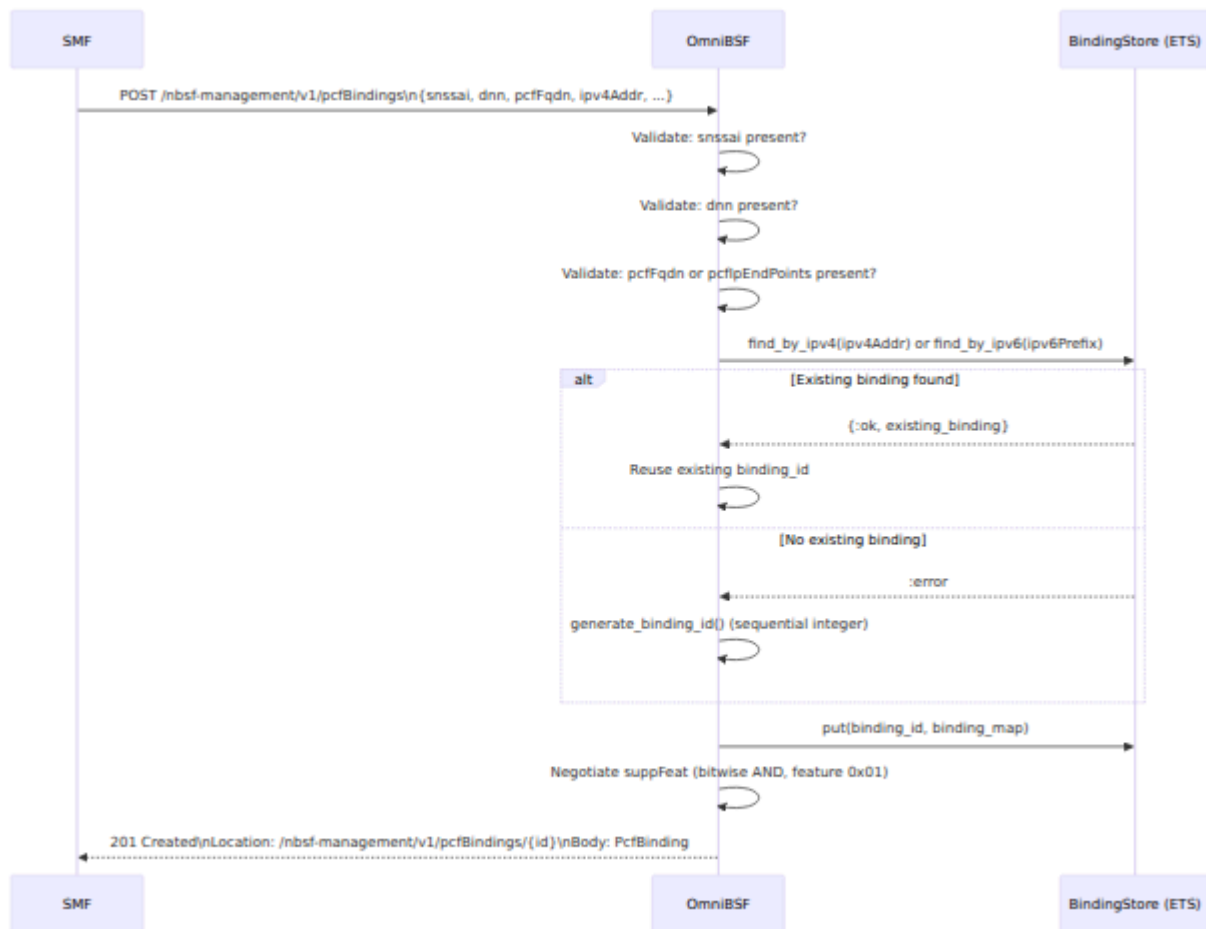
Parameter	Default	Type	Description
			submitted to the NRF during registration.
mnc	"70"	string	Mobile Network Code. Used in the NF profile submitted to the NRF during registration.
heartbeat_interval	10_000	integer (ms)	Interval between NRF heartbeat (NFUpdate) requests. A value of 10_000 (10 s) is suitable for most deployments reduce if NRF TTLs are shorter.

Key Procedures

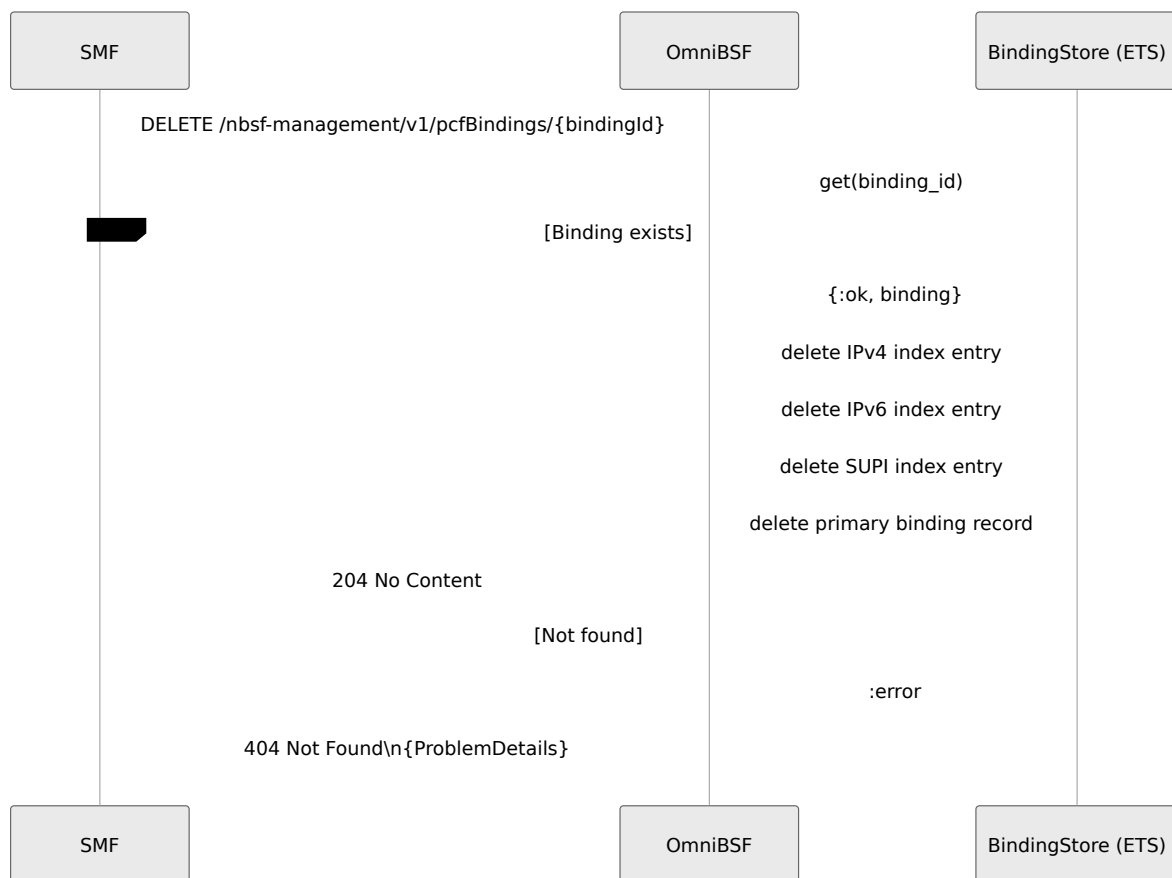
PCF Binding Registration (POST)



PCF Binding Discovery (GET)



PCF Binding Deregistration (DELETE)



NRF Registration and Heartbeat

OmniBSF registers as a BSF NF profile with the NRF on startup using the `Omni5gEx` shared library. It sends periodic NFUpdate heartbeat requests at the configured `heartbeat_interval`. If NRF registration is lost (e.g., NRF restart), OmniBSF will attempt re-registration automatically.

Observability

Telemetry Events

Event	Measurements	Tags	Description
<code>[:omnibsf, :pcf_binding, :register]</code>	count	dnn	Fired on every successful binding registration
<code>[:omnibsf, :pcf_binding, :discover]</code>	count	result (found/not_found)	Fired on every discover request
<code>[:omnibsf, :pcf_binding, :deregister]</code>	count	binding_id	Fired on every successful deregistration
<code>[:omnibsf, :binding, :register]</code>	count	—	Aggregate registration counter
<code>[:omnibsf, :binding, :discover]</code>	count	result	Aggregate discovery counter
<code>[:omnibsf, :binding, :deregister]</code>	count	—	Aggregate deregistration counter
<code>[:omnibsf, :bindings, :active]</code>	count	—	Gauge: current active binding count

Event	Measurements	Tags	Description
<code>[:omni5g, :nrf, :registration]</code>	<code>status</code>	<code>nf_type</code>	NRF registration status (1=registered, 0=not)

Prometheus Metrics

BSF Metrics

Metric	Type	Tags	Description
<code>omni_bsf.pcf_binding.register.count</code>	counter	<code>dnn</code>	PCF binding register count
<code>omni_bsf.pcf_binding.discover.count</code>	counter	<code>result</code>	PCF binding discover count
<code>omni_bsf.pcf_binding.deregister.count</code>	counter	<code>binding_id</code>	PCF binding deregister count
<code>omni_bsf.binding_registers.total</code>	counter	--	Aggregate binding registers count
<code>omni_bsf.binding_discovers.total</code>	counter	<code>result</code>	Aggregate binding discovers count
<code>omni_bsf.binding_deregisters.total</code>	counter	--	Aggregate binding deregisters count
<code>omni_bsf.active_bindings.count</code>	gauge	--	Current active bindings
<code>omni_bsf.nrf.registration.status</code>	gauge	<code>nf_type</code>	NRF registration status (1=registered, 0=not registered)

BEAM VM Metrics

Metric	Type	Description
<code>beam.memory.total</code>	gauge	Total BEAM memory in bytes
<code>beam.memory.processes</code>	gauge	Memory used by Erlang processes
<code>beam.memory.processes_used</code>	gauge	Memory actually used by processes
<code>beam.memory.system</code>	gauge	System 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 in seconds

Log Patterns

Level	Pattern	Meaning
info	PCF binding registered: id=<N> ipv4=<addr> ...	Successful registration
info	PCF binding deregistered: id=<N>	Successful deregistration
debug	PCF binding discovered: id=<N>	Successful discovery
debug	PCF binding not found for query: ...	Discovery miss
warning	PCF binding not found for deregister: id=<N>	DELETE for unknown ID

Known Limitations

ID	Severity	Description
BSF-M1	Medium	<code>PATCH /nbsf-management/v1/pcfBindings/{bindingId}</code> is not implemented. PCF binding updates require a full DELETE followed by a new POST. Per TS 29.521 the PATCH method is optional when the <code>BindingUpdate</code> supported feature is not negotiated, but clients that negotiate feature 0x01 may expect PATCH support.
BSF-M2	Medium	The <code>PcfBinding</code> object returned in POST 201 and GET 200 responses does not include the <code>bindingId</code> field. Consumers must extract the binding ID from the <code>Location</code> header of the POST response.
BSF-M3	Medium	The GET discovery endpoint returns a single <code>PcfBinding</code> object rather than the <code>PcfBindingList</code> array defined in TS 29.521. Consumers that expect the collection form will receive an unexpected response type.
BSF-M4	Medium	The SUPI index uses an ETS <code>:set</code> table, meaning only one binding per SUPI is maintained. If a subscriber has multiple simultaneous PDU sessions across different DNNs, only the most recently registered binding is indexed by SUPI. IPv4/IPv6 indexes are unaffected.
BSF-L1	Low	<code>macAddr48</code> query parameter is not supported. MAC-address-based lookup (used for 5G-LAN and some fixed-access scenarios) always returns 204.
BSF-L2	Low	Query parameters <code>pcfSetId</code> , <code>pcfRegionId</code> , and <code>bindLevel</code> are accepted but ignored. No filtering by PCF set or region is applied.

ID	Severity	Description
BSF-L3	Low	Binding IDs are sequential integers (1, 2, 3, ...) rather than UUIDs or opaque strings. This leaks the total number of bindings created since startup to any consumer that inspects <code>Location</code> headers.
BSF-L5	Low	Bindings have no TTL or expiry mechanism. A binding created for a PDU session that terminates without sending a DELETE will remain in memory indefinitely. Operator intervention (process restart) is required to purge stale bindings.

Troubleshooting

204 returned when a binding is expected

1. Verify the query parameter matches exactly what was registered. IPv4 and IPv6 lookups are exact-string matches — formatting differences (e.g., leading zeros, prefix length notation) will cause a miss.
2. Check logs for `PCF binding not found for query:` to see what was searched.
3. Confirm the SMF successfully completed the POST registration (look for `PCF binding registered:` in logs).
4. Note that DNN+S-NSSAI lookup performs a full table scan; if `sst` does not match exactly (integer vs string type mismatch in the original POST), the scan will not find the record.

404 on DELETE

The binding ID in the request path does not match any active binding. Either the binding was already deregistered, or OmniBSF restarted (all bindings are lost on restart because storage is in-memory only). Check logs for `PCF binding not found for deregister:`.

400 on POST

The request body is missing one or more mandatory fields. The `detail` field in the `ProblemDetails` response body will identify the specific missing field: "No S-NSSAI", "No DNN", or "No PCF address information".

High memory use / stale bindings accumulating

OmniBSF has no binding TTL (BSF-L5). If PDU sessions terminate without sending DELETE requests, bindings accumulate indefinitely. Monitor the `omni_bsf.active_bindings.count` metric. A controlled restart clears all bindings; re-registration by active SMF instances will repopulate the store.

NRF registration failing

Check the `omni_bsf.nrf.registration.status` metric and logs for NRF-related errors. Verify:

- `nrf_uri` points to a reachable NRF instance.
- The `mcc` and `mnc` values match the PLMN configured in the NRF.
- Network connectivity between OmniBSF's `sbi_addr` and the NRF is working.

Supported features negotiation

OmniBSF advertises and negotiates only feature bit `0x01` (BindingUpdate). Clients that send `suppFeat` values with higher bits set will have those bits cleared in the response. This is expected behaviour per TS 29.521 Section 6.1.6.