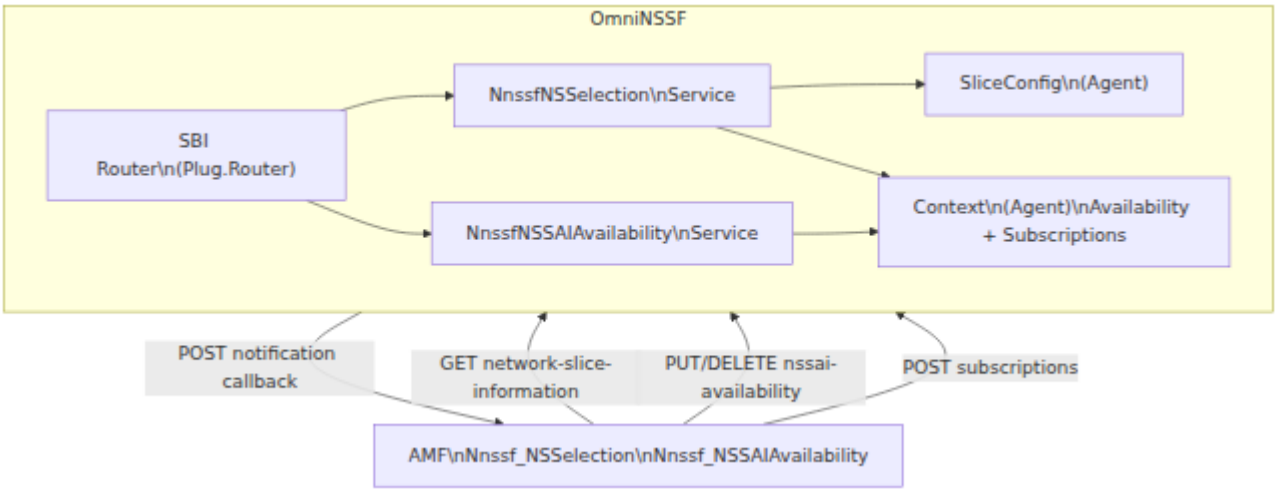
OmniNSSF Operations Guide

Overview

OmniNSSF implements the Network Slice Selection Function (NSSF) of the 5G Core. It provides two SBI services: Nnssf_NSSelection (TS 29.531) which answers AMF queries about which network slices a UE may use, and Nnssf_NSSAIAvailability (TS 29.531) which accepts NSSAI availability reports from AMFs and manages subscriptions to availability change notifications.

Slice configuration — the mapping of S-NSSAIs to NRF URIs, allowed NSSAI per PLMN, and AMF set assignments — is loaded from application configuration at startup. NSSAI availability state reported by AMFs is held in-process in an Agent. Availability change notifications are delivered to subscribers as fire-and-forget HTTP POST calls.

Architecture



3GPP Role and Specification References

Specification	Relevance
TS 23.501	System architecture — NSSF role, network slice concept, S-NSSAI definition (Section 5.15)
TS 23.502	Procedures — slice selection during registration (4.2.3.2), PDU session establishment (4.3.2)
TS 29.531	Nnssf_NSSelection and Nnssf_NSSAIAvailability APIs
TS 23.003	S-NSSAI structure (SST, SD), NSI definition

SBI Endpoints

All endpoints are served under the base URL `{sbi_scheme}://{sbi_addr}:{sbi_port}`.

Method	Path	Servic
GET	`/nnssf-nsselection/v2/network-slice-information`	Nnssf_NSSelec
PUT	`/nnssf-nssaiavailability/v1/nssai-availability/{nfId}`	Nnssf_NSSAIAv
DELETE	`/nnssf-nssaiavailability/v1/nssai-availability/{nfId}`	Nnssf_NSSAIAv
POST	`/nnssf-nssaiavailability/v1/nssai-availability/subscriptions`	Nnssf_NSSAIAv
DELETE	`/nnssf-nssaiavailability/v1/nssai-availability/subscriptions/{subscriptionId}`	Nnssf_NSSAIAv

Request / Response Summary

GET network-slice-information — mandatory query parameters: `nf-id`, `nf-type`. Optional: `slice-info-request-for-registration`, `slice-info-request-

`for-pdu-session`, `snssai`, `tai`, `home-plmn-id`, `supported-features`. Returns `200 OK` with `AuthorizedNetworkSliceInfo`.

PUT nssai-availability/{nfId} — body: `NssaiAvailabilityInfo` (must contain `supportedSnssaiList` or `supportedNssaiAvailabilityData`). Returns `200 OK` with `AuthorizedNssaiAvailabilityInfo`. Triggers notification to all current subscribers.

POST subscriptions — body: `NssaiAvailabilitySubscription` (must contain `nfNssaiAvailabilityUri`). Returns `201 Created` with subscription object and `Location` header.

Configuration Reference

Configuration is read from the application environment key `:omnissf`.

```
config :omnissf,
  sbi_scheme: "http",
  sbi_addr:   "127.0.0.14",
  sbi_port:   7777,
  nrf_uri:    "http://127.0.0.10:7777",
  mcc: "999",
  mnc: "70",
  heartbeat_interval: 10_000,

  nsi_list: [
    %{
      s_nssai: %{sst: 1, sd: "0x000001"},
      nrf_uri: "http://127.0.0.10:7777",
      nsi_id:  "1"
    }
  ],

  allowed_nssai: %{
    "999-70" => [
      %{sst: 1, sd: "0x000001"}
    ]
  },

  amf_set_mapping: %{
    "1-0x000001" => ["1"]
  },

  configured_nssai: %{}
```

Base Parameter Table

Parameter	Type	Default	Description
`sbi_scheme`	string	`"http"`	HTTP scheme for the SBI listening socket (`http` or `https`)
`sbi_addr`	string	`"127.0.0.14"`	IP address the SBI HTTP server binds to
`sbi_port`	integer	`7777`	TCP port the SBI HTTP server listens on
`nrf_uri`	string	`"http://127.0.0.10:7777"`	Base URI of the NRF. Used for NF registration and heartbeat only
`mcc`	string	`"999"`	Mobile Country Code of the home PLMN. Used as the fallback PLMN key

Parameter	Type	Default	Description
			when `home-plmn-id` is absent from a selection query
`mnc`	string	`"70"`	Mobile Network Code of the home PLMN
`heartbeat_interval`	integer (ms)	`10000`	Interval in milliseconds between NRF heartbeat requests

Slice Configuration Parameters

`nsi_list`

A list of Network Slice Instance (NSI) entries. Each entry binds an S-NSSAI to an NRF URI. When the NSSF receives a slice selection query, it looks up the requested S-NSSAI in this list and returns the associated NRF URI in the `nsiInformation` field of the response.

Field	Type	Description
`s_nssai.sst`	integer	Slice/Service Type (1–255). Standard values: 1=eMBB, 2=URLLC, 3=MIoT (TS 23.501 Table 5.15.2.2-1)
`s_nssai.sd`	string or nil	Slice Differentiator as a hex string (e.g., `"0x000001"`). `nil` or absent matches any SD for the given SST (see NSSF-L5)
`nrf_uri`	string	Base URI of the NRF responsible for NF discovery within this slice instance
`nsi_id`	string	Opaque identifier for this NSI, included in the `nsiInformation` response

If no `nsi_list` entry matches the requested S-NSSAI, the NSSF returns `403 Forbidden` with cause `SNSSAI_NOT_FOUND`.

`allowed_nssai`

A map from PLMN key (`"{mcc}-{mnc}"`) to a list of S-NSSAI structs. Controls which slices are included in the `allowedNssaiList` field of the selection response. If the requesting PLMN has no entry, the home PLMN's allowed NSSAI is used as a fallback.

Each S-NSSAI in the list:

Field	Type	Description
`sst`	integer	Slice/Service Type
`sd`	string or nil	Slice Differentiator

`amf_set_mapping`

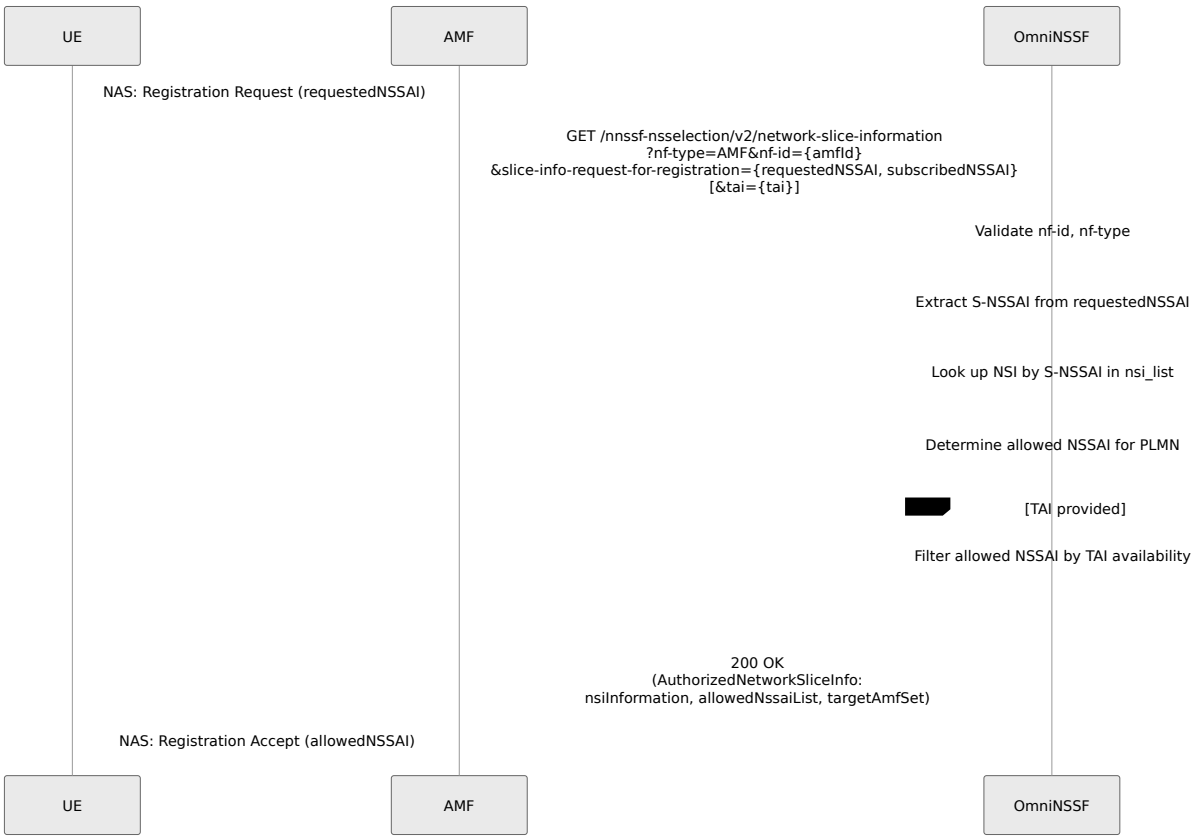
A map from S-NSSAI key (`"{sst}-{sd}"` or `"{sst}"` when no SD) to a list of AMF set ID strings. When populated, the NSSF includes the first entry as `targetAmfSet` in the selection response (see limitation NSSF-L3 — `candidateAmfList` is not populated).

`configured_nssai`

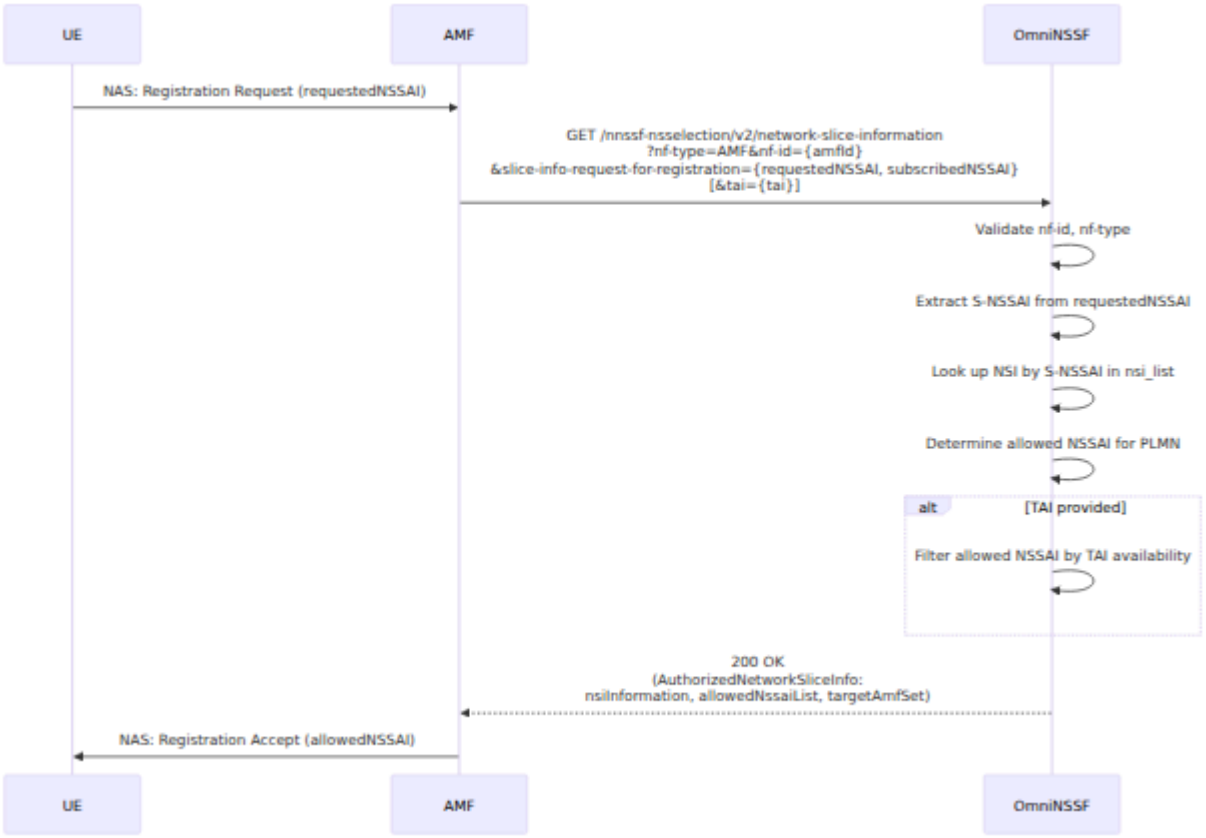
A map from S-NSSAI key to a configured NSSAI structure for roaming scenarios. When present for a given S-NSSAI, the NSSF includes `configuredNssai` in the selection response. Empty by default.

Key Procedures

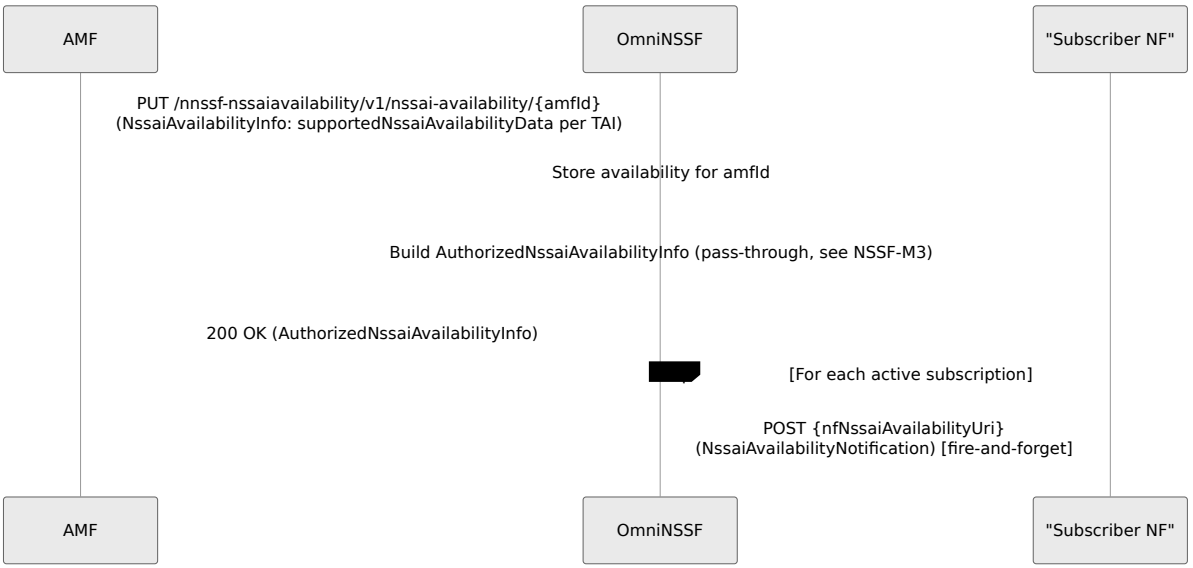
Network Slice Selection — Registration (TS 23.502 Section 4.2.3.2)



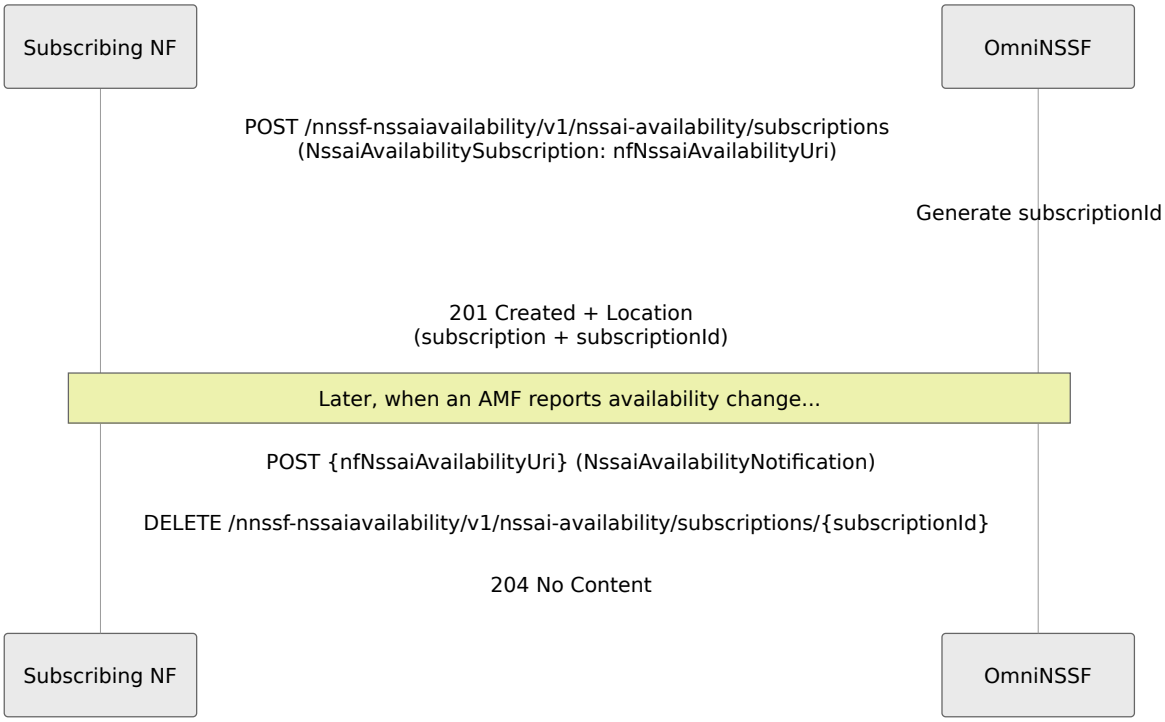
Network Slice Selection — PDU Session (TS 23.502 Section 4.3.2)



NSSAI Availability Reporting (TS 29.531 Section 5.2.2)



NSSAI Availability Subscription / Notification



S-NSSAI Lookup Logic

The NSSF matches a requested S-NSSAI against `nsi_list` using the following rules:

1. SST must match exactly.
2. SD matching: if the `nsi_list` entry has `sd: nil`, it matches any requested SD for that SST (wildcard). If the entry has a specific SD, it is compared case-insensitively after hex normalization.

This means an `nsi_list` entry with `sd: nil` acts as a catch-all for its SST value. See limitation NSSF-L5 for the operational implications of this behavior.

Prometheus Metrics

NSSF Metrics

Metric	Type	Tags
`omni_nssf.nsselection.requests.count`	counter	`result`, `nf_type`
`omni_nssf.nssai_availability.update.count`	counter	`nf_id`
`omni_nssf.nssai_availability.delete.count`	counter	`nf_id`
`omni_nssf.nssai_availability.subscribe.count`	counter	--
`omni_nssf.nssai_availability.unsubscribe.count`	counter	--
`omni_nssf.ns_selection_requests.total`	counter	`result`
`omni_nssf.nssai_availability_updates.total`	counter	--

Metric	Type	Tags
`omni_nssf.nrf.registration.status`	gauge	`nf_type`

BEAM VM Metrics

Metric	Type	Description
`beam.memory.total`	gauge	Total BEAM memory in bytes
`beam.memory.processes`	gauge	Memory used by 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

Known Limitations

ID	Area	Description
NSSF-M2	NSSAI Availability PATCH	There is no `PATCH /nnssf-nssaiavailability/v1/nssai-availability/{nfId}` endpoint. Partial updates to an AMF's availability data are not supported; the full record must be replaced with PUT.
NSSF-M3	Availability Authorization	The NSSF's authorization of reported NSSAI availability is pass-through. The `authorizedNssaiAvailabilityData` in the PUT response is identical to the submitted `supportedNssaiAvailabilityData`. The NSSF does not cross-check reported S-NSSAIs against the configured `nsi_list`.
NSSF-L1	Supported Features	The `supported-features` query parameter is accepted but not processed. No capability negotiation is performed between the AMF and NSSF.
NSSF-L2	Rejected NSSAI	The `AuthorizedNetworkSliceInfo` response does not include `rejectedNssaiInRa` (rejected in registration area) or `rejectedNssaiInTa` (rejected in tracking area) fields. UEs whose requested S-NSSAIs are not available in a specific TA will not receive explicit rejection information.
NSSF-L3	Candidate AMF List	The `candidateAmfList` field is not populated in the selection response. Only `targetAmfSet` (the first entry from `amf_set_mapping`) is included when applicable. Full AMF selection assistance per TS 29.531 is not provided.

ID	Area	Description
NSSF-L4	Subscription PATCH	There is no `PATCH /nnssf-nssaiavailability/v1/nssai-availability/subscriptions/{subscriptionId}` endpoint. Subscription parameters cannot be updated after creation; the subscription must be deleted and re-created.
NSSF-L5	SD Wildcard Matching	An `nsi_list` entry with `s_nssai.sd: nil` matches any SD for the given SST, not just requests without an SD. This means a wildcard entry will also match explicit SD requests. If multiple slices share an SST but differ by SD, each must have its own explicit SD entry in `nsi_list` ordered before any wildcard entry (the first match wins).

Troubleshooting

Slice Selection Returns 403 SNSSAI_NOT_FOUND

No entry in `nsi_list` matches the requested S-NSSAI. Verify:

1. The SST in the request matches an `nsi_list` entry exactly (integer comparison).
2. The SD in the request matches the entry's SD, or the entry has `sd: nil` (wildcard).
3. The `nsi_list` configuration was reloaded after changes — OmniNSSF reads `nsi_list` from application env at startup only. A restart is required to pick up changes.

The NSSF log will show `NSSelection: Cannot find NSI for S-NSSAI [SST:{sst} SD:{sd}]`.

Slice Selection Returns 400 MANDATORY_QUERY_PARAM_MISSING

The `nf-id` or `nf-type` query parameter is absent from the request. Both are mandatory per TS 29.531. Verify the AMF is including these parameters in its selection request.

Allowed NSSAI Missing from Selection Response

The `allowedNssaiList` field is only included in the response when a PLMN key can be determined from the request. The PLMN key is derived from the `home-plmn-id` query parameter, or falls back to the NSSF's own MCC/MNC (`mcc`/`mnc` config). If neither can be parsed, the field is omitted. Verify:

1. `mcc` and `mnc` are correctly configured for the home PLMN.
2. The `allowed_nssai` map contains an entry for the key `"{mcc}-{mnc}"`.

TAI-Based Filtering Removes All Allowed NSSAIs

When a `tai` parameter is provided and no AMF has reported availability data via PUT, the NSSF defaults to allowing all configured NSSAIs (empty availability map = no restriction). Once any AMF reports availability, only S-NSSAIs included in a matching TAI entry in that availability data are returned. If the TAI in the query does not appear in any AMF's reported `supportedNssaiAvailabilityData`, all NSSAIs will be filtered out. To diagnose, check whether AMFs have submitted PUT requests and whether the TAC in those reports matches the TAC in the query.

SD Wildcard Causes Wrong NSI to Be Selected

If an `nsi_list` entry with `sd: nil` is listed before more specific entries, it will match first for any SST request regardless of the SD. Ensure specific SD entries

appear before wildcard entries in the `nsi_list`. See limitation NSSF-L5.

Availability Notifications Not Received by Subscriber

Notifications are sent asynchronously (fire-and-forget) in a spawned Task. Delivery failures are logged as warnings but are not retried. Verify:

1. The `nfNssaiAvailabilityUri` in the subscription is reachable from the NSSF host.
2. The subscriber NF is accepting POST requests at that URI and returning a 2xx response.

Failed notification attempts are logged as `NSSAIAvailability: notification to {uri} failed: {reason}`.

Stale Availability or Subscription State After AMF Restart

NSSAI availability records and subscriptions are stored in-process and are not persistent. They survive OmniNSSF restarts only for the lifetime of the Erlang VM. If the AMF restarts and does not re-register its availability via PUT, the NSSF will continue to serve stale (or absent) availability data for that AMF's NF ID. The AMF should re-submit its NSSAI availability on reconnection. Similarly, subscriptions created before an NSSF restart must be re-created by the subscriber.

Log Correlation

NSSAI selection log lines are prefixed `NSSelection:` and include the requesting `nf-id`. NSSAI availability log lines are prefixed `NSSAIAvailability:` and include the `nfId`. Subscription notification attempts log the callback URI. Use `nf-id` values to correlate selection requests with availability updates from the same AMF.