← Overview

OmniAUSF Configuration

Configuration is read from the application environment key `:omniausf`, populated from `config/runtime.exs`. Most runtime keys are backed by an OS environment variable read at startup, shown below; the two context-reaper keys are read from the `:omniausf` application environment directly and are not env-var backed.

```
config :omniausf,
  udm_uri:                 "http://127.0.0.12:7777",
  sbi_scheme:              "http",
  sbi_addr:                "127.0.0.19",
  sbi_port:                7777,
  nrf_uri:                 "http://127.0.0.1:7777",
  mcc:                     "999",
  mnc:                     "70",
  prometheus_metrics_port: 9568,
  heartbeat_interval:      10_000

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

Parameter Reference

Parameter	Type	Required	Default
`udm_uri`	string	No	`"http://127.0.0.12:77`
`sbi_scheme`	string	No	`"http"`
`sbi_addr`	string	No	`"127.0.0.19"`
`sbi_port`	integer	No	`7777`
`nrf_uri`	string	No	`"http://127.0.0.1:777`

Parameter	Type	Required	Default
`mcc`	string	No	`"999"`
`mnc`	string	No	`"70"`
`prometheus_metrics_port`	integer	No	`9568`
`heartbeat_interval`	integer (ms)	No	`10000`
`context_ttl_seconds`	integer (s)	No	`3600`

Parameter	Type	Required	Default
`context_reap_interval_ms`	integer (ms)	No	`60000`

> **Note on `nrf_uri` default:** `config/runtime.exs` defaults `NRF_URI` to `http://127.0.0.1:7777`. In a standard loopback deployment the NRF is reached at `http://127.0.0.10:7777`; set `NRF_URI` explicitly to match your NRF address.

Logging

Logs are emitted as JSON (`OmniLogger.JsonFormatter`) with all metadata included. Log lines on the authentication path carry session context (`supi`, `auth_ctx_id`, `procedure`) so a single authentication exchange can be traced end-to-end by `auth_ctx_id` or SUPI/SUCI. The log level can be changed at runtime through the Management API (see below).

Management API (Port 8443)

In addition to the SBI listener and metrics port, OmniAUSF exposes an HTTPS management/OAM API on port `8443` (TLS, product name `OmniAUSF`). It is used for operational inspection and runtime control, and is not part of the 3GPP SBI. Key endpoints:

Method	Path	Purpose
GET	`/api/status/api`	API service status
GET	`/api/status/nrf`	NRF registration status
GET	`/api/status/nf`	NF summary (type `AUSF`, `nausf-auth` service, active UE context count)
GET	`/api/ausf_sessions`	List active authentication sessions (context id, SUCI/SUPI, serving network name, auth type/result)
DELETE	`/api/ausf_sessions/{id}`	Clear a specific authentication session by context id
POST	`/api/ausf_sessions`	Clear all active authentication sessions
GET	`/api/statistics`	Authentication session counters (total / success / failure / pending)
GET	`/api/oam/config`	View effective runtime configuration (`sbi_addr`, `sbi_port`, `nrf_uri`, `udm_uri`, NF capacity/priority, log level)
PATCH	`/api/oam/config/{id}`	Update runtime config: `udm_uri`, `nrf_uri`, `nf_capacity`, `nf_priority`, `log_level`
POST	`/api/oam/log_level`	Change the log level at runtime (`emergency`…`debug`)
GET	`/api/oam/resources`	BEAM VM resource usage (memory, process/port counts,

Method	Path	Purpose
		OTP/BEAM version)
GET	`/api/oam/diagnostics`	BEAM VM diagnostics (run queue, I/O, GC, connected nodes)
POST	`/api/oam/nrf/reregister`	Force re-registration with the NRF

Runtime changes made through `PATCH /api/oam/config` and `POST /api/oam/log_level` apply to the running node only and are not persisted; they revert to the configured values on restart.

OAM-only settings

Two values are exposed through `GET`/`PATCH /api/oam/config` but are not startup parameters in the table above - they live in the NRF profile rather than the core `:omniausf` config:

Field	Type	Meaning
`nf_capacity`	integer (0-65535)	Relative capacity advertised in the AUSF's NRF profile, used by consumers as a load-balancing weight when several AUSF instances are registered. A `PATCH` re-advertises the new value to the NRF
`nf_priority`	integer (0-65535)	Relative selection priority advertised in the NRF profile; lower is preferred. A `PATCH` re-advertises the new value to the NRF

`PATCH /api/oam/config` validates these before applying: `nf_capacity`/`nf_priority` must be integers (a decimal string is accepted and parsed), and `log_level` must be a known level. An invalid value rejects the whole request with `400` without applying anything.

Inspecting authentication sessions

`GET /api/ausf_sessions` returns one entry per in-memory UE authentication context, with these fields:

Field	Meaning
`ctx_id`	The `authCtxId` (UUID); pass to `DELETE /api/ausf_sessions/{ctx_id}` to clear one session
`suci`	The `supiOrSuci` the AMF presented at initiation
`supi`	The plain SUPI resolved by the UDM (null until the vector response arrives)
`serving_network_name`	The SN name the session's keys are bound to
`auth_type`	`5G_AKA` once the vector is stored
`auth_result`	Settled outcome, or null while the session is still in flight (pending)
`has_kausf`	Whether `K_AUSF` is held for this session - indicates the vector was fetched and keys derived. The key material itself is never exposed

Because contexts are deleted the moment a confirmation settles, entries in this list are normally short-lived; a session that persists is one the AMF initiated but never confirmed or deregistered.

← Overview

OmniAUSF Metrics

Metrics are exposed at `http://{host}:{prometheus_metrics_port}/metrics` (default port `9568`).

AUSF Metrics

Metric	Type	Labels	Description
`omni_ausf_auth_count`	counter	`result`	UE authentication operations. `result` is one `initiated` (vector obtaine challenge returned), `success` (RES* matched XRES `failure` (UDM error, malform vector, or RES* mismatch), `resync` (an AUTS/SQN resynchronisat was detected and relayed to the UDM), or `resync_invali` (a malformed resync blob wa dropped)
`omni_ausf_nrf_registration_status`	gauge	`nf_type`	NRF registratio status (1 = registered, 0 = not registered)
`omni_ausf_active_contexts_count`	gauge	-	Number of acti UE authentication

Metric	Type	Labels	Description
			contexts. Live gauge driven b the telemetry event `[:omniausf, :ue_contexts, :active]`, emitted on eve context create and delete and on each ContextReaper sweep, so it tracks the current in-fligh context count

> **`omni_ausf_active_contexts_count` is a live gauge.** It is driven by the telemetry event `[:omniausf, :ue_contexts, :active]`, which the AUSF context store emits whenever the set of active contexts changes: on context create, on delete (confirmation, deregister, or authentication-result delete), and on each ContextReaper sweep that reaps idle contexts. The gauge therefore mirrors the current active auth-context count and is safe to alert on and dashboard. The OAM API remains available for cross-checks: `GET /api/status/nf` (includes an active UE context count) or `GET /api/statistics` (total / success / failure / pending session counters).

> `omni_ausf_auth_count` covers the primary authentication flow plus the resynchronisation path. The `resync` and `resync_invalid` labels count SQN resynchronisation attempts detected on the initiation request (validated-and-relayed, and malformed-and-dropped, respectively). The deregister and authentication-result-delete paths emit their own telemetry events but are not currently attached to any exported Prometheus metric.

> **Internal telemetry (not a Prometheus metric).** The context reaper emits a `[:omniausf, :ausf, :context_reap]` telemetry event carrying the

> count of contexts reaped per sweep. This is internal telemetry for operators wiring their own handlers; it is not registered as an exported Prometheus metric.

Example PromQL

- Authentication success rate over 5 minutes: `sum(rate(omni_ausf_auth_count{result="success"}[5m])) / sum(rate(omni_ausf_auth_count{result="initiated"}[5m]))`
- Authentication failures per second: `sum(rate(omni_ausf_auth_count{result="failure"}[5m]))`
- AUSF currently registered with the NRF: `omni_ausf_nrf_registration_status == 1`
- SQN resynchronisation attempts per second (valid vs malformed): `sum(rate(omni_ausf_auth_count{result="resync"}[5m]))` and `sum(rate(omni_ausf_auth_count{result="resync_invalid"}[5m]))`
- Active in-flight authentication contexts: `omni_ausf_active_contexts_count`

For an independent cross-check of the in-flight context count, the OAM API (`GET /api/status/nf` or `GET /api/statistics`) reports the same figure.

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_system`	gauge	System memory (ETS, atoms, code)
`beam_processes_count`	gauge	Number of Erlang processes
`beam_vm_uptime`	gauge	VM uptime in seconds

← Overview

OmniAUSF Operations

OmniAUSF implements the Authentication Server Function (AUSF) of the 5G Core, executing 5G-AKA primary authentication between the AMF (N12) and the UDM (N13). It requests an authentication vector from the UDM, relays the RAND/AUTN challenge (and HXRES*) to the AMF, verifies the UE's response, derives the anchor key K_SEAF, and reports the authentication result back to the UDM.

3GPP Role and Specification References

Specification	Relevance
TS 23.501	System architecture - AUSF role in the 5G System
TS 33.501	Security architecture - 5G-AKA authentication (6.1.3), K_AUSF/K_SEAF key hierarchy, HXRES*/HRES* (Annex A.5), K_SEAF derivation (Annex A.6)
TS 29.509	Nausf_UEAuthentication service API (HTTP SBI)
TS 29.503	Nudm_UEAuthentication service API - authentication vector generation and authentication result reporting
TS 29.510	Nnrf_NFManagement and Nnrf_NFDiscovery service APIs - NF registration/heartbeat and UDM discovery
TS 24.501	NAS - serving network name format (`5G:mnc<MNC>.mcc<MCC>.3gppnetwork.org`)

Interfaces

Interface	Peer	Direction	Transport	Purpose
N12 (Nausf_UEAuthentication)	AMF	Inbound	HTTP SBI	UE authentica initiation and AKA confirmat
N13 (Nudm_UEAuthentication)	UDM	Outbound	HTTP SBI	Authentication vector genera and authentic result reportin
Nnrf	NRF	Outbound	HTTP SBI	NF registratio heartbeat, an peer-NF (UDM discovery (`Nnrf_NFDisc` TS 29.510)

SBI Endpoints

All SBI endpoints are served under the base URL `{sbi_scheme}://{sbi_addr}:{sbi_port}` and use `Content-Type: application/json` on request. The successful create response uses `application/3gppHal+json` per TS 29.509.

Method	Path	Service	
POST	`/nausf-auth/v1/ue-authentications`	Nausf_UEAuthentication	I a – s s
PUT	`/nausf-auth/v1/ue-authentications/{authCtxId}/5g-aka-confirmation`	Nausf_UEAuthentication	C A r
DELETE	`/nausf-auth/v1/ue-authentications/{authCtxId}/5g-aka-confirmation`	Nausf_UEAuthentication	D a /
POST	`/nausf-auth/v1/ue-authentications/deregister`	Nausf_UEAuthentication	R c R

Request / Response Summary

Initiate UE authentication - request body carries `supiOrSuci` (SUPI or SUCI; the field `suci` is also accepted) and `servingNetworkName`; an optional `resynchronizationInfo` object is passed through to the UDM for SQN resynchronisation. The AUSF requests a 5G-AKA vector from the UDM, then returns `201 Created` with a `UEAuthenticationCtx` body:

- `authType: "5G_AKA"`
- `5gAuthData`: `{ rand, autn, hxresStar }` (hex strings)
- `_links.5g-aka.href`: the absolute confirmation URL for this context
- `servingNetworkName`

A `Location` header points at the created context resource (`.../ue-authentications/{authCtxId}`).

5G-AKA confirmation - request body carries `resStar` (hex). The AUSF compares the decoded `resStar` against the stored `XRES*` (TS 33.501 6.1.3.2; the AMF is responsible for the HRES*/HXRES* hash check). It records the outcome in the UDM (`auth-events`) and always returns `200 OK` with a `ConfirmationDataResponse`:

- On success: `{ authResult: "AUTHENTICATION_SUCCESS", supi, kseaf }` (K_SEAF as a hex string).
- On failure: `{ authResult: "AUTHENTICATION_FAILURE", supi }`. Note that an authentication failure is signalled in the body via `authResult`, **not** by an HTTP error status.

The authentication context is deleted after confirmation regardless of outcome.

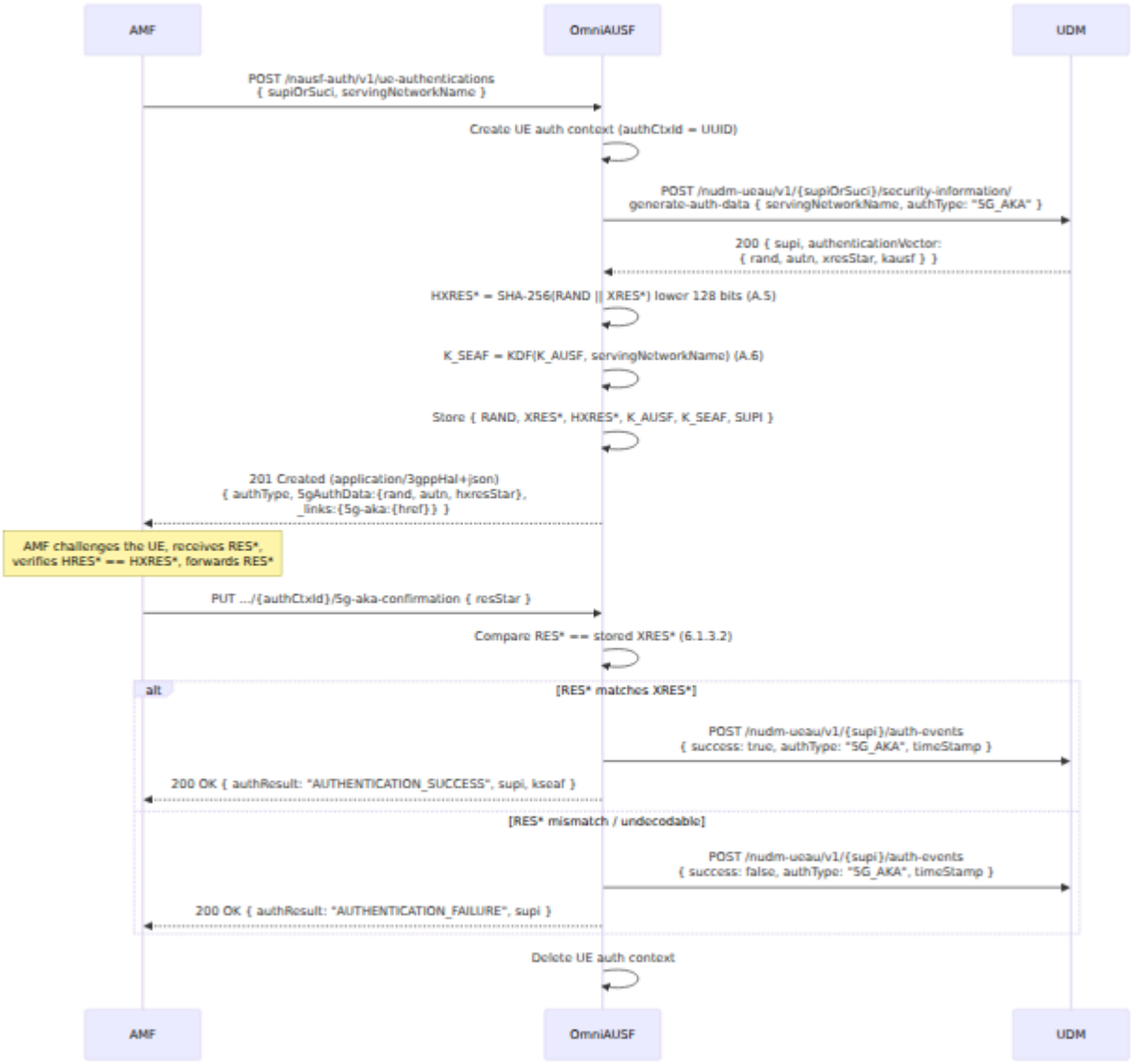
Error Responses

Errors are returned as `ProblemDetails` bodies.

HTTP Status	Cause	Condition
400	`MANDATORY_IE_MISSING`	A required field is missing (`supiOrSuci`/`servingNetworkName` on initiate; `resStar` on confirmation; `supi` on deregister)
404	`USER_NOT_FOUND`	The UDM returned 404 for the subscriber during vector generation
404	`CONTEXT_NOT_FOUND`	The `authCtxId` in a confirmation request matches no active context
500	`SYSTEM_FAILURE`	UDM unreachable, or the UDM returned a malformed authentication vector

Key Procedures

5G-AKA Authentication (TS 33.501 Section 6.1.3)



Initiation. On `POST .../ue-authentications` the AUSF creates a context, then requests a 5G-AKA vector from the UDM at `/nudm-ueau/v1/{supiOrSuci}/security-information/generate-auth-data`, including the AUSF NF instance id and any `resynchronizationInfo` supplied by the AMF. The UDM returns an authentication vector (`rand`, `autn`, `xresStar`, `kausf`) and the resolved `supi` (de-concealed from the SUCI). The AUSF computes `HXRES*` from RAND and XRES*, derives `K_SEAF` from `K_AUSF` bound to the serving network name, stores the material, and returns the challenge to the AMF. The

vector fields are hex-decoded defensively - a missing or malformed vector produces a clean `500 SYSTEM_FAILURE` rather than a crash.

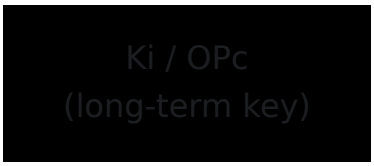
Confirmation. On `PUT .../{authCtxId}/5g-aka-confirmation` the AUSF looks up the context, decodes `resStar`, and compares it directly to the stored `XRES*`. It reports the result to the UDM via `auth-events` (best-effort; a failed report is logged but does not change the response to the AMF) and returns the outcome in the body. The context is then deleted. An unknown `authCtxId` returns `404 CONTEXT_NOT_FOUND`.

Serving network name. Built from `mcc`/`mnc` as `5G:mnc<MNC>.mcc<MCC>.3gppnetwork.org` per TS 24.501, with two-digit MNCs zero-padded to three digits. The serving network name binds `K_SEAF` to the serving network, so the AMF and AUSF must agree on the PLMN identity.

SQN resynchronisation. When the AMF includes `resynchronizationInfo` (the `RAND` and `AUTS` produced by the UE) in the initiation request, the AUSF detects the resync condition, validates that both `rand` and `auts` are well-formed hex, and relays the validated pair to the UDM/ARPF in the `generate-auth-data` request so the UDM can re-derive the sequence number (TS 33.501 6.1.3.3). Malformed resync input is dropped and not forwarded. The AUSF does not itself compute the sequence number, but it is not a blind pass-through: it inspects and validates the resync blob before relaying it.

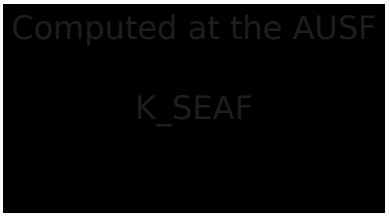
Key Hierarchy and Derivation Split (TS 33.501 Annex A)

The 5G-AKA key hierarchy is split between the UDM and the AUSF. The AUSF never sees the long-term key (Ki) or CK/IK - it receives `K_AUSF` already derived by the UDM inside the authentication vector, and derives only the anchor key `K_SEAF` from it. The AMF (not the AUSF) continues the chain down to `K_AMF` and the access-stratum keys.



UDM / ARPF

K_SEAF = KDF(K_AUSF,
SNN)
FC 0x6C (A.6)



K_AMF = KDF(K_SEAF,
SUPI, ABBA)
FC 0x6D (A.7)

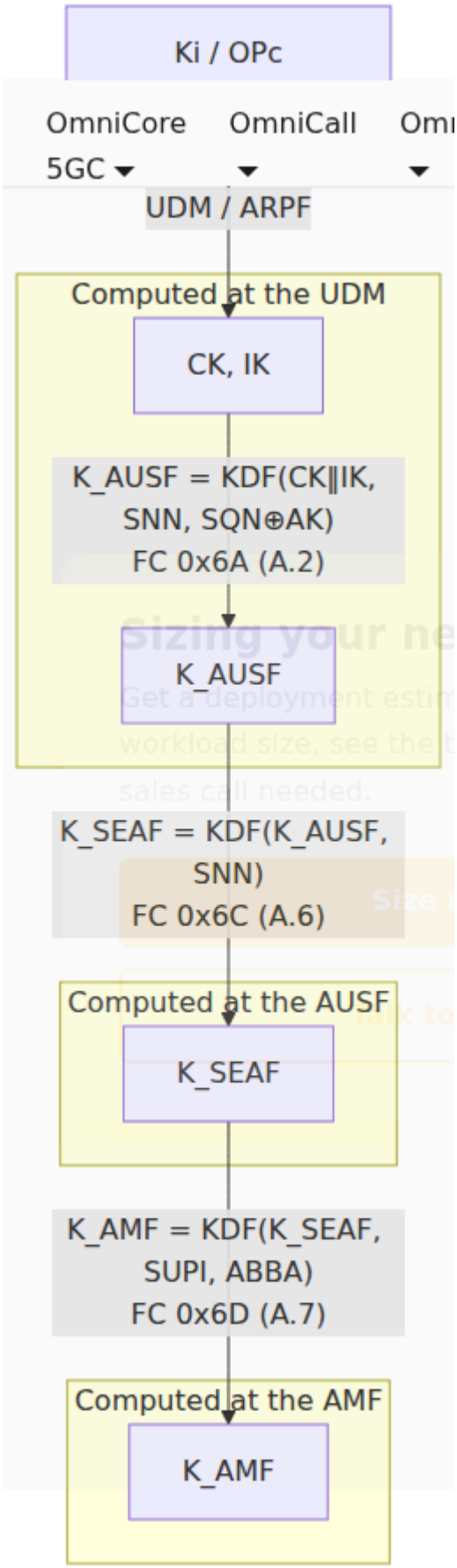
Key / value	Where derived	Derivation	Purpose in the AUSF
`K_AUSF`	UDM (ARPF)	`KDF(CK||IK, servingNetworkName, SQN⊕AK)`, FC `0x6A` (A.2)	Received in the vector; the AUSF anchor key from which `K_SEAF` is derived. Held in context, never returned to the AMF
`K_SEAF`	AUSF	`KDF(K_AUSF, servingNetworkName)`, FC `0x6C` (A.6)	The serving-network anchor key. Returned to the AMF only on `AUTHENTICATION_SUCCESS`; the AMF derives `K_AMF` from it
`HXRES*`	AUSF	lower 128 bits of `SHA-256(RAND || XRES*)` (A.5)	Returned to the AMF at initiation so the AMF can match the UE's `HRES*` before forwarding `RES*`
`XRES*`	UDM (ARPF)	`KDF(CK||IK, SNN, RAND, XRES)`, FC `0x6B` (A.4)	Received in the vector; the expected response the AUSF compares `RES*` against at confirmation

Because `K_SEAF` is bound to the `servingNetworkName` (and `K_AUSF` upstream is bound to the same SN name and to `SQN⊕AK`), the AMF, AUSF and UDM must all agree on the PLMN identity for the derived keys to match the UE's. A serving-network-name mismatch surfaces as an `AUTHENTICATION_FAILURE` at confirmation, not as an explicit error.

Delete 5G-AKA Authentication Result (TS 29.509)

The AMF can explicitly delete an authentication result / context without a confirmation exchange. This maps to `Delete5gAkaAuthenticationResult` in TS

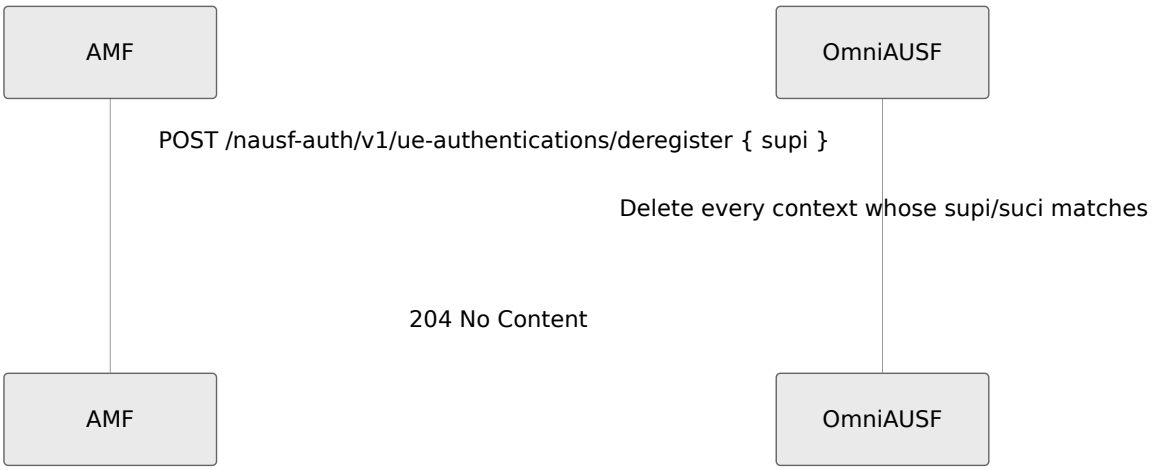
29.509.


Ki / OPc
OmniCore 5GC
OmniCall
UDM / ARPF
Computed at the UDM
CK, IK
K_AUSF = KDF(CK‖IK, SNN, SQN⊕AK)
FC 0x6A (A.2)
K_AUSF
K_SEAF = KDF(K_AUSF, SNN)
FC 0x6C (A.6)
Computed at the AUSF
K_SEAF
K_AMF = KDF(K_SEAF, SUPI, ABBA)
FC 0x6D (A.7)
Computed at the AMF
K_AMF

The AUSF deletes the context unconditionally (an unknown `authCtxId` is a no-op) and always returns `204 No Content`; no UDM call is made.

Deregister (TS 29.509)

Removes any UE security context(s) the AUSF holds for a given SUPI.



`supi` is mandatory (a missing/empty value returns `400 MANDATORY_IE_MISSING`). The AUSF iterates its in-memory contexts and deletes any whose stored `supi` or `suci` equals the requested SUPI; the operation is local only and does not call the UDM. It always returns `204 No Content`, even when no matching context exists.

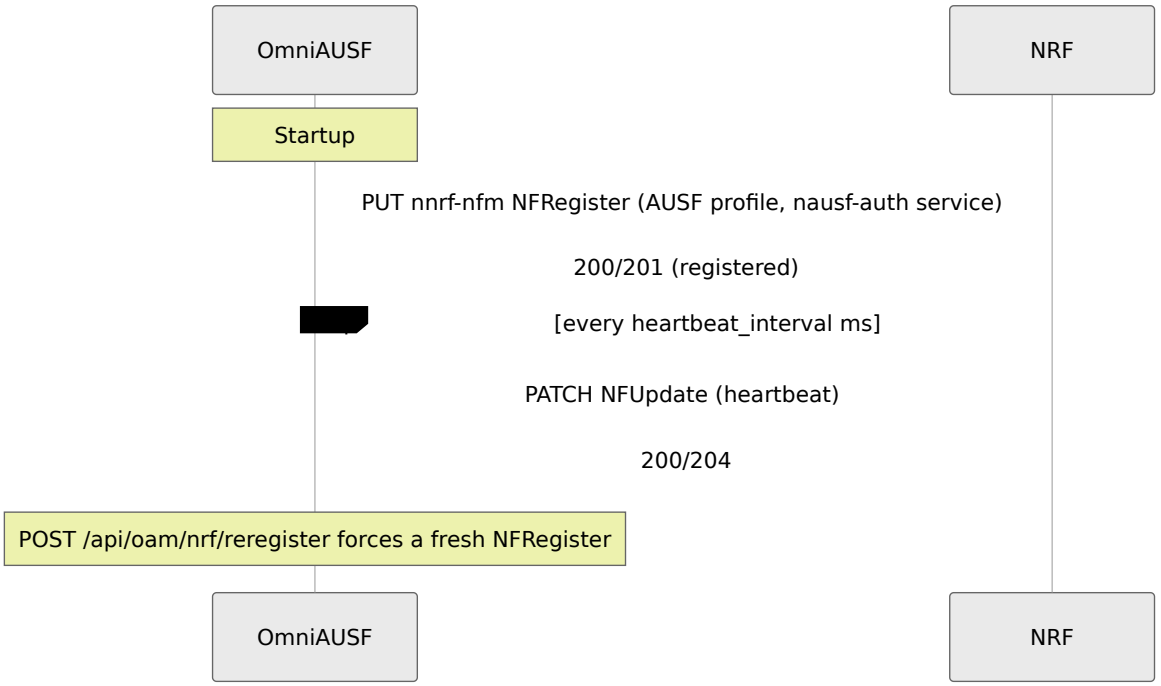
UE Authentication Context

Each initiation creates one in-memory UE authentication context, keyed by a generated `authCtxId` (UUID). The context holds the material needed to complete the exchange and is the unit the OAM session endpoints (`GET /api/ausf_sessions`, `DELETE /api/ausf_sessions/{id}`) list and clear.

Field	Set at	Meaning
`ctx_id`	initiation	The `authCtxId` UUID; appears in the confirmation URL and OAM session list
`suci`	initiation	The `supiOrSuci` originally presented by the AMF (SUCI or SUPI)
`supi`	vector response	The plain SUPI resolved (de-concealed) by the UDM
`serving_network_name`	initiation	The SN name the keys are bound to
`rand`, `xres_star`, `hxres_star`, `kausf`, `kseaf`	vector response	The vector and derived keys held for the lifetime of the exchange
`auth_type`	vector response	`5G_AKA`
`auth_result`	-	Populated for a settled session; a context still in flight has no result and is counted as `pending` in `GET /api/statistics`

A context normally exists only between initiation and settlement. It is deleted on confirmation (success **or** failure), on explicit `DELETE .../5g-aka-confirmation`, on `deregister` for its SUPI, and on process restart. In addition, idle or abandoned contexts are garbage-collected by a supervised periodic reaper: a context the AMF initiates but never confirms is reaped once it exceeds `context_ttl_seconds` (default `3600` seconds), with the reaper sweeping every `context_reap_interval_ms` (default `60000` ms). This bounds the lifetime of abandoned contexts without operator intervention. See Growing active context count for how to inspect and clear contexts manually if needed.

NRF Registration



At startup the AUSF registers an `AUSF` NF profile with the NRF advertising the `nausf-auth` service (`v1`, `1.0.0`), its SBI address/port/scheme, and the configured PLMN. It sends heartbeats every `heartbeat_interval` milliseconds. Registration can be forced again at runtime via `POST /api/oam/nrf/reregister`. The AUSF also uses the NRF to discover the UDM: before each Nudm call it queries `Nnrf_NFDiscovery` (TS 29.510) for a `udm` instance and targets the discovered UDM. Only when discovery yields no instance (or the NRF is unreachable) does it fall back to the statically configured `udm_uri`, so single-UDM and static deployments keep working.

← Overview

OmniAUSF Troubleshooting

Authentication fails with 404 USER_NOT_FOUND

The UDM returned 404 for the subscriber during `generate-auth-data`. Confirm:

1. `udm_uri` is correct and reachable from the OmniAUSF host.
2. The subscriber (IMSI) is provisioned in the UDM/UDR/HSS backend.
3. The SUCI presented by the AMF is well-formed and de-concealable by the UDM.

The AUSF logs `AUSF auth failed for {supiOrSuci}: user not found at UDM`.

Confirmation returns AUTHENTICATION_FAILURE (HTTP 200)

The decoded `resStar` did not equal the stored `XRES*`. This is signalled in the body (`authResult: "AUTHENTICATION_FAILURE"`), not as an HTTP error. Common causes:

- The UE's credentials (Ki/OPc) do not match those provisioned in the backend.
- The RAND/AUTN or RES* were altered in transit, or the AMF/AUSF disagree on the serving network name (which would produce a mismatched vector).
- A malformed/non-hex `resStar` is treated as a mismatch.

The AUSF logs `AUSF 5G-AKA authentication FAILED for {supi}: RES* mismatch`.

Confirmation returns 404 CONTEXT_NOT_FOUND

The `authCtxId` in the PUT request matches no active context. Contexts are deleted after any confirmation (success or failure) and are lost on AUSF restart. The AMF must re-initiate authentication from `POST .../ue-authentications`.

Initiation returns 500 SYSTEM_FAILURE

Either the UDM is unreachable at `udm_uri`, or the UDM returned a malformed authentication vector (missing/non-hex `rand`, `xresStar`, or `kausf`, or a missing `autn`). Verify UDM connectivity and that the UDM's vector generation is healthy. The AUSF logs `AUSF auth failed ...` with the reason, or `AUSF auth failed: malformed authentication vector from UDM`.

AUSF not discoverable / NRF status shows not registered

Check `GET /api/status/nrf` and the `omni_ausf_nrf_registration_status` metric. Verify `nrf_uri` is correct (note the runtime default caveat in the configuration reference) and reachable, then force re-registration with `POST /api/oam/nrf/reregister`.

Growing active context count

In-flight authentication contexts should return to near zero at rest. Contexts are removed on confirmation, delete, or deregister. If the AMF abandons authentications without confirming, those contexts are automatically garbage-

collected by the periodic context reaper once they exceed `context_ttl_seconds` (default `3600` seconds); the reaper sweeps every `context_reap_interval_ms` (default `60000` ms). Stale contexts therefore clear on their own after the TTL without operator intervention.

If you need to inspect or clear contexts sooner, use the OAM API: `GET /api/status/nf` or `GET /api/statistics` for the live counts, `GET /api/ausf_sessions` to list them, and `DELETE /api/ausf_sessions/{id}` (or `POST /api/ausf_sessions` to clear all) to remove them immediately. The `omni_ausf_active_contexts_count` gauge tracks this count directly (it is emitted on context create, delete, and each reaper sweep), so you can watch it in Prometheus as well as via the OAM API. If the count climbs steadily faster than the TTL can drain it, check whether the AMF is repeatedly initiating authentications it never confirms.