← Vue d'ensemble

Configuration d'OmniAUSF

La configuration est lue à partir de la clé d'environnement de l'application `:omniausf`, peuplée à partir de `config/runtime.exs`. La plupart des clés d'exécution sont soutenues par une variable d'environnement du système d'exploitation lue au démarrage, comme indiqué ci-dessous ; les deux clés de reaper de contexte sont lues directement à partir de l'environnement de l'application `:omniausf` et ne sont pas soutenues par des variables d'environnement.

```
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
```

Référence des paramètres

Paramètre	Type	Requis	Par défaut
`udm_uri`	chaîne	Non	`"http://127.0.0.12:7777"`
`sbi_scheme`	chaîne	Non	`"http"`
`sbi_addr`	chaîne	Non	`"127.0.0.19"`
`sbi_port`	entier	Non	`7777`

Paramètre	Type	Requis	Par défaut
`nrf_uri`	chaîne	Non	`"http://127.0.0.1:7777"`
`mcc`	chaîne	Non	`"999"`
`mnc`	chaîne	Non	`"70"`
`prometheus_metrics_port`	entier	Non	`9568`
`heartbeat_interval`	entier (ms)	Non	`10000`
`context_ttl_seconds`	entier (s)	Non	`3600`

Paramètre	Type	Requis	Par défaut
`context_reap_interval_ms`	entier (ms)	Non	`60000`

> **Remarque sur la valeur par défaut de `nrf_uri` :** `config/runtime.exs` définit `NRF_URI` sur `http://127.0.0.1:7777`. Dans un déploiement de boucle de retour standard, le NRF est accessible à `http://127.0.0.10:7777` ; définissez `NRF_URI` explicitement pour correspondre à votre adresse NRF.

Journalisation

Les journaux sont émis au format JSON (`OmniLogger.JsonFormatter`) avec toutes les métadonnées incluses. Les lignes de journal sur le chemin d'authentification portent le contexte de session (`supi`, `auth_ctx_id`, `procedure`) afin qu'un seul échange d'authentification puisse être tracé de bout en bout par `auth_ctx_id` ou SUPI/SUCI. Le niveau de journal peut être changé à l'exécution via l'API de gestion (voir ci-dessous).

API de gestion (Port 8443)

En plus de l'écouteur SBI et du port des métriques, OmniAUSF expose une API de gestion/OAM HTTPS sur le port `8443` (TLS, nom du produit `OmniAUSF`). Elle est utilisée pour l'inspection opérationnelle et le contrôle en temps réel, et ne fait pas partie du SBI 3GPP. Points de terminaison clés :

Méthode	Chemin	Objectif
GET	`/api/status/api`	Statut du service API
GET	`/api/status/nrf`	Statut de l'enregistrement NRF
GET	`/api/status/nf`	Résumé NF (type `AUSF`, service `nausf-auth`, nombre de contextes UE actifs)
GET	`/api/ausf_sessions`	Liste des sessions d'authentification actives (id de contexte, SUCI/SUPI, nom du réseau de service, type/résultat d'authentification)
DELETE	`/api/ausf_sessions/{id}`	Effacer une session d'authentification spécifique par id de contexte
POST	`/api/ausf_sessions`	Effacer toutes les sessions d'authentification actives
GET	`/api/statistics`	Compteurs de sessions d'authentification (total / succès / échec / en attente)
GET	`/api/oam/config`	Voir la configuration d'exécution effective (`sbi_addr`, `sbi_port`, `nrf_uri`, `udm_uri`, capacité/priorité NF, niveau de journal)
PATCH	`/api/oam/config/{id}`	Mettre à jour la configuration d'exécution : `udm_uri`, `nrf_uri`, `nf_capacity`, `nf_priority`, `log_level`

Méthode	Chemin	Objectif
POST	`/api/oam/log_level`	Changer le niveau de journal à l'exécution (`emergency`...`debug`)
GET	`/api/oam/resources`	Utilisation des ressources de la VM BEAM (mémoire, comptes de processus/ports, version OTP/BEAM)
GET	`/api/oam/diagnostics`	Diagnostics de la VM BEAM (file d'attente d'exécution, I/O, GC, nœuds connectés)
POST	`/api/oam/nrf/reregister`	Forcer la réinscription avec le NRF

Les modifications d'exécution effectuées via `PATCH /api/oam/config` et `POST /api/oam/log_level` s'appliquent uniquement au nœud en cours d'exécution et ne sont pas persistées ; elles reviennent aux valeurs configurées au redémarrage.

Paramètres uniquement OAM

Deux valeurs sont exposées via `GET`/`PATCH /api/oam/config` mais ne sont pas des paramètres de démarrage dans le tableau ci-dessus - elles se trouvent dans le profil NRF plutôt que dans la configuration de base `:omniausf` :

Champ	Type	Signification
`nf_capacity`	entier (0-65535)	Capacité relative annoncée dans le profil NRF de l'AUSF, utilisée par les consommateurs comme poids de répartition de charge lorsque plusieurs instances d'AUSF sont enregistrées. Un `PATCH` réannonce la nouvelle valeur au NRF
`nf_priority`	entier (0-65535)	Priorité de sélection relative annoncée dans le profil NRF ; plus bas est préféré. Un `PATCH` réannonce la nouvelle valeur au NRF

`PATCH /api/oam/config` valide ces valeurs avant de les appliquer : `nf_capacity`/`nf_priority` doivent être des entiers (une chaîne décimale est acceptée et analysée), et `log_level` doit être un niveau connu. Une valeur invalide rejette toute la demande avec `400` sans appliquer quoi que ce soit.

Inspection des sessions d'authentification

`GET /api/ausf_sessions` renvoie une entrée par contexte d'authentification UE en mémoire, avec ces champs :

Champ	Signification
`ctx_id`	L'`authCtxId` (UUID) ; à passer à `DELETE /api/ausf_sessions/{ctx_id}` pour effacer une session
`suci`	Le `supiOrSuci` que l'AMF a présenté lors de l'initiation
`supi`	Le SUPI en clair résolu par le UDM (nul jusqu'à l'arrivée de la réponse du vecteur)
`serving_network_name`	Le nom SN auquel les clés de la session sont liées
`auth_type`	`5G_AKA` une fois le vecteur stocké
`auth_result`	Résultat final, ou nul tant que la session est encore en cours (en attente)
`has_kausf`	Indique si `K_AUSF` est détenu pour cette session - indique que le vecteur a été récupéré et que les clés ont été dérivées. Le matériel clé lui-même n'est jamais exposé

Comme les contextes sont supprimés dès qu'une confirmation est établie, les entrées de cette liste sont normalement de courte durée ; une session qui persiste est celle que l'AMF a initiée mais n'a jamais confirmée ou désinscrite.

← Vue d'ensemble

Métriques OmniAUSF

Les métriques sont exposées à `http://{host}:{prometheus_metrics_port}/metrics` (port par défaut `9568`).

Métriques AUSF

Métrique	Type	Étiquettes	Descri
`omni_ausf_auth_count`	compteur	`result`	Opération d'authent UE. `resul` l'un de `initiate` (vecteur d défi retou `success` (correspon XRES*), `f` (erreur UD vecteur malformé non-correspon RES*), `re` (une resynchro AUTS/SQN détectée relayée au ou `resync_i` (un blob d resynchro malformé supprimé)

Métrique	Type	Étiquettes	Descri
`omni_ausf_nrf_registration_status`	jauge	`nf_type`	État d'enregist NRF (1 = enregistré non enreg
`omni_ausf_active_contexts_count`	jauge	-	Nombre d contextes d'authent UE actifs. en direct par l'évén de téléme `[:omniau :ue_conte :active]` chaque cr et suppre contexte chaque ba de ContextRe donc elle nombre a contextes cours

> **`omni_ausf_active_contexts_count` est une jauge en direct.** Elle est pilotée par l'événement de télémétrie `[:omniausf, :ue_contexts, :active]`, que le magasin de contexte AUSF émet chaque fois que l'ensemble des contextes actifs change : lors de la création de contexte, lors de la suppression (confirmation, désenregistrement ou suppression de résultat d'authentification), et à chaque balayage de ContextReaper qui récolte les contextes inactifs. La jauge reflète donc le nombre actuel de contextes d'authentification actifs et est sûre pour les alertes et les

> tableaux de bord. L'API OAM reste disponible pour des vérifications croisées : `GET /api/status/nf` (inclut un compte de contexte UE actif) ou `GET /api/statistics` (compteurs de sessions totales / réussies / échouées / en attente).

> `omni_ausf_auth_count` couvre le flux d'authentification principal ainsi que le chemin de resynchronisation. Les étiquettes `resync` et `resync_invalid` comptent les tentatives de resynchronisation SQN détectées lors de la demande d'initiation (validées-et-relayées, et malformées-et-supprimées, respectivement). Les chemins de désenregistrement et de suppression de résultat d'authentification émettent leurs propres événements de t��lémétrie mais ne sont actuellement pas attachés à une métrique Prometheus exportée.

> **Télémétrie interne (pas une métrique Prometheus).** Le collecteur de contexte émet un événement de télémétrie `[:omniausf, :ausf, :context_reap]` portant le compte des contextes récoltés par balayage. C'est une télémétrie interne pour les opérateurs qui câblent leurs propres gestionnaires ; elle n'est pas enregistrée comme une métrique Prometheus exportée.

Exemple PromQL

- Taux de réussite de l'authentification sur 5 minutes : `sum(rate(omni_ausf_auth_count{result="success"}[5m])) / sum(rate(omni_ausf_auth_count{result="initiated"}[5m]))`
- Échecs d'authentification par seconde : `sum(rate(omni_ausf_auth_count{result="failure"}[5m]))`
- AUSF actuellement enregistré avec le NRF : `omni_ausf_nrf_registration_status == 1`
- Tentatives de resynchronisation SQN par seconde (valide vs malformé) : `sum(rate(omni_ausf_auth_count{result="resync"}[5m]))` et `sum(rate(omni_ausf_auth_count{result="resync_invalid"}[5m]))`
- Contextes d'authentification actifs en cours : `omni_ausf_active_contexts_count`

Pour une vérification croisée indépendante du nombre de contextes en cours, l'API OAM (`GET /api/status/nf` ou `GET /api/statistics`) rapporte le même chiffre.

Métriques BEAM VM

Métrique	Type	Description
`beam_memory_total`	jauge	Mémoire totale BEAM en octets
`beam_memory_processes`	jauge	Mémoire utilisée par les processus Erlang
`beam_memory_system`	jauge	Mémoire système (ETS, atomes, code)
`beam_processes_count`	jauge	Nombre de processus Erlang
`beam_vm_uptime`	jauge	Temps de fonctionnement de la VM en secondes

← Vue d'ensemble

Opérations OmniAUSF

OmniAUSF implémente la fonction de serveur d'authentification (AUSF) du cœur 5G, exécutant l'authentification primaire 5G-AKA entre l'AMF (N12) et le UDM (N13). Il demande un vecteur d'authentification au UDM, relaye le défi RAND/AUTN (et HXRES*) à l'AMF, vérifie la réponse de l'UE, dérive la clé d'ancrage K_SEAF et rapporte le résultat de l'authentification au UDM.

Rôle 3GPP et références de spécification

Spécification	Pertinence
TS 23.501	Architecture système - rôle de l'AUSF dans le système 5G
TS 33.501	Architecture de sécurité - authentification 5G-AKA (6.1.3), hiérarchie de clés K_AUSF/K_SEAF, HXRES*/HRES* (Annexe A.5), dérivation de K_SEAF (Annexe A.6)
TS 29.509	API de service Nausf_UEAuthentication (HTTP SBI)
TS 29.503	API de service Nudm_UEAuthentication - génération de vecteur d'authentification et rapport de résultat d'authentification
TS 29.510	APIs de service Nnrf_NFManagement et Nnrf_NFDiscovery - enregistrement NF/battement de cœur et découverte UDM
TS 24.501	NAS - format du nom du réseau de service (`5G:mnc<MNC>.mcc<MCC>.3gppnetwork.org`)

Interfaces

Interface	Pair	Direction	Transport	But
N12 (Nausf_UEAuthentication)	AMF	Entrant	HTTP SBI	Initiation de l'authentificati et confirmatio AKA
N13 (Nudm_UEAuthentication)	UDM	Sortant	HTTP SBI	Génération de vecteur d'authentificat rapport de rés d'authentificat
Nnrf	NRF	Sortant	HTTP SBI	Enregistremen battement de et découverte pair-NF (UDM) (`Nnrf_NFDisc` TS 29.510)

Points de terminaison SBI

Tous les points de terminaison SBI sont servis sous l'URL de base `{sbi_scheme}://{sbi_addr}:{sbi_port}` et utilisent `Content-Type: application/json` lors de la demande. La réponse de création réussie utilise `application/3gppHal+json` selon la TS 29.509.

Méthode	Chemin	Service
POST	`/nausf-auth/v1/ue-authentications`	Nausf_UEAuthentication
PUT	`/nausf-auth/v1/ue-authentications/{authCtxId}/5g-aka-confirmation`	Nausf_UEAuthentication
DELETE	`/nausf-auth/v1/ue-authentications/{authCtxId}/5g-aka-confirmation`	Nausf_UEAuthentication
POST	`/nausf-auth/v1/ue-authentications/deregister`	Nausf_UEAuthentication

Résumé de la demande / réponse

Initier l'authentification UE - le corps de la demande contient `supiOrSuci` (SUPI ou SUCI ; le champ `suci` est également accepté) et `servingNetworkName` ; un objet optionnel `resynchronizationInfo` est transmis au UDM pour la resynchronisation SQN. L'AUSF demande un vecteur 5G-AKA au UDM, puis retourne `201 Created` avec un corps `UEAuthenticationCtx` :

- `authType: "5G_AKA"`
- `5gAuthData`: `{ rand, autn, hxresStar }` (chaînes hexadécimales)
- `_links.5g-aka.href`: l'URL de confirmation absolue pour ce contexte
- `servingNetworkName`

Un en-tête `Location` pointe vers la ressource de contexte créée (`.../ue-authentications/{authCtxId}`).

Confirmation 5G-AKA - le corps de la demande contient `resStar` (hex). L'AUSF compare le `resStar` décodé avec le `XRES*` stocké (TS 33.501 6.1.3.2 ; l'AMF est responsable de la vérification du hachage HRES*/HXRES*). Il enregistre le résultat dans le UDM (`auth-events`) et retourne toujours `200 OK` avec un `ConfirmationDataResponse` :

- En cas de succès : `{ authResult: "AUTHENTICATION_SUCCESS", supi, kseaf }` (K_SEAF sous forme de chaîne hexadécimale).
- En cas d'échec : `{ authResult: "AUTHENTICATION_FAILURE", supi }`. Notez qu'un échec d'authentification est signalé dans le corps via `authResult`, **pas** par un statut d'erreur HTTP.

Le contexte d'authentification est supprimé après confirmation, quel que soit le résultat.

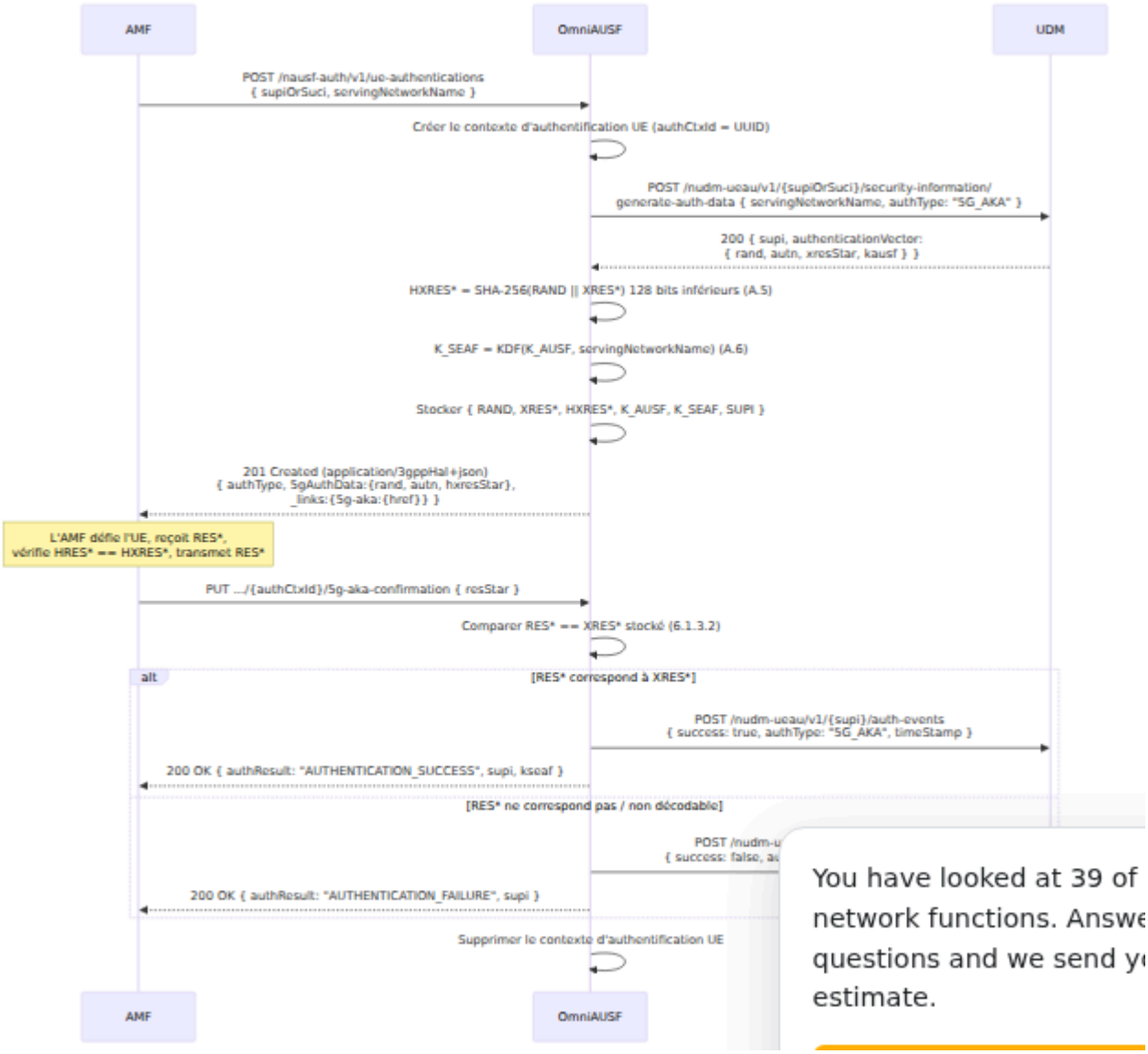
Réponses d'erreur

Les erreurs sont retournées sous forme de corps `ProblemDetails`.

Statut HTTP	Cause	Condition
400	`MANDATORY_IE_MISSING`	Un champ requis est manquant (`supiOrSuci`/`servingNetworkName` lors de l'initiation ; `resStar` lors de la confirmation ; `supi` lors de la désinscription)
404	`USER_NOT_FOUND`	Le UDM a retourné 404 pour l'abonné lors de la génération du vecteur
404	`CONTEXT_NOT_FOUND`	L'`authCtxId` dans une demande de confirmation ne correspond à aucun contexte actif
500	`SYSTEM_FAILURE`	UDM inaccessible, ou le UDM a retourné un vecteur d'authentification mal formé

Procédures clés

Authentification 5G-AKA (Section 6.1.3 de la TS 33.501)



Initiation. Lors de `POST .../ue-authentications`, l'AUSF crée un contexte, puis demande un vecteur 5G-AKA au UDM à `/nudm-ueau/v1/{supiOrSuci}/security-information/generate-auth-data`, incluant l'identifiant d'instance NF de l'AUSF et toute `resynchronizationInfo` fournie par l'AMF. Le UDM retourne un vecteur d'authentification (`rand`, `autn`, `xresStar`, `kausf`) et le `supi` résolu (déconcelé du SUCI). L'AUSF calcule `HXRES*` à partir de RAND et XRES*, dérive `K_SEAF` à partir de `K_AUSF` lié au nom du réseau de service, stocke le matériel et retourne le défi à l'AMF. Les champs du vecteur sont décodés en hex de manière défensive - un vecteur

manquant ou mal formé produit un `500 SYSTEM_FAILURE` propre plutôt qu'un crash.

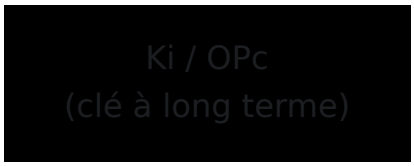
Confirmation. Lors de `PUT .../{authCtxId}/5g-aka-confirmation`, l'AUSF recherche le contexte, décode `resStar` et le compare directement au `XRES*` stocké. Il rapporte le résultat au UDM via `auth-events` (meilleure effort ; un rapport échoué est enregistré mais ne change pas la réponse à l'AMF) et retourne le résultat dans le corps. Le contexte est ensuite supprimé. Un `authCtxId` inconnu retourne `404 CONTEXT_NOT_FOUND`.

Nom du réseau de service. Construit à partir de `mcc`/`mnc` comme `5G:mnc<MNC>.mcc<MCC>.3gppnetwork.org` selon la TS 24.501, avec des MNC à deux chiffres complétés par des zéros pour former trois chiffres. Le nom du réseau de service lie `K_SEAF` au réseau de service, donc l'AMF et l'AUSF doivent s'accorder sur l'identité PLMN.

Resynchronisation SQN. Lorsque l'AMF inclut `resynchronizationInfo` (le `RAND` et `AUTS` produits par l'UE) dans la demande d'initiation, l'AUSF détecte la condition de resynchronisation, valide que `rand` et `auts` sont tous deux des hex bien formés, et relaye la paire validée au UDM/ARPF dans la demande `generate-auth-data` afin que le UDM puisse redériver le numéro de séquence (TS 33.501 6.1.3.3). Les entrées de resynchronisation mal formées sont supprimées et ne sont pas transmises. L'AUSF ne calcule pas lui-même le numéro de séquence, mais ce n'est pas un simple passage : il inspecte et valide le blob de resynchronisation avant de le relayer.

Hiérarchie des clés et dérivation (Annexe A de la TS 33.501)

La hiérarchie des clés 5G-AKA est divisée entre le UDM et l'AUSF. L'AUSF ne voit jamais la clé à long terme (Ki) ou CK/IK - il reçoit `K_AUSF` déjà dérivé par le UDM à l'intérieur du vecteur d'authentification, et ne dérive que la clé d'ancrage `K_SEAF` à partir de celle-ci. L'AMF (et non l'AUSF) continue la chaîne jusqu'à `K_AMF` et les clés de stratum d'accès.



UDM / ARPF



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



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

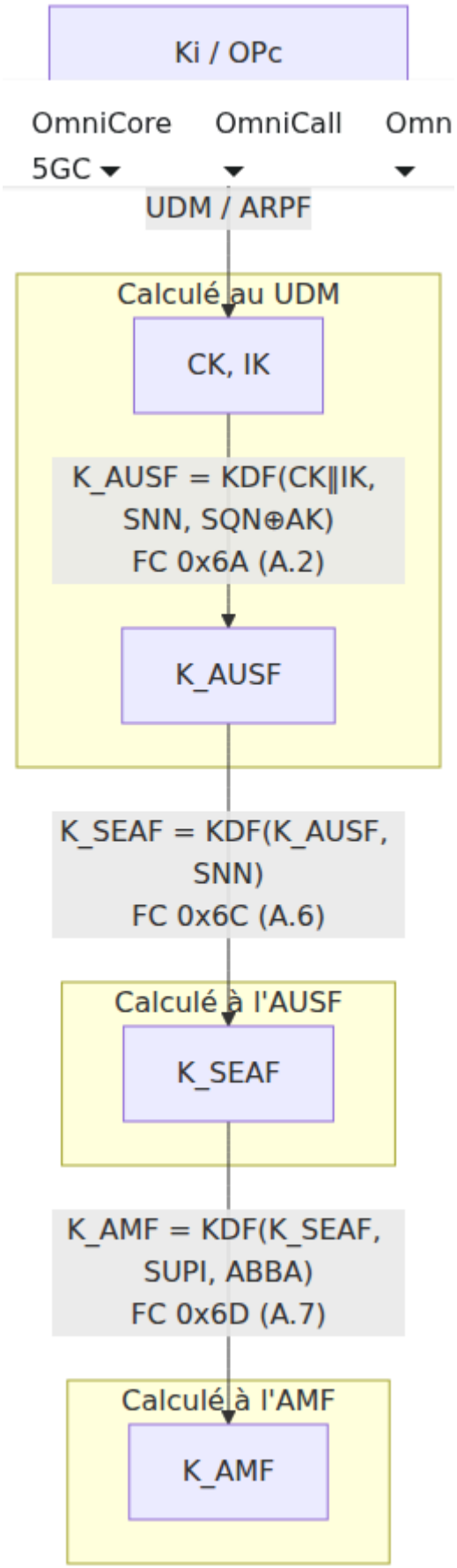


Clé / valeur	Où dérivée	Dérivation	But dans l'AUSF
`K_AUSF`	UDM (ARPF)	`KDF(CK‖IK, servingNetworkName, SQN⊕AK)`, FC `0x6A` (A.2)	Reçue dans le vecteur ; la clé d'ancrage AUSF à partir de laquelle `K_SEAF` est dérivée. Conservée dans le contexte, jamais retournée à l'AMF
`K_SEAF`	AUSF	`KDF(K_AUSF, servingNetworkName)`, FC `0x6C` (A.6)	La clé d'ancrage du réseau de service. Retournée à l'AMF uniquement en cas de `AUTHENTICATION_SUCCESS` ; l'AMF dérive `K_AMF` à partir de celle-ci
`HXRES*`	AUSF	128 bits inférieurs de `SHA-256(RAND ‖ XRES*)` (A.5)	Retournée à l'AMF lors de l'initiation afin que l'AMF puisse faire correspondre le `HRES*` de l'UE avant de transmettre `RES*`
`XRES*`	UDM (ARPF)	`KDF(CK‖IK, SNN, RAND, XRES)`, FC `0x6B` (A.4)	Reçue dans le vecteur ; la réponse attendue à laquelle l'AUSF compare `RES*` lors de la confirmation

Parce que `K_SEAF` est lié au `servingNetworkName` (et `K_AUSF` en amont est lié au même nom de SN et à `SQN⊕AK`), l'AMF, l'AUSF et le UDM doivent tous s'accorder sur l'identité PLMN pour que les clés dérivées correspondent à celles de l'UE. Un décalage dans le nom du réseau de service se manifeste comme un `AUTHENTICATION_FAILURE` lors de la confirmation, et non comme une erreur explicite.

Supprimer le résultat d'authentification 5G-AKA (TS 29.509)

L'AMF peut explicitement supprimer un résultat d'authentification / contexte sans échange de confirmation. Cela correspond à `Delete5gAkaAuthenticationResult` dans la TS 29.509.

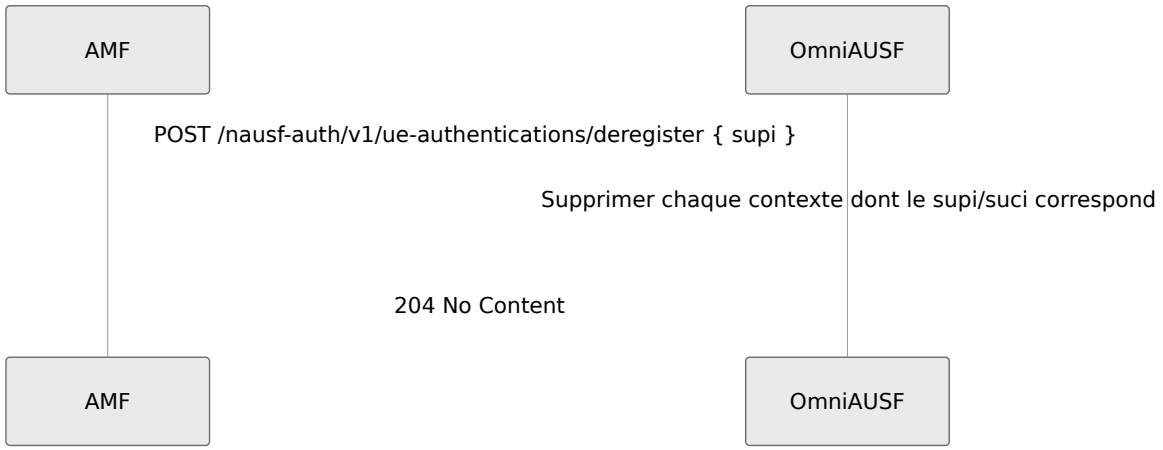


L'AUSF supprime le contexte inconditionnellement (un `authCtxId` inconnu est une opération sans effet) et retourne toujours `204 No Content` ; aucun appel

au UDM n'est effectué.

Désinscription (TS 29.509)

Supprime tout contexte de sécurité UE que l'AUSF détient pour un SUPI donné.



`supi` est obligatoire (une valeur manquante/vide retourne `400 MANDATORY_IE_MISSING`). L'AUSF parcourt ses contextes en mémoire et supprime ceux dont le `supi` ou `suci` stocké est égal au SUPI demandé ; l'opération est uniquement locale et n'appelle pas le UDM. Il retourne toujours `204 No Content`, même lorsqu'aucun contexte correspondant n'existe.

Contexte d'authentification UE

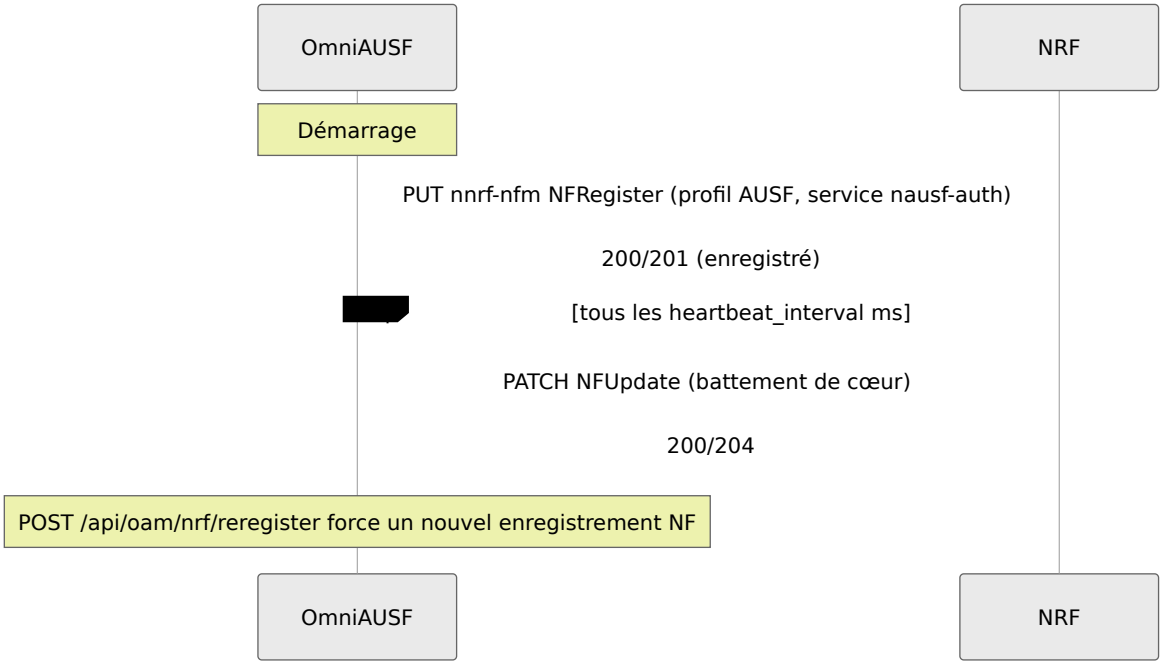
Chaque initiation crée un contexte d'authentification UE en mémoire, identifié par un `authCtxId` généré (UUID). Le contexte contient le matériel nécessaire pour compléter l'échange et est l'unité que les points de terminaison de session OAM (`GET /api/ausf_sessions`, `DELETE /api/ausf_sessions/{id}`) listent et nettoient.

Champ	Défini à	Signification
`ctx_id`	initiation	L'UUID `authCtxId` ; apparaît dans l'URL de confirmation et la liste des sessions OAM
`suci`	initiation	Le `supiOrSuci` présenté à l'origine par l'AMF (SUCI ou SUPI)
`supi`	réponse du vecteur	Le SUPI en clair résolu (déconcelé) par le UDM
`serving_network_name`	initiation	Le nom du réseau SN auquel les clés sont liées
`rand`, `xres_star`, `hxres_star`, `kausf`, `kseaf`	réponse du vecteur	Le vecteur et les clés dérivées conservées pendant la durée de l'échange
`auth_type`	réponse du vecteur	`5G_AKA`
`auth_result`	-	Rempli pour une session réglée ; un contexte encore en vol n'a pas de résultat et est compté comme `pending` dans `GET /api/statistics`

Un contexte existe normalement uniquement entre l'initiation et le règlement. Il est supprimé lors de la confirmation (succès **ou** échec), lors de la suppression explicite `DELETE .../5g-aka-confirmation`, lors de la désinscription pour son SUPI, et lors du redémarrage du processus. De plus, les contextes inactifs ou abandonnés sont collectés par un nettoyeur périodique supervisé : un contexte que l'AMF initie mais ne confirme jamais est récolté une fois qu'il dépasse `context_ttl_seconds` (par défaut `3600` secondes), le nettoyeur balayant tous

les `context_reap_interval_ms` (par défaut `60000` ms). Cela limite la durée de vie des contextes abandonnés sans intervention de l'opérateur. Voir Augmentation du nombre de contextes actifs pour savoir comment inspecter et nettoyer les contextes manuellement si nécessaire.

Enregistrement NRF



Au démarrage, l'AUSF enregistre un profil NF `AUSF` auprès du NRF en annonçant le service `nausf-auth` (`v1`, `1.0.0`), son adresse/port/schéma SBI, et le PLMN configuré. Il envoie des battements de cœur toutes les `heartbeat_interval` millisecondes. L'enregistrement peut être forcé à nouveau à l'exécution via `POST /api/oam/nrf/reregister`. L'AUSF utilise également le NRF pour découvrir le UDM : avant chaque appel Nudm, il interroge `Nnrf_NFDiscovery` (TS 29.510) pour une instance `udm` et cible le UDM découvert. Ce n'est que lorsque la découverte ne donne aucune instance (ou que le NRF est inaccessible) qu'il revient à l'`udm_uri` configuré statiquement, de sorte que les déploiements à UDM unique et statiques continuent de fonctionner.

← Vue d'ensemble

Dépannage OmniAUSF

L'authentification échoue avec 404 USER_NOT_FOUND

Le UDM a retourné 404 pour l'abonné lors de `generate-auth-data`. Confirmez :

1. `udm_uri` est correct et accessible depuis l'hôte OmniAUSF.
2. L'abonné (IMSI) est provisionné dans le backend UDM/UDR/HSS.
3. Le SUCI présenté par l'AMF est bien formé et déchiffrable par le UDM.

Les journaux de l'AUSF indiquent `AUSF auth failed for {supiOrSuci}: user not found at UDM`.

La confirmation retourne AUTHENTICATION_FAILURE (HTTP 200)

Le `resStar` décodé n'était pas égal au `XRES*` stocké. Cela est signalé dans le corps (`authResult: "AUTHENTICATION_FAILURE"`), et non comme une erreur HTTP. Causes courantes :

- Les identifiants de l'UE (Ki/OPc) ne correspondent pas à ceux provisionnés dans le backend.
- Le RAND/AUTN ou RES* ont été altérés en transit, ou l'AMF/AUSF ne s'accordent pas sur le nom du réseau de service (ce qui produirait un vecteur non correspondant).
- Un `resStar` mal formé/non hexadécimal est traité comme un non-correspondance.

Les journaux de l'AUSF indiquent `AUSF 5G-AKA authentication FAILED for {supi}: RES* mismatch`.

La confirmation retourne 404 CONTEXT_NOT_FOUND

L'`authCtxId` dans la requête PUT ne correspond à aucun contexte actif. Les contextes sont supprimés après toute confirmation (succès ou échec) et sont perdus lors du redémarrage de l'AUSF. L'AMF doit réinitier l'authentification à partir de `POST .../ue-authentications`.

L'initiation retourne 500 SYSTEM_FAILURE

Soit le UDM est injoignable à `udm_uri`, soit le UDM a retourné un vecteur d'authentification mal formé (manquant/non hexadécimal `rand`, `xresStar`, ou `kausf`, ou un `autn` manquant). Vérifiez la connectivité UDM et que la génération de vecteurs du UDM est saine. Les journaux de l'AUSF indiquent `AUSF auth failed ...` avec la raison, ou `AUSF auth failed: malformed authentication vector from UDM`.

AUSF non découvrable / le statut NRF montre non enregistré

Vérifiez `GET /api/status/nrf` et la métrique `omni_ausf_nrf_registration_status`. Vérifiez que `nrf_uri` est correct (notez la mise en garde par défaut à l'exécution dans la référence de configuration) et accessible, puis forcez la réinscription avec `POST /api/oam/nrf/reregister`.

Compte de contexte actif croissant

Les contextes d'authentification en cours devraient revenir près de zéro au repos. Les contextes sont supprimés lors de la confirmation, de la suppression ou de la désinscription. Si l'AMF abandonne les authentifications sans confirmation, ces contextes sont automatiquement collectés par le nettoyeur de contexte périodique une fois qu'ils dépassent `context_ttl_seconds` (par défaut `3600` secondes) ; le nettoyeur balaie tous les `context_reap_interval_ms` (par défaut `60000` ms). Les contextes obsolètes se nettoient donc d'eux-mêmes après le TTL sans intervention de l'opérateur.

Si vous devez inspecter ou supprimer des contextes plus tôt, utilisez l'API OAM : `GET /api/status/nf` ou `GET /api/statistics` pour les comptes en direct, `GET /api/ausf_sessions` pour les lister, et `DELETE /api/ausf_sessions/{id}` (ou `POST /api/ausf_sessions` pour tout supprimer) pour les retirer immédiatement. Le compteur `omni_ausf_active_contexts_count` suit ce compte directement (il est émis lors de la création, de la suppression de contexte et de chaque balayage du nettoyeur), donc vous pouvez le surveiller dans Prometheus ainsi que via l'API OAM. Si le compte augmente de manière constante plus rapidement que le TTL ne peut l'épuiser, vérifiez si l'AMF initie à plusieurs reprises des authentifications qu'il ne confirme jamais.