Architecture d'OmniTWAG

Vue d'ensemble

OmniTWAG est une passerelle d'accès WLAN de confiance (TWAG) selon la section 16 de la norme 3GPP TS 23.402. Elle relie les points d'accès WLAN de confiance au cœur de paquet évolué (EPC), fournissant authentification, gestion de session et tunneling du plan de données.

Table des matières

- Architecture des composants et des processus
- Arbre de supervision
- Référence des composants
- Modes de fonctionnement
- Intégration AP (Tunnel GRE)
- Comportement DHCP
- Authentification
- Cycle de vie de la session (routage à domicile)
- Interfaces Diameter
- Résumé de la configuration
- Dérivation de clé
- Pages connexes

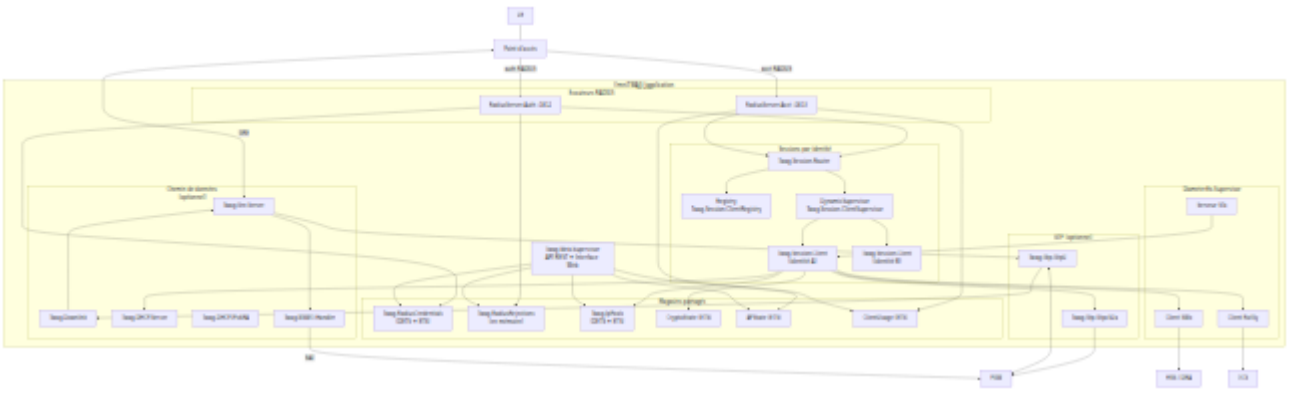
Architecture des composants et des processus

OmniTWAG fonctionne comme une seule application BEAM (Erlang VM). Le superviseur de niveau supérieur démarre un ensemble fixe de services

partagés et, en fonction de la configuration, un ensemble d'enfants optionnels de chemin de données et d'interface. L'état par abonné vit dans un GenServer pour chaque identité, créé à la demande.

Les principaux groupes sont :

- **Web et OAM** : l'API REST et l'interface web (`Twag.Web.Supervisor`).
- **Sessions par identité** : un GenServer `Twag.Session.Client` pour chaque abonné, trouvé via un `Registry` et démarré par un `DynamicSupervisor`. `Twag.Session.Router` effectue la recherche ou la création.
- **Magasins partagés** : `Twag.RadiusCredentials` (secrets partagés durables), `Twag.RadiusRejections` (sources rejetées en mémoire), `Twag.IpPools` (pools d'adresses), et les magasins `CryptoState`, `APState`, et `ClientUsage` soutenus par ETS.
- **Écouteurs RADIUS** : `RadiusServer.Auth` (port 1812) et `RadiusServer.Acct` (port 1813).
- **Diameter** : `DiameterEx.Supervisor` gère les pairs SWx (vecteurs d'authentification), STa (ré-auth et abandon), et Ro/Gy (facturation en ligne).
- **GTP** : `Twag.Gtp.GtpcS2a` (contrôle) et `Twag.Gtp.GtpU` (plan utilisateur), démarré uniquement lorsque `gtp.enabled` est vrai.
- **Chemin de données** : `Twag.Gre.Server`, `Twag.Downlink`, `Twag.DHCP.Server`, `Twag.DHCP.IPv6RA`, et `Twag.NSWO.Handler`, chacun démarré uniquement lorsque son bloc de configuration est activé.



Arbre de supervision

Le superviseur racine utilise une stratégie `:one_for_one` (maximum 10 redémarrages en 60 secondes). Les enfants démarrent dans cet ordre, de sorte que les secrets par IP source se résolvent avant l'arrivée du premier paquet RADIUS :



Les enfants marqués `*` ne sont démarrés que lorsque leur bloc de configuration est activé (voir Résumé de la configuration).

Référence des composants

Composant	Type	Démarré quand
`Twag.Web.Supervisor`	Superviseur	Toujours
`DiameterEx.Supervisor`	Superviseur	Toujours
`Twag.Session.ClientRegistry`	Registry	Toujours
`Twag.Session.ClientSupervisor`	DynamicSupervisor	Toujours
`Twag.Session.Router`	Module	À la demande
`Twag.Session.Client`	GenServer	Par identité
`Twag.RadiusCredentials`	GenServer	Toujours

Composant	Type	Démarré quand
`Twag.RadiusRejections`	GenServer	Toujours
`Twag.IpPools`	GenServer	Toujours
`RadiusServer.Auth`	GenServer	Toujours
`RadiusServer.Acct`	GenServer	Toujours
`CryptoState`	GenServer (ETS)	Toujours

Composant	Type	Démarré quand
`APState`	GenServer (ETS)	Toujours
`ClientUsage`	GenServer (ETS)	Toujours
`Twag.Gtp.GtpcS2a`	GenServer	`gtp.enabled`
`Twag.Gtp.GtpU`	GenServer	`gtp.enabled`
`Twag.Downlink`	GenServer	`downlink.enabled`
`Twag.Gre.Server`	GenServer	`gre.enabled`
`Twag.DHCP.Server`	GenServer	`dhcp.enabled`

Composant	Type	Démarré quand
`Twag.DHCP.IPv6RA`	GenServer	`ipv6_ra.enabled`
`Twag.NSWO.Handler`	GenServer	`nswo.enabled`

Modes de fonctionnement

OmniTWAG prend en charge deux modes de plan de données. Le mode utilisé pour une session est résolu à partir des informations d'identification AP correspondantes (voir Secrets partagés RADIUS), avec la configuration globale `gtp` / `nswo` comme valeur par défaut.

Mode routé à domicile (Tunnel GTP)

Le trafic est tunnélisé via GTP-C/GTP-U (interface S2a) vers le PGW. Le PGW alloue l'adresse IP de l'UE, applique la politique/facturation et fournit l'accès à Internet.

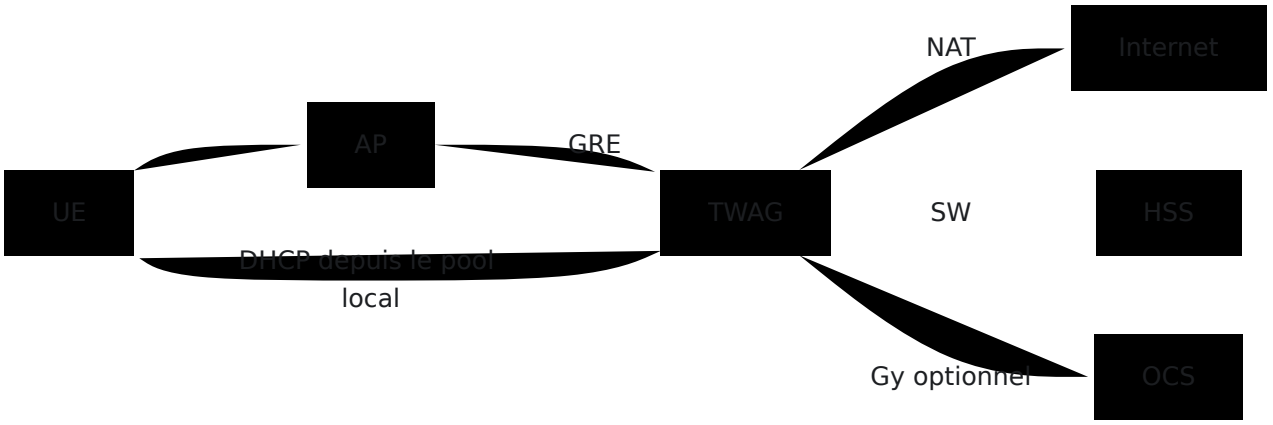


No thanks

- **IP UE** : Allouée par le PGW dans la réponse de création de session (IE PAA).
- **Livraison IP** : Le TWAG sert DHCP à l'UE via GRE, distribuant l'IP allouée par le PGW.
- **Uplink** : L'AP envoie le trafic de l'UE via GRE. Le TWAG encapsule en GTP-U vers le PGW.
- **Downlink** : Le PGW envoie via GTP-U. Le TWAG décapsule et envoie via GRE vers l'AP, puis vers l'UE.
- **Facturation** : Gérée par le PGW (PCEF) via Gy vers l'OCS.
- **Configuration** : `gtp.enabled: true`, `gre.enabled: true`.

Mode de rupture locale (NSWO)

Le trafic sort directement du TWAG vers Internet. Aucun PGW n'est impliqué. Le TWAG effectue NAT/maskerade et effectue éventuellement sa propre facturation en ligne via Gy.



- **IP UE** : Allouée à partir d'un pool IP local du TWAG (`Twag.IpPools`).
- **Livraison IP** : Le TWAG sert DHCP à l'UE via GRE, depuis le pool.
- **Uplink** : L'AP envoie via GRE. Le TWAG applique NAT vers Internet.
- **Downlink** : Le chemin Internet revient par NAT TWAG, puis GRE vers l'AP, puis vers l'UE.
- **Facturation** : Le TWAG agit comme PCEF via Gy vers l'OCS (optionnel, désactivé par défaut). Voir Facturation en ligne.
- **Configuration** : `nswo.enabled: true`, `gtp.enabled: false`.

La rupture locale est sélectionnée en définissant `nswo.enabled: true` (avec `gtp.enabled: false`), ou par AP via l'information d'identification `breakout_type`. Lorsqu'elle est activée, le gestionnaire NSWO est démarré, les sessions reçoivent une adresse d'un pool local au lieu d'une adresse assignée par le PGW, et l'upstream est NAT/maskerade directement vers Internet. Voir Données utilisateur et chemin de données pour le détail du chemin de données.

Int��gration AP (Tunnel GRE)

L'AP tunnelise les trames L2 de l'UE vers le TWAG via GRE (protocole IP 47). Cela fournit le lien point à point requis entre l'UE et le TWAG selon la section

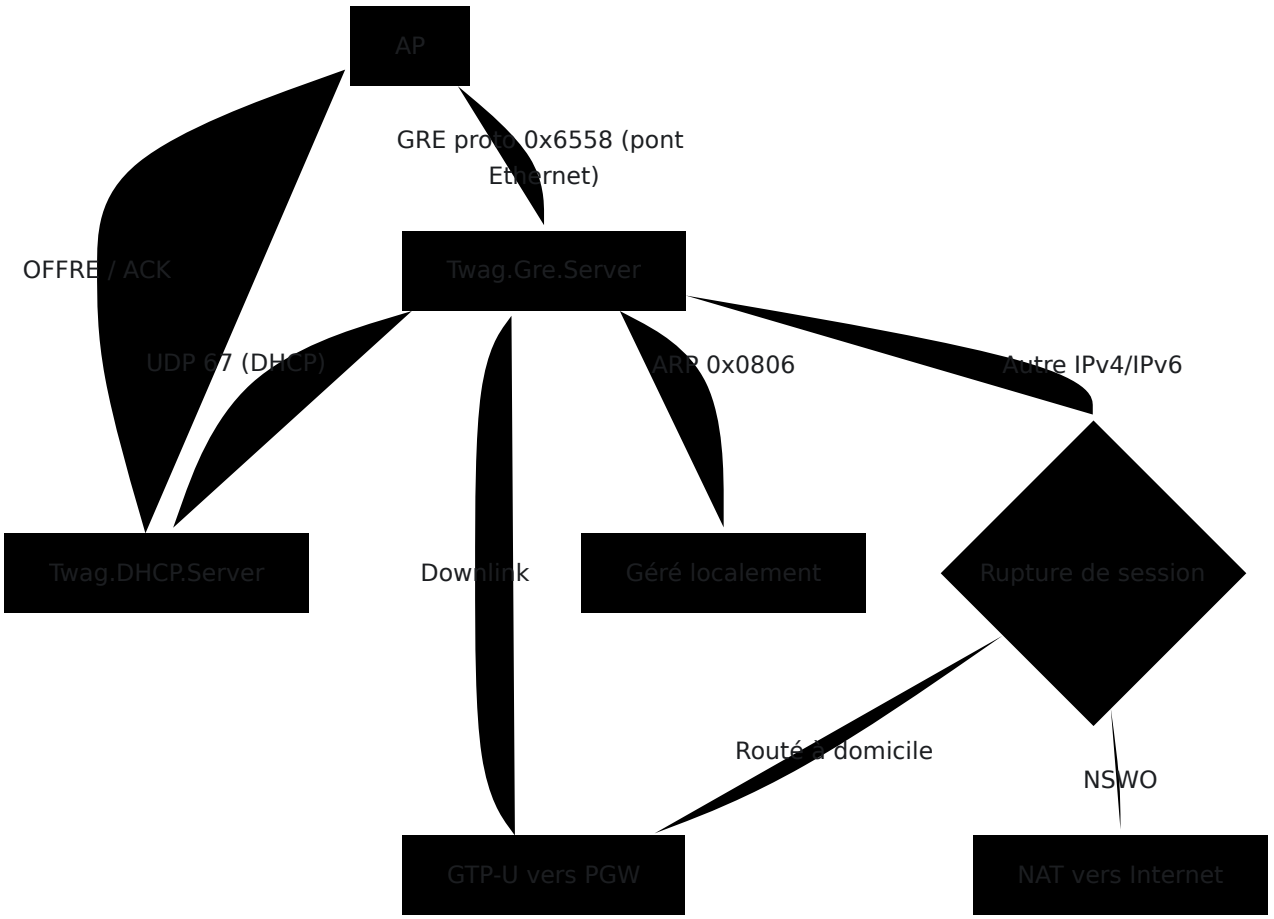
16.1.2 de la norme TS 23.402.

Configuration AP Cisco

```
dot11 ssid ons
   tunnel CWAG
   authentication open eap eap_methods
   authentication key-management wpa
   accounting acct_methods

dot11 tunnel CWAG
    type gre
    source {AP_IP}
    destination {TWAG_IP}
    mss 1410
    mtu 1500
```

Flux de données GRE



Comportement DHCP

Mode	Source IP	Serveur DHCP	Remarques
Routé à domicile	PGW (PAA dans la réponse de création de session)	TWAG pré-alloue l'IP PGW, le sert via DHCP sur GRE	DNS depuis PCO PGW
Rupture locale	Pool local du TWAG (`Twag.IpPools`)	TWAG alloue depuis le pool lié lors de la DISCOVER (distribution de pool)	DNS depuis la configuration du pool

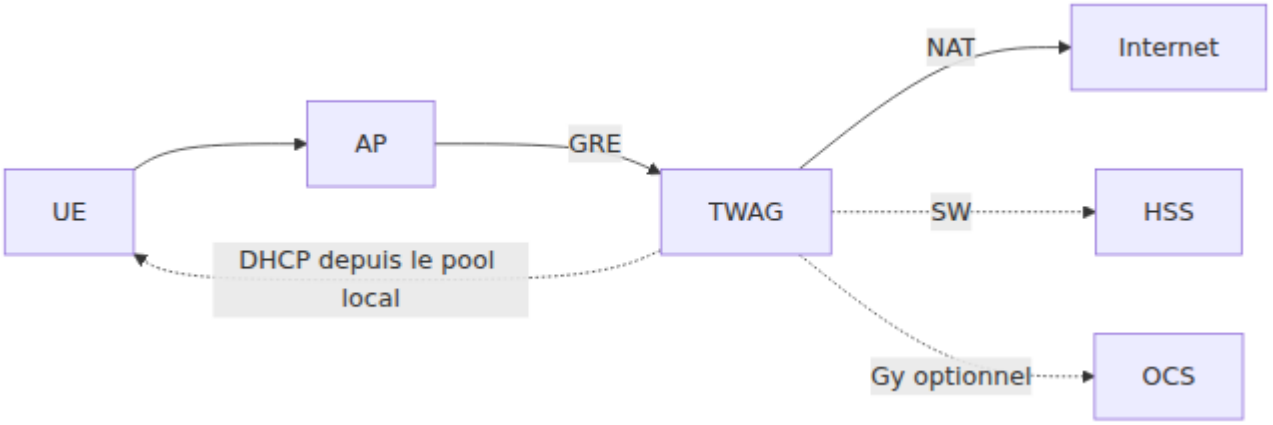
Le serveur DHCP TWAG :

1. Reçoit un DISCOVER DHCP de l'UE via le tunnel GRE.
2. Sert l'OFFRE. En mode routé à domicile, il s'agit de l'IP allouée par le PGW que la session a pré-allouée. En mode de rupture locale, lorsque le MAC est lié à un pool, le serveur distribue une nouvelle adresse du pool.
3. Gère la DEMANDE/ACK pour compléter l'échange.
4. Passe tout le trafic DHCP à travers le tunnel GRE vers et depuis l'AP.

Authentification

OmniTWAG agit comme un TWAG combiné et un serveur AAA 3GPP, effectuant l'authentification EAP-AKA localement et récupérant les vecteurs d'authentification du HSS via Diameter SWx. La machine d'état EAP de chaque abonné s'exécute à l'intérieur de son propre `Twag.Session.Client`.

Flux d'authentification



Avant qu'un paquet soit traité, l'écouteur RADIUS autorise l'IP source contre `Twag.RadiusCredentials`. Une source sans sous-réseau de credential correspondant est rejetée et enregistrée comme `unauthorized_source`. Une source qui correspond mais échoue le contrôle de Message-Authenticator est enregistrée comme `bad_secret`. Voir Rejets RADIUS.

Modes d'authentification

Mode	Configuration	Source de vecteur	Traitement EAP
`:swx`	`auth_mode: :swx`	HSS via SWx MAR/MAA	Local (TWAG)
`:local`	`auth_mode: :local`	Clés de test locales (Milenage)	Local (TWAG)

Méthode EAP

Configurée via `eap_method`. Par défaut `:eap_aka`. Si l'UE refuse la méthode offerte, le TWAG négocie avec la méthode préférée supportée par l'UE. EAP-SIM et la ré-authentification EAP (ERP) sont également supportées. Voir Authentification.

Cycle de vie de la session (routage

à domicile)


Auth EAP -> Acceptation d'accès (MSK à l'AP)
L'AP envoie Accounting-Start
Post-acceptation sur Twag.Session.Client
Demande de création de session GTP-C -> PGW
PGW alloue IP, réponse de création de session
Enregistrer le tunnel GTP-U (TEIDs locaux + distants)
Pré-allouer l'IP PGW dans DHCP pour le MAC de l'UE







Une session est interrompue pour l'une des plusieurs raisons, chacune enregistrée sur la métrique `twag.session.teardown` : `accounting_stop`, `admin_disconnect`, `network_disconnect` (STa ASR), `identity_timeout` (authentification commencée mais jamais complétée), et `liveness_timeout`. Voir Référence des métriques et Sessions.

Interfaces Diameter

Interface	ID App	Direction	But
SWx	16777265	TWAG -> HSS	Récupération de vecteur d'authentification (MAR/MAA)
STa	16777250	Entrant vers TWAG	RAR (ré-auth), ASR (abandon), STR (terminer)
Ro/Gy	4	TWAG -> OCS	Facturation en ligne (NSWO uniquement)

Routage DRA

Le DRA doit router SWx vers le HSS :

```
%{rule_name: "swx_to_hss", match: :all,
  filters: [{:application_id, 16777265}, {:avp, {296, "
{realm}"}}],
  route: %{peers: ["{hss_host}"]}}
```

Résumé de la configuration

Chaque paramètre est documenté dans le tableau qui suit le bloc.

```
config :omnitwag,
  auth_mode: :swx,
  eap_method: :eap_aka,
  identity_timeout_ms: 600_000,

  gtp: %{
    enabled: true,
    pgw_ip: "10.179.1.21",
    local_ip: "10.179.1.41",
    pgw_port: 2123,
    gtpu_port: 2152,
    echo_enabled: true
  },

  gre: %{
    enabled: true,
    local_ip: "10.179.1.41"
  },

  downlink: %{
    enabled: true
  },

  dhcp: %{
    enabled: true,
    interface: "gre"
  },

  ipv6_ra: %{
    enabled: false
  },

  nswo: %{
    enabled: false,
    local_pool: "10.100.0.0/16",
    nat_interface: "eth0"
  },

  online_charging: %{
    mode: :disabled
  },

  rate_limit: %{
```

```
    max_failures: 10,
    window_ms: 60_000
  },

  prometheus: %{
    port: 9568
  }
```

Paramètre	Par défaut	Description
`auth_mode`	`:swx`	`:swx` récupère des vecteurs du HSS ; `:local` utilise des clés de test Milenage locales.
`eap_method`	`:eap_aka`	Méthode EAP offerte (`:eap_aka` ou `:eap_aka_prime`).
`identity_timeout_ms`	`600000`	Temps qu'une session peut rester à la phase d'identité avant d'être interrompue (raison de démontage `identity_timeout`).
`gtp.enabled`	`true`	Démarrer les enfants GTP-C/GTP-U (mode routé à domicile).
`gtp.pgw_ip`	`127.0.0.1`	Adresse PGW pour GTP-C/GTP-U.
`gtp.local_ip`	aucun	Adresse locale TWAG pour GTP.
`gtp.pgw_port`	`2123`	Port GTP-C PGW.
`gtp.gtpu_port`	`2152`	Port GTP-U local.

Paramètre	Par défaut	Description
`gtp.echo_enabled`	`true`	Envoyer des keepalives d'écho GTP au PGW.
`gre.enabled`	`false`	Démarrer le serveur GRE pour les tunnels AP.
`gre.local_ip`	`0.0.0.0`	Adresse locale à laquelle le serveur GRE se lie.
`downlink.enabled`	`true`	Démarrer le transmetteur de downlink.
`dhcp.enabled`	`false`	Démarrer le serveur DHCPv4.
`dhcp.interface`	aucun	`"gre"` pour le mode tunnel, ou un nom de NIC pour la livraison directe.
`ipv6_ra.enabled`	`false`	Démarrer les annonces de routeur IPv6 et DHCPv6.
`nswo.enabled`	`false`	Démarrer le gestionnaire de rupture locale NSWO.

Paramètre	Par défaut	Description
`nswo.local_pool`	aucun	CIDR de secours pour l'adressage de rupture locale.
`nswo.nat_interface`	aucun	NIC de sortie pour NAT/maskerade.
`online_charging.mode`	`:disabled`	Mode Gy : `:disabled`, `:authorize_only`, ou `:online_charging`.
`rate_limit.max_failures`	`10`	Échecs d'authentification par NAS dans la fenêtre avant que les demandes d'accès ne soient rejetées.
`rate_limit.window_ms`	`60000`	Fenêtre de limitation de taux en millisecondes.
`prometheus.port`	`9568`	Port des métriques Prometheus.
`radius_config.acct_interim_interval`	`300`	Intervalle de comptabilité intérimaire (secondes) annoncé à l'AP dans

Paramètre	Par défaut	Description
		Acceptation d'accès.

Dérivation de clé

Pour EAP-AKA', selon 3GPP TS 33.402 clause 6.2 et Annexe A.2, et RFC 5448:

Étape	Nœud (`:swx` mode)	Résultat
1	TWAG	SWx MAR avec `SIP-Authentication-Scheme` = `EAP-AKA'` et `ANID` = nom du réseau
2	HSS	RAND, AUTN avec bit de séparation AMF 1, XRES
3	HSS	`CK'`, `IK'` = KDF(`CK`, `IK`, FC 0x20, nom du réseau, SQN XOR AK)
4	TWAG	`MK` = PRF'(`IK'`, `CK'`, "EAP-AKA'", Identité)
5	TWAG	`K_aut`, `MSK`, `EMSK` à partir de `MK`

En mode `:local`, le TWAG exécute également les étapes 2 et 3, avec les clés de test configurées. Voir Authentification pour la dérivation complète.

Pour EAP-AKA (RFC 4187):

```
HSS fournit : RAND, AUTN, XRES, CK, IK
TWAG dérive : MK = SHA1(Identité || IK || CK)
              K_encr, K_aut, MSK, EMSK à partir de PRF(MK)
```

Le nom du réseau (ANID) est `WLAN` selon la clause 8.1.1.2 de la norme TS 24.302. Voir Nom du réseau.

Pages connexes

- Authentification
- Secrets partagés RADIUS
- Rejets RADIUS
- Pools IP
- Sessions
- Comptabilité par AP
- Données utilisateur et chemin de données
- Facturation en ligne
- Référence des métriques

Vue d'ensemble

Authentification (EAP-AKA / EAP-AKA' / EAP-SIM)

Vue d'ensemble

OmniTWAG authentifie chaque abonné avec une méthode EAP qui fonctionne sur RADIUS entre le point d'accès et le TWAG. Le TWAG agit comme un serveur AAA combiné TWAG et 3GPP. Il exécute la machine d'état EAP et obtient les vecteurs d'authentification du HSS via Diameter SWx, ou à partir de clés de test locales. En cas de succès, le TWAG livre la Clé de Session Maître (MSK) au point d'accès sous forme de clés MPPE à l'intérieur de l'Access-Accept RADIUS. Le point d'accès complète ensuite le handshake 802.11 en 4 étapes avec l'abonné.

Pour le contexte système plus large, voir Architecture. Pour savoir comment le secret partagé RADIUS est résolu par point d'accès, voir Secrets Partagés RADIUS. L'authentification se déroule en premier. La politique de rupture et de facturation par point d'accès est appliquée après que l'authentification ait réussi. Pour la session authentifiée et son cycle de vie de statut, voir Sessions. Pour la session GTP-C S2a que le TWAG crée après l'authentification, voir S2a GTP-C. Pour la facturation qui suit une authentification réussie, voir Facturation en Ligne (Gy).

Table des Matières

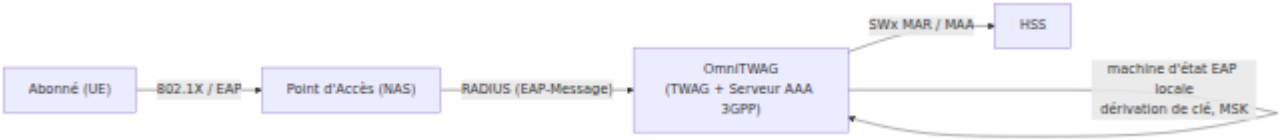
- Vue d'ensemble
- Méthodes EAP Supportées
- Machine d'État EAP et Phases
- Séquence d'Authentification
- Sourcing des Vecteurs d'Auth
- Gestion de l'Identité et NAI

- Dérivation et Livraison des Clés
- Resynchronisation
- Ré-authentification Rapide (ERP)
- Gestion des Échecs et Codes de Résultat
- Configuration
- Observabilité
- Dépannage

Vue d'ensemble

Le point d'accès commence un échange EAP IEEE 802.1X avec l'abonné. Le point d'accès encapsule chaque message EAP dans un attribut RADIUS `EAP-Message` et l'envoie au TWAG. Le TWAG exécute une machine d'état d'authentification par identité d'abonné. La machine d'état dirige l'échange EAP vers un résultat, puis retourne l'un des trois résultats à la couche RADIUS.

Résultat	Réponse RADIUS	Signification
Challenge	`Access-Challenge`	L'échange nécessite un autre tour EAP.
Acceptation	`Access-Accept`	L'authentification a réussi. La réponse contient la MSK sous forme de clés MPPE.
Rejet	`Access-Reject`	L'authentification a échoué. La réponse contient un EAP-Failure.



Le TWAG conserve l'état cryptographique par identité. Il garde la réponse attendue (XRES), la clé d'authentification (`K_aut`), le RAND, la MSK et la MSK étendue (EMSK) entre les tours EAP. Après que le TWAG ait livré les clés MPPE, le TWAG efface le matériel clé sensible.

Méthodes EAP Supportées

Le TWAG prend en charge trois méthodes EAP. Chaque méthode utilise le même format d'attribut TLV, mais une valeur de type EAP différente et une hiérarchie de clés différente.

Méthode	Type EAP	Spécification	Vecteurs	K_aut	MAC	Dé d
EAP-AKA'	50 (`0x32`)	RFC 5448	quintet UMTS	32 octets	HMAC-SHA-256, tronqué à 16 octets	`CK'`/ PRF' SHA-
EAP-AKA	23 (`0x17`)	RFC 4187	quintet UMTS	16 octets	HMAC-SHA1, tronqué à 16 octets	`MK =` `SHA1` `| IK` puis 186-
EAP-SIM	18 (`0x12`)	RFC 4186	2 ou 3 triplets GSM	16 octets	HMAC-SHA1, tronqué à 16 octets	`MK =` `SHA1` `| Kc` `NONC` `vers` PRF I 2

Sélection et Négociation de Méthode

Le TWAG propose une méthode par défaut. Définissez la méthode avec la clé de configuration `eap_method`. La valeur par défaut est `:eap_aka_prime` (EAP-AKA').

L'abonné peut rejeter la méthode proposée avec un EAP-NAK (type EAP 3). Le NAK énumère les types EAP que l'abonné préfère. Le TWAG lit la liste et sélectionne le premier type qu'il prend en charge. Le TWAG prend en charge ces types dans le NAK : `50` (EAP-AKA'), `23` (EAP-AKA), et `18` (EAP-SIM).

- Si le NAK énumère un type pris en charge, le TWAG redémarre l'échange avec cette méthode. Pour EAP-AKA' et EAP-AKA, le TWAG envoie une demande d'identité AKA avec `AT_PERMANENT_ID_REQ`. Pour EAP-SIM, le TWAG envoie une demande EAP-SIM/Start.
- Si le NAK n'énumère aucun type pris en charge, le TWAG rejette l'abonné.

Machine d'État EAP et Phases

Le TWAG exécute une machine d'état par identité d'abonné, à l'intérieur d'un processus client par identité. La machine d'état suit une phase EAP. La phase détermine quel message EAP le TWAG attend ensuite et quelle télémétrie il émet.

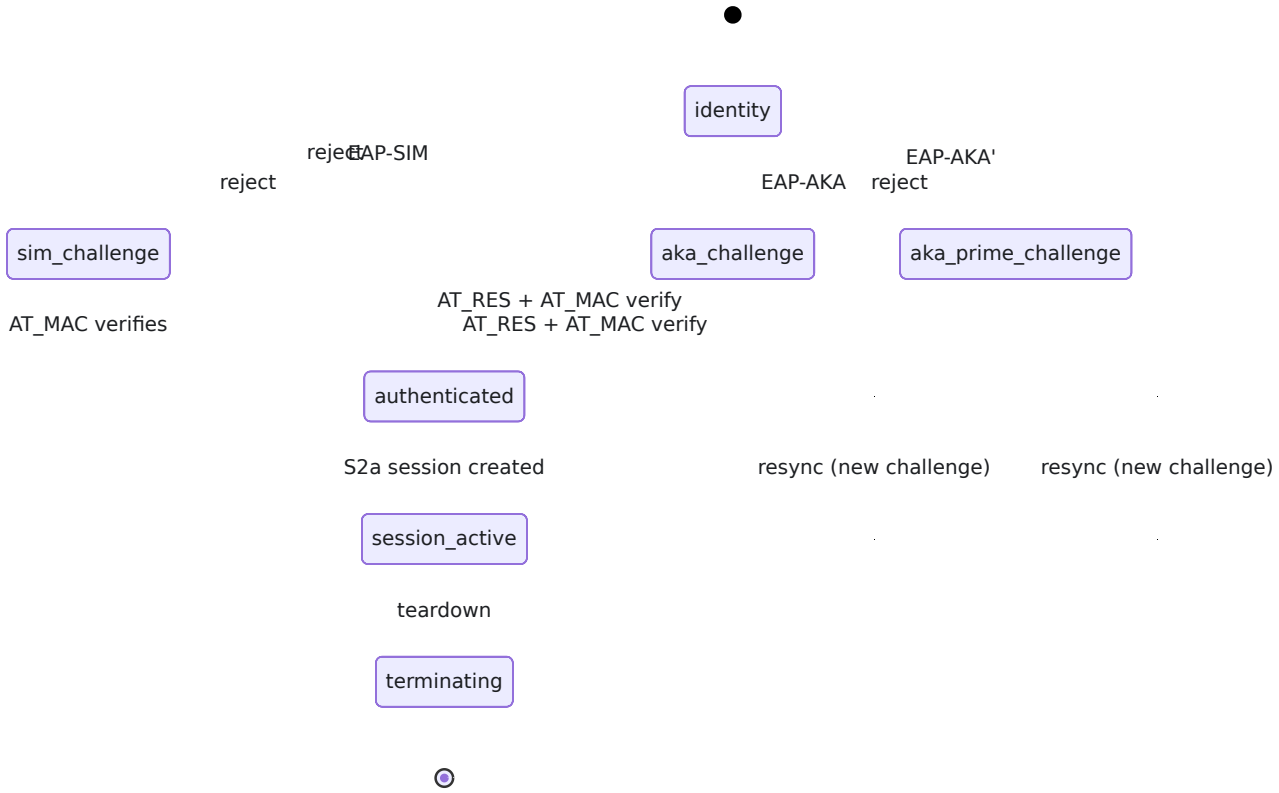
Phase (`eap_phase`)	Signification
`identity`	La phase initiale. Le TWAG attend un EAP-Identity, ou le premier tour AKA-Identity ou EAP-SIM/Start. Un tour échoué renvoie l'état à `identity`.
`aka_challenge`	Le TWAG a envoyé un défi EAP-AKA et attend la réponse AKA-Challenge.
`aka_prime_challenge`	Le TWAG a envoyé un défi EAP-AKA' et attend la réponse AKA-Challenge.
`sim_challenge`	Le TWAG a envoyé un défi EAP-SIM et attend la réponse SIM/Challenge.
`authenticated`	L'échange EAP a réussi. Le TWAG a la MSK et est sur le point d'établir la session S2a.
`session_active`	La session S2a est active. L'abonné peut transmettre des données utilisateur. La comptabilité RADIUS peut également promouvoir une session à cette phase.

Deux autres valeurs de phase existent dans le code :

- `resyncing` est réservé comme une phase d'authentification en cours. En pratique, un tour de resynchronisation reconstruit le défi et renvoie la phase à `aka_challenge` ou `aka_prime_challenge`, donc l'état ne se fixe pas sur `resyncing`.
- `terminating` marque une session qui est en cours de démantèlement.

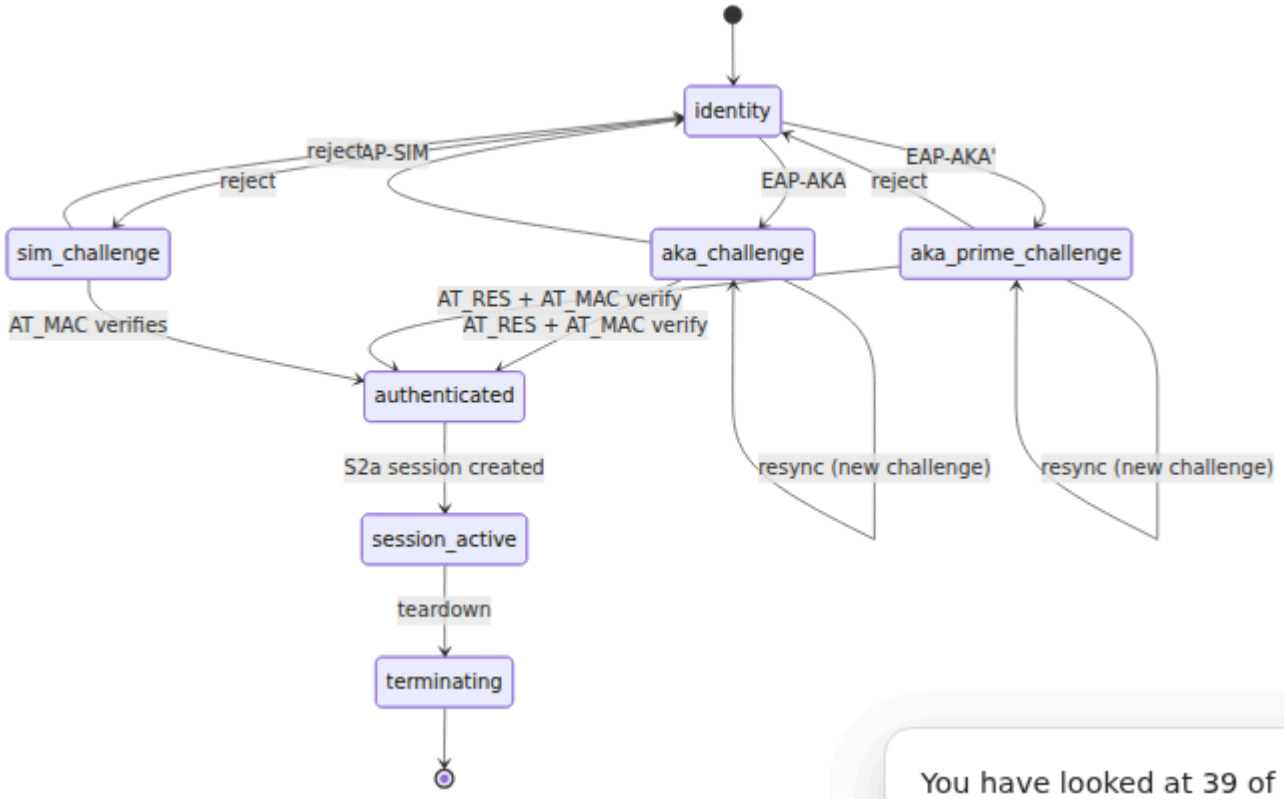
Une session qui reste dans une phase d'authentification en cours (`identity`, `aka_challenge`, `aka_prime_challenge`, `sim_challenge`, ou `resyncing`) au-delà du délai d'identité est supprimée. Définissez le délai avec `identity_timeout_ms`. Voir Configuration.

La page Sessions associe ces phases au statut d'authentification de session affiché dans l'UI et l'API.



Séquence d'Authentification

La séquence complète comporte une phase d'identité et une phase de défi. L'exemple ci-dessous montre EAP-AKA'. EAP-AKA a la même structure. EAP-SIM utilise un tour de démarrage à la place du tour AKA-Identity, et il n'envoie pas `AT_RES`.



Phase d'identité

1. L'abonné envoie une réponse EAP-Identity qui contient l'Identifiant d'Accès Réseau (NAI).
2. Pour EAP-AKA' ou EAP-AKA, le TWAG répond avec une demande d'identité AKA qui contient `AT_PERMANENT_ID_REQ`. Cela demande à l'abonné son identité permanente.
3. Pour EAP-SIM, le TWAG répond avec une demande EAP-SIM/Start qui contient le `AT_VERSION_LIST` (version 1 uniquement).
4. L'abonné renvoie l'identité permanente (AKA-Identity avec `AT_IDENTITY`), ou la réponse de démarrage avec `AT_NONCE_MT` pour EAP-SIM.

Phase de défi

1. Le TWAG extrait l'IMSI de l'identité et obtient les vecteurs d'authentification. Voir Sourcing des Vecteurs d'Auth.
2. Le TWAG dérive les clés de session, y compris la MSK. Voir Dérivation et Livraison des Clés.
3. Le TWAG construit le défi. Pour EAP-AKA' le défi contient `AT_RAND`, `AT_AUTN`, `AT_KDF_INPUT` (le nom du réseau), `AT_KDF`, `AT_BIDDING`, et `AT_MAC`. Pour

EAP-AKA le défi contient `AT_RAND`, `AT_AUTN`, et `AT_MAC`. Pour EAP-SIM le défi contient `AT_RAND` (2 ou 3 RANDs) et `AT_MAC`.

4. L'abonné vérifie le réseau (AUTN) et le MAC, puis renvoie la réponse.
5. Le TWAG vérifie la réponse. Voir Vérification de la réponse.
6. En cas de succès, le TWAG renvoie un EAP-Success et la MSK. En cas d'échec, le TWAG renvoie un EAP-Failure.

Vérification de la réponse

Le TWAG vérifie la réponse au défi dans un ordre fixe. L'ordre est le même pour EAP-AKA et EAP-AKA'.

1. **Vérification de l'état.** Si le `K_aut` ou XRES stocké est manquant, le TWAG rejette.
2. **Vérification des attributs.** Si `AT_RES` ou `AT_MAC` est manquant, le TWAG rejette.
3. **Vérification du MAC.** Le TWAG met à zéro le champ `AT_MAC`, recompute le MAC sur le paquet avec `K_aut`, et compare. EAP-AKA utilise HMAC-SHA1. EAP-AKA' utilise HMAC-SHA-256. Les deux tronquent à 16 octets.
4. **Vérification de la réponse.** Le TWAG compare `AT_RES` avec le XRES stocké. La comparaison est **en temps constant** pour prévenir les attaques par timing. Le TWAG utilise un contrôle de longueur et une comparaison d'octets en temps constant.
5. **Résultat.** Si le MAC et la réponse correspondent, le TWAG accepte. Sinon, le TWAG rejette.

Pour EAP-SIM, l'abonné n'envoie pas `AT_RES`. La vérification de `AT_MAC` est la seule preuve que l'abonné détient les clés correctes. Si le MAC est vérifié, le TWAG accepte.

Sourcing des Vecteurs d'Auth

Le TWAG obtient les vecteurs d'authentification à partir de l'une de deux sources. Définissez la source avec la clé de configuration `auth_mode`.

Mode	`auth_mode`	Source du vecteur	Utiliser
SWx	`:swx`	HSS, via Diameter SWx (MAR/MAA)	Production
Local	toute valeur autre que `:swx` (par défaut `:local`)	Milenage local à partir de clés de test provisionnées	Test

Dans les deux modes, le TWAG exécute la machine d'état EAP localement. Seule la source du vecteur diffère. Toute valeur `auth_mode` qui n'est pas `:swx` sélectionne le chemin local Milenage.

> **Le passage STa est un chemin séparé.** Un fournisseur `Twag.Auth.StaAuth` peut relayer des charges utiles EAP brutes à un serveur AAA 3GPP via Diameter STa (DER/DEA), de sorte que le serveur AAA exécute l'échange EAP-AKA et renvoie la MSK. Ce fournisseur existe mais n'est pas sur le chemin d'authentification RADIUS en direct. Le chemin en direct récupère les vecteurs avec `:swx` ou `:local` et exécute la machine d'état EAP sur le TWAG.

SWx vers le HSS

En mode `:swx`, le TWAG envoie une Demande d'Authentification Multimédia (MAR) au HSS via l'interface SWx (3GPP TS 29.273, ID d'application `16777265`). Le MAR contient :

- `User-Name`: l'IMSI.
- `SIP-Auth-Data-Item` avec `SIP-Authentication-Scheme` défini sur la méthode EAP en cours : `EAP-AKA'` ou `EAP-AKA`. Pour une resynchronisation, l'élément contient également `SIP-Authorization`. Cela contient le RAND suivi de l'AUTS.
- `SIP-Number-Auth-Items`: 1.
- `ANID`: le nom du réseau. La valeur par défaut est `WLAN`. Pour `EAP-AKA'`, le HSS l'utilise pour dériver `CK'`/`IK'`. Voir 3GPP TS 29.273 clause 8.1.2.1.
- `RAT-Type`: 0.

- `Auth-Session-State`: 1 (NO_STATE_MAINTAINED).
- `Vendor-Specific-Application-Id` avec l'ID d'application d'authentification SWx `16777265` et Vendor-Id `10415`.

Le HSS renvoie une Réponse d'Authentification Multimédia (MAA). Lorsque le `Result-Code` est `2001` (succès), le TWAG lit le premier `SIP-Auth-Data-Item` et extrait le vecteur :

- `SIP-Authentication-Scheme` = le schéma du vecteur : `EAP-AKA'` ou `EAP-AKA`.
- `SIP-Authenticate` = RAND (16 octets) suivi de AUTN (16 octets).
- `SIP-Authorization` = XRES.
- `Confidentiality-Key` = CK pour `EAP-AKA`, ou CK' pour `EAP-AKA'`.
- `Integrity-Key` = IK pour `EAP-AKA`, ou IK' pour `EAP-AKA'`.

Si le `Result-Code` n'est pas `2001`, ou si les champs du vecteur sont manquants, le TWAG considère la réponse comme un refus d'authentification. Le TWAG rejette l'abonné avec un EAP-Failure et enregistre un refus, pas un crash.

Si le TWAG a demandé `EAP-AKA'` et que le vecteur n'est pas un vecteur EAP-AKA', le TWAG rejette également l'abonné. Cela s'applique lorsque le schéma est `EAP-AKA`, ou lorsque le bit de séparation AMF dans l'AUTN est 0. Voir hiérarchie de clés EAP-AKA'.

Signalisation d'erreur SWx (Experimental-Result)

SWx signale la plupart des erreurs dans un AVP `Experimental-Result` groupé (Vendor-Id `10415`), et non dans le `Result-Code` de base. Un MAA échoué peut donc ne pas contenir de base `Result-Code` du tout. Le TWAG lit également le `Experimental-Result-Code`, et il enregistre le nom lisible par l'homme afin que l'opérateur voit la véritable cause. Voir Dépannage pour les codes.

Un cas courant est `DIAMETER_ERROR_USER_UNKNOWN` (5001) ou `DIAMETER_ERROR_USER_NO_NON_3GPP_SUBSCRIPTION` (5450) : l'abonné n'est pas activé pour l'accès non-3GPP sur SWx, donc l'authentification échoue même si le même abonné peut être connu sur S6a pour l'accès EPC.

Le DRA doit router l'application SWx vers le HSS. Voir Architecture : Routage DRA.

Milenage Local

En mode `:local`, le TWAG génère le vecteur avec l'algorithme Milenage (3GPP TS 35.206) à partir des clés de test provisionnées (`test_keys`). Les clés sont `ki`, `opc`, `amf`, et `sqn`. Le TWAG calcule RAND, AUTN, XRES, CK, et IK dans le processus. Utilisez ce mode uniquement pour des tests en laboratoire.

Vecteurs pour EAP-SIM

EAP-SIM a besoin de 2 ou 3 triplets GSM. Le TWAG dérive les triplets à partir des mêmes vecteurs Milenage. Pour chaque vecteur, le TWAG prend RAND, prend les 4 premiers octets de XRES comme SRES, et dérive le `Kc` GSM à partir de CK et IK avec la fonction de conversion c3 (3GPP TS 33.102 Annexe C.3) : `Kc = CK[0..7] XOR CK[8..15] XOR IK[0..7] XOR IK[8..15]`.

Gestion de l'Identité et NAI

L'identité de l'abonné est un Identifiant d'Accès Réseau (NAI). Le NAI a la forme `{username}@{realm}`, par exemple `0{IMSI}@wlan.mnc{MNC}.mcc{MCC}.3gppnetwork.org`.

Extraction de l'IMSI

Le TWAG extrait l'IMSI du NAI en deux étapes :

1. Le TWAG prend le nom d'utilisateur, c'est-à-dire la partie avant le `@`.
2. Le TWAG supprime un chiffre de type d'identité en tête. Un `0` en tête (identité permanente, EAP-AKA/AKA') ou un `6` en tête correspond à l'IMSI nu. Les chiffres restants sont l'IMSI.

Le TWAG accepte un IMSI de 14 ou 15 chiffres. L'IMSI est le `User-Name` dans le SWx MAR. Le TWAG utilise également l'IMSI pour dériver le MCC et le MNC pour la session GTP-C : le MCC est les 3 premiers chiffres, et le MNC est les 2 chiffres suivants.

Nom du réseau (EAP-AKA')

EAP-AKA' lie les clés au nom du réseau. 3GPP appelle cette valeur l'**Identité du Réseau d'Accès (ANID)**. Le TWAG envoie le nom du réseau à l'abonné dans `AT_KDF_INPUT`. Le TWAG envoie la même chaîne au HSS dans l'AVP `ANID` du SWx MAR. Le HSS et l'abonné l'utilisent tous deux pour dériver `CK'`/`IK'`, donc les deux valeurs doivent être identiques.

La valeur par défaut est `WLAN`. C'est l'ANID pour un réseau d'accès WLAN dans 3GPP TS 24.302 clause 8.1.1.2, sans chaîne ANID supplémentaire. La clé de configuration `network_name` remplace la valeur par défaut. Laissez la clé non définie à moins que le HSS n'attende un ANID différent.

Dérivation et Livraison des Clés

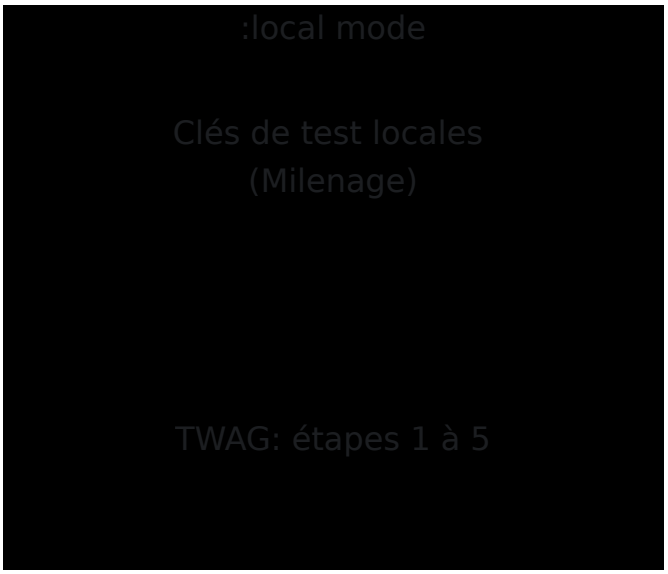
Chaque méthode dérive une Clé de Session Maître (MSK) de 64 octets et une MSK Étendue (EMSK) de 64 octets. Le TWAG livre la MSK au point d'accès. Le TWAG conserve l'EMSK pour une ré-authentification rapide.

Hiérarchie de clés EAP-AKA'

Selon RFC 5448 et 3GPP TS 33.402 clause 6.2 et Annexe A :

1. `Key = CK \| IK`.
2. `S = FC(0x20) \| network_name \| len(network_name) \| (SQN XOR AK) \| len(SQN XOR AK)`.
3. `CK' \| IK' = HMAC-SHA-256(Key, S)`. `CK'` est les 16 premiers octets. `IK'` est les 16 derniers octets.
4. `MK = PRF'(IK' \| CK', "EAP-AKA'" \| Identity)`, 208 octets. PRF' utilise HMAC-SHA-256.
5. Le TWAG divise `MK`: `K_encr` (16), `K_aut` (32), `K_re` (32), `MSK` (64), `EMSK` (64).

La source du vecteur décide où s'exécutent les étapes 1 à 3 :


:local mode
Clés de test locales
(Milenage)
TWAG: étapes 1 à 5


:swx mode
MAR: Scheme=EAP-
AKA', ANID
HSS: bit de séparation
AMF = 1
étapes 1 à 3 (CK', IK')
MAA: RAND, AUTN,
XRES, CK', IK'
TWAG: étapes 4 et 5

En mode `:swx`, le HSS exécute les étapes 1 à 3. Le TWAG demande un vecteur EAP-AKA' dans le MAR. Le HSS définit le bit de séparation AMF à 1 et dérive `CK'`/`IK'` avec l'ANID. Le MAA contient `CK'` dans `Confidentiality-Key` et `IK'` dans `Integrity-Key`. Le TWAG utilise ces clés telles que reçues et exécute les étapes 4 et 5. `CK` et `IK` ne quittent pas le HSS pour un vecteur AKA'. Voir 3GPP TS 33.402 clause 6.2, étape 10.

En mode `:local`, le TWAG est la source du vecteur. Le TWAG exécute les cinq étapes avec les clés de test configurées.

Si le HSS répond à une demande EAP-AKA' avec le schéma `EAP-AKA`, ou avec le bit de séparation AMF 0, le TWAG rejette l'abonné avec un EAP-Failure. La ligne de log commence par `HSS vector unusable for EAP-AKA'`. L'abonné rejette un tel défi dans tous les cas, selon la clause 6.2 étape 15 de TS 33.402. Le rejet précoce désigne le HSS comme la cause au lieu d'une `Client-Error` provenant du terminal.

Hiérarchie de clés EAP-AKA

Selon RFC 4187 :

1. `MK = SHA1(Identity \| IK \| CK)`.
2. Le TWAG étend `MK` à 160 octets avec le PRF FIPS 186-2.
3. Le TWAG divise la sortie : `K_encr` (16), `K_aut` (16), `MSK` (64), `EMSK` (64).

Hiérarchie de clés EAP-SIM

Selon RFC 4186 :

1. `MK = SHA1(Identity \| Kc1 \| Kc2 \| [Kc3] \| NONCE_MT \| Version_List \| Selected_Version)`.
2. Le TWAG étend `MK` à 160 octets avec le PRF FIPS 186-2.
3. Le TWAG divise la sortie : `K_encr` (16), `K_aut` (16), `MSK` (64), `EMSK` (64).

MSK à MPPE au point d'accès

Le TWAG livre la MSK de 64 octets au point d'accès dans l'Access-Accept RADIUS. Le TWAG divise la MSK et encode chaque moitié comme une clé MPPE Microsoft (RFC 2548, RFC 3748):

- `MS-MPPE-Recv-Key` = `MSK[0..31]` (premiers 32 octets).
- `MS-MPPE-Send-Key` = `MSK[32..63]` (derniers 32 octets).

Le TWAG chiffre chaque clé avec le secret partagé RADIUS et l'Authenticator de demande, conformément à la RFC 2548. Le TWAG utilise un sel frais de 2 octets avec le bit haut défini pour chaque clé. Le TWAG signe l'Access-Accept avec le secret partagé. Voir RADIUS Shared Secrets pour savoir comment le TWAG résout le secret.

Le point d'accès déchiffre les clés MPPE, récupère le MSK et utilise le MSK comme la clé maître pairwise pour le handshake 802.11 en 4 étapes avec l'abonné.

L'Access-Accept nécessite un MSK valide de 64 octets. Si le MSK est manquant ou a la mauvaise taille, le TWAG envoie un Access-Reject à la place.

Resynchronisation

L'abonné peut signaler que son numéro de séquence (SQN) est désynchronisé avec le réseau. L'abonné envoie un AKA-Synchronization-Failure (sous-type `0x04`) avec `AT_AUTS`. `AT_AUTS` fait 14 octets : 6 octets de SQN dissimulé et 8 octets de MAC-S.

Le TWAG gère la resynchronisation comme suit :

1. Le TWAG construit les informations de resynchronisation : le RAND stocké concaténé avec les 14 octets d'AUTS.
2. Le TWAG exécute la resynchronisation. Cela récupère le SQN de l'abonné avec la fonction Milenage f5*, vérifie MAC-S et renvoie un numéro de séquence mis à jour.
3. En cas de succès, le TWAG génère un nouveau vecteur avec le SQN mis à jour, construit un nouveau défi et envoie un nouvel Access-Challenge.

4. En cas d'échec, le TWAG rejette l'abonné avec un EAP-Failure.

Le TWAG gère la resynchronisation pour EAP-AKA et EAP-AKA'. La resynchronisation utilise les clés Milenage locales. La valeur `AT_AUTS` doit faire exactement 14 octets, sinon le TWAG génère une erreur et le tour échoue.

Ré-authentification rapide (ERP)

Le TWAG prend en charge le Protocole de Ré-authentification EAP (ERP) conformément à RFC 6696. L'ERP permet à un abonné de se ré-authentifier sans un échange EAP complet. L'économie provient d'une chaîne de clés que le TWAG dérive de l'EMSK lors de la première authentification complète.

Après une authentification complète, le TWAG dérive et stocke les clés de ré-authentification :

- `rRK` = clé racine de ré-authentification, dérivée de l'EMSK.
- `rIK` = clé d'intégrité de ré-authentification, dérivée de `rRK`.

Toutes les dérivations de clés ERP utilisent HMAC-SHA-256.

Lorsque l'abonné envoie un message EAP-Initiate/Re-auth (type EAP 41), le TWAG procède comme suit :

1. Si le TWAG n'a pas de `rRK` ou `rIK` stocké, le TWAG revient à une authentification complète. Le TWAG envoie une demande AKA-Identity fraîche.
2. Si le TWAG a les clés, le TWAG analyse le message et vérifie le tag d'authentification avec `rIK`.
3. Si le tag est vérifié, le TWAG dérive le MSK de ré-authentification (`rMSK`) à partir de `rRK` et du numéro de séquence du message, construit une réponse EAP-Finish/Re-auth (type EAP 42) et accepte. Le `rMSK` devient le nouveau MSK pour les clés MPPE.
4. Si le tag ne vérifie pas, ou si l'analyse échoue, le TWAG revient à une authentification complète.

Gestion des échecs et codes de

résultat

Résultats EAP

Condition	Action du TWAG
Échec de SWx MAR, ou vecteurs incomplets	La récupération de vecteur est refusée. Le TWAG rejette proprement l'abonné avec un EAP-Failure, et enregistre un refus plutôt qu'un crash.
EAP-AKA' demandé, mais le schéma MAA est `EAP-AKA`	Access-Reject avec EAP-Failure. La ligne de log commence par `HSS vector unusable for EAP-AKA'`. Le HSS ne prend pas en charge les vecteurs EAP-AKA'.
EAP-AKA' demandé, mais le bit de séparation AMF dans l'AUTN est 0	Access-Reject avec EAP-Failure. Même ligne de log. Le HSS a émis un vecteur que l'abonné doit rejeter (TS 33.402 clause 6.2 étape 15).
Échec de la vérification `AT_MAC`	Access-Reject avec EAP-Failure.
`AT_RES` ne correspond pas à XRES	Access-Reject avec EAP-Failure.
`K_aut` ou état XRES manquant	Access-Reject avec EAP-Failure.
Attribut `AT_RES` ou `AT_MAC` manquant	Access-Reject avec EAP-Failure.
L'abonné envoie AKA-Authentication-Reject (sous-type `0x02`)	Access-Reject. L'abonné a rejeté le réseau.

Condition	Action du TWAG
L'abonné envoie AKA-Client-Error (sous-type `0x0E`)	Access-Reject.
EAP-NAK sans méthode prise en charge	Access-Reject.
MSK manquant ou pas 64 octets à Accept	Access-Reject.
Message EAP non traité	Access-Reject avec EAP-Failure.

Codes de résultat Diameter SWx

Le code de succès est un `Result-Code` de base. Les codes d'erreur ci-dessous arrivent dans le AVP `Experimental-Result`, pas dans le `Result-Code` de base. Voir la signalisation d'erreur SWx et Dépannage.

Code	Nom	Significat
2001	DIAMETER_SUCCESS	Le MAA po un vecteur valide.
5001	DIAMETER_ERROR_USER_UNKNOWN	Le HSS n'a pas d'abonnem non-3GPP/SWx pour cet IM L'abonné p encore êtr connu sur S6a/EPC.
5004	DIAMETER_ERROR_NOT_SUPPORTED_USER_DATA	Les donné utilisateur demandée ne sont pa prises en charge.
5420	DIAMETER_ERROR_UNKNOWN_EPS_SUBSCRIPTION	Pas d'abonnem EPS pour c abonné.
5450	DIAMETER_ERROR_USER_NO_NON_3GPP_SUBSCRIPTION	L'abonné n pas d'abonnem non-3GPP.
Autre	(Erreur Diameter)	L'obtentio vecteur échoue. Le

Code	Nom	Significat
		tour est rejeté.

Référence : RFC 6733 Section 7.1, 3GPP TS 29.273, 3GPP TS 29.229.

Événements initiés par le réseau (STa)

Le serveur AAA 3GPP peut envoyer des événements au TWAG via l'interface STa (3GPP TS 29.273, ID d'application `16777250`). Le TWAG répond à chaque événement avec `Result-Code` `2001`.

Message	Déclencheur	Action du TWAG
ASR (Abort-Session-Request)	Le serveur AAA souhaite une déconnexion.	Le TWAG déconnecte la session de l'abonné, puis répond ASA.
STR (Session-Termination-Request)	Détachement initié par le HSS.	Le TWAG déconnecte la session de l'abonné, puis répond STA.
RAR (Re-Auth-Request)	Le serveur AAA souhaite une ré-authentification.	Le TWAG répond RAA avec succès et enregistre la demande. La ré-authentification complète de la session en cours n'est pas encore déclenchée à partir de RAR.

Les trois réponses portent l'Origin-Host et l'Origin-Realm de l'identité Diameter. Voir Configuration.

Configuration

Définissez les paramètres d'authentification dans la configuration de l'application `:omnitwag`. L'identité Diameter (`host` et `realm`) provient de la

configuration `:diameter_ex`, et le TWAG l'utilise comme Origin-Host et Origin-Realm sur SWx et STa.

```
config :omnitwag,
  # Source de vecteur : :swx (production) ou :local (test)
  auth_mode: :swx,

  # Méthode EAP proposée : :eap_aka_prime (par défaut), :eap_aka,
ou :eap_sim
  eap_method: :eap_aka_prime,

  # Nom du réseau EAP-AKA' (ANID). Le TWAG l'envoie dans
AT_KDF_INPUT et dans le
  # SWx MAR. La valeur par défaut est "WLAN" (TS 24.302 clause
8.1.1.2). Gardez-le non défini dans
  # les déploiements normaux.
  network_name: "WLAN",

  # Récoltez une session toujours bloquée dans une phase
d'authentification en cours après ce nombre
  # de millisecondes. Par défaut 600000 (10 minutes).
  identity_timeout_ms: 600_000,

  # Champs TWAN-Identifier, ajoutés à STa DER/AAR lorsqu'ils sont
définis
  twan_ssid: "ons",
  twan_bssid: "00:11:22:33:44:55",

  # Clés de test locales (utilisées uniquement lorsque auth_mode:
:local)
  test_keys: %{
    ki: "11111111111111111111111111111111",
    opc: "22222222222222222222222222222222",
    amf: "8000",
    sqn: 1200
  }
```

Paramètre	Type	Requis	Par défaut	Descripti
`auth_mode`	Atome	Non	`:local`	Source de vecteur. `:sw` obtient des vecteurs du via SWx. `:lo` génère des vecteurs à pa de `test_key` Définissez `:s` en productio
`eap_method`	Atome	Non	`:eap_aka_prime`	La méthode proposée. Ur `:eap_aka_pr` `:eap_aka`, `:eap_sim`. L'abonné peu négocier une méthode pris en charge différente av EAP-NAK.
`network_name`	Chaîne	Non	`WLAN`	Le nom du réseau EAP-A (ANID). Le TV l'envoie dans `AT_KDF_INPU` en tant qu'AV `ANID` dans le SWx MAR. `WL` est la valeur définie par T 24.302 claus

Paramètre	Type	Requis	Par défaut	Descripti
				8.1.1.2 pour l'accès WLAN
`identity_timeout_ms`	Entier	Non	`600000`	Temps en milliseconde qu'une sessi peut rester d une phase d'authentific en cours ava que le TWAG la récolte.
`twan_ssid`	Chaîne	Non	-	Le SSID WLA Lorsqu'il est défini, le TW ajoute un TW Identifier aux messages ST DER et AAR.
`twan_bssid`	Chaîne	Non	-	Le BSSID du point d'accès Combiné ave `twan_ssid` p former le TW Identifier.
`test_keys`	Carte	Non	valeurs par défaut intégrées	Clés Milenag locales. Utilis uniquement lorsque `auth_mode` e `:local`. Voir key paramet

Paramètres de clés de test

Le TWAG lit `test_keys` uniquement en mode `:local`. Ne pas utiliser `test_keys` en production.

Paramètre	Type	Requis	Par défaut	Description
`ki`	Chaîne (hex)	Oui (mode local)	`1111...`	Clé d'abonné `Ki`, 16 octets sous forme de 32 caractères hexadécimaux.
`opc`	Chaîne (hex)	Oui (mode local)	`2222...`	Clé de variante d'opérateur `OPc`, 16 octets sous forme de 32 caractères hexadécimaux.
`amf`	Chaîne (hex)	Oui (mode local)	`8000`	Champ de gestion d'authentification, 2 octets sous forme de 4 caractères hexadécimaux.
`sqn`	Entier	Oui (mode local)	`1200`	Le numéro de séquence de départ.

Observabilité

Le TWAG émet des compteurs de télémétrie via la machine d'état EAP et le gestionnaire STa. Le point de terminaison Prometheus expose les compteurs. Voir le Metrics Reference pour le point de terminaison et des exemples de requêtes.

Métriques EAP

Métrique	Signification
`eap_aka_identity_count`	Échanges d'identité EAP-AKA/AKA' (Phase 1).
`eap_aka_challenge_count`	Échanges de défi EAP-AKA/AKA' (Phase 2).
`eap_aka_sync_failure_count`	Échecs de synchronisation AKA (événements de resynchronisation).
`eap_aka_auth_success_count`	Authentifications EAP-AKA/AKA' réussies.
`eap_aka_auth_reject_count`	Authentifications EAP-AKA/AKA' rejetées.

Le TWAG émet également des événements de télémétrie pour EAP-SIM (`start`, `auth_success`, `auth_reject`), pour AKA `auth_reject` et `client_error`, et pour ERP (`initiate`, `success`, `fallback`, `mac_failure`). Voir le Metrics Reference.

Métriques de réponse RADIUS

Métrique	Signification
`radius_access_challenge_count`	Réponses Access-Challenge (tour EAP en cours).
`radius_access_accept_count`	Réponses Access-Accept (authentification réussie).
`radius_access_reject_count`	Réponses Access-Reject (authentification échouée).

Métriques d'événements STa

Le gestionnaire STa émet un compteur de télémétrie pour chaque événement initié par le réseau : `[:sta, :asr, :received, :count]`, `[:sta, :str, :received, :count]`, et `[:sta, :rar, :received, :count]`, plus des compteurs de session `disconnect` et `reauth`.

Contexte de journal

Le TWAG définit les métadonnées de journal `identity` et `imsi` pour chaque session. Utilisez ces champs pour tracer un abonné à travers le flux d'authentification.

Dépannage

Symptôme	Cause probable	
L'authentification échoue et le journal montre `DIAMETER_ERROR_USER_UNKNOWN (5001)`	Le HSS n'a pas d'abonnement non-3GPP/SWx pour cet IMSI. L'abonné peut être connu sur S6a/EPC mais n'est pas activé pour l'accès non-3GPP.	Activ non- l'abo
L'authentification échoue et le journal montre `DIAMETER_ERROR_USER_NO_NON_3GPP_SUBSCRIPTION (5450)`	Le profil de l'abonné n'a pas d'abonnement non-3GPP.	Prov l'abo 3GP
Le journal montre `Result-Code=nil` lors d'un MAA échoué	SWx a renvoyé une erreur dans l'AVP `Experimental-Result`, pas dans le `Result-Code` de base.	Lise `Expe` `Resu` mên journ impr par l
L'authentification ne se termine jamais et la session est récoltée	L'abonné a stagné dans une phase d'authentification en cours au-delà du délai d'identité.	Vérif l'éch `iden` si né

Symptôme	Cause probable	
L'abonné envoie AKA-Authentication-Reject	L'UE n'a pas réussi à authentifier le réseau (vérification AUTN).	Vérif du r et l'A corr SIM ou q HSS mod
Échec répété de AKA-Synchronization-Failure	Le SQN de l'abonné a dérivé du SQN du réseau.	Le T auto resy cela la ge le HS Resy
Access-Reject avec un EAP-Success valide attendu	Le MSK était manquant ou pas 64 octets à Accept.	Conf sour renv com MAA l'ens loca
L'authentification EAP-AKA' échoue avant le défi et le journal montre `HSS vector unusable for EAP-AKA'`	Le HSS a répondu au MAR avec un vecteur EAP-AKA : schéma `EAP-AKA`, ou bit de séparation AMF 0. Le HSS ne prend pas en charge les	Mett conf pour vect conf clau 29.2 solu cont défi

Symptôme	Cause probable	
	vecteurs EAP-AKA', ou il a ignoré l'`ANID`.	`eap_` `:eap`
Le téléphone répond au défi EAP-AKA' avec AKA-Client-Error	Le téléphone n'a pas pu vérifier le défi. La cause habituelle est un décalage de nom de réseau : le HSS a dérivé `CK'`/`IK'` avec un ANID différent de la valeur `AT_KDF_INPUT`.	Conf utilis Garc non deux `WLAN`

Pour les rejets au niveau RADIUS qui se produisent avant ou autour de l'échange EAP (mauvais secret partagé, point d'accès inconnu), voir RADIUS Shared Secrets et la référence d'état des Sessions.

Flux d'appels

Vue d'ensemble

Cette page montre les flux de messages de bout en bout pour OmniTWAG. Chaque flux est un diagramme de séquence entre l'équipement utilisateur (UE), le point d'accès WiFi (AP), le TWAG et les pairs du réseau central : le HSS (SWx), le PGW (S2a) et l'OCS (Gy).

Lisez les pages par fonctionnalité pour les détails derrière chaque étape : Authentification, S2a GTP-C, Plan utilisateur, Gestion des adresses IP, Pools IP, Chargement en ligne, Identifiants RADIUS et Sessions.

Table des matières

- Acteurs et interfaces
- Attachement réussi - tunnelé, facturé
- Attachement - NSWO avec un pool IP
- Attachement - NSWO avec distribution DHCP
- Attachement - rupture locale (plan de données AP)
- Échec d'authentification - abonné non provisionné pour SWx
- Comptabilité et rapport d'utilisation
- Épuisement de quota et réattribution
- Démontage - déconnexion de l'abonné
- Démontage - coup de pied de l'opérateur depuis l'API
- Démontage - délai d'activité
- Variantes de mode de facturation

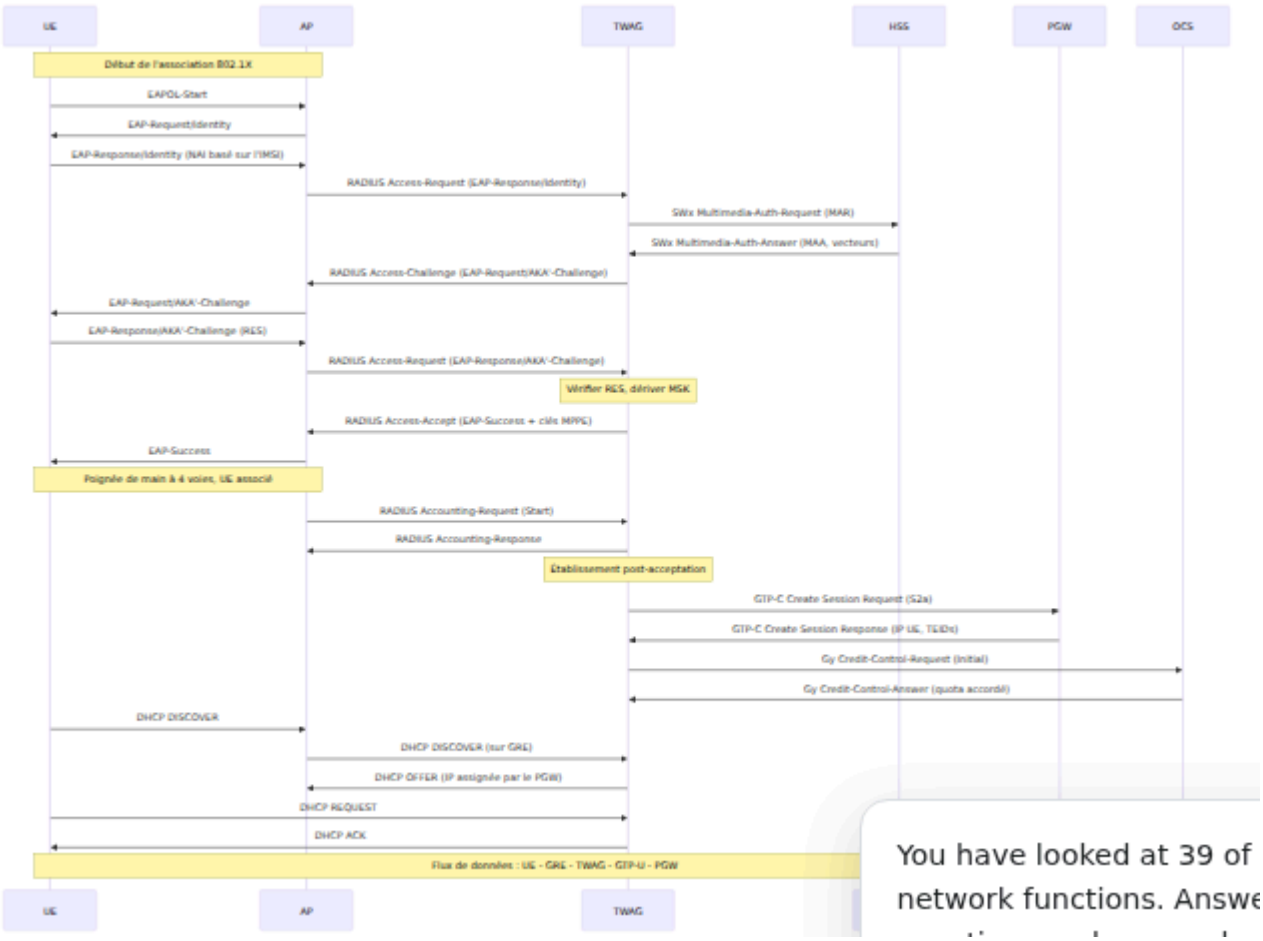
Acteurs et interfaces

Acteur	Rôle	Interface vers le TWAG
UE	Appareil abonné avec une SIM	802.11 / EAPOL (via l'AP)
AP	Point d'accès WiFi et client RADIUS	RADIUS (UDP 1812 auth, 1813 comptabilité)
TWAG	Passerelle d'accès WiFi de confiance et serveur AAA 3GPP	-
HSS	Serveur d'abonnés à domicile	Diameter SWx (vecteurs d'authentification)
PGW	Passerelle de paquets	GTP-C/GTP-U sur S2a (tunnelé uniquement)
OCS	Système de facturation en ligne	Diameter Gy (facturé uniquement)

Le type de rupture et le mode de facturation d'une session proviennent des identifiants de l'AP. Voir Identifiants RADIUS.

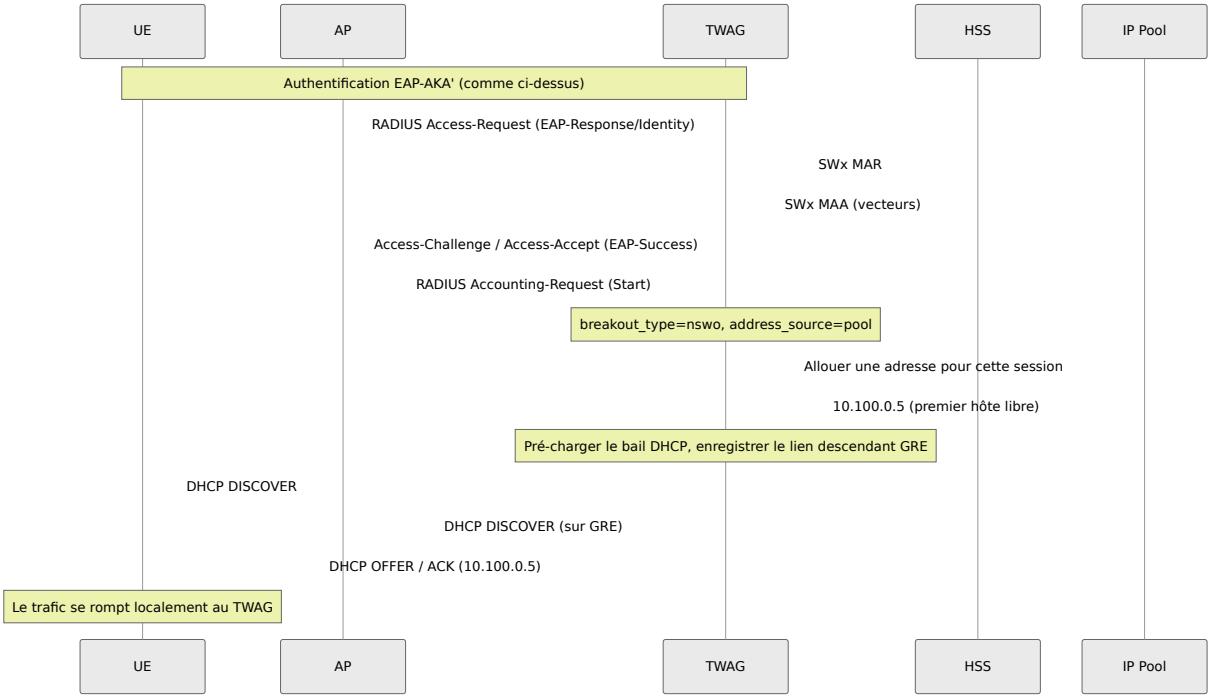
Attachement réussi - tunnelé, facturé

C'est le chemin heureux complet pour `breakout_type: tunneled` et `charging_mode: charged`. Le TWAG authentifie l'abonné avec EAP-AKA', établit une session GTP avec le PGW, délivre l'adresse IP assignée par le PGW via DHCP et ouvre une session de facturation Gy.



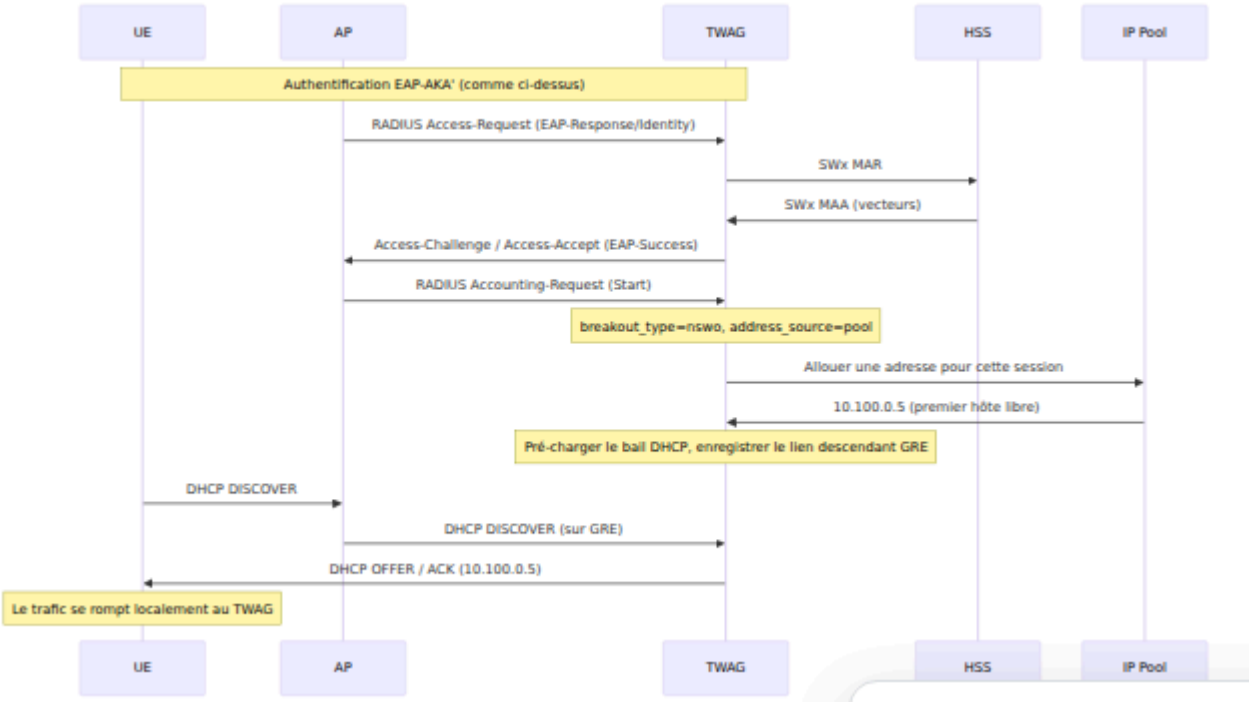
Attachement - NSWO avec un pool IP

Pour `breakout_type: nswo` et `address_source: pool`, le TWAG ne contacte pas le PGW. Le TWAG pré-alloue une adresse fixe à partir du pool IP associé et la délivre via DHCP. Le trafic de l'abonné se rompt localement.



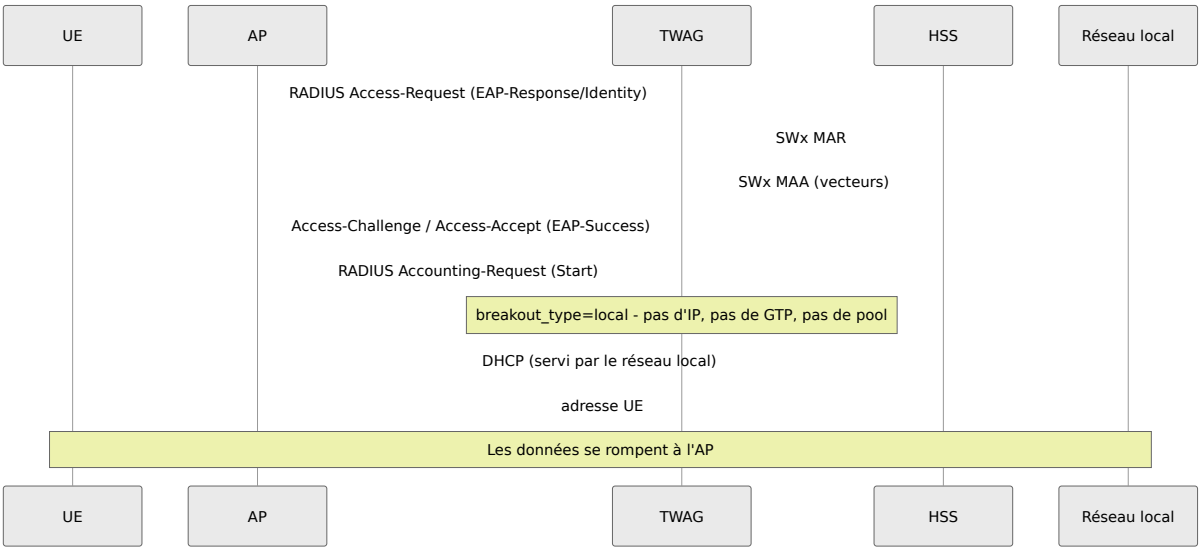
Attachement - NSWO avec distribution DHCP

Pour `breakout_type: nswo` et `address_source: dhcp`, le TWAG lie le MAC du client au pool. Le serveur DHCP distribue la prochaine adresse libre du pool lorsque l'UE envoie un DISCOVER. Voir Pools IP.



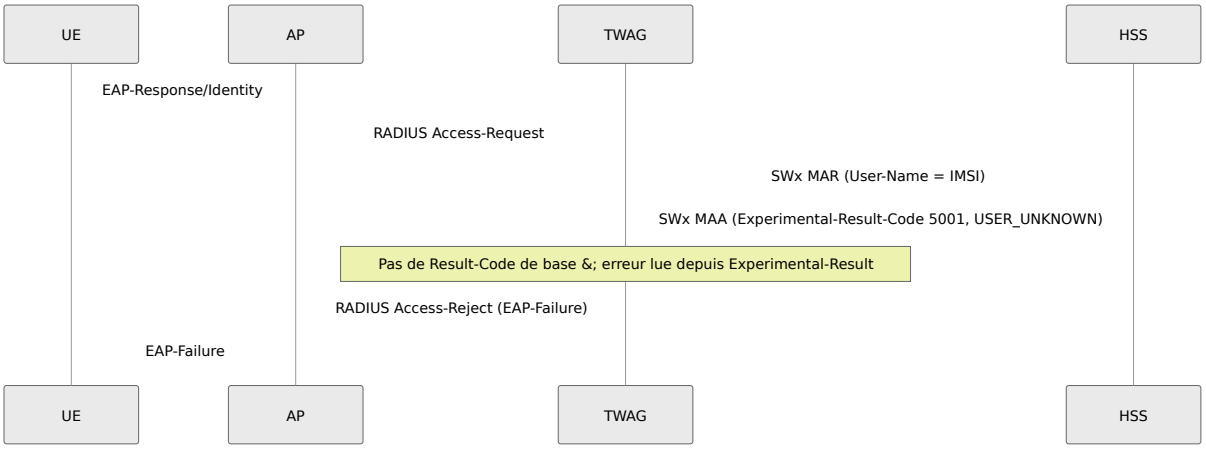
Attachement - rupture locale (plan de données AP)

Pour `breakout_type: local`, l'AP possède le plan de données. Le TWAG authentifie l'abonné et n'assigne pas d'IP. L'UE obtient son adresse du réseau local. Le TWAG n'établit pas de session GTP et pas de bail DHCP.



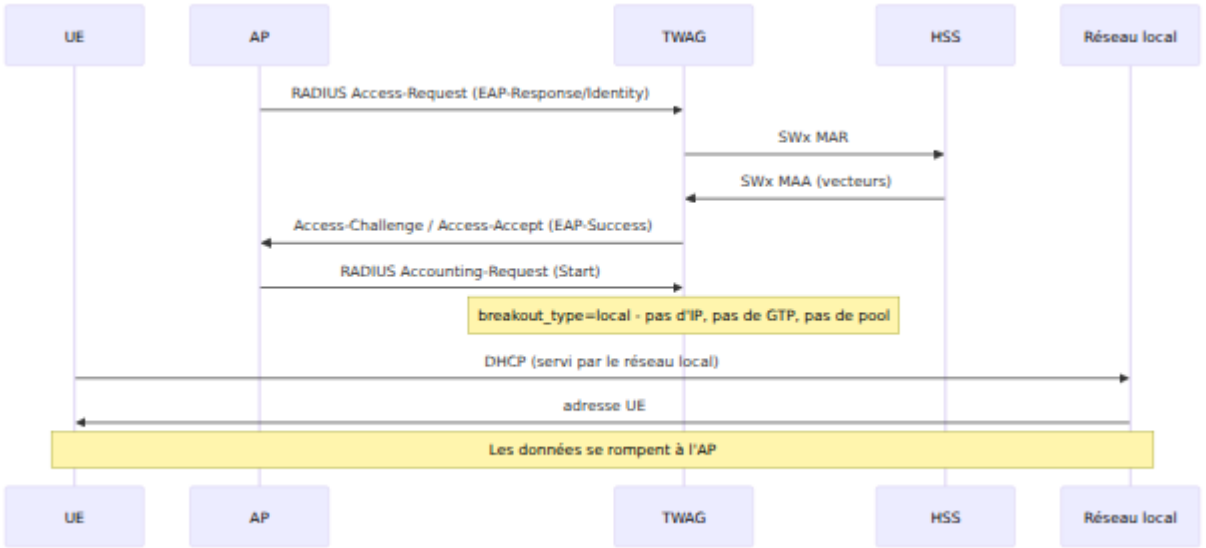
Échec d'authentification - abonné non provisionné pour SWx

Le HSS signale une erreur SWx dans l'AVP `Experimental-Result` groupé, pas dans le `Result-Code` de base. Un abonné connu sur S6a peut toujours échouer SWx si le HSS n'a pas d'abonnement non-3GPP pour l'IMSI. Le TWAG rejette la session. Voir Authentification.



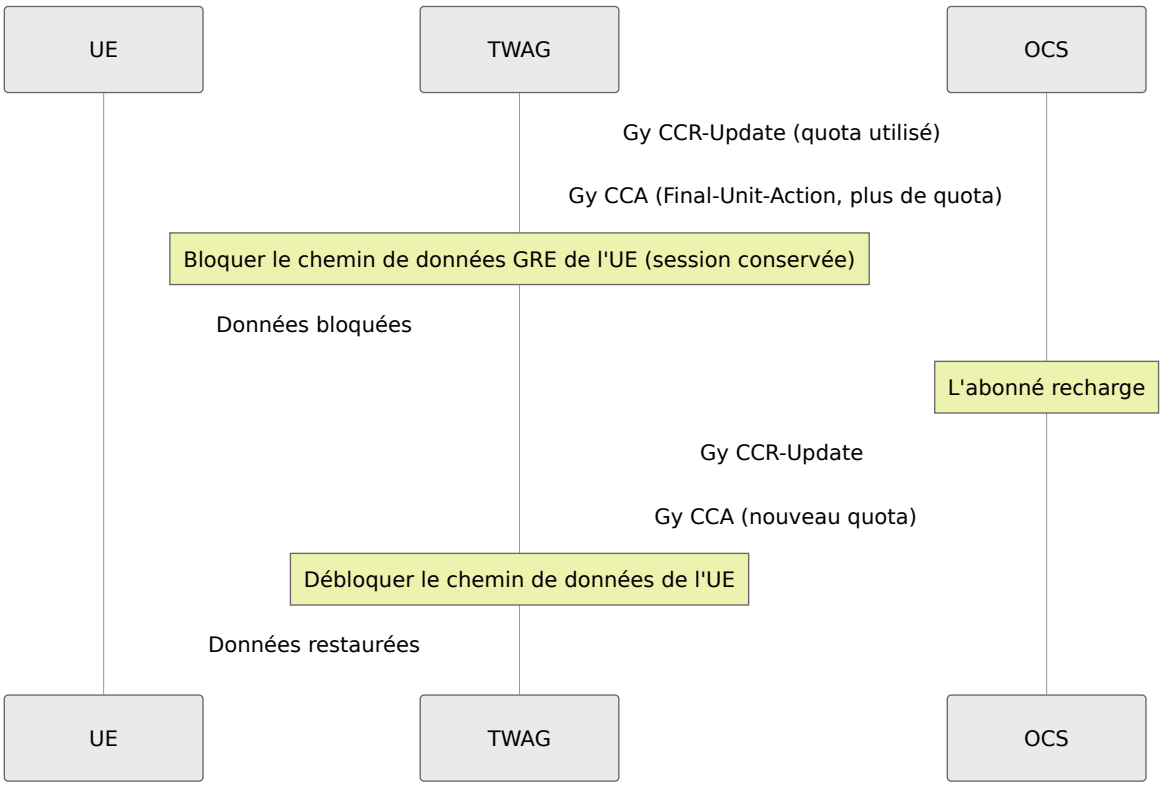
Comptabilité et rapport d'utilisation

L'AP envoie la comptabilité RADIUS. Le TWAG fusionne le trafic dans la session (voir Sessions) et, en mode `charged`, rapporte l'utilisation à l'OCS sur un seuil de quota ou une expiration de minuterie de validité, pas sur chaque mise à jour intermédiaire. Voir Chargement en ligne.



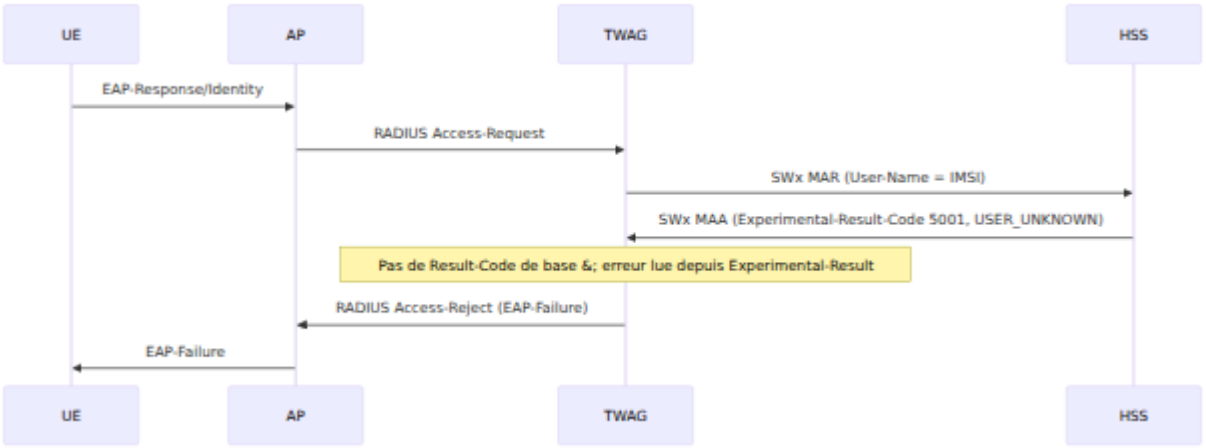
Épuisement de quota et réattribution

Lorsque l'OCS n'accorde plus de quota, ou définit une Action-Unit-Finale, le TWAG bloque le chemin de données de l'UE. Le TWAG conserve la session, le bail et la comptabilité afin qu'un rechargement puisse débloquer l'UE. Voir Chargement en ligne.



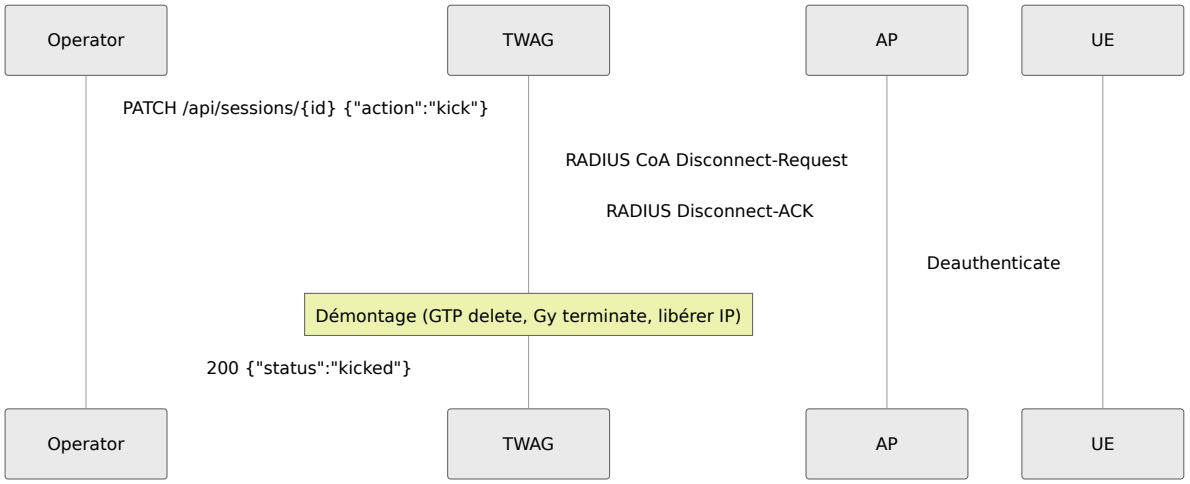
Démontage - déconnexion de l'abonné

Lorsque l'UE se déconnecte, l'AP envoie un Accounting-Stop. Le TWAG détruit la session : il supprime la session GTP (tunnelée), envoie un Gy CCR-Terminate (facturé), libère l'adresse et efface l'état de la session.



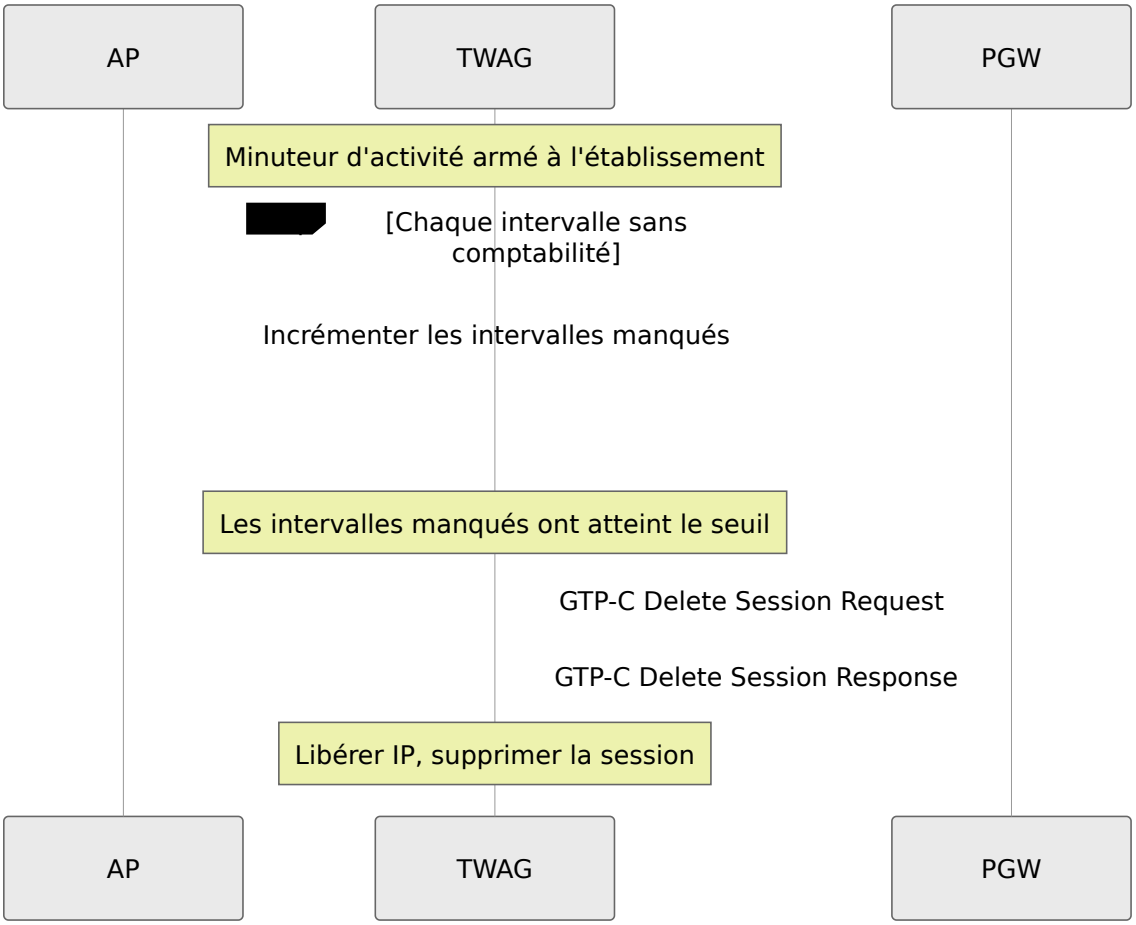
Démontage - coup de pied de l'opérateur depuis l'API

Un opérateur peut supprimer une session via l'API. Le TWAG envoie un RADIUS CoA-Disconnect (Disconnect-Message) à l'AP, qui déconnecte l'UE, puis détruit la session. Voir Sessions.



Démontage - délai d'activité

Le TWAG surveille une session établie avec un minuteur d'activité. Les mises à jour intermédiaires de comptabilité réinitialisent le minuteur. Si l'AP n'envoie pas de mises à jour pendant le nombre configuré d'intervalles, le TWAG déclare la session morte et la détruit. Une session qui ne termine jamais l'authentification est récoltée par un délai d'identité séparé. Voir Sessions.

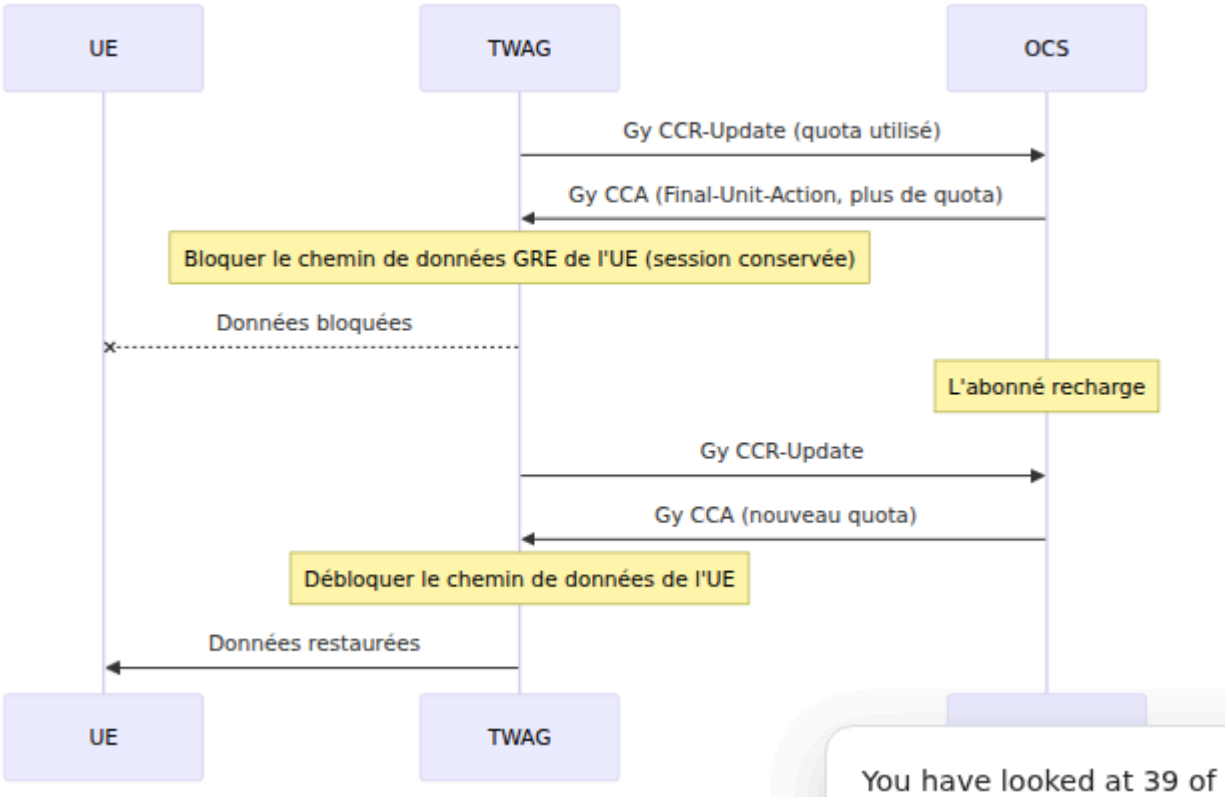


Variantes de mode de facturation

Le `charging_mode` sur l'identifiant de l'AP change les étapes post-acceptation. Voir Identifiants RADIUS.

charging_mode	Session de données	Gy
`charged`	Établi (selon `breakout_type`)	CCR-Initial / Update / Terminate
`uncharged`	Établi (selon `breakout_type`)	Pas de session Gy
`auth_only`	Non établi	Pas de session Gy

Pour `uncharged`, le flux est l'attachement tunnelé ou NSWO ci-dessus, mais le TWAG n'envoie pas de messages Gy. Pour `auth_only`, le flux s'arrête après l'Access-Accept et l'Accounting-Start : le TWAG n'établit pas de session de données.



Vue d'ensemble

Modes de connexion, WLCP et mobilité

Vue d'ensemble

Ce guide explique les modes de connexion TWAG, le protocole de contrôle WLAN (WLCP) et le comportement de transfert. Il décrit ce que fait le logiciel OmniTWAG aujourd'hui. Il marque également chaque domaine qui est partiel ou pas encore intégré, afin que les opérateurs ne planifient pas autour d'un comportement que le code ne gère pas.

Le TWAG suit le modèle WLAN de confiance dans la section 16 de 3GPP TS 23.402. Pour le plan de données, l'authentification et la signalisation S2a sur lesquels ces modes reposent, voir :

- Architecture : plan de données, cycle de vie de session et interfaces
- Authentification : EAP-AKA / EAP-AKA' et matériel clé
- S2a GTP-C : Création de session, suppression de session et signalisation de porteur
- Sessions : la session authentifiée et son cycle de vie d'état
- Plan utilisateur : GRE, GTP-U et livraison DHCP
- Secrets partagés RADIUS : séparation par point d'accès et facturation appliquée après authentification

Table des matières

- Vue d'ensemble et comparaison des modes
- Négociation de mode
- Mode de connexion unique transparent (TSCM)
- Mode de connexion unique (SCM)
- Mode de connexion multiple (MCM) et WLCP
- Transfert et mobilité
- Configuration

- Observabilité
- État de mise en œuvre

Vue d'ensemble et comparaison des modes

Le TWAG prend en charge trois modes de connexion pour un UE qui accède à l'EPC via un WLAN de confiance. Le mode contrôle comment l'UE apprend son adresse IP, si l'UE peut signaler un APN et si l'UE peut maintenir plus d'une connexion PDN.

Propriété	TSCM	SCM	MCM
Code de mode	`0`	`1`	`2`
Nom complet	Mode de connexion unique transparent	Mode de connexion unique	Mode de connexion multiple
Connexions PDN par UE	Une	Une	Plusieurs
Livraison d'adresse IP	DHCP	EAP	EAP / WLCP
Sélection d'APN signalée par l'UE	Non	Oui	Oui (par connexion)
Transfert avec préservation de l'IP	Non	Oui	Oui
Porteurs dédiés	Non	Oui	Oui
Protocole de contrôle de l'UE	Aucun	Aucun	WLCP (TS 24.244)
UE conscient de S2a	Non	Oui	Oui

TSCM est le mode par défaut. TSCM correspond au comportement pré-Release-12, où l'UE ne signale pas le réseau. Le TWAG sélectionne l'APN, crée une connexion PDN et délivre l'adresse IP allouée par le PGW via DHCP.

Les codes de mode `0`, `1` et `2` suivent l'AVP TWAN-Connection-Mode dans 3GPP TS 29.273.

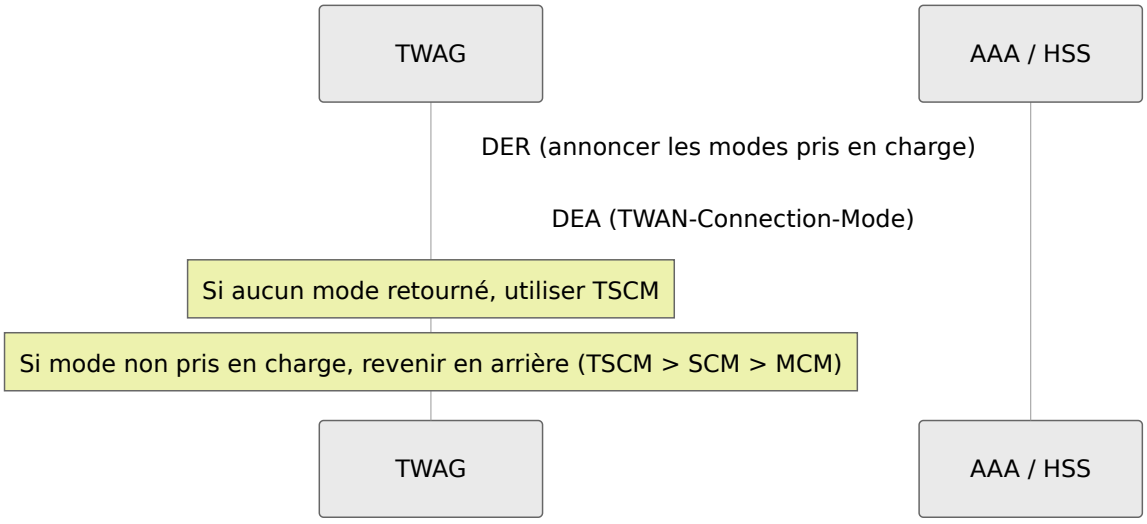


Négociation de mode

Le mode de connexion est négocié lors de l'authentification, sur l'échange STa qui transporte EAP. Voir Authentification pour le flux EAP lui-même. La conception est :

1. Le TWAG annonce les modes qu'il prend en charge au serveur AAA. L'ensemble annoncé provient de la clé de configuration `connection_modes`.
2. Le serveur AAA renvoie le mode sélectionné dans la réponse DEA STa, dans l'AVP TWAN-Connection-Mode.
3. Si le serveur AAA n'indique pas de mode, le TWAG utilise TSCM comme mode par défaut.

Si le serveur AAA sélectionne un mode que le TWAG ne prend pas en charge, le TWAG revient en arrière. L'ordre de retour préfère TSCM, puis SCM, puis MCM. Le TWAG sélectionne le premier mode pris en charge dans cet ordre.



Le TWAG lit le mode sélectionné à partir de l'une de ces clés dans la réponse AAA, dans l'ordre : `twan_connection_mode`, `TWAN-Connection-Mode`, `connection_mode` ou `Connection-Mode`. La valeur peut être un code numérique (`0`/`1`/`2`) ou un atome (`:tscm`/`:scm`/`:mcm`). Une valeur non reconnue par défaut à TSCM.

Sur le chemin STa, le DER STa annonce le mode pris en charge préféré du TWAG dans l'AVP `TWAN-Connection-Mode`. L'analyse DEA STa lit le mode négocié et par défaut à TSCM lorsque le DEA n'en porte aucun.

> **État : le mode négocié n'est pas appliqué sur le chemin de session en direct.** Le client STa (`StaClient`) annonce le mode pris en charge dans le DER et lit le mode négocié à partir du DEA. Le module de mode de connexion est présent et testé unitairement. Cependant, le chemin d'authentification de passage STa n'est pas le chemin d'authentification RADIUS en direct, et le client en direct n'applique jamais le mode négocié : le point d'entrée `store_connection_mode` n'a pas d'appelant sur le chemin d'attachement, et le chemin d'attachement et de données se comporte toujours comme TSCM (IP via DHCP, pas de WLCP, pas de transfert signalé par l'UE ou d'APN). Aujourd'hui, chaque session en direct suit le chemin TSCM. Voir État de mise en œuvre.

Mode de connexion unique transparent (TSCM)

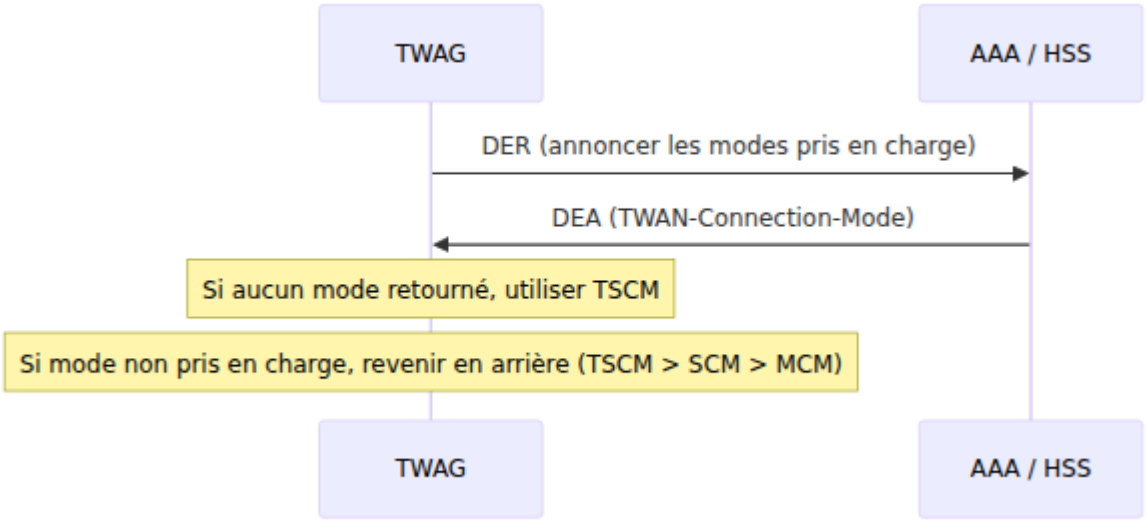
TSCM est le mode par défaut et le comportement que le chemin de session en direct exécute aujourd'hui. L'UE ne signale pas le réseau. TSCM est décrit dans le cycle de vie de session de l'Architecture et est résumé ici.

Dans TSCM :

- L'UE maintient une connexion PDN.
- Le TWAG sélectionne l'APN. L'UE ne peut pas demander un APN.
- L'UE reçoit son adresse IP via DHCP, servie par le TWAG sur le tunnel GRE. Voir Plan utilisateur.
- Il n'y a pas de signalisation WLCP.
- Le transfert avec préservation de l'IP, les porteurs dédiés et la sélection d'APN signalée par l'UE ne sont pas disponibles.

Le TWAG crée automatiquement la session S2a après authentification. Il envoie une demande de création de session au PGW avec le drapeau d'indication de transfert désactivé. Il pré-alloue ensuite l'adresse IP assignée par le PGW dans

son serveur DHCP pour l'adresse MAC de l'UE. Voir S2a GTP-C pour le détail de la création de session et Sessions pour la session résultante et son cycle de vie d'état. La facturation pour la session, lorsqu'elle est activée, suit via Gy. Voir Facturation en ligne (Gy).



Mode de connexion unique (SCM)

SCM maintient une seule connexion PDN par UE, mais l'UE est conscient de S2a. SCM ajoute ces capacités par rapport à TSCM :

- L'UE reçoit son adresse IP via EAP, pas DHCP.
- L'UE peut sélectionner un APN non par défaut.
- L'UE conserve son adresse IP lors du transfert 3GPP vers WLAN et WLAN vers 3GPP.
- Le PGW peut créer des porteurs dédiés.

Le module de mode de connexion rapporte ces capacités SCM. Il classe SCM comme un mode qui délivre l'IP via EAP, prend en charge le transfert, prend en charge la sélection d'APN et prend en charge les porteurs dédiés.

> **État : SCM n'est pas encore un chemin d'exécution distinct.** Le modèle de capacité pour SCM existe, mais le code d'établissement de session ne branche pas encore sur le mode négocié. Un UE que le serveur AAA place dans SCM est toujours servi via le chemin TSCM (PDN auto-créé,

> IP via DHCP). Ne comptez pas sur l'IP fournie par EAP ou la sélection d'APN signalée par l'UE pour SCM dans cette version.

Mode de connexion multiple (MCM) et WLCP

MCM permet à un UE de maintenir plusieurs connexions PDN en même temps. L'UE gère chaque connexion avec le protocole de contrôle WLAN (WLCP), défini dans 3GPP TS 24.244. Après authentification, le TWAG ne crée pas automatiquement une connexion PDN. Au lieu de cela, l'UE envoie des demandes de connexion PDN WLCP. Le TWAG mappe chaque connexion PDN à une session S2a GTP-C séparée vers le PGW.

Transport WLCP

Le TWAG exécute un serveur WLCP sur le port UDP `2156`, le port WLCP standard. Le serveur achemine chaque paquet vers un gestionnaire de session WLCP par UE, basé sur l'identité de l'UE. Le serveur mappe l'IP source de l'UE et le port à l'identité après authentification.

En production, la signalisation WLCP s'exécute sur DTLS selon la section 16.1.4A.3.1 de TS 23.402. Le serveur actuel gère uniquement la couche UDP simple. DTLS n'est pas encore appliqué.

Format de message WLCP

Chaque message WLCP a un en-tête fixe, suivi d'une liste d'éléments d'information (IE).

```
Type de message (1 octet) | Longueur (2 octets) | ID de
transaction (1 octet) | IEs...
```

Chaque IE a ce format :

```
Type d'IE (1 octet) | Longueur d'IE (2 octets) | Valeur d'IE
(variable)
```

Le champ `Longueur` compte le message complet, y compris l'en-tête de 4 octets.

Types de messages WLCP

Code	Message	Direction	Objectif
`0x01`	Demande de connexion PDN	UE vers TWAG	Demander une nouvelle connexion PDN
`0x02`	Acceptation de connexion PDN	TWAG vers UE	Connexion créée, avec IP / DNS / QoS
`0x03`	Rejet de connexion PDN	TWAG vers UE	Connexion refusée, avec une cause
`0x04`	Demande de déconnexion PDN	UE vers TWAG	Libérer une connexion PDN
`0x05`	Acceptation de déconnexion PDN	TWAG vers UE	Connexion libérée
`0x06`	Notification de données descendantes	TWAG vers UE	Informer l'UE des données descendantes en attente

Éléments d'information WLCP

Code	IE	Remarques
`0x01`	APN	Chaîne de nom de point d'accès
`0x02`	Type de PDN	`1` IPv4, `2` IPv6, `3` IPv4v6
`0x03`	PCO	Options de configuration de protocole
`0x04`	Adresse PDN	Adresse IP allouée à l'UE (IPv4, IPv6 ou IPv4v6)
`0x05`	QoS de porteur	Valeurs QCI et MBR / GBR
`0x06`	TFT	Modèle de flux de trafic
`0x07`	Cause	Valeur de cause de résultat
`0x08`	ID de porteur	Identité de porteur EPS (EBI)
`0x09`	Adresse DNS	Une ou plusieurs adresses de serveur DNS

Valeurs de cause WLCP

Code	Cause	Signification
`0x10`	demande_acceptée	Demande acceptée
`0x20`	apn_non_autorisé	APN non autorisé pour cet UE
`0x21`	pas_de_ressources	Pas de ressources disponibles
`0x22`	apn_inconnu	APN inconnu
`0x23`	échec_auth_utilisateur	Échec de l'authentification de l'utilisateur
`0x24`	échec_réseau	Échec du réseau (par exemple échec S2a)
`0x25`	type_de_pdn_non_supporté	Type de PDN demandé non supporté
`0x30`	demande_rejetée	Demande rejetée

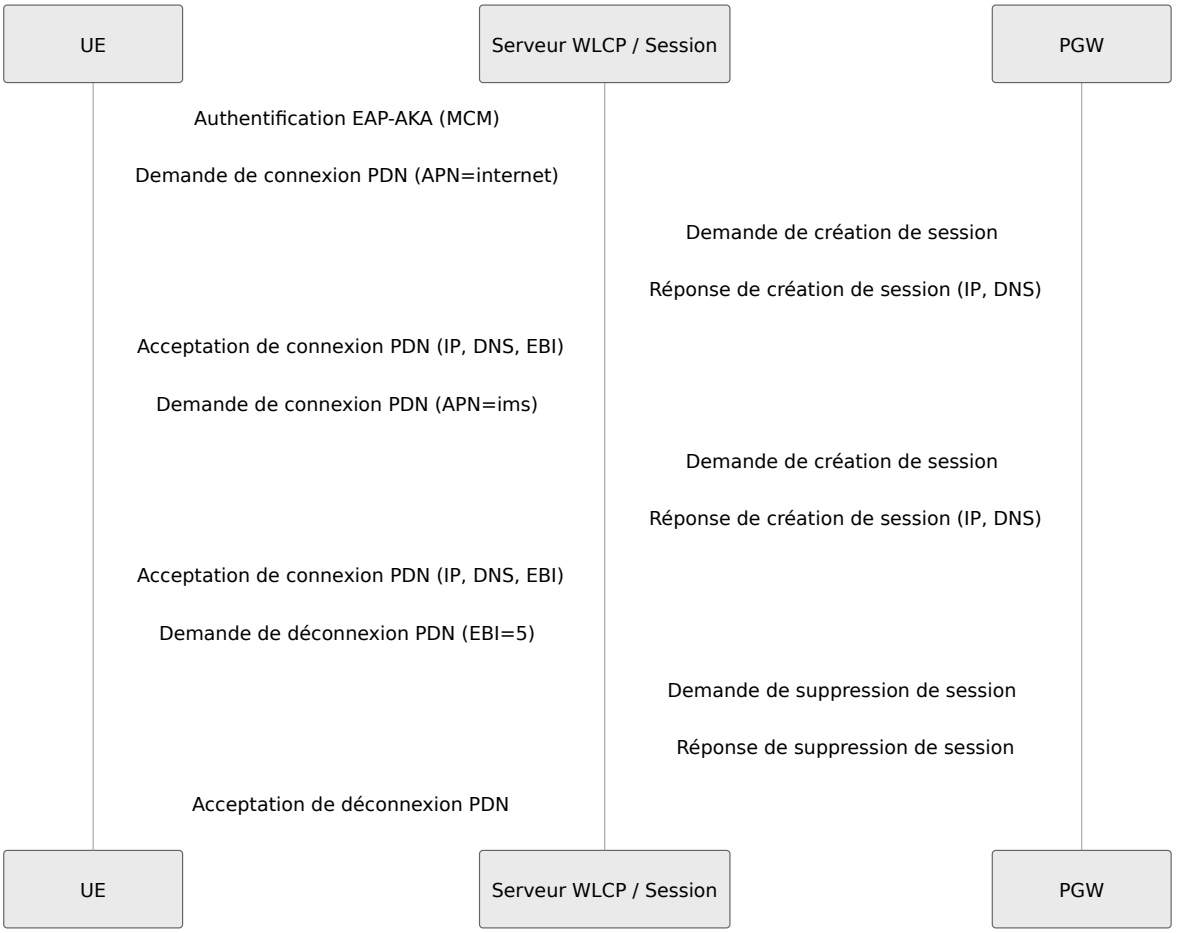
Configuration de la connexion PDN

Lorsqu'une demande de connexion PDN arrive, le gestionnaire de session WLCP effectue ces étapes :

1. Il lit l'APN, le type de PDN et le PCO à partir du message. Si l'APN est absent, il utilise `internet`. Si le type de PDN est absent, il utilise IPv4.
2. Il vérifie que l'APN est autorisé. Voir `allowed_apns` dans Configuration.
3. Si l'UE n'a pas d'IMSI connu, il rejette la demande avec la cause `échec_auth_utilisateur`.
4. Il crée une session S2a GTP-C vers le PGW pour cet APN, avec un type RAT `3` (WLAN) et le drapeau d'indication de transfert désactivé.
5. En cas de succès, il enregistre le tunnel GTP-U et stocke la connexion PDN, indexée sur l'EBI. Il envoie ensuite une acceptation de connexion PDN avec

l'adresse IP de l'UE et, le cas échéant, les serveurs DNS, QoS de porteur et EBI.

6. En cas d'échec S2a, il envoie un rejet de connexion PDN avec la cause `échec_réseau`.



Déconnexion de connexion PDN

Lorsqu'une demande de déconnexion PDN arrive, le gestionnaire trouve la connexion PDN par EBI. Si l'EBI est inconnu, il répond avec un rejet de connexion PDN et la cause `demande_rejetée`. Si l'EBI est connu, il supprime la session S2a GTP-C, désenregistre le tunnel GTP-U, supprime la connexion et répond avec une acceptation de déconnexion PDN. Lorsque l'UE n'a plus de connexions PDN, le gestionnaire de session WLCP s'arrête.

Si le PGW supprime un porteur, le gestionnaire supprime la connexion PDN correspondante et libère le tunnel GTP-U local. Il ne envoie pas de demande de suppression de session dans ce cas, car le PGW a déjà libéré le porteur.

Dérivation de clé WLCP

La clé WLCP `K_WLCP` protège la signalisation WLCP. Elle est dérivée de la clé de session maître (MSK) produite par l'authentification EAP. La dérivation suit la section 6.2 et l'annexe A de 3GPP TS 33.402.

```
K_WLCP = HMAC-SHA-256(MSK, FC | P0 | L0)

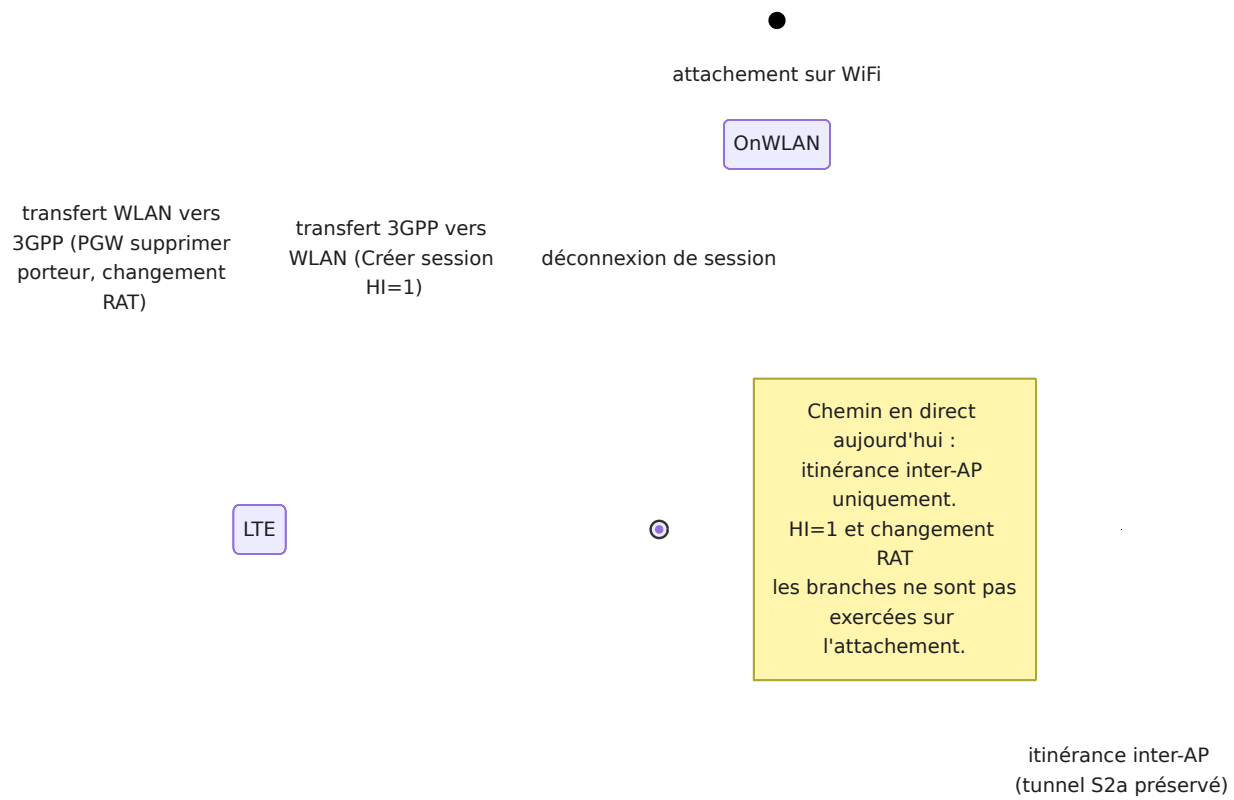
FC = 0x21
P0 = adresse IP du TWAG (4 octets pour IPv4, 16 octets pour IPv6)
L0 = longueur de P0 (2 octets, big-endian)
```

La sortie est de 32 octets (256 bits). Le TWAG lie la clé à sa propre adresse IP via le paramètre `P0`. Une variante simplifiée dérive la clé à partir de la MSK et d'une étiquette fixe lorsque l'IP du TWAG n'est pas encore connue.

> **État : MCM et WLCP sont partiels et non intégrés.** Le codec WLCP, le serveur UDP WLCP, le gestionnaire de session par UE et la dérivation de clé existent tous et sont testés unitairement. Ils ne sont pas démarrés par l'application et ne sont pas pilotés par le flux d'authentification. Les éléments suivants ne sont pas implémentés :
>
> - Protection DTLS pour WLCP. Seul le UDP simple est géré.
> - Gestion des porteurs dédiés à l'intérieur de MCM. Le codec transporte les valeurs QoS et TFT, mais le gestionnaire n'agit pas sur la création de porteur dédié initiée par le PGW.
> - Transfert à l'intérieur de MCM.
> - Le message de notification de données descendantes est défini dans le codec mais n'est pas généré par le gestionnaire de session.
>
> Traitez MCM comme un aperçu. Ne l'activez pas pour les UEs de production.

Transfert et mobilité

Le TWAG modélise trois scénarios de mobilité par TS 23.402. Seule la mobilité inter-AP est active sur le chemin de session en direct aujourd'hui.



Transfert 3GPP vers WLAN

Lorsque l'UE passe de LTE à WiFi, le TWAG doit conserver la session IP existante. Pour ce faire, le TWAG envoie une demande de création de session avec le drapeau d'indication de transfert (HI) activé. Le PGW préserve alors l'adresse IP existante et le contexte de porteur, au lieu d'allouer une nouvelle adresse. Voir S2a GTP-C pour le détail de la création de session et de l'IE d'indication.

La demande de création de session S2a prend en charge le drapeau d'indication de transfert. Le gestionnaire de transfert peut acheminer un événement de transfert-in vers le processus client UE avec le drapeau de transfert activé.

> **État : le chemin de session principal ne définit pas le drapeau de transfert.** L'établissement automatique de session envoie toujours une demande de création de session avec le drapeau d'indication de transfert désactivé. En conséquence, la préservation de l'IP lors du transfert 3GPP vers WLAN n'est pas exercée par le chemin d'attachement normal dans cette version.

Transfert WLAN vers 3GPP

Lorsque l'UE passe de WiFi à LTE, le PGW envoie une demande de suppression de porteur. La valeur de cause indique un changement de RAT. Le TWAG traite ces valeurs de cause comme un transfert de changement de RAT :

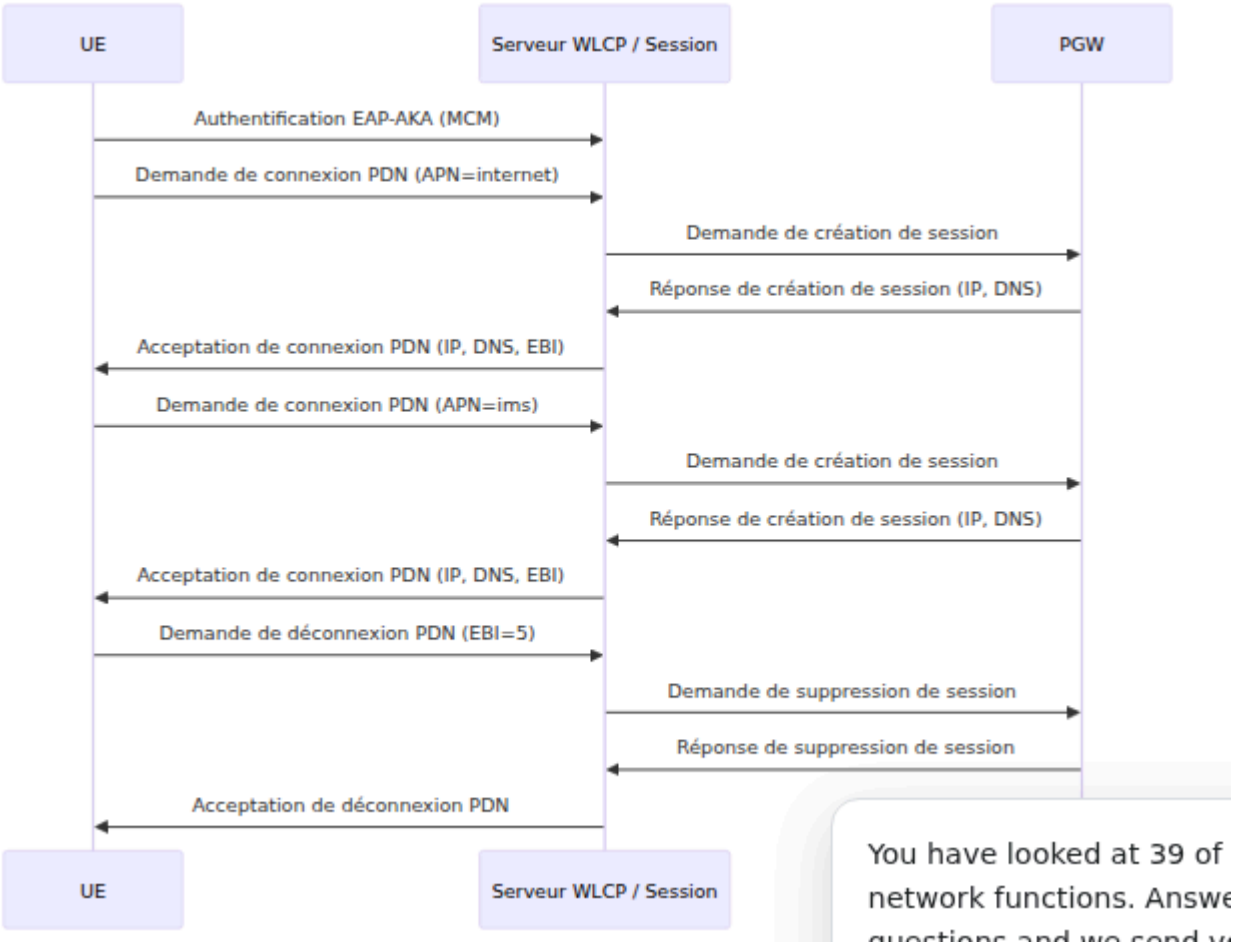
Code de cause	Nom
`4`	RAT changé (3GPP à non-3GPP)
`5`	Désactivation ISR
`10`	Accès changé de non-3GPP à 3GPP

Lors d'une suppression de porteur de changement de RAT, le TWAG nettoie la session WiFi localement. Il ne envoie pas de suppression de session GTP-C, car le PGW a déjà libéré le porteur. L'UE conserve son adresse IP du côté LTE.

> **État : le gestionnaire de suppression de porteur ne branche pas sur la cause aujourd'hui.** Le processus client UE gère une suppression de porteur PGW en exécutant un nettoyage local, puis s'arrête, quelle que soit la cause. La classification de changement de RAT existe comme aide, mais le client n'utilise pas la cause pour choisir un chemin de nettoyage différent.

Mobilité inter-AP

Lorsque l'UE itinère entre des AP WiFi à l'intérieur du même TWAN, le tunnel S2a GTP reste actif. L'UE conserve la même identité, mais arrive avec une adresse NAS-IP différente et un Called-Station-Id. Le TWAG met à jour les données de suivi AP sur le processus client UE et ne déconnecte pas la session. Cela maintient l'adresse IP de l'UE stable lors du changement d'AP.



La mobilité inter-AP est le seul chemin de mobilité que le code de session en direct exécute. Il met à jour les champs du processus client UE pour NAS-IP et Called-Station-Id, et enregistre le nouvel AP dans l'état de suivi AP. Voir Comptabilité par AP pour les données de suivi AP.

Configuration

Modes de connexion

Définissez les modes que le TWAG annonce avec la clé `connection_modes`. La valeur est une liste d'atomes de mode. La valeur par défaut est `[:tscm]`.

```
config :omnitwag,
  # Modes de connexion que ce TWAG annonce au serveur AAA.
  connection_modes: [:tscm, :scm]
```

Paramètre	Type	Requis	Par défaut	Description
`connection_modes`	Liste	Non	`[:tscm]`	Modes que le TWAG annonce. Utilisez `:tscm`, `:scm` et `:mcm`. L'ordre n'a pas d'importance. Les codes annoncés sont triés avant utilisation.

> **Remarque.** L'ensemble annoncé est lu par le module de mode de connexion et le client STa envoie le mode préféré dans le DER STa. Le chemin de session RADIUS en direct n'agit pas encore sur le mode négocié. Voir État de mise en œuvre. Gardez le défaut `[:tscm]` pour la production jusqu'à ce que SCM et MCM soient intégrés.

Serveur WLCP (MCM)

Les paramètres du serveur WLCP se trouvent sous la clé `:wlcp`. Le serveur WLCP n'est pertinent que pour MCM.

```
config :omnitwag,
  wlcp: %{
    enabled: true,
    port: 2156,
    allowed_apns: :all
  }
```

Paramètre	Type	Requis	Par défaut	Description
`enabled`	Booléen	Non	`false`	Activer le serveur WLCP. Laissez `false` sauf si vous testez MCM.
`port`	Entier	Non	`2156`	Port UDP pour WLCP. `2156` est le port WLCP standard.
`allowed_apns`	Atome ou Liste	Non	`:all`	Liste d'autorisation APN pour les demandes PDN WLCP. Utilisez `:all` pour autoriser chaque APN, ou une liste de chaînes APN, par exemple `["internet", "ims"]`. Une demande pour un APN non dans la liste est rejetée avec la cause `apn_non_autorisé`.

> **Remarque.** Le serveur WLCP n'est pas démarré par le superviseur d'application dans cette version. La clé `enabled` est définie pour une utilisation future.

Configuration connexe

Le chemin de configuration PDN MCM réutilise les paramètres GTP pour ses sessions S2a. Voir le résumé de configuration de l'Architecture pour le bloc `gtp`, y compris `pgw_ip`, `local_ip`, `mcc` et `mnc`.

Observabilité

Le serveur WLCP et les gestionnaires de session émettent des événements de télémétrie. Ces événements alimentent le pipeline de métriques. Voir

Métriques pour le catalogue de métriques.

Événement de télémétrie	Émis lorsque
`[:wlcp, :packet, :received]`	Un paquet WLCP arrive. Contient `bytes`.
`[:wlcp, :packet, :malformed]`	Un paquet échoue à décoder.
`[:wlcp, :packet, :unregistered]`	Un paquet arrive d'une adresse source non mappée.
`[:wlcp, :pdn, :request]`	Une demande de connexion PDN est reçue.
`[:wlcp, :pdn, :accept]`	Une connexion PDN est établie.
`[:wlcp, :pdn, :reject]`	Une connexion PDN est rejetée. Contient la cause.
`[:wlcp, :pdn, :disconnect_request]`	Une demande de déconnexion PDN est reçue.
`[:wlcp, :pdn, :disconnect_accept]`	Une connexion PDN est libérée.
`[:twag, :handover, :in]`	Un événement de transfert 3GPP vers WLAN est routé. Contient `source`.
`[:twag, :handover, :out]`	Un changement de RAT WLAN vers 3GPP est géré. Contient `cause`.
`[:twag, :handover, :ap_change]`	Un événement de mobilité inter-AP est géré.

Le serveur WLCP et les gestionnaires de session enregistrent également au niveau d'information. Ils enregistrent chaque demande de connexion PDN, acceptation, rejet et déconnexion, avec l'identité de l'UE, l'APN et l'EBI. Le gestionnaire de transfert enregistre chaque transfert-in, changement de RAT et changement d'AP, avec l'identité de l'UE.

État de mise en œuvre

Ce tableau résume ce qui fonctionne sur le chemin de session en direct, ce qui existe mais n'est pas intégré, et ce qui n'est pas implémenté. Utilisez-le pour planifier autour du comportement réel.

Fonctionnalité	État
TSCM (par défaut, PDN auto-créé, IP via DHCP)	En direct
Mobilité inter-AP (tunnel S2a préservé)	En direct
S2a Création de session avec drapeau d'indication de transfert	Pris en charge dans l'encodeur S2a, mais le chemin d'attachement efface toujours
Logique de négociation de mode de connexion et retour en arrière	Présent en tant que module, testé unitairement
DER STa annonce les modes pris en charge	Câblé (le DER STa transporte le TWAN-Connection-Mode préféré, plus SSID / BSSID)
DEA STa lit le mode sélectionné	Câblé (StaClient analyse le mode négocié, par défaut à TSCM)
Mode négocié appliqué à la session en direct	Non câblé (`store_connection_mode` n'a pas d'appelant sur le chemin d'attachement)
SCM en tant que chemin d'exécution distinct (EAP IP, sélection d'APN)	Non intégré (servi en tant que TSCM)
Codec WLCP (tous les types de messages et IEs)	Présent, testé unitairement
Serveur UDP WLCP et gestionnaire de session par UE	Présent, pas démarré par l'application

Fonctionnalité	État
Dérivation de clé WLCP (`K_WLCP`, FC=0x21)	Présent, testé unitairement
MCM plusieurs connexions PDN concurrentes	Présent dans le gestionnaire, non intégré de bout en bout
Protection DTLS pour WLCP	Non implémenté (UDP simple uniquement)
Gestion des porteurs dédiés dans MCM	Non implémenté
Génération de notification de données descendantes WLCP	Non implémenté (codec uniquement)
Transfert dans MCM	Non implémenté
Préservation de l'IP 3GPP vers WLAN sur le chemin d'attachement	Non exercé (l'attachement efface le drapeau HI)
Nettoyage WLAN vers 3GPP par cause de changement de RAT	Non ramifié (le nettoyage de suppression de porteur est indépendant de la cause)

Gestion des adresses IP (DHCP / DHCPv6 / IPv6 RA / PCO)

Vue d'ensemble

OmniTWAG attribue à chaque UE une adresse IP et les paramètres réseau qui l'accompagnent. Ce sous-système couvre DHCPv4, DHCPv6, la publicité de routeur IPv6 (SLAAC) et la distribution des adresses DNS et P-CSCF. Le sous-système n'invente pas une adresse au hasard. Il délivre une adresse qu'une autre partie du système a déjà choisie. La source dépend du mode par AP.

- En **mode tunnelé (routé à domicile)**, le PGW alloue l'adresse via S2a GTP, et OmniTWAG délivre cette même adresse à l'UE. Ce design suit 3GPP TS 23.402 Section 16.2.
- En **mode NSWO (local breakout)**, il n'y a pas de PGW. OmniTWAG alloue l'adresse lui-même à partir d'un **pool IP** local défini par l'opérateur. Voir Pools IP.

Le mode, la source d'adresse et le pool proviennent tous des informations d'identification de l'AP qui correspondent à la source RADIUS. Voir Informations d'identification RADIUS pour savoir comment un AP est mappé à un `breakout_type`, un `address_source` et un `ip_pool_id`.

Pour les modes de fonctionnement et le chemin de données GRE, voir la référence User Plane. Pour le plan de contrôle GTP-C qui transporte le PAA et le PCO, voir S2a GTP-C. Pour la machine d'état par UE qui gère l'allocation, voir Sessions.

Table des matières

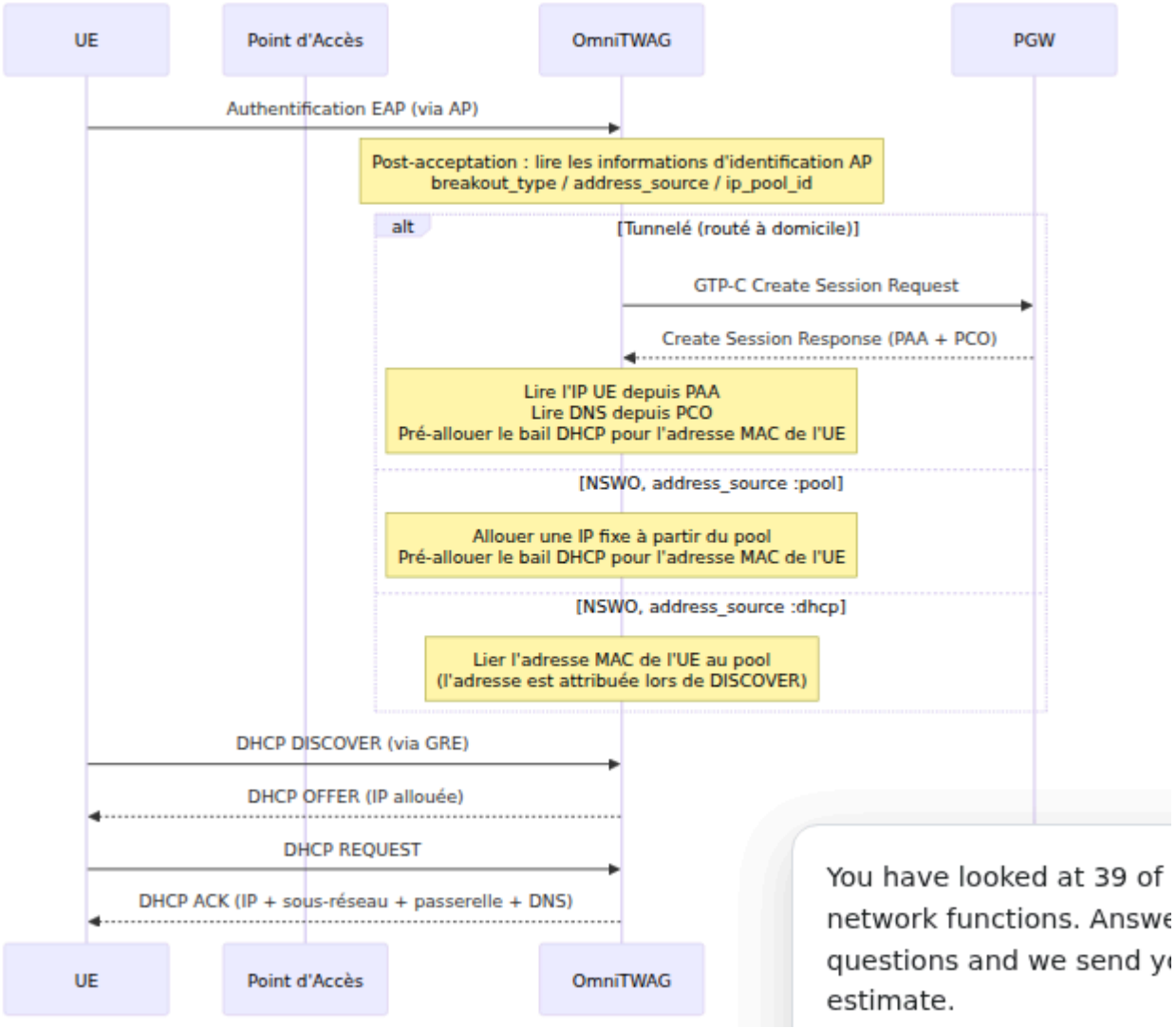
- Vue d'ensemble : qui alloue l'adresse
- Sources d'adresse

- Sessions tunnelées : adresse attribuée par le PGW
- Sessions NSWO : pool et dhcp
- DHCPv4
- DHCPv6
- Publicité de routeur IPv6 (SLAAC)
- Distribution DNS et PCO
- Pools d'adresses et épuisement
- Configuration
- Observabilité
- Dépannage

Vue d'ensemble : qui alloue l'adresse

OmniTWAG exécute un serveur DHCPv4 léger, un serveur DHCPv6 sans état et avec état, et un serveur de publicité de routeur IPv6. Ces serveurs ne scannent pas une plage pour une adresse libre dans le cas tunnelé. Ils délivrent l'adresse que le PGW a déjà allouée. En mode NSWO, le même serveur DHCPv4 sert une adresse que OmniTWAG a prise d'un pool IP local.

Le flux d'adresse commence lorsque l'UE s'authentifie. Après l'Access-Accept, le processus de session par UE exécute une étape d'établissement post-acceptation. Cette étape lit les informations d'identification de l'AP pour décider du mode et alloue ensuite l'adresse par le bon chemin.



Sources d'adresse

Les informations d'identification de l'AP définissent la source d'adresse à travers deux champs :

- `breakout_type` : `:tunneled`, `:nswo`, ou `:local`.
- `address_source` : `:pool` ou `:dhcp`. Cela s'applique uniquement lorsque `breakout_type` est `:nswo`.

Mode	Informations d'identification	Qui choisit l'adresse UE	Comment elle atteint l'UE
Tunnelé (routé à domicile)	`breakout_type::tunneled`	Le PGW, dans la réponse de création de session GTP-C PAA.	Bail DHCPv4 pré-alloué, DHCPv6, ou RA IPv6.
NSWO, fixe	`breakout_type::nswo`, `address_source::pool`	OmniTWAG, à partir du pool `ip_pool_id`, lors de l'établissement.	Bail DHCPv4 pré-alloué.
NSWO, attribué	`breakout_type::nswo`, `address_source::dhcp`	OmniTWAG, à partir du pool `ip_pool_id`, lors du DHCP DISCOVER.	Distribution DHCPv4 à partir du pool lors de DISCOVER.
AP-local	`breakout_type::local`	Le réseau local derrière l'AP.	OmniTWAG n'attribue aucune adresse.

Pour les champs d'informations d'identification et comment une source RADIUS est mappée à une information d'identification, voir Informations d'identification RADIUS. Pour le magasin de pool, voir Pools IP.

Sessions tunnelées : adresse attribuée par le PGW

Pour une session tunnelée, OmniTWAG établit une session S2a avec le PGW avec une demande de création de session GTP-C. Le PGW répond avec une réponse de création de session. La réponse contient deux éléments que ce sous-système utilise :

- L'**Allocation d'Adresse PDN (PAA)**. C'est l'adresse IP de l'UE.
- Les **Options de Configuration de Protocole (PCO)**. Cela contient des serveurs DNS, des adresses P-CSCF, et la MTU du lien.

OmniTWAG lit le PAA et le PCO à partir de la réponse. OmniTWAG pré-alloue ensuite un bail DHCP pour l'adresse MAC de l'UE. Lorsque l'UE envoie une demande DHCP, OmniTWAG répond avec cette adresse pré-allouée. L'UE ne reçoit jamais une adresse que le PGW n'a pas allouée. Voir S2a GTP-C pour l'échange de création de session.

Le type PAA détermine quelle adresse OmniTWAG délivre.

Type PAA	Adresse délivrée à l'UE
`ipv4`	L'adresse IPv4 du PAA.
`ipv6`	L'adresse IPv6 du PAA.
`ipv4v6`	L'adresse IPv4 du PAA.

Si la réponse de création de session n'a pas de PAA, OmniTWAG ne pré-alloue pas de bail. Dans ce cas, l'UE ne reçoit aucune adresse.

Sessions NSWO : pool et dhcp

Une session NSWO n'a pas de PGW, donc OmniTWAG possède l'espace d'adresses. L'adresse provient du pool IP nommé par l'identifiant `ip_pool_id` des informations d'identification. Un pool contient un CIDR IPv4, une passerelle, une liste de serveurs DNS et un temps de bail. Voir Pools IP.

Le champ `address_source` des informations d'identification sélectionne comment l'adresse du pool atteint l'UE.

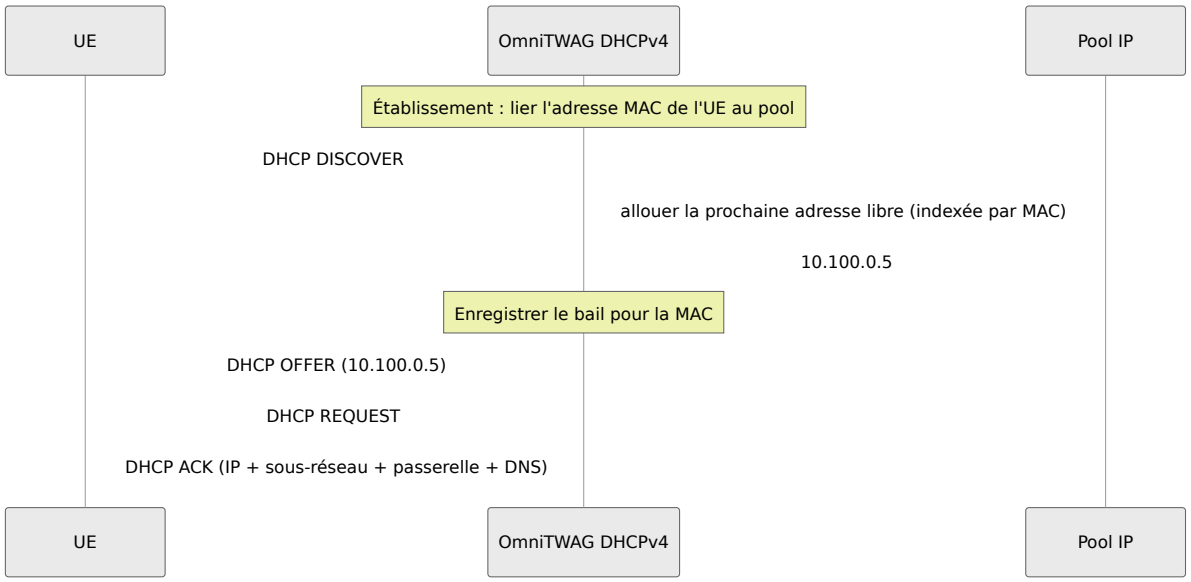
`address_source: :pool` (fixe)

Lors de l'établissement post-acceptation, OmniTWAG alloue une adresse à partir du pool, indexée par l'identité de l'abonné. L'allocation est idempotente,

donc la même identité conserve la même adresse lors de la réétablissement. OmniTWAG pré-alloue un bail DHCPv4 pour l'adresse MAC de l'UE avec cette adresse, la passerelle du pool, les serveurs DNS du pool, et un masque de sous-réseau dérivé du CIDR du pool. Il enregistre également l'UE dans le chemin de données GRE à ce moment. Lorsque l'UE envoie DISCOVER, le serveur propose l'adresse pré-allouée.

`address_source: :dhcp` (attribué lors de DISCOVER)

Lors de l'établissement, OmniTWAG n'alloue pas encore d'adresse. Il **lie** l'adresse MAC de l'UE au pool. Lors du premier DHCP DISCOVER de cette MAC, le serveur DHCPv4 alloue la prochaine adresse libre de la plage du pool, enregistre un bail, et l'offre. C'est le nouveau chemin dynamique d'attribution. L'adresse est choisie au moment de DISCOVER, et non lors de l'établissement.



Lors de la destruction de la session, OmniTWAG libère l'allocation du pool et, pour `:dhcp`, supprime la liaison du pool et le bail dynamique. Si le pool est manquant ou épuisé, ou si l'AP n'a pas d'`ip_pool_id`, OmniTWAG n'attribue aucune adresse et l'UE s'auto-attribue.

DHCPv4

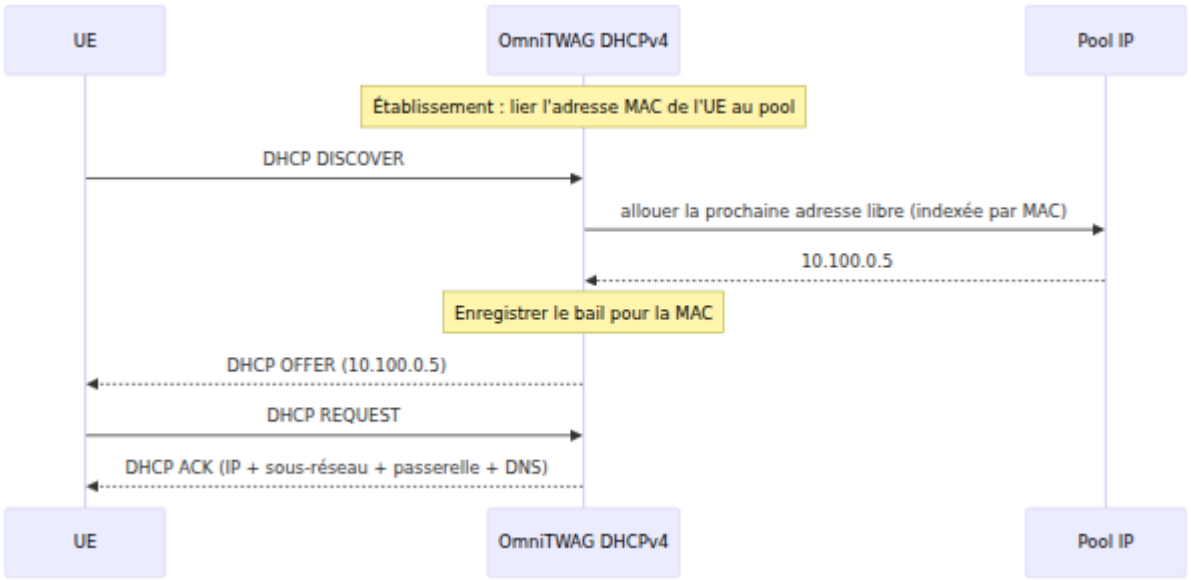
Le serveur DHCPv4 écoute sur le port UDP `67`. Il stocke chaque bail dans une table ETS en mémoire, indexée par l'adresse MAC de l'UE. Un bail atteint la

table par l'un des deux chemins.

- **Pré-alloué.** Le processus de session insère un bail lorsque le PGW alloue une adresse (tunnelée) ou lorsque OmniTWAG alloue une adresse fixe du pool (NSWO `address_source: :pool`).
- **Attribué lors de DISCOVER.** Pour NSWO `address_source: :dhcp`, le processus de session lie plutôt l'adresse MAC de l'UE à un pool. Le serveur conserve une liaison par MAC en mémoire. Lors du premier DISCOVER d'une MAC liée sans bail, le serveur alloue la prochaine adresse libre du pool, enregistre le bail, et l'offre.

Cycle de vie du bail

Pour un bail pré-alloué, le serveur sert l'adresse qui existe déjà dans la table pour la MAC demandante. Pour une MAC liée, le serveur crée le bail lors de DISCOVER en attribuant à partir du pool.



OmniTWAG gère ces types de messages DHCP.

Message	Code type	Comportement d'OmniTWAG
DISCOVER	1	Si un bail existe pour la MAC, OmniTWAG envoie une OFFER. Si aucun bail n'existe mais que la MAC est liée à un pool, OmniTWAG attribue la prochaine adresse libre du pool, enregistre un bail, et envoie une OFFER. Si aucun bail et aucune liaison n'existent, OmniTWAG enregistre un avertissement et n'envoie rien.
REQUEST	3	Si un bail existe et que l'IP demandée (option 50) correspond au bail, ou si la demande n'a pas d'IP demandée, OmniTWAG envoie un ACK. Si l'IP demandée ne correspond pas, OmniTWAG envoie un NAK. Si aucun bail n'existe, OmniTWAG envoie un NAK.
RELEASE	7	OmniTWAG marque le bail comme `libéré`. OmniTWAG conserve l'enregistrement du bail. La connexion PDN reste active, conformément à TS 23.402 §16.1.5.2 NOTE 2.
DECLINE	4	OmniTWAG enregistre un avertissement. OmniTWAG conserve l'enregistrement du bail.
INFORM	8	OmniTWAG envoie un ACK avec uniquement la configuration. L'ACK contient la passerelle, le masque de sous-réseau et le DNS, mais pas d'adresse (`yiaddr` est `0.0.0.0`).

Pour INFORM, OmniTWAG utilise la `default_gateway` et le `default_subnet_mask` de la configuration. OmniTWAG utilise `8.8.8.8` comme serveur DNS dans la réponse INFORM.

Contenu du bail

Chaque bail contient l'IP de l'UE, le masque de sous-réseau, la passerelle, les serveurs DNS, et la durée du bail. Pour une session tunnelée, le processus de session définit l'IP de l'UE et les serveurs DNS à partir de la réponse du PGW, et le masque de sous-réseau, la passerelle, et la durée du bail à partir de la configuration DHCP. Pour une session NSWO, ces valeurs proviennent du pool IP : l'IP de la plage du pool, le masque de sous-réseau du préfixe CIDR du pool, et la passerelle, les serveurs DNS, et la durée du bail de l'enregistrement du pool. Voir Pools IP et Configuration.

Un enregistrement de bail a également un temps `allocated_at` et un temps `expires_at`. Le temps `expires_at` est `allocated_at` plus la durée du bail. OmniTWAG utilise ces valeurs pour rapporter l'état du bail. Un récolteur en arrière-plan s'exécute toutes les 60 secondes et supprime tout bail dont le `expires_at` est passé. Un bail avec une durée de bail de zéro n'a pas de `expires_at` et n'expire jamais. Un bail est également supprimé lorsque la session est détruite.

Options DHCP dans une réponse

OmniTWAG construit chaque OFFER et ACK avec ces options DHCP.

Option	Code	Contenu
Type de message	53	OFFER ou ACK.
Identifiant du serveur	54	L'IP du serveur DHCP.
Masque de sous-réseau	1	Le masque de sous-réseau du bail.
Routeur	3	La passerelle du bail.
Serveur DNS de nom récursif	6	Les serveurs DNS du bail. Omitted if the list is empty.
Temps de bail d'adresse IP	51	La durée du bail en secondes. Omitted if the lease time is zero.
Adresse de diffusion	28	Calculée à partir de l'IP et du masque de sous-réseau. Envoyée uniquement si l'UE liste cette option dans sa liste de demande de paramètres (option 55).

Transport DHCP sur GRE

Un point d'accès qui utilise un tunnel GRE transfère la demande DHCP de l'UE à l'intérieur d'un trame Ethernet encapsulée GRE. Le serveur GRE extrait la charge utile DHCP et la transmet au serveur DHCPv4. OmniTWAG étiquette le paquet avec l'adresse de l'AP source. Lorsque OmniTWAG construit la réponse, il enveloppe la charge utile DHCP dans une trame UDP/IP/Ethernet et renvoie la trame à travers le tunnel GRE vers le même AP. La source est `10.0.0.1:67` et la destination est l'adresse de diffusion sur le port client `68`.

Une demande DHCP qui arrive directement sur le socket UDP (pas via GRE) est répondue sur le socket UDP. Si le port source n'est pas le port client standard

`68`, OmniTWAG répond à l'adresse et au port source. Ce chemin prend en charge les outils de test et les relais. Sinon, OmniTWAG suit RFC 2131 : il utilise l'adresse de diffusion si le drapeau de diffusion est défini ou si `ciaddr` est zéro, et il utilise `ciaddr` si cette adresse est définie.

DHCPv6

Le serveur DHCPv6 écoute sur le port UDP `547` via IPv6. Il prend en charge à la fois un mode sans état et un mode avec état.

- **Mode sans état** répond à une demande d'information (type 11) avec une réponse d'information (type 12). La réponse contient les serveurs DNS. Cela suit RFC 3736 et RFC 3646. Un UE utilise ce mode lorsque il définit le drapeau de configuration autre (O) à partir de son traitement RA.
- **Mode avec état** répond à une demande (type 1) avec une annonce (type 2), et une demande (type 3) avec une réponse (type 7). Ce mode délivre une adresse IPv6 pré-allouée dans une option IA_NA. Cela suit RFC 3315.

Le serveur pré-alloue une adresse avec état par DUID client. Le plan de contrôle GTP insère l'allocation. Le serveur conserve ces allocations dans une table en mémoire, indexée par le DUID client.

Gestion des messages DHCPv6

Message	Type	Comportement d'OmniTWAG
Demande d'information	11	OmniTWAG envoie une réponse d'information avec le DUID du serveur, l'ID client écho, et les serveurs DNS.
Demande	1	OmniTWAG envoie une annonce. Si le DUID client a une allocation, l'annonce contient l'adresse IA_NA. Sinon, l'annonce contient un statut `NoAddrsAvail` (code 2).
Demande	3	OmniTWAG envoie une réponse. La réponse confirme l'adresse IA_NA pour le DUID client.
Autres types	-	OmniTWAG ignore le message.

Chaque réponse écho l'ID de transaction client, conformément à RFC 3315 §15. Chaque réponse qui inclut une adresse allouée définit les temporisateurs DHCPv6 à partir de la durée du bail. T1 est la moitié de la durée du bail. T2 est quatre cinquièmes de la durée du bail.

DUID du serveur

OmniTWAG construit un DUID de serveur de type DUID-EN (Numéro d'Entreprise). Il utilise le numéro d'entreprise 3GPP `10415`. L'identifiant est dérivé du nom du nœud, donc il est stable pour un nœud donné.

Options DHCPv6 dans une réponse

Option	Code	Contenu
Identifiant client	1	L'ID client écho de la demande, si présent.
Identifiant serveur	2	Le DUID du serveur.
Association d'identité pour les adresses non temporaires (IA_NA)	3	L'adresse IPv6 allouée, ou un statut `NoAddrsAvail`. Mode avec état uniquement.
Serveur de noms DNS récursif	23	Les serveurs DNS configurés. Omitted if the list is empty.
Liste de recherche de domaine	24	La liste de recherche configurée. Envoyée uniquement si `dhcpv6_domain_list` est définie.

Publicité de routeur IPv6 (SLAAC)

Le serveur RA IPv6 délivre des préfixes IPv6 alloués par le PGW aux UE avec autoconfiguration d'adresse sans état (SLAAC). Il implémente ICMPv6 Router Advertisement conformément à RFC 4861.

Le plan de contrôle GTP ajoute un préfixe par interface. Le serveur conserve les préfixes en mémoire, regroupés par nom d'interface. Le serveur envoie une publicité de routeur dans deux cas :

1. Sur un minuteur périodique. L'intervalle par défaut est de `30` secondes.
2. À la demande, lorsque le plan de contrôle appelle une RA sur une interface.

Le serveur analyse également les messages de sollicitation de routeur (ICMPv6 type 133).

Contenu de la publicité de routeur

Chaque publicité de routeur contient ces champs et options.

Champ ou option	Valeur
Limite de saut actuelle	À partir de `hop_limit`. Par défaut `64`.
Durée de vie du routeur	À partir de `router_lifetime`. Par défaut `1800` secondes.
Drapeaux M et O	Tous deux `0`. La RA ne signale pas la configuration d'adresse gérée ou d'autres configurations.
Adresse de couche de liaison source (option 1)	La MAC source, si c'est une valeur de 6 octets.
MTU (option 5)	Inclus uniquement si une MTU est fournie.
Informations de préfixe (option 3)	Une par préfixe. Les drapeaux L (sur lien) et A (autonome) sont tous deux définis.
Serveur DNS récursif, RDNSS (option 25)	Les serveurs DNS des préfixes, conformément à RFC 8106. Omitted if there are no DNS servers.

Chaque option d'information de préfixe contient le préfixe, la longueur du préfixe, la durée de vie valide (par défaut `86400` secondes), et la durée de vie préférée (par défaut `14400` secondes). Comme le drapeau A est défini, l'UE forme sa propre adresse IPv6 à partir du préfixe. La durée de vie RDNSS est la plus grande durée de vie valide parmi les préfixes.

Distribution DNS et PCO

OmniTWAG délivre des informations DNS et P-CSCF de plusieurs manières. La source est le PCO du PGW lorsque cela est possible, et la configuration sinon.

PCO du PGW

Le parseur PCO lit les Options de Configuration de Protocole à partir de la réponse de création de session, conformément à 3GPP TS 24.008. Le parseur retourne ces champs.

Champ	ID de protocole PCO	Signification
`dns_ipv4`	`0x000D`	Adresses de serveurs DNS IPv4.
`dns_ipv6`	`0x0003`	Adresses de serveurs DNS IPv6.
`pcscf_ipv4`	`0x000C`	Adresses P-CSCF IPv4.
`pcscf_ipv6`	`0x0001`	Adresses P-CSCF IPv6.
`mtu`	`0x0010`	MTU de lien IPv4.
`ip_alloc_nas`	`0x000A`	Un drapeau. Il est `true` si le PGW signale l'allocation IP via la signalisation NAS.

Le parseur conserve également chaque option analysée dans une liste `raw`. Si le binaire PCO est malformé en cours de route, le parseur retourne les champs qu'il a lus jusqu'à présent. Il ne échoue pas sur toute l'analyse.

OmniTWAG peut également construire une demande PCO pour la demande de création de session. La demande énumère les options que OmniTWAG souhaite que le PGW retourne, par exemple les serveurs DNS IPv4 et la MTU du lien.

Distribution DNS à l'UE

Les serveurs DNS atteignent l'UE par le transport que l'UE utilise.

Transport	Source DNS	Remarques
DHCPv4 (option 6)	Les serveurs DNS IPv4 du PCO.	Si le PCO n'a pas de serveur DNS IPv4, OmniTWAG utilise `8.8.8.8`.
DHCPv6 (option 23)	La liste des serveurs DNS DHCPv6.	Définie à partir du PCO du PGW ou de `dhcpv6_dns_servers`. Par défaut, utilise Google Public DNS IPv6 si non définie.
RA IPv6 (RDNSS, option 25)	Les serveurs DNS IPv6 attachés au préfixe annoncé.	Livré avec le préfixe.

Le bail DHCPv4 prend les serveurs DNS IPv4 à partir de la liste PCO `dns_ipv4`. Si cette liste est vide, OmniTWAG revient à `8.8.8.8` pour le bail.

> **Remarque sur P-CSCF.** Le parseur PCO lit les adresses P-CSCF à partir du PGW. Les chemins de livraison DHCP, DHCPv6 et RA actuels n'ajoutent pas d'option DHCP P-CSCF ou d'option RA. Les valeurs P-CSCF sont analysées et disponibles, mais elles ne sont pas livrées à l'UE par ce sous-système dans le code audité.

Pools d'adresses et épuisement

Le comportement d'épuisement dépend du mode.

- **Tunnelé (routé à domicile).** Il n'y a pas de pool d'adresses local. Le PGW possède l'espace d'adresses. OmniTWAG délivre exactement l'adresse que le PGW alloue pour chaque session. Il n'y a pas de plage locale à épuiser. Si le PGW ne retourne pas de PAA, l'UE ne reçoit aucune adresse, et OmniTWAG enregistre la condition.
- **NSWO.** OmniTWAG alloue à partir d'un pool IP local, donc un pool peut s'épuiser. Un pool est un CIDR IPv4 d'au plus un `/30`. L'alloueur ignore les

adresses réseau et de diffusion et attribue le premier hôte libre. Lorsque chaque hôte est utilisé, l'alloueur retourne `pool_exhausted`. Pour `address_source: :pool`, l'UE s'auto-attribue. Pour `address_source: :dhcp`, le DISCOVER reste sans réponse et OmniTWAG enregistre l'épuisement. Les allocations sont en mémoire et se reconstruisent à mesure que les sessions se rétablissent, de la même manière que les baux DHCP. Voir Pools IP pour la définition et la capacité du pool.

Le helper `Twag.Cidr` effectue les calculs de sous-réseau pour la plateforme. Il analyse les chaînes CIDR IPv4 et IPv6 et teste si une adresse est à l'intérieur d'un sous-réseau. Le sous-système RADIUS utilise ce helper pour l'autorisation d'IP source. Le magasin de pool IP l'utilise pour dériver le masque de sous-réseau du pool, la plage d'hôtes utilisables, et la passerelle par défaut.

Configuration

Les serveurs de gestion IP lisent leur configuration à partir de l'application `:omnitwag`. Les principaux blocs sont `dhcp` et `ipv6_ra`. DHCPv6 lit un petit ensemble de clés de niveau supérieur.

```
config :omnitwag,
  # Serveur DHCPv4
  dhcp: %{
    enabled: true,
    interface: "wlan0",
    server_ip: "10.0.0.1",
    default_lease_time: 3600,
    default_subnet_mask: "255.255.255.0",
    default_gateway: "10.0.0.1"
  },

  # Publicité de routeur IPv6
  ipv6_ra: %{
    enabled: true,
    interface: "wlan0",
    ra_interval: 30,
    hop_limit: 64,
    router_lifetime: 1800
  },

  # DHCPv6 (clés de niveau supérieur optionnelles)
  dhcpv6_dns_servers: ["2001:4860:4860::8888",
"2001:4860:4860::8844"],
  dhcpv6_domain_list: ["example.com"]
```

Paramètres DHCPv4 (`dhcp`)

Paramètre	Type	Requis	Par défaut	Descriptio
`enabled`	Booléen	Non	`true`	Activer le serveur DHCPv4. Définir sur `false` pour l désactiver, p exemple dan un environneme de test.
`interface`	Chaîne	Non	`wlan0`	Interface réseau pour service DHCI Utiliser `gre` pour le mode tunnel GRE.
`server_ip`	Chaîne	Non	`10.0.0.1`	L'identifiant serveur DHC OmniTWAG envoie cette valeur dans l'option DHC 54 et comme source des réponses encapsulées GRE.
`default_lease_time`	Entier	Non	`3600`	Durée du bai en secondes OmniTWAG

Paramètre	Type	Requis	Par défaut	Descriptio
				applique cett valeur à chaque bail pré-alloué.
`default_subnet_mask`	Chaîne	Non	`255.255.255.0`	Masque de sous-réseau pour le bail e pour le calcu de diffusion RFC 2131.
`default_gateway`	Chaîne	Non	`10.0.0.1`	Passerelle pa défaut pour l bail (option DHCP 3).

Paramètres IPv6 RA (`ipv6_ra`)

Paramètre	Type	Requis	Par défaut	Description
`enabled`	Booléen	Non	`true`	Activer le serveur RA IPv6. Définir sur `false` pour le désactiver.
`interface`	Chaîne	Non	`wlan0`	Interface réseau pour les publicités de routeur.
`ra_interval`	Entier	Non	`30`	Intervalle en secondes entre les publicités de routeur périodiques.
`hop_limit`	Entier	Non	`64`	Champ Limite de Saut actuel dans la publicité de routeur.
`router_lifetime`	Entier	Non	`1800`	Durée de vie du routeur en secondes dans la publicité de routeur.

Paramètres DHCPv6 (niveau supérieur)

Paramètre	Type	Requis	Par défaut	Des
`dhcpv6_dns_servers`	Liste de Chaînes	Non	DNS Public Google IPv6 (`2001:4860:4860::8888`, `2001:4860:4860::8844`)	Serv DNS pour répo DHC PCO peut égal défir liste l'exé
`dhcpv6_domain_list`	Liste de Chaînes	Non	(non défini)	Liste rech DNS répo DHC (opti Si nc défir Omn ome l'opt

Observabilité

Interface Web

Le panneau de contrôle a une page **Pool DHCP** à `/dhcp`. La page liste chaque bail DHCPv4 avec l'adresse MAC de l'UE, l'adresse IP, l'heure de début du bail, l'heure d'expiration du bail, et le statut. Le statut est `Actif` ou `Expiré`. La page

affiche un compte à rebours en direct pour chaque bail actif. Sélectionnez une ligne de bail pour voir les détails complets du bail : masque de sous-réseau, passerelle, serveurs DNS, durée du bail, et horodatages. La page se rafraîchit toutes les secondes. Si le serveur DHCPv4 n'est pas en cours d'exécution, la page affiche `Serveur non en cours d'exécution`.

API REST

L'API REST sert les données de bail DHCPv4 sous le tag `DHCP`.

Méthode	Chemin	Description
GET	`/dhcp` (index)	Lister tous les baux DHCPv4 avec un compte résumé.
GET	`/dhcp/{id}` (show)	Obtenir un bail par adresse MAC. L'`id` est la MAC séparée par des deux-points.

La réponse de l'index a un drapeau `enabled`, la liste `leases`, un total `count`, un compte `active`, et un compte `expired`. Si le serveur DHCPv4 n'est pas en cours d'exécution, l'index retourne `enabled: false` avec une liste vide. Le point de terminaison show retourne `404` si aucun bail ne correspond à la MAC, et `503` si le serveur n'est pas en cours d'exécution.

Chaque objet de bail a ces champs.

Champ	Description
`mac`	Adresse MAC de l'UE, séparée par des deux-points.
`ip`	Adresse IPv4 attribuée.
`subnet_mask`	Masque de sous-réseau du bail.
`gateway`	Passerelle du bail.
`dns_servers`	Liste des serveurs DNS.
`lease_time`	Durée du bail en secondes.
`allocated_at`	Heure de début du bail, en secondes depuis l'époque Unix.
`expires_at`	Heure d'expiration du bail, en secondes depuis l'époque Unix.
`status`	`actif` ou `expiré`.
`time_remaining`	Secondes jusqu'à l'expiration, ou `0` si expiré.

Événements de télémétrie

Les serveurs de gestion IP émettent des événements de télémétrie. Un opérateur peut attacher ces événements à un exportateur de métriques. Voir la Référence des métriques pour le point de terminaison Prometheus.

Événement	Source	Signification
`[:dhcp, :lease, :allocate]`	DHCPv4	Un bail a été pré-alloué pour une MAC.
`[:dhcp, :lease, :deallocate]`	DHCPv4	Un bail a été supprimé lors de la destruction.
`[:dhcp, :lease, :reap]`	DHCPv4	Un bail expiré a été supprimé par le récolteur en arrière-plan.
`[:dhcp, :discover]`	DHCPv4	Un DISCOVER a été répondu par une OFFER d'un bail existant.
`[:dhcp, :discover, :pool_allocated]`	DHCPv4	Un DISCOVER d'une MAC liée à un pool a été répondu par l'attribution d'une nouvelle adresse de pool (NSWO `:dhcp`).
`[:dhcp, :discover, :no_allocation]`	DHCPv4	Un DISCOVER est arrivé pour une MAC sans bail et sans liaison au pool.
`[:dhcp, :ack]`	DHCPv4	Une REQUEST a été répondue par un ACK.
`[:dhcp, :nak]`	DHCPv4	Une REQUEST a été répondue par un NAK.
`[:dhcp, :release]`	DHCPv4	Un client a libéré son bail.
`[:dhcp, :decline]`	DHCPv4	Un client a refusé son bail.
`[:dhcp, :inform]`	DHCPv4	Un INFORM a été répondu.
`[:dhcp, :packet, :malformed]`	DHCPv4	Un paquet DHCP malformé a été reçu.

Événement	Source	Signification
`[:dhcpv6, :info_request, :received]`	DHCPv6	Une demande d'information a été reçue.
`[:dhcpv6, :solicit, :received]`	DHCPv6	Une demande a été reçue.
`[:dhcpv6, :request, :received]`	DHCPv6	Une demande a été reçue.
`[:dhcpv6, :info_reply, :sent]`	DHCPv6	Une réponse d'information a été envoyée.
`[:dhcpv6, :advertise, :sent]`	DHCPv6	Une annonce a été envoyée.
`[:dhcpv6, :reply, :sent]`	DHCPv6	Une réponse a été envoyée.
`[:ipv6_ra, :prefix, :add]`	IPv6 RA	Un préfixe a été ajouté à une interface.
`[:ipv6_ra, :prefix, :remove]`	IPv6 RA	Un préfixe a été supprimé d'une interface.
`[:ipv6_ra, :advertisement, :sent]`	IPv6 RA	Une publicité de routeur a été envoyée.

Dépannage

L'UE ne reçoit pas d'adresse IP

Symptômes : L'UE s'authentifie mais ne reçoit pas d'adresse IPv4. La page Pool DHCP montre aucun bail pour l'adresse MAC de l'UE. L'événement `[:dhcp, :discover, :no_allocation]` se déclenche.

Causes possibles (tunnelées) :

- La réponse de création de session GTP n'avait pas de PAA, donc OmniTWAG n'a pas pré-alloué de bail.
- La session GTP-C ne s'est pas établie, donc aucun bail n'a été créé.

Causes possibles (NSWO) :

- Les informations d'identification de l'AP n'ont pas d'`ip_pool_id`, donc OmniTWAG n'attribue aucune adresse.
- Le pool est épuisé, donc aucune adresse libre n'a pu être attribuée.
- Pour `address_source: :dhcp`, le DISCOVER est arrivé avant que la MAC soit liée au pool, ou la liaison n'a pas été exécutée car la session n'avait pas encore de MAC.

Cause commune :

- Le serveur DHCPv4 n'est pas en cours d'exécution.

Résolution :

1. Vérifiez les journaux d'OmniTWAG pour le chemin d'établissement (tunnelé contre NSWO) et l'IP de l'UE.
2. Pour les sessions tunnelées, confirmez que le PGW retourne un PAA pour l'APN.
3. Pour les sessions NSWO, confirmez que les informations d'identification de l'AP nomment un `ip_pool_id` valide et que le pool a une capacité libre. Voir Pools IP et Informations d'identification RADIUS.
4. Confirmez que la page Pool DHCP montre le serveur DHCPv4 comme en cours d'exécution.

L'UE reçoit une adresse mais pas de DNS

Symptômes : L'UE a une adresse IPv4 mais ne peut pas résoudre de noms.

Causes possibles :

- Le PCO du PGW n'a pas retourné de serveur DNS IPv4. Dans ce cas, OmniTWAG utilise `8.8.8.8`.
- L'UE ne peut pas atteindre le serveur DNS que le PGW a fourni.

Résolution :

1. Lisez les serveurs DNS du bail sur la page Pool DHCP ou via l'API REST.
2. Confirmez que le PCO du PGW retourne le serveur DNS attendu.
3. Confirmez que le serveur DNS est accessible depuis le chemin de données de l'UE.

L'UE reçoit un NAK

Symptômes : L'UE envoie une REQUEST et reçoit un NAK. L'événement `[:dhcp, :nak]` se déclenche.

Causes possibles :

- L'UE a demandé une IP (option 50) qui ne correspond pas à son bail pré-alloué.
- Le bail a été supprimé, par exemple après une destruction de session.

Résolution :

1. Confirmez qu'un bail existe pour l'adresse MAC de l'UE.
2. Confirmez que l'UE demande la même IP que celle que le PGW ou le pool a allouée.
3. Si la session a été détruite, laissez l'UE redémarrer l'échange DHCP après qu'il se soit ré-authentifié.

Pages connexes

- Vue d'ensemble
- Pools IP
- Informations d'identification RADIUS
- Sessions
- S2a GTP-C
- User Plane
- Chargement en ligne (Gy)

Vue d'ensemble

Pools IP

Vue d'ensemble

OmniTWAG attribue des adresses IPv4 aux sessions NSWO (Non-Seamless WLAN Offload) à partir d'un **pool IP**. NSWO est une sortie locale ancrée par TWAG : OmniTWAG termine le trafic utilisateur et le sort localement. Un pool IP définit un bloc CIDR IPv4, une passerelle, des serveurs DNS et un temps de location. L'opérateur ajoute, modifie et supprime des pools à la demande via l'API REST ou l'interface web du panneau de contrôle.

Les définitions de pool sont persistantes. OmniTWAG les stocke dans un fichier DETS, de sorte que les pools survivent à un redémarrage. Les allocations d'adresses sont en mémoire. OmniTWAG reconstruit les allocations au fur et à mesure que les sessions se rétablissent après un redémarrage.

Table des matières

- Concepts
- Comment une session obtient une adresse
- Allocation d'adresses
- Modèle de pool
- API REST
- Interface web
- Dépannage

Concepts

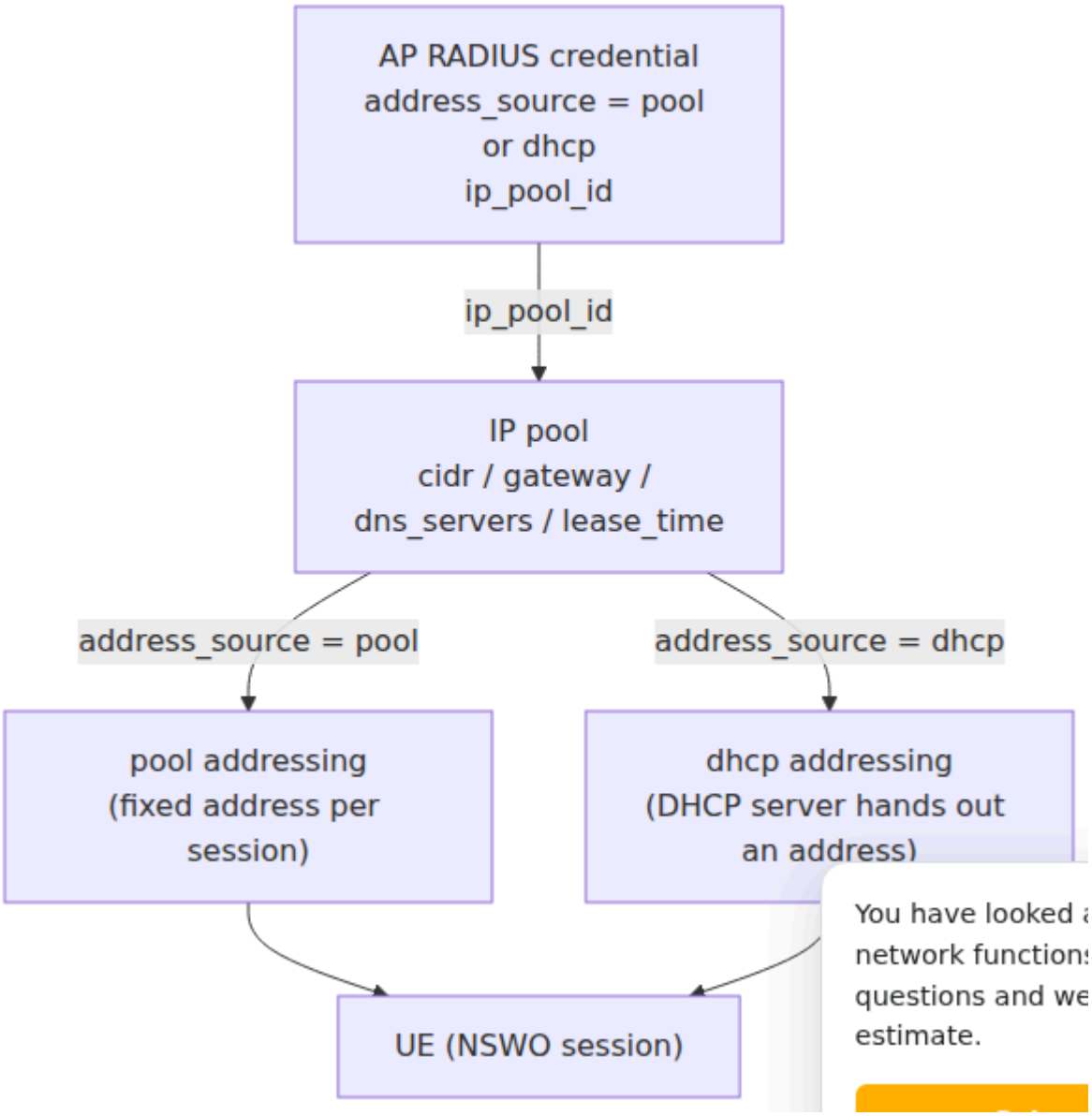
Un **pool** est un ensemble d'adresses provenant d'un bloc CIDR IPv4. Chaque pool a une passerelle, une liste de serveurs DNS et un temps de location.

Un pool IP est lié à un point d'accès via les informations d'identification RADIUS de l'AP. L'information d'identification a un champ `ip_pool_id`. Ce champ contient l'`id` du pool. Voir Secrets partagés RADIUS pour le modèle

d'information d'identification. Les sessions NSWO de ce point d'accès sont adressées à partir du pool lié.

Chaque information d'identification a également un champ `address_source`. Ce champ sélectionne comment OmniTWAG attribue l'adresse à l'UE (équipement utilisateur). Il existe deux sources d'adresses :

Source d'adresse	Comportement
`pool`	OmniTWAG pré-alloue une adresse fixe par session à partir du pool. L'adresse est choisie lors de la configuration de la session.
`dhcp`	Le serveur DHCP d'OmniTWAG attribue une adresse à partir de la plage du pool en réponse à un DHCP DISCOVER de l'UE.

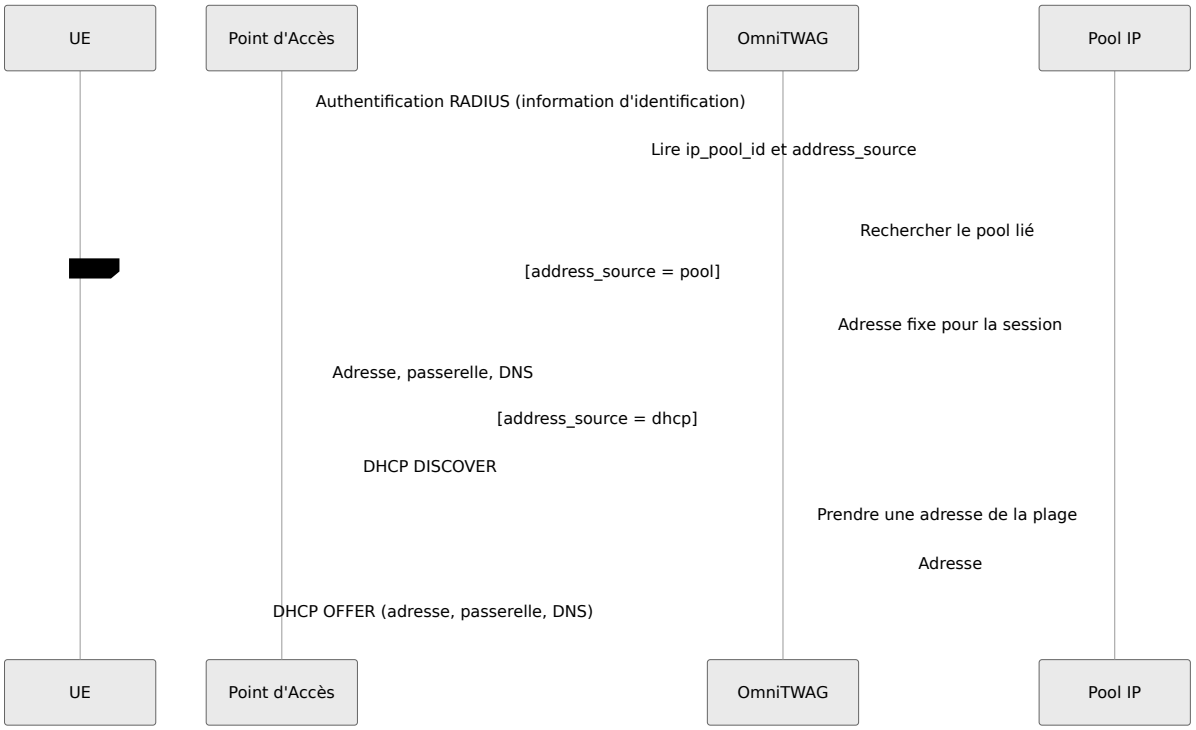


Comment une session obtient une adresse

OmniTWAG adresse chaque session NSWO à partir du pool qui est lié à l'information d'identification de l'AP. Les étapes sont :

1. Le point d'accès s'authentifie via ses informations d'identification RADIUS.
2. OmniTWAG lit le champ `ip_pool_id` sur l'information d'identification pour trouver le pool.
3. OmniTWAG lit le champ `address_source` sur l'information d'identification.

4. Si `address_source` est `pool`, OmniTWAG pré-alloue une adresse fixe pour la session à partir du pool.
5. Si `address_source` est `dhcp`, OmniTWAG attend un DHCP DISCOVER de l'UE. Le serveur DHCP attribue alors une adresse à partir de la plage du pool.
6. OmniTWAG donne à l'UE l'adresse, la passerelle et les serveurs DNS du pool.



Allocation d'adresses

OmniTWAG alloue des hôtes à partir du CIDR du pool. L'allocation ignore l'adresse réseau et l'adresse de diffusion. Le premier hôte utilisable est l'adresse réseau plus un.

Le préfixe CIDR du pool doit être `/30` ou plus grand pour avoir des hôtes utilisables. Un bloc `/30` a 2 hôtes utilisables. Un préfixe plus petit a moins d'hôtes utilisables, donc OmniTWAG le rejette.

Lorsque chaque hôte utilisable est alloué, OmniTWAG retourne `pool_exhausted`. Aucune autre session du point d'accès lié ne peut obtenir une adresse jusqu'à ce qu'une allocation soit libérée.

Le champ `allocated` sur le pool contient le nombre actuel d'allocations. OmniTWAG calcule ce nombre à partir des allocations en mémoire.

Modèle de pool

Chaque pool a ces champs.

Champ	Type	Description
`id`	String	Identifiant unique (UUID). Le serveur définit cette valeur.
`name`	String	Étiquette lisible par l'homme, par exemple `WiFi invité de Melbourne`.
`cidr`	String	Bloc CIDR IPv4 pour le pool, par exemple `10.60.0.0/24`. Le préfixe doit être `/30` ou plus grand pour avoir des hôtes utilisables.
`gateway`	String	Adresse de passerelle IPv4 donnée à l'UE. Par défaut, c'est le premier hôte utilisable du CIDR si omis.
`dns_servers`	Liste de String	Adresses de serveurs DNS IPv4 données à l'UE.
`lease_time`	Integer	Temps de location en secondes. Par défaut, c'est `3600`.
`default`	Boolean	Lorsque `true`, ce pool est la plage par défaut DHCP. Le serveur DHCP attribue à partir de celui-ci pour tout client sans location pré-allouée et sans liaison de pool. Un seul pool peut être par défaut à la fois. Par défaut, c'est `false`.
`allocated`	Integer	Nombre d'allocations actuelles. Le serveur calcule cette valeur.
`inserted_at`	Integer	Temps de création du pool, en millisecondes depuis l'époque Unix. Le serveur définit cette valeur.

Champ	Type	Description
`updated_at`	Integer	Temps de dernière modification du pool, en millisecondes depuis l'époque Unix. Le serveur définit cette valeur.

L'opérateur définit `name`, `cidr`, `gateway`, `dns_servers` et `lease_time`. Le serveur définit `id`, `allocated`, `inserted_at` et `updated_at`.

API REST

L'API REST d'OmniTWAG sert ces points de terminaison via HTTPS sur le port API configuré (`8444` par défaut). Le chemin de base est `/ip_pools`.

Méthode	Chemin	Action	Description
GET	`/ip_pools`	index	Lister tous les pools. Le résultat est paginé et peut être recherché par `name` ou `cidr`.
GET	`/ip_pools/{id}`	show	Obtenir un pool par `id`.
POST	`/ip_pools`	create	Créer un pool.
PUT	`/ip_pools/{id}`	update	Modifier un pool.
DELETE	`/ip_pools/{id}`	delete	Supprimer un pool.

Corps de la requête

`POST` et `PUT` acceptent un corps JSON avec ces champs.

Champ	Type	Requis	Description
`name`	String	Non	Étiquette lisible par l'homme.
`cidr`	String	Oui	Bloc CIDR IPv4. Le préfixe doit être `/30` ou plus grand.
`gateway`	String	Non	Adresse de passerelle IPv4. Par défaut, c'est le premier hôte utilisable du CIDR si omis.
`dns_servers`	Liste de String	Non	Adresses de serveurs DNS IPv4.
`lease_time`	Integer	Non	Temps de location en secondes. Par défaut, c'est `3600`.
`default`	Boolean	Non	Marquer ce pool comme la plage par défaut DHCP. Le définir efface le drapeau sur tout autre pool.

`PUT` ne modifie que les champs dans le corps. Les champs qui ne sont pas dans le corps conservent leur valeur actuelle.

Réponses

Statut	Signification
`200 OK`	La requête a réussi (index, show, update, delete).
`201 Created`	Le pool a été créé.
`404 Not Found`	Aucun pool n'a l'`id` donné.
`422 Unprocessable Entity`	Le `cidr` n'est pas un bloc CIDR IPv4 valide, ou le préfixe est trop petit pour avoir des hôtes utilisables.

Exemple : lister les pools

Requête :

```
GET /ip_pools
```

Réponse :


```
{
  "count": 1,
  "ip_pools": [
    {
      "id": "0d1e2f3a-4b5c-6d7e-8f90-1a2b3c4d5e6f",
      "name": "WiFi invité de Melbourne",
      "cidr": "10.60.0.0/24",
      "gateway": "10.60.0.1",
      "dns_servers": ["8.8.8.8", "8.8.4.4"],
      "lease_time": 3600,
      "allocated": 12,
      "inserted_at": 1734512400000,
      "updated_at": 1734512400000
    }
  ]
}
```


Exemple : créer un pool

Requête :

```
POST /ip_pools
```

```
{
  "name": "WiFi invité de Melbourne",
  "cidr": "10.60.0.0/24",
  "gateway": "10.60.0.1",
  "dns_servers": ["8.8.8.8", "8.8.4.4"],
  "lease_time": 3600
}
```

Une requête réussie retourne `201 Created` avec le nouveau pool, y compris son `id`. Si `gateway` est omis, le pool utilise le premier hôte utilisable du CIDR (`10.60.0.1` dans cet exemple).

Exemple : un CIDR trop petit

Requête :

```
{
  "name": "Pool cassé",
  "cidr": "10.60.0.0/31"
}
```

Réponse (`422 Unprocessable Entity`) :

```
{
  "error": "cidr trop petit : 10.60.0.0/31 n'a pas d'hôtes
utilisables"
}
```

Pool par défaut

Un pool peut être la plage par défaut DHCP. Le serveur DHCP attribue une adresse à partir du pool par défaut pour tout client qui n'a pas de location pré-allouée et pas de liaison de pool explicite. Utilisez le pool par défaut pour donner une plage définie par l'API au service DHCP de base, pas seulement aux sessions NSWO.

Définissez le par défaut avec le champ `default`, ou la case à cocher sur la page web. Un seul pool peut être par défaut à la fois. Lorsque vous marquez un pool comme par défaut, OmniTWAG efface le drapeau sur le pool par défaut précédent.

Chaque location active enregistre sa source. Voir la vue Baux IP : un bail montre `PGW (tunnélisé)`, un `pool NSWO` nommé, ou le `pool par défaut`.

Interface web

Le panneau de contrôle a une page d'administration **Pools IP**. La page liste chaque pool avec son nom, CIDR, passerelle et nombre alloué. La page dispose de contrôles pour ajouter, modifier et supprimer un pool.

L'ancienne page **Pool DHCP** est maintenant étiquetée **Baux IP**. La page des Baux IP montre les baux actifs.

Dépannage

Un pool est épuisé

Symptômes : De nouvelles sessions NSWO d'un point d'accès ne peuvent pas obtenir d'adresse. OmniTWAG retourne `pool_exhausted`.

Causes possibles :

- Chaque hôte utilisable dans le CIDR du pool est alloué.
- Le CIDR du pool est trop petit pour le nombre de sessions actives.

Résolution :

1. Vérifiez le nombre `allocated` par rapport au nombre d'hôtes utilisables dans le CIDR.
2. Élargissez le CIDR du pool pour ajouter plus d'hôtes utilisables, ou libérez des allocations obsolètes en mettant fin aux sessions inactives.

3. Confirmez que le temps de location n'est pas si long que les adresses libérées restent allouées.

Un point d'accès n'utilise pas de pool

Symptômes : Les sessions d'un point d'accès ne sont pas adressées à partir d'un pool IP.

Causes possibles :

- Le point d'accès utilise une sortie locale appartenant au point d'accès, pas un NSWO ancré par TWAG. Un point d'accès à sortie locale appartenant à l'AP adresse ses propres clients et n'utilise pas un pool OmniTWAG.
- L'information d'identification de l'AP n'a pas de `ip_pool_id`, donc aucun pool n'est lié.

Résolution :

1. Confirmez que le point d'accès utilise NSWO ancré par TWAG. Seules les sessions NSWO utilisent un pool.
2. Si le point d'accès doit utiliser un pool, définissez `ip_pool_id` sur ses informations d'identification RADIUS à l'`id` du pool. Voir Secrets partagés RADIUS.

L'UE ne reçoit pas d'adresse

Symptômes : L'UE se connecte mais ne reçoit pas d'adresse IPv4.

Causes possibles :

- L'`address_source` de l'information d'identification ne correspond pas à la façon dont l'UE demande une adresse. Avec `dhcp`, l'UE doit envoyer un DHCP DISCOVER. Avec `pool`, OmniTWAG pré-alloue l'adresse.
- Le pool lié est épuisé.
- Le `ip_pool_id` sur l'information d'identification pointe vers un pool qui n'existe pas.

Résolution :

1. Confirmez que l'`address_source` sur l'information d'identification correspond au comportement de l'UE.
2. Confirmez que le pool lié a des adresses libres. Vérifiez le nombre `allocated`.
3. Confirmez que le `ip_pool_id` sur l'information d'identification pointe vers un pool valide.

Vue d'ensemble

Référence des Métriques TWAG

Vue d'ensemble

TWAG expose des métriques Prometheus sur le port **9568** (défini par `prometheus.port`). Les métriques de type Gauge sont rafraîchies par un poller toutes les 5 secondes. Les métriques de type Counter sont émises par le chemin de code qui les déclenche.

Table des Matières

- Point de terminaison des métriques
- Métriques d'authentification RADIUS
- Métriques d'autorisation et de limitation de taux RADIUS
- Métriques de comptabilité RADIUS
- Métriques EAP
- Gauges Client, AP et Session
- Métriques de débit de données
- Métriques de session et de chemin de données TWAG
- Métriques de facturation en ligne Gy
- Métriques de diamètre STa
- Métrique de message de diamètre
- Métriques NSWO
- Métriques DHCP
- Métriques DHCPv6 et RA IPv6
- Métriques WLCP
- Métriques Erlang VM
- Exemples de requêtes Grafana
- Règles d'alerte
- Pages connexes

Point de terminaison des métriques

```
http://<twag-host>:9568/metrics
```

Prometheus aplatit chaque nom d'événement en un nom de métrique en remplaçant les points par des underscores. Par exemple, l'événement `[:radius, :access, :request, :count]` est exposé sous le nom `radius_access_request_count`.

Métriques d'authentification RADIUS

Ces compteurs suivent les flux de requêtes/réponses RADIUS Access-Request par RFC 2865.

Métrique	Type	Étiquettes	Description
`radius_access_request_count`	Counter	none	Paquets RADIUS Access-Request reçus.
`radius_access_accept_count`	Counter	none	Paquets RADIUS Access-Accept envoyés (authentification réussie).
`radius_access_reject_count`	Counter	none	Paquets RADIUS Access-Reject envoyés (authentification échouée).
`radius_access_challenge_count`	Counter	none	Paquets RADIUS Access-Challenge envoyés (échange EAP en cours).

Exemples de requêtes :

```
# Taux de requêtes par seconde
rate(radius_access_request_count[5m])

# Taux de défis (négociations EAP actives)
rate(radius_access_challenge_count[5m])

# Taux de succès d'authentification
sum(rate(radius_access_accept_count[5m])) /
  (sum(rate(radius_access_accept_count[5m])) +
sum(rate(radius_access_reject_count[5m])))
```

Métriques d'autorisation et de limitation de taux RADIUS

Ces compteurs suivent les paquets qui sont abandonnés ou signalés avant ou pendant l'authentification. Voir Rejets RADIUS et Secrets partagés RADIUS.

Métrique	Type	Étiquettes	
`radius_auth_unauthorized_source_count`	Counter	`source`	Pac qui sou (AP est
`radius_auth_bad_secret_count`	Counter	`source`, `credential_id`	Pac cor ma Me (ma
`radius_access_rate_limited_count`	Counter	`nas_ip`	Rec aba NA `ra` dan

Exemples de requêtes :

```
# AP inconnus apparaissant (ajouter un credential pour ceux-ci)
increase(radius_auth_unauthorized_source_count[1h])

# AP avec un mauvais secret partagé
sum by (source) (increase(radius_auth_bad_secret_count[1h]))

# Appareils NAS limités par le taux
sum by (nas_ip) (increase(radius_access_rate_limited_count[15m]))
```

Métriques de comptabilité RADIUS

Suit les messages de comptabilité RADIUS par RFC 2866.

radius_accounting_request_count

Type : Counter

Étiquette	Description
`status_type`	Type de statut de comptabilité (voir ci-dessous).

Valeurs de type de statut : `start`, `stop`, `interim_update`, `accounting_on`, `accounting_off`.

Exemples de requêtes :

```
# Début de session par minute
rate(radius_accounting_request_count{status_type="start"}[5m]) *
60

# Arrêt de session par minute
rate(radius_accounting_request_count{status_type="stop"}[5m]) * 60
```

Métriques EAP

Ces compteurs suivent les phases EAP-AKA par RFC 4187, plus EAP-SIM, EAP ré-authentification (ERP), et échecs EAP génériques.

M��trique	Type	Description
`eap_aka_identity_count`	Counter	Échanges d'identité EAP-AKA (Phase 1).
`eap_aka_challenge_count`	Counter	Échanges de défi EAP-AKA (Phase 2).
`eap_aka_sync_failure_count`	Counter	Échecs de synchronisation EAP-AKA (SQN hors synchronisation, déclenche une resynchronisation).
`eap_aka_auth_success_count`	Counter	Authentifications EAP-AKA réussies.
`eap_aka_auth_reject_count`	Counter	Authentifications EAP-AKA rejetées (UE n'a pas réussi à authentifier le réseau / échec AUTN).
`eap_aka_client_error_count`	Counter	Notifications d'erreur client EAP-AKA reçues de l'UE.
`eap_auth_rejected_count`	Counter	Refus d'authentification attendus (par exemple HSS USER_UNKNOWN) qui rejettent l'UE proprement.
`eap_process_crash_count`	Counter	Erreurs inattendues lors du traitement d'un message EAP (UE rejetée, session préservée).
`eap_sim_start_count`	Counter	Échanges de démarrage EAP-SIM.

M��trique	Type	Description
`eap_sim_auth_success_count`	Counter	Authentifications EAP-SIM réussies.
`eap_sim_auth_reject_count`	Counter	Authentifications EAP-SIM rejetées.
`eap_erp_initiate_count`	Counter	Initiations de ré-authentification EAP ERP.
`eap_erp_success_count`	Counter	Ré-authentifications EAP ERP réussies.
`eap_erp_mac_failure_count`	Counter	Échecs de vérification MAC EAP ERP.
`eap_erp_fallback_count`	Counter	Recul EAP ERP vers une authentification complète. Étiquette `reason`.

Exemples de requêtes :

```
# Échecs de synchronisation (devraient être rares - alerter si >
0)
increase(eap_aka_sync_failure_count[1h])

# Authentifications AKA réussies par minute
rate(eap_aka_auth_success_count[5m]) * 60

# Échecs de traitement EAP (devraient rester constants)
increase(eap_process_crash_count[1h])
```

Gauges Client, AP et Session

Ces gauges sont échantillonnées toutes les 5 secondes par le poller.

Métrique	Type	Étiquettes	Description
`radius_active_clients_count`	Gauge	none	Clients actuelleme authentifiés (processus `Twag.Session.Cli` en direct).
`radius_access_points_count`	Gauge	none	Points d'accès enregistrés (entrée `APState`).
`radius_clients_per_ap_count`	Gauge	`nas_ip`, `ssid`	Clients connectés point d'accès.
`radius_active_sessions_count`	Gauge	`status`	Sessions regroupé par statut de comptabilité (`star` `stop`, `interim_update`).
`radius_active_sessions_total`	Gauge	none	Sessions en statut `start` ou `interim_update`.

Exemples de requêtes :

```
# Top 5 AP les plus occupés
topk(5, radius_clients_per_ap_count)

# Clients par SSID
sum by (ssid) (radius_clients_per_ap_count)

# Alerte sur une chute soudaine des sessions actives
delta(radius_active_sessions_total[5m]) < -10
```

Métriques de débit de données

Ces gauges sont cumulées sur toutes les sessions suivies à partir des compteurs de comptabilité RADIUS.

Métrique	Type	Unité	Description
`radius_data_bytes_in`	Gauge	bytes	Total des octets reçus à travers les sessions (Acct-Input-Octets).
`radius_data_bytes_out`	Gauge	bytes	Total des octets envoyés à travers les sessions (Acct-Output-Octets).
`radius_session_total_time`	Gauge	secondes	Temps total de session à travers les sessions (Acct-Session-Time).

Exemples de requêtes :

```
# Débit total en Mo
(radius_data_bytes_in + radius_data_bytes_out) / 1024 / 1024

# Durée moyenne de session
radius_session_total_time / radius_active_sessions_total
```

Métriques de session et de chemin de données TWAG

Ces compteurs suivent les événements du cycle de vie par session émis par `Twag.Session.Client`.

Métrique	Type	Étiquettes	De

`twag_session_teardown_count`	Counter	`reason`	Arrêts
			Raison
			`accour`
			`admin_`
			`networ`
			`identi`
			`livene`

`twag_bearer_modify_command_count`	Counter	none	Événer Modify Comma à une s
`twag_bearer_update_count`	Counter	none	Événer Update appliqu sessior
`twag_detach_cleanup_start_count`	Counter	none	Débuts de déta
`twag_detach_cleanup_complete_count`	Counter	none	Achève nettoya détach
`twag_detach_pgw_delete_count`	Counter	none	Requêt Delete du déta
`twag_detach_liveness_timeout_count`	Counter	none	Détach déclen délai d vivacit

Métrique	Type	Étiquettes	De
`twag_liveness_arp_probe_success_count`	Counter	none	Probes vivacit
`twag_liveness_arp_probe_failure_count`	Counter	none	Probes vivacit
`twag_handover_in_count`	Counter	`source`	Événer transfe
`twag_handover_out_count`	Counter	`cause`	Événer transfe
`twag_handover_ap_change_count`	Counter	none	Événer change intra-s

Exemples de requêtes :

```
# Raisons d'arrêt au cours de la dernière heure
sum by (reason) (increase(twag_session_teardown_count[1h]))

# Sessions récoltées pour ne jamais avoir complété
l'authentification
increase(twag_session_teardown_count{reason="identity_timeout"}
[1h])

# Sessions abandonnées parce que l'UE est restée silencieuse
increase(twag_detach_liveness_timeout_count[1h])
```

> Remarque : les événements de quota Gy `twag_gy_quota_blocked` et `twag_gy_quota_unblocked` sont émis par `Twag.Session.Client` mais ne sont pas déclarés comme métriques Prometheus aujourd'hui, donc ils n'apparaissent pas sur le point de terminaison `/metrics`. Ils sont visibles dans les journaux structurés.

Métriques de facturation en ligne Gy

Ces compteurs suivent le contrôle de crédit Diameter Ro/Gy, utilisé uniquement pour le breakout local NSWO. Voir Facturation en ligne.

Métrique	Type	Étiquettes	Description
`twag_gy_ccr_sent_count`	Counter	`type`	Requêtes de contrôle de crédit envoyées à l'OCS. `type` est `initial`, `update`, ou `terminate`.
`twag_gy_ccr_error_count`	Counter	`type`	Échecs d'envoi ou de construction de CCR.
`twag_gy_cca_received_count`	Counter	`type`, `result`	Réponses de contrôle de crédit reçues de l'OCS.

Exemples de requêtes :

```
# Taux d'envoi de CCR par type
sum by (type) (rate(twag_gy_ccr_sent_count[5m]))

# Échecs de CCR
sum by (type) (increase(twag_gy_ccr_error_count[1h]))

# Résultats de CCA
sum by (result) (rate(twag_gy_cca_received_count[5m]))
```

Métriques de diamètre STa

Ces compteurs suivent les requêtes entrantes STa traitées par le TWAG.

Métrique	Type	Étiquettes	Description
`sta_asr_received_count`	Counter	none	Requêtes Abor Session reçues
`sta_rar_received_count`	Counter	none	Requêtes Re-Auth reçues.
`sta_str_received_count`	Counter	none	Requêtes Session-Termination reçues.
`sta_session_disconnect_count`	Counter	`reason`	Déconnexions de session STa
`sta_session_reauth_count`	Counter	`re_auth_type`	Ré-authentificatio de session STa

Métrique de message de diamètre

diameter_message_count

Type : Counter

Étiquette	Description
`application`	Application de diamètre (par exemple `swx`, `sta`).
`command`	Nom de la commande (par exemple `MAR`, `MAA`).
`direction`	`inbound` ou `outbound`.

Exemples de requêtes :

```
# Taux de messages de diamètre par application
sum by (application) (rate(diameter_message_count[5m]))
```

Métriques NSWO

Métrique	Type	Description
`nswo_allocate_count`	Counter	Allocations d'IP locales NSWO.
`nswo_release_count`	Counter	Libérations d'IP locales NSWO.

Métriques DHCP

Ces compteurs suivent le serveur DHCPv4. L'événement `discover.pool_allocated` couvre le chemin de distribution NSWO, où le serveur alloue une nouvelle adresse à partir d'un pool lié lors de la découverte.

Métrique	Type	Description
`dhcp_lease_allocate_count`	Counter	Allocations de baux.
`dhcp_lease_deallocate_count`	Counter	Désallocations de baux.
`dhcp_discover_count`	Counter	Paquets DISCOVER servis à partir d'un bail pré-alloué.
`dhcp_discover_no_allocation_count`	Counter	DISCOVERs sans bail et sans pool lié (aucune adresse à offrir).
`dhcp_discover_pool_allocated_count`	Counter	DISCOVERs servis en distribuant une nouvelle adresse à partir d'un pool lié (NSWO).
`dhcp_ack_count`	Counter	Paquets ACK envoyés.
`dhcp_nak_count`	Counter	Paquets NAK envoyés.
`dhcp_release_count`	Counter	Paquets RELEASE reçus.
`dhcp_decline_count`	Counter	Paquets DECLINE reçus.
`dhcp_inform_count`	Counter	Paquets INFORM reçus.
`dhcp_packet_malformed_count`	Counter	Paquets DHCP malformés.

Exemples de requêtes :

```
# Taux de distribution de pool (adressage NSWO)
rate(dhcp_discover_pool_allocated_count[5m])

# DISCOVERs qui n'ont pas pu être servis
increase(dhcp_discover_no_allocation_count[15m])
```

> Remarque : le serveur DHCP émet également `dhcp_lease_reap` lorsqu'il récolte un bail expiré. Cet événement n'est pas déclaré comme métrique Prometheus aujourd'hui et n'apparaît pas sur le point de terminaison `/metrics`.

Métriques DHCPv6 et RA IPv6

Métrique	Type	Description
`dhcpv6_info_request_received_count`	Counter	Paquets DHCPv6 Information-Request reçus.
`dhcpv6_solicit_received_count`	Counter	Paquets DHCPv6 Solicit reçus.
`dhcpv6_request_received_count`	Counter	Paquets DHCPv6 Request reçus.
`dhcpv6_info_reply_sent_count`	Counter	Paquets DHCPv6 Information-Reply envoyés.
`dhcpv6_advertise_sent_count`	Counter	Paquets DHCPv6 Advertise envoyés.
`dhcpv6_reply_sent_count`	Counter	Paquets DHCPv6 Reply envoyés.
`ipv6_ra_prefix_add_count`	Counter	Ajouts de préfixes RA IPv6.
`ipv6_ra_prefix_remove_count`	Counter	Suppressions de préfixes RA IPv6.
`ipv6_ra_advertisement_sent_count`	Counter	Annonces de routeur IPv6 envoyées.

Métriques WLCP

Ces compteurs suivent le protocole de contrôle WLAN utilisé pour le mode multi-connexion.

Métrique	Type	Unité	Description
`wlcp_packet_received_count`	Counter	none	Paquets WLCP reçus.
`wlcp_packet_received_bytes`	Sum	bytes	Total des octets de paquets WLCP reçus.
`wlcp_packet_malformed_count`	Counter	none	Paquets WLCP malformés.
`wlcp_packet_unregistered_count`	Counter	none	Paquets WLCP provenant de clients non enregistrés.
`wlcp_pdn_request_count`	Counter	none	Requêtes de connexion PDN.
`wlcp_pdn_accept_count`	Counter	none	Acceptations de connexion PDN.
`wlcp_pdn_reject_count`	Counter	none	Rejets de connexion PDN.
`wlcp_pdn_disconnect_request_count`	Counter	none	Requêtes de déconnexion PDN.
`wlcp_pdn_disconnect_accept_count`	Counter	none	Acceptations de

Métrique	Type	Unité	Description
			déconnexion PDN.

Métriques Erlang VM

Ces métriques donnent de la visibilité sur le runtime BEAM.

Mémoire

Métrique	Description
`vm_memory_total`	Mémoire totale allouée (octets).
`vm_memory_processes`	Mémoire utilisée par les processus Erlang.
`vm_memory_processes_used`	Mémoire activement utilisée par les processus.
`vm_memory_system`	Mémoire utilisée par le runtime.
`vm_memory_atom`	Mémoire utilisée par les atomes.
`vm_memory_atom_used`	Mémoire activement utilisée par les atomes.
`vm_memory_binary`	Mémoire utilisée par les binaires.
`vm_memory_code`	Mémoire utilisée par le code chargé.
`vm_memory_ets`	Mémoire utilisée par les tables ETS (état de session et d'AP).

Système

Métrique	Description
`vm_system_info_process_count`	Nombre actuel de processus Erlang.
`vm_system_info_port_count`	Nombre actuel de ports.
`vm_system_info_atom_count`	Nombre actuel d'atomes.
`vm_system_info_schedulers`	Total des threads de planification.
`vm_system_info_schedulers_online`	Threads de planification en ligne.

Planificateurs

Métrique	Description
`vm_statistics_run_queue`	Longueur totale de la file d'attente d'exécution.
`vm_total_run_queue_lengths_total`	Longueur totale de la file d'attente d'exécution du planificateur.
`vm_total_run_queue_lengths_cpu`	Longueur de la file d'attente d'exécution du planificateur CPU.
`vm_total_run_queue_lengths_io`	Longueur de la file d'attente d'ex��cution du planificateur IO.

Exemples de requêtes :

```
# Mémoire totale en Mo
vm_memory_total / 1024 / 1024

# Saturation de la file d'attente d'exécution (devrait être
faible)
avg_over_time(vm_statistics_run_queue[5m])
```

Exemples de requêtes Grafana

Vue d'ensemble de l'authentification

```
# Acceptations par seconde
sum(rate(radius_access_accept_count[5m]))

# Rejets par seconde
sum(rate(radius_access_reject_count[5m]))

# Pourcentage de ratio de succès
100 * sum(rate(radius_access_accept_count[5m])) /
  (sum(rate(radius_access_accept_count[5m])) +
sum(rate(radius_access_reject_count[5m])))
```

Statistiques par AP

```
# Requêtes par AP (Access-Request n'a pas d'étiquette nas_ip ;
utilisez la comptabilité à la place)
sum by (nas_ip) (rate(radius_accounting_request_count[5m]))

# Clients par AP
radius_clients_per_ap_count
```

Analyse des rejets et de la limitation de taux

```
# Rejets d'authentification par classe de raison
sum(increase(radius_access_reject_count[1h]))
sum(increase(radius_auth_unauthorized_source_count[1h]))
sum(increase(radius_auth_bad_secret_count[1h]))

# Raisons d'arrêt
sum by (reason) (increase(twag_session_teardown_count[1h]))
```

Utilisation des données

```
# Débit total en Mo
(radius_data_bytes_in + radius_data_bytes_out) / 1024 / 1024
```

Règles d'alerte

Alertes critiques

```
groups:
  - name: twag_critical
    rules:
      - alert: HighAuthenticationFailureRate
        expr: >
          sum(rate(radius_access_reject_count[5m])) /
          (sum(rate(radius_access_accept_count[5m])) +
sum(rate(radius_access_reject_count[5m]))) > 0.1
        for: 5m
        labels:
          severity: critical
        annotations:
          summary: "Taux d'échec d'authentification élevé (> 10%)"

      - alert: EAPSyncFailures
        expr: increase(eap_aka_sync_failure_count[5m]) > 0
        for: 1m
        labels:
          severity: warning
        annotations:
          summary: "Échecs de synchronisation EAP-AKA détectés"

      - alert: UnknownRadiusSources
        expr: increase(radius_auth_unauthorized_source_count[15m])
> 0
        for: 5m
        labels:
          severity: warning
        annotations:
          summary: "Paquets RADIUS provenant de sources sans
credential correspondant"

      - alert: BadRadiusSecret
        expr: increase(radius_auth_bad_secret_count[15m]) > 0
        for: 5m
        labels:
          severity: warning
        annotations:
```

```
          summary: "Paquets RADIUS rejetés pour un mauvais secret
partagé"

      - alert: NoActiveClients
        expr: radius_active_clients_count == 0
        for: 15m
        labels:
          severity: warning
        annotations:
          summary: "Aucun client actif pendant 15 minutes"
```

Alertes de ressources

```
groups:
  - name: twag_resources
    rules:
      - alert: HighMemoryUsage
        expr: vm_memory_total > 1073741824
        for: 10m
        labels:
          severity: warning
        annotations:
          summary: "L'utilisation de la mémoire TWAG dépasse 1 Go"

      - alert: HighRunQueue
        expr: avg_over_time(vm_statistics_run_queue[5m]) > 10
        for: 5m
        labels:
          severity: warning
        annotations:
          summary: "Une file d'attente de planification élevée
indique une saturation du CPU"
```

Pages connexes

- Architecture
- Rejets RADIUS
- Secrets partagés RADIUS

- Sessions
- Facturation en ligne
- Comptabilité par AP

Vue d'ensemble

Chargement en ligne (Gy)

Vue d'ensemble

OmniTWAG peut charger des sessions de données en temps réel contre un Système de Chargement en Ligne (OCS). OmniTWAG utilise l'interface Diameter Ro (Gy) pour envoyer des Demandes de Contrôle de Crédit (CCR) et recevoir des Réponses de Contrôle de Crédit (CCA). Cela permet à un opérateur d'appliquer un contrôle de crédit prépayé au trafic WLAN que le TWAG dessert.

Le chargement en ligne s'applique aux sessions où le TWAG possède le chemin de données, c'est-à-dire **NSWO (sortie locale)** et le trafic ancré par le TWAG. Dans ce cas, le TWAG est le point où le trafic sort vers Internet, donc le TWAG agit comme le point d'application du contrôle de chargement. En mode **routé à domicile (GTP)**, le PGW effectue le chargement, donc un opérateur laisse normalement le mode de chargement global `:disabled`. Voir la référence Plan Utilisateur pour les modes de plan de données.

Deux contrôles décident si Gy fonctionne pour un UE donné :

- Le **mode de chargement global** (`online_charging.mode`). Cela détermine quelles Demandes de Contrôle de Crédit OmniTWAG peut envoyer.
- Le **mode de chargement par AP** sur les informations d'identification AP (`charging_mode`). Cela détermine, par point d'accès, si OmniTWAG ouvre une session Gy pour cet UE. Voir Informations d'identification RADIUS.

Les deux contrôles doivent permettre le chargement pour qu'un CCR-Initial soit envoyé. Voir Mode de Chargement par AP.

Table des Matières

- Concepts
- Modes de Chargement

- Mode de Chargement par AP
- Flux de Contrôle de Crédit
- Quota, Utilisation et Réautorisation
- Application du Quota
- Groupe de Tarification et Unités de Service
- Configuration
- Comportement Opérationnel et Gestion des Échecs
- Observabilité
- Tables de Référence

Concepts

L'interface Gy est une application Diameter qui transporte des messages de contrôle de crédit entre un élément de réseau et un OCS. OmniTWAG utilise le dictionnaire 3GPP Ro avec Auth-Application-Id `4`, tel que défini dans RFC 4006 et 3GPP TS 32.299. OmniTWAG utilise l'Id de Contexte de Service `32251@3gpp.org`, qui est le contexte de chargement PS 3GPP.

OmniTWAG envoie trois types de Demandes de Contrôle de Crédit pendant une session :

- **CCR-Initial (CCR-I)** : au début de la session. OmniTWAG demande à l'OCS de commencer le contrôle de crédit pour l'abonné.
- **CCR-Update (CCR-U)** : pendant la session. OmniTWAG rapporte les données utilisées et demande plus de quota.
- **CCR-Termination (CCR-T)** : à la fin de la session. OmniTWAG rapporte les données finales utilisées et ferme la session de contrôle de crédit.

L'OCS répond à chaque demande avec une Réponse de Contrôle de Crédit (CCA). Une CCA contient un `Result-Code`, un octroi optionnel de quota (`Granted-Service-Unit`), et un `Validity-Time` optionnel.

OmniTWAG garde un état Gy séparé pour chaque session d'abonné. L'état Gy contient le `Session-Id` Diameter, le `CC-Request-Number` actuel, les compteurs d'utilisation cumulés, et le dernier quota accordé par l'OCS. OmniTWAG crée

cet état lorsqu'il envoie le CCR-I, et il rejette l'état lorsque la session se termine.

Modes de Chargement

Le paramètre `online_charging.mode` est le mode de chargement **global**. Il sélectionne l'un des trois modes et contrôle quelles Demandes de Contrôle de Crédit OmniTWAG peut envoyer. Le `charging_mode` par AP (voir Mode de Chargement par AP) décide ensuite, pour chaque UE, si OmniTWAG ouvre une session Gy.

Mode	CCR-I au début	CCR-U pendant la session	CCR-T à la fin
`:disabled`	Non	Non	Non
`:authorize_only`	Oui	Non	Oui
`:online_charging`	Oui	Oui	Oui

`:disabled`

OmniTWAG ne contacte pas l'OCS. C'est le mode par défaut. Utilisez ce mode pour le fonctionnement Routé à Domicile (GTP), où le PGW effectue le chargement.

`:authorize_only`

OmniTWAG envoie un CCR-I au début de la session et un CCR-T à la fin de la session. OmniTWAG ne send pas de messages CCR-U intermédiaires. Utilisez ce mode pour vérifier le solde de l'abonné au début et pour rapporter l'utilisation totale à la fin, sans gestion de quota en direct.

`:online_charging`

OmniTWAG envoie un CCR-I au début, un CCR-U à chaque mise à jour intermédiaire de comptabilité, et un CCR-T à la fin. Utilisez ce mode pour une gestion complète du quota, où l'OCS accorde et rafraîchit le quota pendant la session.

Mode de Chargement par AP

Chaque information d'identification AP contient un champ `charging_mode`. Cela est séparé du `online_charging.mode` global. OmniTWAG résout l'information d'identification à partir de la source RADIUS de la session, donc chaque point d'accès peut avoir son propre comportement. Voir Informations d'identification RADIUS.

`charging_mode` par AP	Signification	Effet sur Gy
`:charged`	Charger le trafic de ce point d'accès. C'est le mode par défaut.	OmniTWAG ouvre une session Gy (CCR-Initial) pour l'UE lorsque le mode global le permet également.
`:uncharged`	Servir ce point d'accès mais ne pas le charger.	OmniTWAG établit la session de données mais n'ouvre aucune session Gy. Aucun CCR n'est envoyé.
`:auth_only`	Authentifier uniquement, pas de session de données.	OmniTWAG authentifie l'abonné et s'arrête. Il ne construit pas de session de données et n'ouvre aucune session Gy.

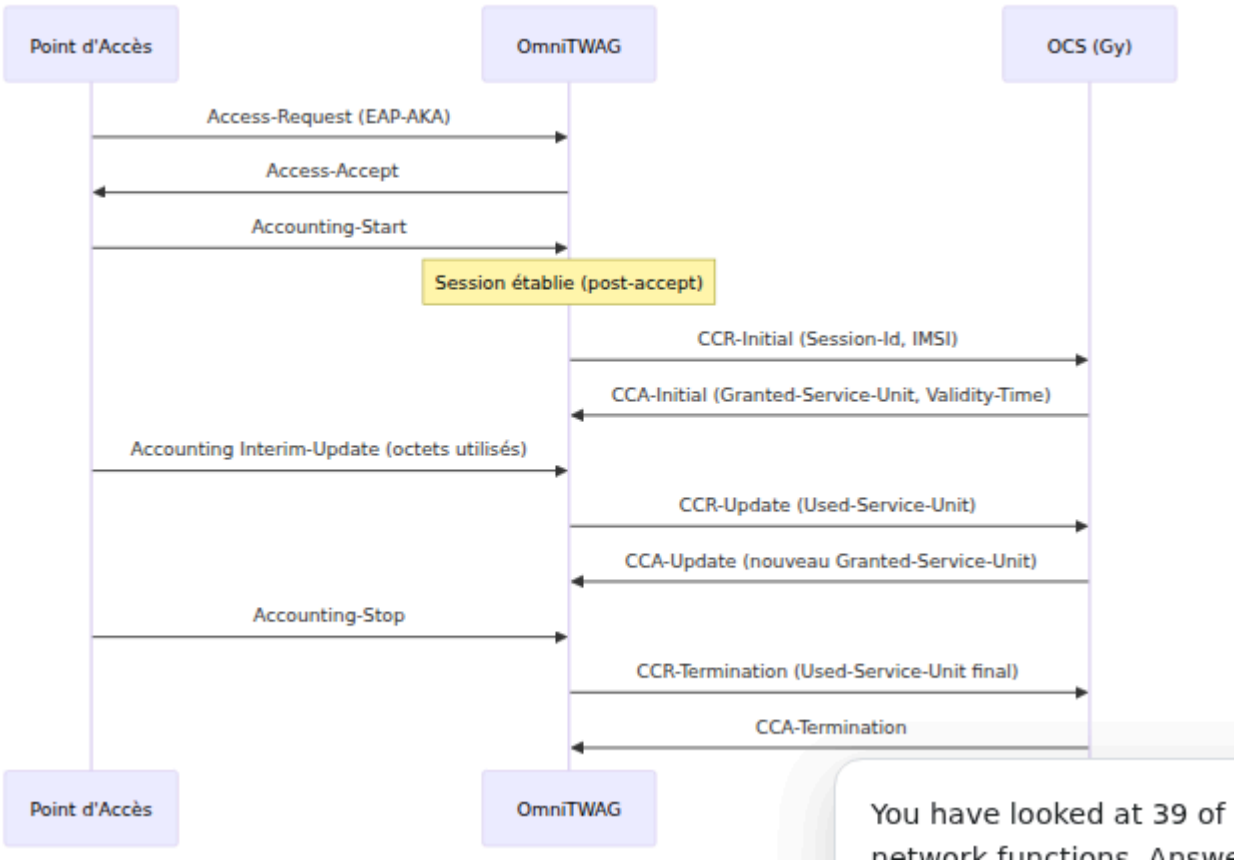
OmniTWAG envoie un CCR-Initial uniquement lorsque **toutes** ces conditions sont remplies :

- Le `online_charging.mode` global n'est pas `:disabled` (le client Gy est activé).
- La session a un IMSI.
- L'information d'identification AP `charging_mode` est `:charged`.

Si une condition échoue, OmniTWAG n'ouvre pas de session Gy pour l'UE, et aucun CCR-U ou CCR-T ultérieur n'est envoyé. Le mode par AP gate donc Gy indépendamment du mode global : même avec le mode global réglé sur `:online_charging`, un AP marqué `:uncharged` ne produit aucun trafic Gy.

Flux de Contrôle de Crédit

La session de contrôle de crédit suit la session d'abonné. OmniTWAG déclenche chaque Demande de Contrôle de Crédit à partir d'un point spécifique dans le cycle de vie de la session RADIUS.



Début de session (CCR-Initial)

OmniTWAG envoie le CCR-I après avoir établi la session. L'établissement de la session se déroule lors de l'étape post-acceptation, que le serveur RADIUS déclenche après l'Access-Accept et l'Accounting-Start. OmniTWAG envoie le CCR-I uniquement lorsque le mode de chargement global est activé, la session a un IMSI, et l'information d'identification AP `charging_mode` est `:charged`. Voir Mode de Chargement par AP.

Le CCR-I contient l'IMSI de l'abonné dans un AVP `Subscription-ID` de type `1` (END_USER_IMSI). Le CCR-I définit `Multiple-Services-Indicator` sur `1` et inclut un groupe `Multiple-Services-Credit-Control` (MSCC) vide avec un `Requested-Service-Unit` vide et le `Rating-Group` configuré.

Mises à jour intermédiaires (CCR-Update)

En mode `:online_charging`, chaque mise à jour intermédiaire de comptabilité RADIUS intègre les octets rapportés et le temps de session dans les compteurs d'utilisation Gy. OmniTWAG n'envoie pas de CCR-U à chaque intermédiaire. Il envoie un CCR-U uniquement lorsque l'un de ces déclencheurs se produit :

- Le volume utilisé a dépassé la fraction configurée du volume accordé (le `quota.update_threshold`, par défaut `0.8`).
- Le `Validity-Time` du quota accordé a expiré.

Un intermédiaire qui ne franchit pas un déclencheur enregistre simplement l'utilisation localement, donc l'OCS n'est pas consulté à chaque intermédiaire. Lorsque un CCR-U est envoyé, OmniTWAG lit les octets utilisés et le temps de session à partir des attributs de comptabilité et les rapporte dans un AVP `Used-Service-Unit`. Le CCR-U inclut également un `Requested-Service-Unit` vide, afin que l'OCS puisse accorder plus de quota.

Un CCR-U est également envoyé en dehors du chemin de comptabilité lorsque le minuteur de validité du quota se déclenche. Voir Application du Quota.

OmniTWAG lit ces attributs de comptabilité RADIUS pour le CCR-U :

Attribut RADIUS	Champ Gy
`Acct-Input-Octets`	`CC-Input-Octets`
`Acct-Output-Octets`	`CC-Output-Octets`
`Acct-Session-Time`	`CC-Time`

OmniTWAG calcule `CC-Total-Octets` comme la somme des octets d'entrée et de sortie.

Fin de session (CCR-Termination)

OmniTWAG envoie un CCR-T lorsque la session se termine, par exemple lors d'un Accounting-Stop RADIUS. OmniTWAG rapporte l'utilisation cumulative finale dans un AVP `Used-Service-Unit` et définit le `Termination-Cause` sur `1` (DIAMETER_LOGOUT). OmniTWAG envoie le CCR-T à la fois en mode `:authorize_only` et en mode `:online_charging`.

Quota, Utilisation et Réautorisation

Quota accordé

Chaque CCA peut contenir un `Granted-Service-Unit` à l'intérieur du premier groupe MSCC. OmniTWAG stocke les valeurs accordées dans l'état Gy :

- `CC-Total-Octets`, `CC-Input-Octets`, `CC-Output-Octets` : le volume de données accordé.
- `CC-Time` : le temps accordé.
- `Validity-Time` : le temps en secondes pendant lequel l'octroi reste valide. OmniTWAG transforme cela en une date limite absolue et arme un minuteur contre cela.

OmniTWAG lit également le `Final-Unit-Indication` MSCC et le `Result-Code` au niveau MSCC de chaque CCA. Une `Final-Unit-Action` de TERMINATE, REDIRECT ou RESTRICT_ACCESS, ou un `Result-Code` MSCC de `4010` (END_USER_SERVICE_DENIED) ou `4012` (CREDIT_LIMIT_REACHED), est stocké comme un signal d'épuisement de crédit. Voir Application du Quota.

OmniTWAG met à jour l'octroi stocké à chaque CCA-I et CCA-U réussis.

Rapport d'utilisation

OmniTWAG garde des compteurs d'utilisation cumulés dans l'état Gy. Les compteurs de comptabilité RADIUS sont cumulés pour la session, donc chaque mise à jour intermédiaire rapporte le total jusqu'à présent. OmniTWAG enregistre les compteurs rapportés et les envoie comme le `Used-Service-Unit`. Le CCR-U définit le `Reporting-Reason` sur `4` (VALIDITY_TIME). Le CCR-T définit le `Reporting-Reason` sur `2` (FINAL).

Réautorisation

OmniTWAG se réautorise sur deux voies :

- **Sur le trafic.** Une mise à jour intermédiaire de comptabilité RADIUS qui franchit le seuil de quota ou trouve la validité expirée déclenche un CCR-U. L'intervalle intermédiaire est une propriété du point d'accès, donc un intervalle intermédiaire plus court donne une mesure plus fine.
- **Sur le temps.** OmniTWAG arme un minuteur de validité de quota à partir du `Validity-Time` accordé. Si le minuteur se déclenche avec peu ou pas de trafic, OmniTWAG envoie tout de même un CCR-U pour se réautoriser. Si l'OCS n'envoie pas de `Validity-Time`, OmniTWAG peut revenir au `quota.validity_time` configuré. En l'absence des deux, aucun minuteur de validité n'est armé.

Le `CC-Request-Number` augmente d'un après chaque CCA réussi. Cette valeur donne à chaque demande dans la session de contrôle de crédit un numéro de séquence unique, comme requis par RFC 4006.

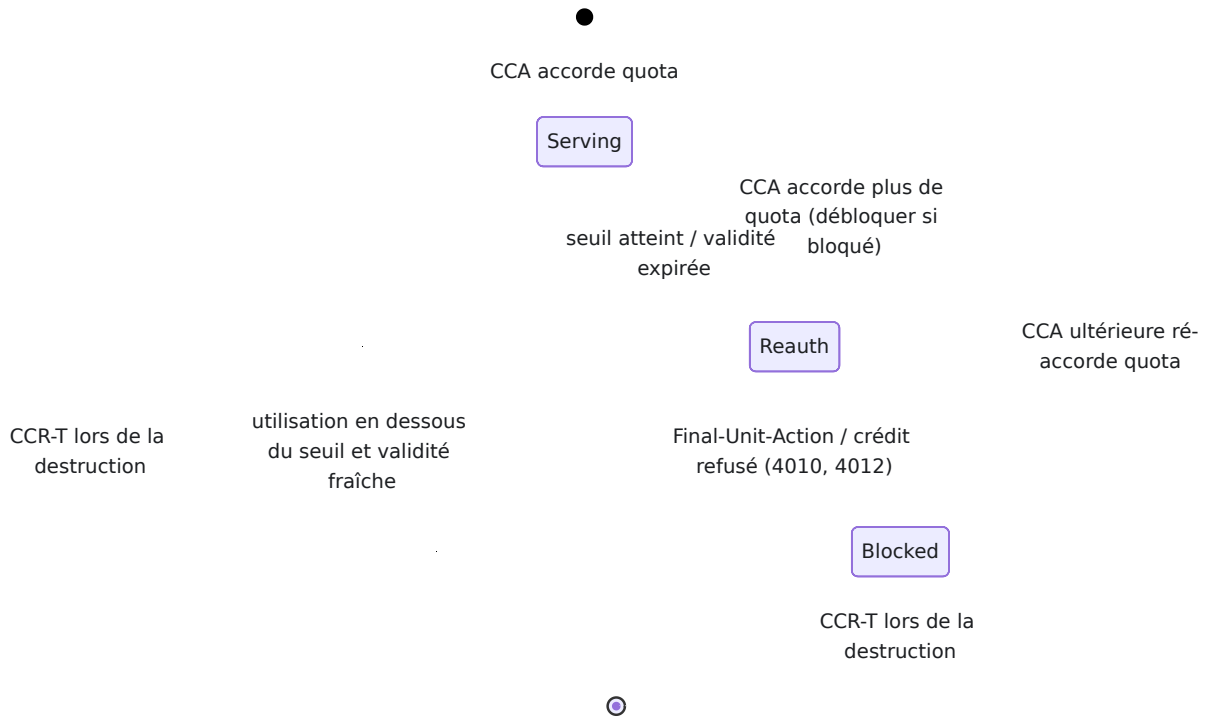
Application du Quota

OmniTWAG applique l'épuisement de crédit en mode `:online_charging`. Le TWAG n'a pas de fonction de plan utilisateur séparée, donc son levier d'application est le chemin de données GRE : il désenregistre l'UE du chemin de données, ce qui coupe l'ascendant et le descendant pour cet UE.

Une réponse CCR-U déclenche l'application lorsque l'état Gy montre un crédit épuisé, c'est-à-dire lorsque l'OCS :

- signale toute `Final-Unit-Action` (TERMINATE, REDIRECT ou RESTRICT_ACCESS), ou
- retourne un `Result-Code` MSCC de `4010` ou `4012`.

Parce que le TWAG ne peut pas honorer REDIRECT ou RESTRICT_ACCESS sur son chemin de données, toute `Final-Unit-Action` est traitée comme un arrêt. Lorsque l'application se déclenche, OmniTWAG bloque le chemin de données de l'UE mais garde la session, le bail IP et la comptabilité actifs. Une réautorisation ultérieure qui accorde un quota utilisable restaure le chemin de données.



L'application est un meilleur effort et échoue ouvertement sur les erreurs de transport : un CCR-U qui échoue au niveau Diameter ne bloque jamais l'UE. Le

blocage se produit uniquement sur un refus explicite de crédit de l'OCS ou un signal d'épuisement. Voir Comportement Opérationnel et Gestion des Échecs.

Groupe de Tarification et Unités de Service

OmniTWAG utilise un seul groupe de tarification pour toute la session. Le paramètre `rating_group` définit la valeur `Rating-Group` dans le MSCC du CCR-I. La valeur par défaut est `1`. OmniTWAG ne divise pas le trafic en plusieurs groupes de tarification ou identifiants de service. OmniTWAG demande un octroi de volume combiné avec un `Requested-Service-Unit` vide, donc l'OCS décide de l'octroi.

Configuration

Définissez les paramètres de chargement en ligne sous l'application `:omnitwag`, dans le bloc `online_charging` de `config/runtime.exs`.

```
config :omnitwag,
  online_charging: %{
    # Mode de chargement global : :disabled, :authorize_only, ou
:online_charging
    mode: :disabled,

    # Domaine Diameter OCS (Destination-Realm pour CCR)
    ocs_realm: "epc.mnc001.mcc001.3gppnetwork.org",

    # Groupe de tarification pour le groupe Multiple-Services-
Credit-Control
    rating_group: 1,

    # Gestion du quota (utilisé en mode :online_charging)
    quota: %{
      # Fraction du volume accordé à laquelle un CCR-Update est
déclenché
      update_threshold: 0.8,

      # Temps de validité de secours (secondes) lorsque l'OCS
n'envoie pas de Validity-Time
      validity_time: 3600
    }
  }
```

Paramètres

Paramètre	Type	Requis	Par défaut	Descriptio
`mode`	Atome	Non	`:disabled`	Mode de chargement g Un des `:disab` `:authorize_o` ou `:online_char` Voir Modes de Chargement.
`ocs_realm`	Chaîne	Non	Domaine Diameter	Domaine Diam de l'OCS. OmniTWAG ut cela comme `Destination-Realm` dans ch CCR. Si non dé OmniTWAG ut le domaine Diameter loca
`rating_group`	Entier	Non	`1`	Valeur `Rating Group` que OmniTWAG m dans le MSCC CCR-Initial.
`quota.update_threshold`	Flottant	Non	`0.8`	Fraction (0.0 à du volume acc à laquelle une à jour intermédiaire

Paramètre	Type	Requis	Par défaut	Descriptio
				déclenche un Update.
`quota.validity_time`	Entier	Non	(non défini)	Validité de quo de secours en secondes, utili pour armer le minuteur de validité lorsqu l'OCS n'envoie de `Validity-T` Si non défini, aucun minuteu secours n'est armé.

Le `charging_mode` par AP ne fait pas partie de ce bloc. Il vit sur chaque information d'identification AP. Voir Informations d'identification RADIUS.

L'`Origin-Host` et l'`Origin-Realm` Diameter pour la session Gy proviennent de l'identité Diameter dans la configuration `:diameter_ex` (`host` et `realm`). Le `Destination-Host` n'est pas configuré. OmniTWAG apprend le `Destination-Host` à partir de l'`Origin-Host` du premier CCA, et il ajoute cet AVP aux messages CCR-U et CCR-T ultérieurs.

Transport Diameter et l'application Ro

Le client Gy envoie via la pile Diameter partagée. Configurez l'OCS comme un pair Diameter, et activez l'application `:ro`, dans la configuration `:diameter_ex`.

```
config :diameter_ex,
  diameter: %{
    service_name: :omnitouch_twag,
    host: "amanaki",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    request_timeout: 5000,
    applications: [
      %{
        application_name: :ro,
        application_dictionary: :diameter_gen_3gpp_ro,
        vendor_specific_application_ids: [
          %{vendor_id: 0, auth_application_id: 4,
acct_application_id: nil}
        ]
      }
    ],
    peers: [
      %{
        host: "ocs.epc.mnc001.mcc001.3gppnetwork.org",
        realm: "epc.mnc001.mcc001.3gppnetwork.org",
        ip: "10.179.2.233",
        port: 3868,
        transport: :diameter_tcp,
        initiate_connection: true
      }
    ]
  }
```

Paramètre	Type	Requis	Par défaut	De
`service_name`	Atome	Oui	-	Nom du Diamete OmniTW demand service.
`host`	Chaîne	Oui	-	Identité (hôte). C construit Gy à par `realm`.
`realm`	Chaîne	Oui	-	Domaine local. Or cela com `Realm`, e `Destina` lorsque pas défi
`request_timeout`	Entier	Non	5000	Temps e à attend avant qu échoue.
`application_name`	Atome	Oui	-	Nom de Diamete `:ro` pou
`application_dictionary`	Atome	Oui	-	Dictionn pour l'ap Réglez s `:diamet` pour Gy.

Paramètre	Type	Requis	Par défaut	De
`vendor_specific_application_ids`	Liste	Oui	-	ID d'app spécifiqu fourniss utilisez et `auth_ap 4`.
`peers`	Liste	Oui	-	Liste des Diamete en tant Paramèt

Paramètres des Pairs

Paramètre	Type	Requis	Par défaut	Descriptio
`host`	Chaîne	Oui	-	Identité Diameter du pair (FQDN). Doit correspondre l'identité de l'OCS.
`realm`	Chaîne	Oui	-	Domaine Diameter du pair. Réglez s le domaine d l'OCS.
`ip`	Chaîne	Oui	-	Adresse IP de l'OCS pour la connexion TC
`port`	Entier	Non	3868	Port Diamete de l'OCS. Le port standar est 3868.
`transport`	Atome	Non	`:diameter_tcp`	Protocole de transport. Utilisez `:diameter_t` pour TCP.
`initiate_connection`	Booléen	Non	false	Lorsque `true` OmniTWAG commence la connexion au pair.

> **Remarque :** Le bloc `online_charging` dans le `config/runtime.exs` expédié montre également `ocs_host`, `service_identifier`, un `request_timeout` de niveau supérieur, et les clés `quota` `initial_request` et `subsequent_request`. Le code de chargement actuel ne lit pas ces clés. Il lit `mode`, `ocs_realm`, `rating_group`, `quota.update_threshold`, et `quota.validity_time`. Le délai d'attente de la demande Diameter provient du `:diameter_ex` `request_timeout`. OmniTWAG demande un quota avec un `Requested-Service-Unit` vide, donc `initial_request` et `subsequent_request` ne façonnent pas le CCR aujourd'hui. Ne comptez pas sur les clés non lues jusqu'à ce que le code les prenne en charge.

Comportement Opérationnel et Gestion des Échecs

Comportement de faille ouverte

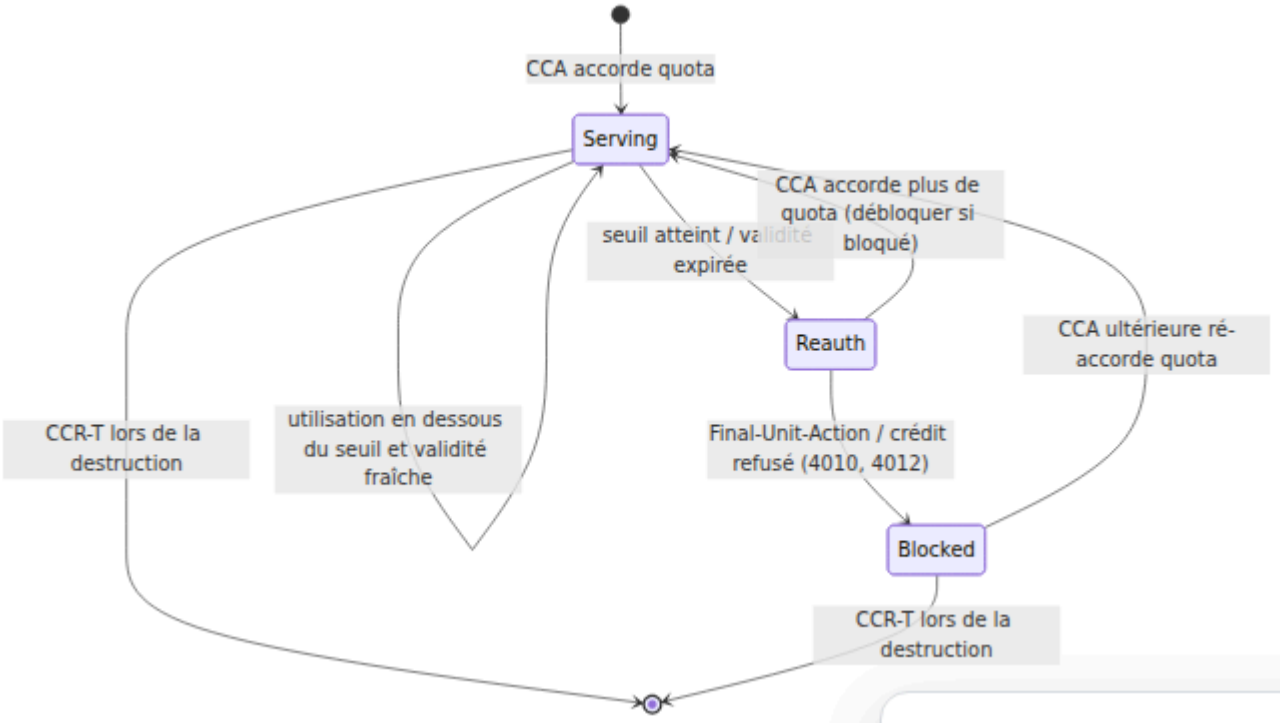
OmniTWAG traite un échec de transport Gy comme non fatal. Si une Demande de Contrôle de Crédit échoue au niveau Diameter, la session d'abonné continue et OmniTWAG ne bloque pas le trafic.

- **OCS injoignable ou délai d'attente** : Le CCR échoue au transport Diameter. OmniTWAG enregistre un avertissement. Pour un CCR-I, OmniTWAG garde toujours l'état Gy, afin qu'il puisse envoyer un CCR-T plus tard. Pour un CCR-U, OmniTWAG garde l'état Gy précédent et ne bloque pas l'UE. Pour un CCR-T, OmniTWAG enregistre l'avertissement et termine la destruction.
- **Code de résultat d'erreur CCA** : L'OCS retourne un `Result-Code` au niveau de la commande qui n'est pas `2001` (DIAMETER_SUCCESS). OmniTWAG enregistre l'erreur et traite la demande comme échouée. OmniTWAG garde la session active.

L'application est séparée de l'échec de transport. En mode `:online_charging`, OmniTWAG bloque le chemin de données de l'UE lorsque l'OCS refuse explicitement le crédit ou signale une `Final-Unit-Action` dans un MSCC CCA. Voir Application du Quota. Il ne bloque jamais sur un échec de transport, et il

ne refuse jamais l'accès en mode `:authorize_only`. Planifiez l'intégration de l'OCS avec cette séparation à l'esprit.

Résumé du cycle de vie de la session de contrôle de crédit



Gestion des codes de résultat

OmniTWAG lit le code de résultat à deux niveaux.

- **`Result-Code` au niveau de la commande.** OmniTWAG traite `2001` comme un succès. Il traite également un `Result-Code` manquant comme un succès, car certaines réponses de pair arrivent sous forme de record brut sans un code de résultat décodé. Toute autre valeur au niveau de la commande est une erreur : OmniTWAG l'enregistre, ne réessaie pas, et garde la session active.
- **`Result-Code` au niveau MSCC.** À l'intérieur du premier groupe MSCC, un `Result-Code` de `4010` (END_USER_SERVICE_DENIED) ou `4012` (CREDIT_LIMIT_REACHED) est interprété comme un signal de refus de crédit. En mode `:online_charging`, cela déclenche l'application et bloque le chemin de données de l'UE. Voir Application du Quota.

Observabilité

OmniTWAG émet des événements de télémétrie pour chaque Demande et réponse de Contrôle de Crédit. Les événements portent une valeur de métadonnées `type` de `:initial`, `:update`, ou `:terminate`.

Événement de Télémétrie	Quand	Métadonnées
`[:twag, :gy, :ccr, :sent]`	OmniTWAG envoie un CCR	`type`
`[:twag, :gy, :cca, :received]`	OmniTWAG reçoit un CCA	`type`, `result` (`:success` ou `:error`)
`[:twag, :gy, :ccr, :error]`	Un CCR échoue au transport	`type`
`[:twag, :gy, :quota, :blocked]`	Le chemin de données de l'UE a été bloqué en raison de l'épuisement du crédit	`identity`
`[:twag, :gy, :quota, :unblocked]`	Le chemin de données de l'UE a été restauré après un ré-accord	`identity`

> **Remarque :** Ces événements de Télémétrie Gy ne sont pas enregistrés en tant que métriques Prometheus dans la version actuelle. La Référence des Métriques ne les exporte pas encore. Pour observer le comportement de Gy aujourd'hui, utilisez les journaux de l'application. Le client Gy journalise chaque CCR-I, CCR-U, et CCR-T au niveau INFO, et chaque échec au niveau WARN ou ERROR. Les lignes de journal portent l'`Session-Id` Diameter et, pour le CCR-I, l'IMSI.

Tables de Référence

Valeurs CC-Request-Type

Valeur	Nom	Message
1	INITIAL_REQUEST	CCR-Initial
2	UPDATE_REQUEST	CCR-Update
3	TERMINATION_REQUEST	CCR-Termination

Les valeurs sont définies dans RFC 4006 §8.3.

Valeurs de Reporting-Reason utilisées

Valeur	Nom	Où OmniTWAG l'utilise
2	FINAL	CCR-Termination
4	VALIDITY_TIME	CCR-Update

Code de Résultat

Code	Niveau	Nom	Comportement d'OmniTWAG
2001	Commande	DIAMETER_SUCCESS	Considéré comme un succès. L'état Gy avance.
(absent)	Commande	-	Considéré comme un succès, pour les réponses de record brut.
Toute autre	Commande	(erreur)	Enregistré comme une erreur. La session continue.
4010	MSCC	END_USER_SERVICE_DENIED	Lu comme crédit refusé. Applique (bloque l'UE) en mode `:online_charging`.
4012	MSCC	CREDIT_LIMIT_REACHED	Lu comme crédit refusé. Applique (bloque l'UE) en mode `:online_charging`.

Valeurs de Final-Unit-Action

Le `Final-Unit-Indication` MSCC porte une `Final-Unit-Action`. OmniTWAG n'a pas de fonction de plan utilisateur, donc il ne peut pas honorer REDIRECT ou RESTRICT_ACCESS sur le chemin de données. Il traite toute `Final-Unit-Action` comme un arrêt et bloque l'UE.

Valeur	Nom	Comportement d'OmniTWAG
0	TERMINATE	Bloque le chemin de données de l'UE (session conservée).
1	REDIRECT	Bloque le chemin de données de l'UE (redirection non supportée).
2	RESTRICT_ACCESS	Bloque le chemin de données de l'UE (restriction non supportée).

Le `Final-Unit-Indication` est défini dans 3GPP TS 32.299 et RFC 4006 §5.6.

Application Diameter

ID	Interface	Dictionnaire	Direction	Référence
4	Ro / Gy	`diameter_gen_3gpp_ro`	TWAG vers OCS	RFC 4006, 3GPP TS 32.299

Références de Spécification

Sujet	Spécification
Application de Contrôle de Crédit Diameter	RFC 4006
Applications de chargement Diameter (Ro)	3GPP TS 32.299
Architecture de chargement	3GPP TS 32.240
TWAG / accès non-3GPP (S2a, NSWO)	3GPP TS 23.402
Protocole de Base Diameter	RFC 6733

Pages Associées

- Vue d'ensemble
- Informations d'identification RADIUS
- Pools IP
- Gestion des Adresses IP
- Sessions
- S2a GTP-C
- Plan Utilisateur

Vue d'ensemble

Comptabilité par point d'accès

Vue d'ensemble

OmniTWAG agrège les données de comptabilité RADIUS pour chaque point d'accès. L'opérateur voit le trafic total et le nombre de sessions pour chaque point d'accès. Ces données sont séparées des enregistrements de comptabilité par session. OmniTWAG clé les totaux par point d'accès par l'`NAS-IP-Address` du point d'accès. Lorsqu'un point d'accès omet l'`NAS-IP-Address` (par exemple certains modèles UBNT), OmniTWAG revient à l'IP source UDP du paquet RADIUS, de sorte que la comptabilité s'attache à la même entrée de point d'accès que l'authentification a créée.

Table des matières

- Concepts
- Comment les totaux sont comptés
- Champs enregistrés
- Nombre de clients actifs
- Où voir les données
- Relation avec les données par session
- Dépannage
- Pages connexes

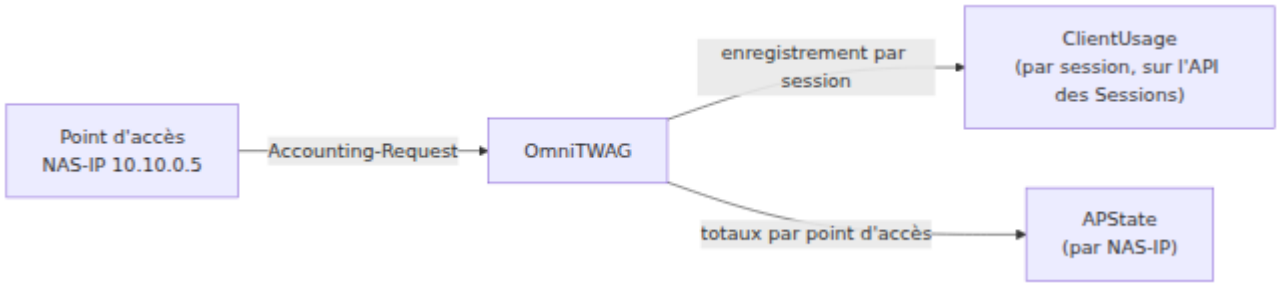
Concepts

Un point d'accès envoie des paquets RADIUS Accounting-Request pour chaque session client. Un paquet a un `Acct-Status-Type` qui indique l'événement :

- **Démarrer** (`1`) : une session a commencé.

- **Mise à jour intermédiaire** (`3`) : une session est active, avec les compteurs jusqu'à présent.
- **Arrêter** (`2`) : une session s'est arrêtée, avec les compteurs finaux.

OmniTWAG lit les compteurs d'octets et de paquets ainsi que l'`NAS-IP-Address` de chaque paquet. OmniTWAG ajoute ces valeurs aux totaux en cours pour ce point d'accès. Cela donne à l'opérateur une vue par point d'accès du trafic et des sessions.



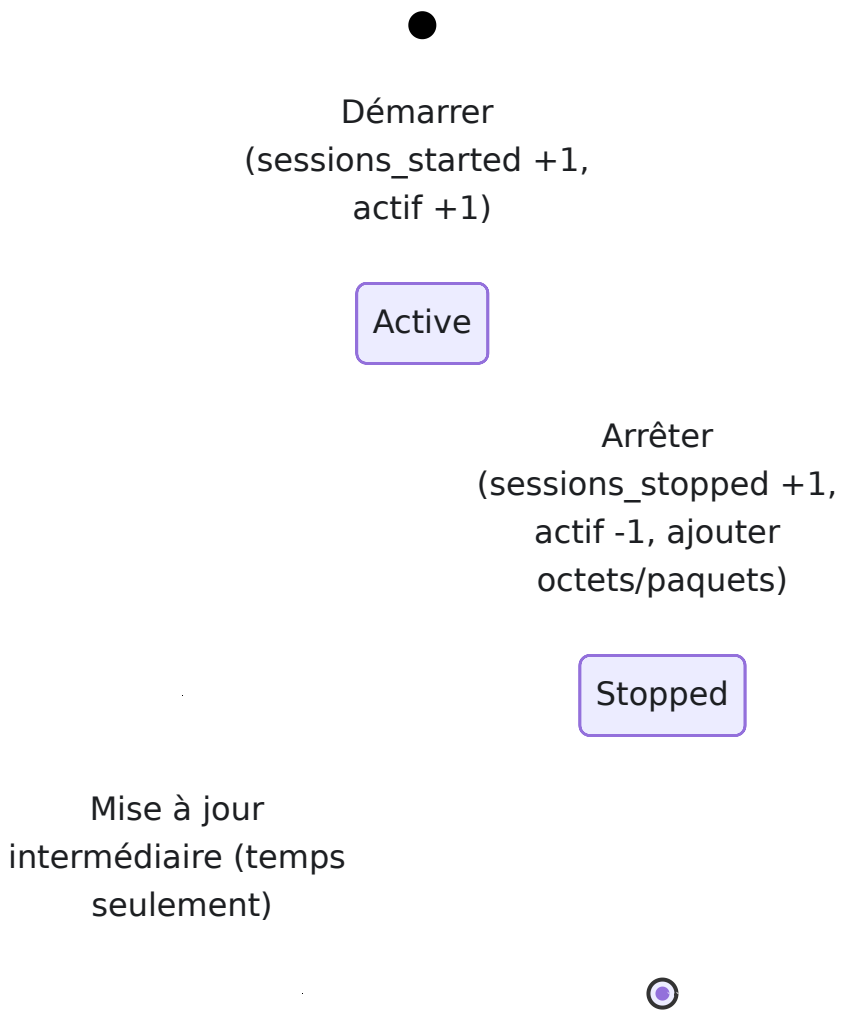
Comment les totaux sont comptés

Les compteurs d'octets et de paquets RADIUS sont **cumulatifs pour chaque session**. Une mise à jour intermédiaire et un arrêt rapportent tous deux le total pour la session jusqu'à présent, et non le changement depuis le dernier paquet. Si OmniTWAG ajoutait les compteurs de chaque paquet, il compterait le même trafic plus d'une fois.

Pour éviter le double comptage, OmniTWAG ajoute les compteurs d'octets et de paquets aux totaux par point d'accès **uniquement lors d'un arrêt**. Un arrêt rapporte le total final pour une session terminée. Les règles sont :

- Lors de **Démarrer** : augmenter `acct_sessions_started` de un. Augmenter `acct_active_sessions` de un.
- Lors de **Arrêter** : augmenter `acct_sessions_stopped` de un. Diminuer `acct_active_sessions` de un, mais pas en dessous de zéro. Ajouter les compteurs d'octets et de paquets finaux aux totaux par point d'accès.
- Lors de **Mise à jour intermédiaire** : mettre à jour le dernier temps de comptabilité uniquement. Ne pas changer les totaux.

Si OmniTWAG reçoit un arrêt pour une session qu'il n'a pas vue commencer, le compte de sessions actives ne descend pas en dessous de zéro. OmniTWAG ajoute toujours les derniers octets et paquets. Si OmniTWAG reçoit un Accounting-Request pour un point d'accès qu'il n'a pas vu auparavant, OmniTWAG crée l'entrée par point d'accès. Cela signifie qu'un point d'accès uniquement comptable est toujours suivi.



Champs enregistrés

OmniTWAG enregistre ces champs pour chaque point d'accès, en plus des métadonnées du point d'accès telles que SSID et `Called-Station-Id`.

Champ	Type	Description
`acct_input_octets`	Entier	Total des octets entrants, cumulés sur les sessions arrêtées.
`acct_output_octets`	Entier	Total des octets sortants, cumulés sur les sessions arrêtées.
`acct_input_packets`	Entier	Total des paquets entrants, cumulés sur les sessions arrêtées.
`acct_output_packets`	Entier	Total des paquets sortants, cumulés sur les sessions arrêtées.
`acct_sessions_started`	Entier	Nombre de sessions qui ont commencé sur ce point d'accès.
`acct_sessions_stopped`	Entier	Nombre de sessions qui se sont arrêtées sur ce point d'accès.
`acct_active_sessions`	Entier	Mesure des sessions ouvertes dérivée de la comptabilité : `started` moins `stopped`, jamais moins que zéro. Elle reflète uniquement ce que la comptabilité RADIUS rapporte, donc elle peut surcompter lorsque un point d'accès manque des paquets d'arrêt (par exemple après un redémarrage) et reste obsolète jusqu'à ce que les arrêts arrivent ou que l'entrée expire. Ce n'est **pas** le chiffre "Clients actifs" affiché sur la page des Points d'accès — voir Nombre de clients actifs.
`last_accounting_at`	Entier	Heure du dernier paquet de comptabilité, en millisecondes depuis l'époque Unix.

La direction des compteurs d'octets et de paquets suit la définition RADIUS dans RFC 2866. L'entrée est le trafic du client vers le réseau. La sortie est le trafic du réseau vers le client.

Nombre de clients actifs

Le chiffre **Clients actifs** sur la page des Points d'accès, et le champ `client_count` sur l'API du point d'accès, rapportent combien de clients sont **connectés en ce moment**. Cette valeur n'est pas le compteur de comptabilité `acct_active_sessions`. Elle est dérivée des processus de session en mémoire en direct que OmniTWAG exécute pour chaque client authentifié, regroupés par l'`NAS-IP-Address` du point d'accès.

Chaque processus de session surveille la vivacité à partir des mises à jour intermédiaires de comptabilité. Lorsqu'un client cesse d'envoyer des mises à jour pendant un nombre configuré d'intervalles, le processus se termine, que un RADIUS Accounting-Stop arrive ou non. Parce que le compte de clients actifs est un total de ces processus en direct, il descend à zéro de lui-même lorsque un point d'accès se déconnecte — il ne dépend pas de l'envoi d'un arrêt par le point d'accès, et il ne monte pas après un redémarrage du point d'accès.

Utilisez `client_count` (le chiffre en direct) comme le compte des clients actuellement connectés. Utilisez `acct_active_sessions` lorsque vous souhaitez spécifiquement la mesure des sessions ouvertes rapportée par la comptabilité, en gardant à l'esprit ses mises en garde ci-dessus.

Où voir les données

Les champs par point d'accès font partie de l'enregistrement du point d'accès. L'opérateur les lit à deux endroits :

- **API REST** : le point de terminaison `GET /access_points` retourne chaque point d'accès avec ses champs de comptabilité par point d'accès. Le point de terminaison `GET /access_points/{id}` retourne un point d'accès par son `NAS-IP-Address`.

- **Interface Web** : la page **Points d'accès** montre chaque point d'accès. Développez un point d'accès pour voir ses données complètes, y compris les champs de comptabilité par point d'accès.

Relation avec les données par session

Les totaux par point d'accès sont un agrégat. Pour chaque RADIUS Accounting-Request, OmniTWAG écrit deux enregistrements :

- Les **totaux par point d'accès** décrits ici, conservés dans `APState` et clés par NAS-IP.
- Un **enregistrement par session**, conservé dans `ClientUsage` et clés par `Acct-Session-Id`. C'est la vue de trafic et de comptabilité par abonné en direct, et elle est exposée sur l'API des Sessions, pas sur l'enregistrement du point d'accès. Voir Sessions.

Les totaux par point d'accès comptent les octets et les paquets uniquement lors d'un arrêt, donc ils sont en retard par rapport à la vue par session en direct pendant une session active. Utilisez l'API des Sessions lorsque vous avez besoin de l'utilisation actuelle d'un seul abonné, et les totaux par point d'accès lorsque vous avez besoin du trafic et des comptes de sessions à vie pour un point d'accès.

L'entrée du point d'accès est créée soit par authentification (le point d'accès est enregistré à partir de la demande d'accès, voir Secrets partagés RADIUS) ou, pour les points d'accès uniquement comptables, par le premier Accounting-Request. Un point d'accès n'apparaît jamais si ses paquets sont perdus parce que l'IP source ne correspondait à aucun sous-réseau de crédentiel.

Dépannage

Les totaux ne changent pas

Symptômes : Les totaux d'octets et de paquets pour un point d'accès restent à zéro, même si des clients utilisent le réseau.

Causes possibles :

- Le point d'accès n'envoie pas de paquets d'arrêt. OmniTWAG ajoute des octets et des paquets uniquement lors d'un arr��t.
- Le point d'accès n'envoie pas de comptabilité du tout.

Résolution :

1. Confirmez que le point d'accès envoie des paquets Accounting-Request à OmniTWAG.
2. Confirmez que le point d'accès envoie un arrêt à la fin d'une session.
3. Confirmez que le secret partagé de comptabilité sur le point d'accès correspond à la crédentiale enregistrée. Voir Secrets partagés RADIUS.

Le compte de sessions actives semble trop élevé

Symptômes : La valeur `acct_active_sessions` est supérieure au nombre réel de clients actifs.

Causes possibles :

- Le point d'accès a envoyé des paquets de démarrage mais n'a pas envoyé les paquets d'arrêt correspondants, par exemple après un redémarrage du point d'accès.

Résolution :

1. Confirmez que le point d'accès envoie un arrêt pour chaque session.
2. Définissez un intervalle de mise à jour intermédiaire sur le point d'accès, afin qu'OmniTWAG voie les sessions actives.

3. Le compte actif se corrige de lui-même à mesure que de nouveaux paquets d'arrêt arrivent. Le compte ne descend pas en dessous de zéro.

Cela n'affecte que `acct_active_sessions`. Le chiffre **Clients actifs** sur la page des Points d'accès (et le champ API `client_count`) n'est pas affecté, car il s'agit d'un total de processus de session en direct plutôt que du compteur de comptabilité. Voir Nombre de clients actifs.

Un point d'accès n'apparaît pas

Symptômes : Un point d'accès qui envoie de la comptabilité n'apparaît pas dans la liste.

Causes possibles :

- OmniTWAG a perdu les paquets parce que l'IP source ne correspondait à aucun sous-réseau de crédentiel.

Résolution :

1. Vérifiez la métrique `radius_auth_unauthorized_source_count`. Voir Référence des métriques et Rejets RADIUS.
2. Ajoutez une crédentiale pour le sous-réseau du point d'accès. Voir Secrets partagés RADIUS.

Pages connexes

- Vue d'ensemble
- Sessions
- Secrets partagés RADIUS
- Rejets RADIUS
- Référence des métriques
- Architecture

Vue d'ensemble

Références de performance d'OmniTWAG

Débit d'authentification et de signalisation du plan de contrôle du Trusted WLAN Access Gateway.

Environnement de test

CPU	Intel(R) Core(TM) i5-5250U CPU @ 1.60GHz (2 cœurs)
Runtime	Elixir 1.17.0 / Erlang OTP 27 (JIT) (erts 15.0.1)
Mesure	1 s de préchauffage + 3 s de mesure + 1 s de mémoire par op
Commit	`main` @ `01ec4e8`

Le débit est rapporté en tant que **ips** (itérations par seconde : plus c'est élevé, plus c'est rapide). **Moy** et **Médiane** sont la latence par appel en microsecondes ; la médiane est la meilleure estimation de la latence typique à l'état stable. **Mémoire/op** est la mémoire du tas allouée par appel.

Authentification complète (de bout en bout)

La figure principale : chaque scénario enchaîne la séquence complète par attachement (décodage de l'identité EAP, gestion du vecteur d'authentification, hiérarchie de clés EAP-AKA', construction de défi, MAC, décodage de réponse et vérification de MAC), donc son débit (**ips**) est **des attaches complètes, c'est-à-dire des sessions, par seconde par cœur**. *Mode local* calcule le

vecteur d'authentification sur la boîte (Milenage) ; *mode STa* le reçoit du HSS via SWx/STa.

Scénario	Débit (ips)	Moy (µs)	Médiane (µs)	Mémoire/op
Attachement complet EAP-AKA', mode STa (vecteur du HSS)	17,311	57.77	52.90	7,048 B
Attachement complet EAP-AKA', mode local (vecteur calculé sur la boîte)	12,493	80.05	73.31	10,560 B

Cryptographie

Opérations cryptographiques par authentification : génération de vecteur d'authentification SIM (Milenage), hiérarchie de clés EAP-AKA' (CK'/IK', clés de session, MAC), clés de ré-authentification rapide ERP, et la fonction de dérivation de clé générique 3GPP.

Opération	Débit (ips)	Moy (µs)	Médiane (µs)	Mémoire/op
Dérivation de clé WLCP (EMSK -> clé WLCP)	262,975	3.80	3.54	72 B
Construction du paquet Challenge EAP-AKA' (encodage)	257,991	3.88	0.87	456 B
Dérivation EAP-AKA' CK'/IK'	225,191	4.44	3.93	328 B
Calcul MAC EAP-AKA' (HMAC-SHA-256/16)	223,503	4.47	4.11	72 B
KDF générique 3GPP (HMAC-SHA-256)	198,596	5.04	4.32	248 B
Dérivation ERP rMSK (par ré-auth)	113,269	8.83	8.01	376 B
Ré-authentification rapide complète ERP (rRK -> rIK -> rMSK)	61,496	16.26	14.97	520 B
Milenage compute_all (RES, CK, IK, AK, MAC-A)	52,451	19.07	16.11	3,560 B
Dérivation EAP-AKA' keys (MK -> toutes les clés de session)	31,696	31.55	28.93	1,552 B
Cryptographie d'authentification complète EAP-AKA' (CK'/IK' + toutes les clés)	28,380	35.24	32.59	1,808 B

Opération	Ce qu'elle fait
Construction du paquet Challenge EAP-AKA'	Encode la demande EAP/AKA'-Challenge (AT_RAND, AT_AUTN, AT_KDF, AT_MAC).
Dérivation de clé WLCP	Dérive la clé de protection WLCP à partir de l'EMSK (mode multi-connexion).
KDF générique 3GPP	Fonction de dérivation de clé HMAC-SHA-256 utilisée dans toute la hiérarchie de clés.
Calcul MAC EAP-AKA'	HMAC-SHA-256 AT_MAC sur un paquet EAP, tronqué à 16 octets.
Dérivation EAP-AKA' CK'/IK'	Transforme CK/IK en CK'/IK' lié au nom du réseau d'accès.
Dérivation ERP rMSK	Clé de session par ré-auth pour la ré-authentification EAP (RFC 6696).
Ré-authentification rapide complète ERP	Chaîne de dérivation complète rRK → rIK → rMSK pour une ré-auth rapide.
Milenage compute_all	Un vecteur d'authentification AKA : RES, CK, IK, AK, MAC-A.
Dérivation EAP-AKA' keys	Développe la clé maître en K_encr, K_aut, K_re, MSK, EMSK.
Cryptographie d'authentification complète EAP-AKA'	Dérivation CK'/IK' plus l'expansion complète de la clé de session.

Codecs de paquets

Opérations d'encodage/décodage du plan de signalisation : analyse des paquets EAP, gestion des messages WLCP (TS 24.244) pour le mode multi-connexion, et l'attribut de clé MPPE construit sur chaque RADIUS Access-Accept.

Opération	Débit (ips)	Moy (µs)	Médiane (µs)	Mémoire/op
Décodage WLCP (Demande de connexion PDN)	4,229,558	0.24	0.20	448 B
Encodage WLCP (Demande de connexion PDN)	2,492,441	0.40	0.24	160 B
Aller-retour encodage + décodage WLCP	1,989,407	0.50	0.40	608 B
Décodage EAP paquet d'identité	336,816	2.97	2.66	1,136 B
Construction MPPE VSA (Send-Key, RC4 + MD5)	135,026	7.41	5.32	1,296 B

Opération	Ce qu'elle fait
Décodage WLCP	Analyse une demande de connexion PDN WLCP depuis le réseau.
Encodage WLCP	Sérialise une demande de connexion PDN WLCP vers le réseau.
Aller-retour encodage + décodage WLCP	Encodage puis décodage d'un message WLCP.
Décodage EAP paquet d'identité	Analyse une réponse d'identité EAP (extraction NAI).
Construction MPPE VSA	Construit le VSA RADIUS MS-MPPE-Send-Key (RC4 + MD5) portant le MSK.

Secrets partagés RADIUS (par groupe d'IP source)

Vue d'ensemble

OmniTWAG authentifie chaque client RADIUS (un point d'accès ou NAS) avec un secret partagé. Cette fonctionnalité permet à un opérateur de gérer **de nombreux** secrets partagés, chacun lié à un ou plusieurs sous-réseaux d'IP source. Un groupe de points d'accès partageant un sous-réseau partage également un secret. L'opérateur ajoute, modifie et supprime ces identifiants à la demande via l'API REST ou l'interface web du panneau de contrôle. Le stockage est persistant, donc les identifiants survivent à un redémarrage.

Table des matières

- Concepts
- Comment l'authentification utilise un identifiant
- Modèle d'identifiant
- Démarrage et facturation par AP
- API REST
- Interface web
- Configuration
- Migration d'un seul secret statique
- Métriques
- Dépannage

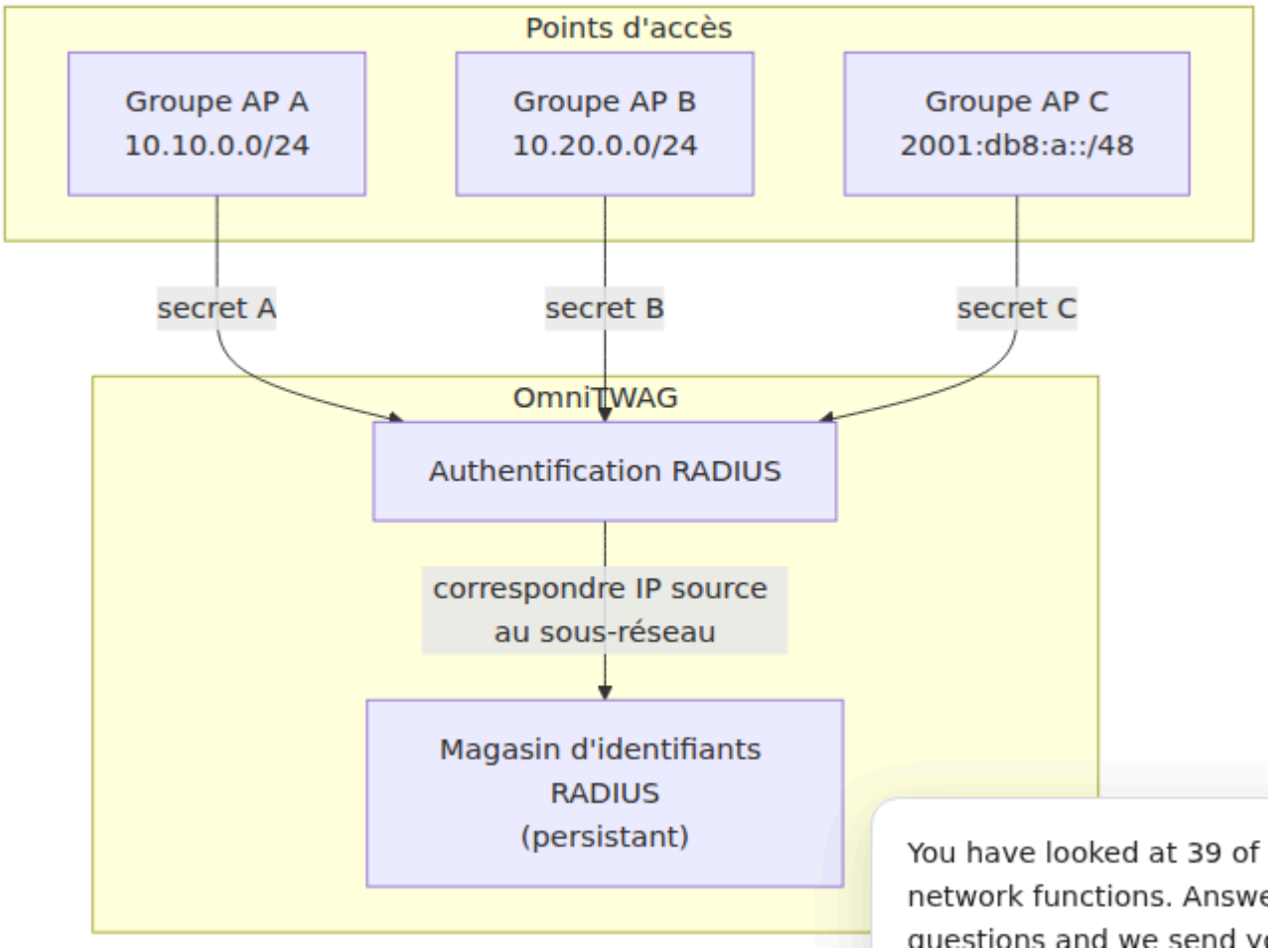
Concepts

Un **identifiant** lie un secret partagé à un ou plusieurs **sous-réseaux source**. Un sous-réseau source est un bloc CIDR. Le sous-réseau peut être IPv4, IPv6 ou

un mélange des deux.

L'**IP source** est l'adresse IP du point d'accès qui envoie le paquet RADIUS. OmniTWAG lit l'IP source à partir du paquet UDP reçu. OmniTWAG n'utilise pas l'attribut `NAS-IP-Address` pour la sélection du secret, car un attaquant peut définir cet attribut sur n'importe quelle valeur.

Un sous-réseau d'identifiant **autorise** également le point d'accès. Si l'IP source correspond à un sous-réseau d'identifiant, le point d'accès est autorisé. Si l'IP source ne correspond à aucun sous-réseau d'identifiant, OmniTWAG rejette le paquet. Cela signifie que lorsqu'un opérateur ajoute un identifiant pour un nouveau point d'accès, le point d'accès est autorisé en même temps.

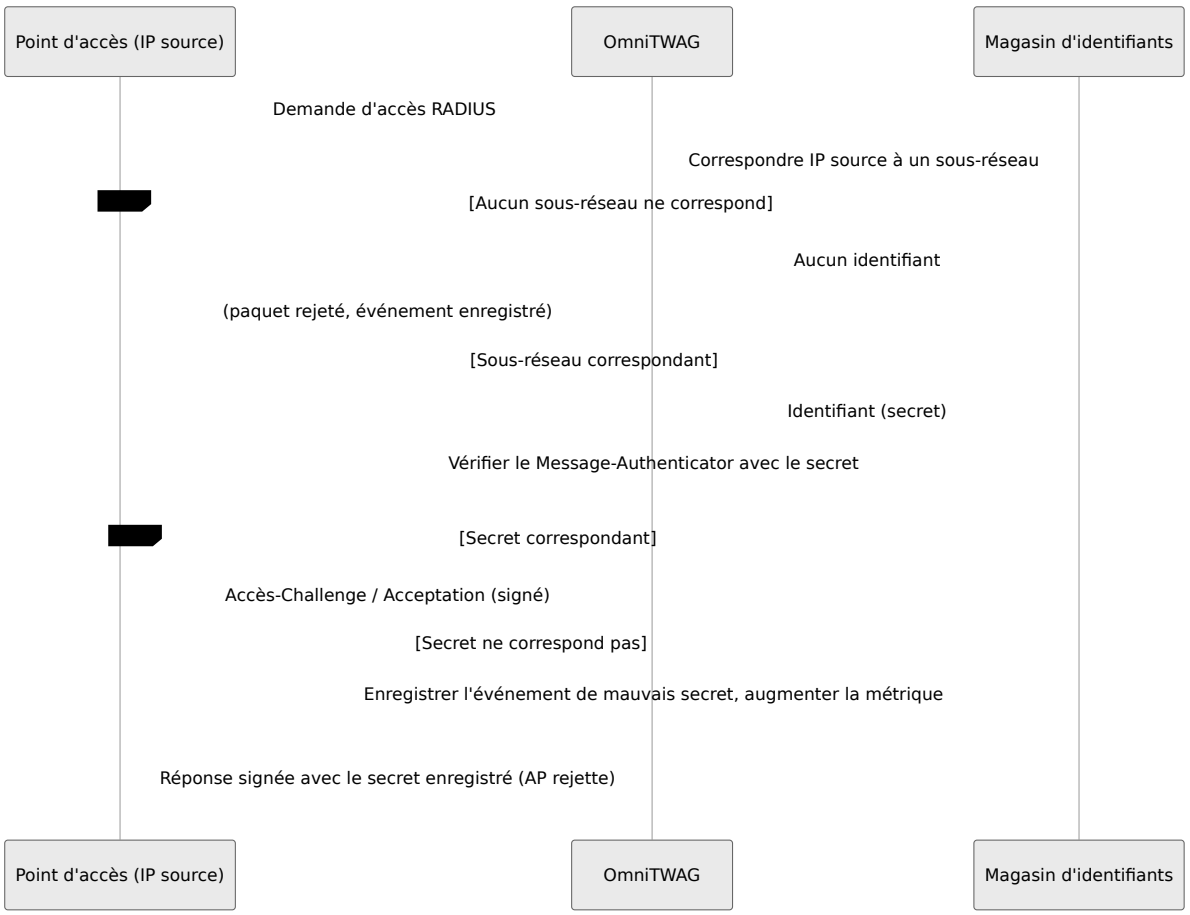


Comment l'authentification utilise un identifiant

OmniTWAG résout le secret partagé pour chaque paquet RADIUS entrant. Les étapes sont :

1. OmniTWAG lit l'IP source du paquet.
2. OmniTWAG trouve l'identifiant dont le sous-réseau correspond à l'IP source. Si plus d'un sous-réseau correspond, OmniTWAG utilise le sous-réseau avec le **préfixe le plus long** (la correspondance la plus spécifique).
3. Si aucun sous-réseau ne correspond, OmniTWAG rejette le paquet. La source n'est pas autorisée.
4. Si un sous-réseau correspond, OmniTWAG utilise le secret de cet identifiant pour décoder le paquet, vérifier le `Message-Authenticator`, chiffrer les clés MPPE et signer la réponse.

Si un sous-réseau correspond mais que le `Message-Authenticator` ne vérifie pas, le point d'accès détient un secret différent de celui enregistré. OmniTWAG enregistre cet événement et augmente une métrique. OmniTWAG traite toujours le paquet. La réponse est signée avec le secret enregistré, donc le point d'accès rejette la réponse et l'authentification échoue. Ce comportement rend un secret mal configuré facile à trouver. Il ne rejette pas le paquet, donc un cas limite valide n'est jamais rejeté par erreur.



Modèle d'identifiant

Chaque identifiant a ces champs.

Champ	Type	Description
`id`	Chaîne	Identifiant unique (UUID). Le serveur définit cette valeur.
`name`	Chaîne	Étiquette lisible par l'homme, par exemple `Points d'accès CBD de Melbourne`.
`secret`	Chaîne	Secret partagé RADIUS.
`subnets`	Liste de Chaînes	Un ou plusieurs blocs CIDR (IPv4, IPv6 ou un mélange), par exemple `["10.10.0.0/24", "2001:db8:a::/48"]`.
`breakout_type`	Chaîne	Gestion du plan de données pour les sessions de ce point d'accès : `tunnélisé`, `nswo` ou `local`. Par défaut `tunnélisé`. Voir Démarrage et facturation par AP.
`address_source`	Chaîne	Pour le démarrage `nswo`, comment l'IP UE est attribuée : `pool` ou `dhcp`. Par défaut `pool`. Voir Démarrage et facturation par AP.
`charging_mode`	Chaîne	Comportement de facturation en ligne Gy : `charged`, `uncharged` ou `auth_only`. Par défaut `charged`. Voir Démarrage et facturation par AP.
`ip_pool_id`	Chaîne	`id` optionnel d'un pool IP qui adresse les sessions `nswo` de ce point d'accès. `null` signifie pas de pool.
`inserted_at`	Entier	Heure à laquelle l'identifiant a été créé, en millisecondes depuis l'époque Unix. Le serveur définit cette valeur.

Champ	Type	Description
`updated_at`	Entier	Heure à laquelle l'identifiant a été modifié pour la dernière fois, en millisecondes depuis l'époque Unix. Le serveur définit cette valeur.

L'opérateur définit `name`, `secret`, `subnets`, `breakout_type`, `address_source`, `charging_mode` et `ip_pool_id`. Le serveur définit `id`, `inserted_at` et `updated_at`.

Démarrage et facturation par AP

Chaque identifiant contrôle également le plan de données et la facturation pour les points d'accès du groupe. Les valeurs par défaut préservent le comportement standard : un tunnel GTP vers le PGW avec facturation en ligne.

breakout_type

`breakout_type` définit comment le trafic des abonnés quitte le TWAG.

Valeur	Description
`tunnélisé`	OmniTWAG établit une session GTP-C/GTP-U vers le PGW et transfère le trafic des abonnés via le tunnel. C'est la valeur par défaut. Voir Interface GTP-C S2a.
`nswo`	Déchargement WLAN non transparent. OmniTWAG ancre la session localement et ne tunnelise pas vers le PGW. OmniTWAG attribue l'IP UE à partir d'un pool IP. Le trafic des abonnés transite par le TWAG et se décharge vers le réseau local.
`local`	Le point d'accès possède le plan de données. OmniTWAG authentifie l'abonné et n'attribue pas d'IP. L'UE obtient son adresse du réseau local.

address_source

`address_source` s'applique uniquement lorsque `breakout_type` est `nswo`. Il définit comment l'IP UE est attribuée. Voir Pools IP.

Valeur	Description
`pool`	OmniTWAG pré-alloue une adresse fixe à partir du pool IP associé pour chaque session. C'est la valeur par défaut.
`dhcp`	Le serveur DHCP d'OmniTWAG attribue une adresse à partir de la plage du pool lorsque l'UE envoie un DHCP DISCOVER.

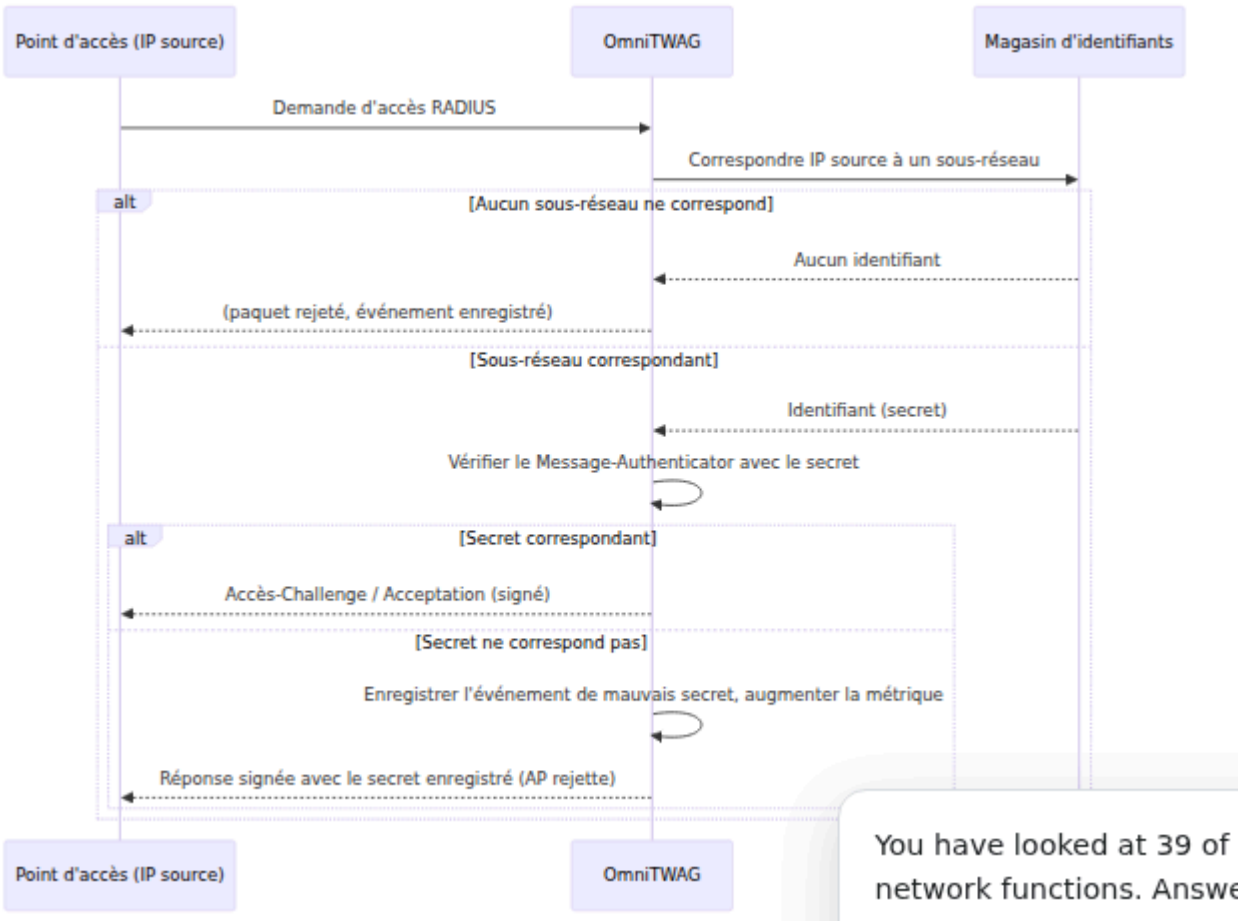
charging_mode

`charging_mode` définit le comportement de facturation en ligne Gy. Voir Facturation en ligne.

Valeur	Description
`charged`	OmniTWAG ouvre une session de facturation en ligne Gy et rapporte l'utilisation au OCS. C'est la valeur par défaut.
`uncharged`	OmniTWAG établit la session de données mais n'ouvre pas de session Gy. Utilisez cette valeur pour un accès à tarif fixe ou non facturé.
`auth_only`	OmniTWAG authentifie l'abonné mais n'établit pas de session de données. Utilisez cette valeur lorsque un autre système fournit la connectivité.

Comment les champs interagissent

`charging_mode: auth_only` a la priorité. Lorsqu'un identifiant utilise `auth_only`, OmniTWAG n'établit pas de session de données, donc `breakout_type` et `address_source` n'ont aucun effet.



API REST

L'API REST d'OmniTWAG sert ces points de terminaison via HTTPS sur le port `api_ex` configuré (`8444` par défaut). Le chemin de base est `/radius_credentials`.

Méthode	Chemin	Action	Description
GET	`/radius_credentials`	index	Lister tous les identifiants.
GET	`/radius_credentials/{id}`	show	Obtenir un identifiant par `id`.
POST	`/radius_credentials`	create	Créer un identifiant.
PUT	`/radius_credentials/{id}`	update	Modifier un identifiant.
DELETE	`/radius_credentials/{id}`	delete	Supprimer un identifiant.

Corps de la requête

`POST` et `PUT` acceptent un corps JSON avec ces champs.

Champ	Type	Requis	Description
`name`	Chaîne	Non	Étiquette lisible par l'homme.
`secret`	Chaîne	Oui	Secret partagé RADIUS. Ne doit pas être vide.
`subnets`	Liste de Chaînes	Oui	Un ou plusieurs blocs CIDR valides. Ne doit pas être vide.
`breakout_type`	Chaîne	Non	`tunnélisé`, `nswo` ou `local`. Par défaut `tunnélisé`. Voir Démarrage et facturation par AP.
`address_source`	Chaîne	Non	`pool` ou `dhcp` (pour `nswo`). Par défaut `pool`.
`charging_mode`	Chaîne	Non	`charged`, `uncharged` ou `auth_only`. Par défaut `charged`.
`ip_pool_id`	Chaîne	Non	`id` d'un pool IP pour l'adressage `nswo`. Une chaîne vide signifie pas de pool.

`PUT` ne modifie que les champs dans le corps. Les champs qui ne sont pas dans le corps conservent leur valeur actuelle.

Réponses

Statut	Signification
`200 OK`	La requête a réussi (index, show, update, delete).
`201 Created`	L'identifiant a été créé.
`404 Not Found`	Aucun identifiant n'a l'`id` donné.
`422 Unprocessable Entity`	Le `secret` est vide, un sous-réseau n'est pas un bloc CIDR valide, ou `breakout_type` / `address_source` / `charging_mode` n'est pas une valeur valide.

Exemple : lister les identifiants

Requête :

```
GET /radius_credentials
```

Réponse :

```
{
  "count": 1,
  "radius_credentials": [
    {
      "id": "0d1e2f3a-4b5c-6d7e-8f90-1a2b3c4d5e6f",
      "name": "Points d'accès CBD de Melbourne",
      "secret": "s3cr3t",
      "subnets": ["10.10.0.0/24", "2001:db8:a::/48"],
      "inserted_at": 1734512400000,
      "updated_at": 1734512400000
    }
  ]
}
```

Exemple : créer un identifiant

Requête :

```
POST /radius_credentials
```

```
{
  "name": "Points d'accès CBD de Melbourne",
  "secret": "s3cr3t",
  "subnets": ["10.10.0.0/24", "2001:db8:a::/48"]
}
```

Une requête réussie renvoie `201 Created` avec le nouvel identifiant, y compris son `id`.

Exemple : un sous-réseau invalide

Requête :

```
{
  "secret": "s3cr3t",
  "subnets": ["10.10.0.0/24", "not-a-subnet"]
}
```

Réponse (`422 Unprocessable Entity`):

```
{
  "error": "sous-réseau CIDR invalide(s) : not-a-subnet"
}
```

Interface web

Le panneau de contrôle a une page **Secrets RADIUS** à `/radius_credentials`. La page liste chaque identifiant avec son nom, ses sous-réseaux, un secret masqué et l'heure de la dernière modification. La page dispose d'un formulaire

pour ajouter un identifiant et de contrôles pour modifier ou supprimer un identifiant. Entrez un ou plusieurs sous-réseaux dans le champ des sous-réseaux. Séparez les sous-réseaux par une virgule ou une nouvelle ligne.

Configuration

Le magasin d'identifiants RADIUS conserve ses données dans un fichier DETS. Définissez le chemin du fichier dans le bloc `radius_config`.

```
config :omnitwag,
  radius_config: %{
    # Chemin vers le magasin d'identifiants RADIUS persistant.
    credentials_dets_path: "priv/radius_credentials.dets",
    # Champs de secret unique hérités (voir "Migration" ci-
dessous).
    allowed_source_subnets: [],
    secret: "123456"
  }
```

Paramètre	Type	Requis	Par défaut
`credentials_dets_path`	Chaîne	Non	`priv/radius_credentials.`
`secret`	Chaîne	Non	`123456`
`allowed_source_subnets`	Liste de Chaînes	Non	`[]`

Paramètre	Type	Requis	Par défaut

Migration d'un seul secret statique

Les versions antérieures utilisaient un seul `secret` statique et une liste d'autorisation `allowed_source_subnets` optionnelle. Au premier démarrage, si le magasin d'identifiants est vide, OmniTWAG initialise un identifiant à partir de ces champs hérités. L'initialisation ne s'exécute que lorsque les deux conditions suivantes sont vraies :

- Le magasin n'a pas d'identifiants.
- Le `secret` est défini et `allowed_source_subnets` n'est pas vide.

L'identifiant initialisé utilise le `secret` hérité et les sous-réseaux hérités. Cela permet à un déploiement existant de continuer à fonctionner après une mise à niveau. OmniTWAG enregistre un message lorsqu'il initialise un identifiant.

Si `allowed_source_subnets` est vide, OmniTWAG ne initialise pas d'identifiant. Dans ce cas, OmniTWAG rejette toutes les demandes RADIUS jusqu'à ce que l'opérateur ajoute un identifiant. Ajoutez des identifiants via l'API REST ou l'interface web.

L'authentification par paquet ne lit pas `secret` ou `allowed_source_subnets`. Ces champs ne servent que d'initialisation au premier démarrage. Gérez les identifiants actifs via l'API ou l'interface web.

Métriques

OmniTWAG expose ces compteurs sur le point de terminaison Prometheus. Voir la Référence des métriques pour le point de terminaison et des exemples de requêtes.

radius_auth_unauthorized_source_count

Type : Compteur
Description : Nombre de paquets RADIUS rejetés parce que l'IP source ne correspondait à aucun sous-réseau d'identifiant.

Étiquette	Description
`source`	IP source du paquet rejeté.

radius_auth_bad_secret_count

Type : Compteur
Description : Nombre de paquets RADIUS dont l'IP source correspondait à un identifiant, mais dont le `Message-Authenticator` ne vérifiait pas. Le point d'accès détient un secret différent de celui enregistré.

Étiquette	Description
`source`	IP source du point d'accès.
`credential_id`	`id` de l'identifiant correspondant.

Dépannage

Un point d'accès ne reçoit pas de réponse

Symptômes : Le point d'accès envoie des demandes RADIUS mais ne reçoit aucune réponse. La métrique `radius_auth_unauthorized_source_count` augmente.

Causes possibles :

- Aucun sous-réseau d'identifiant ne contient l'IP source du point d'accès.
- Le point d'accès envoie depuis une IP différente de celle attendue, par exemple une adresse NAT.

Résolution :

1. Trouvez l'IP source du point d'accès dans les journaux d'OmniTWAG.
2. Confirmez qu'un sous-réseau d'identifiant contient cette IP.
3. Si aucun identifiant ne couvre l'IP, ajoutez un identifiant pour le sous-réseau correct.

Un point d'accès est autorisé mais l'authentification échoue

Symptômes : La métrique `radius_auth_bad_secret_count` augmente pour le point d'accès. Les clients ne peuvent pas s'authentifier.

Causes possibles :

- Le point d'accès détient un secret partagé différent de celui de l'identifiant enregistré.

Résolution :

1. Lisez l'étiquette `credential_id` sur la métrique, ou trouvez le message de mauvais secret dans les journaux.
2. Comparez le secret sur le point d'accès avec le secret dans l'identifiant.

3. Corrigez le secret sur le point d'accès, ou mettez à jour l'identifiant pour qu'il corresponde.

Les identifiants sont perdus après un redémarrage

Symptômes : La liste des identifiants est vide après un redémarrage. OmniTWAG rejette toutes les demandes.

Causes possibles :

- Le `credentials_dets_path` pointe vers un répertoire qui n'est pas écrivable, ou vers un chemin qui est effacé au redémarrage.

Résolution :

1. Définissez `credentials_dets_path` sur un répertoire de données persistant et écrivable.
2. Confirmez que le processus OmniTWAG peut écrire à ce chemin.
3. Ajoutez à nouveau les identifiants via l'API ou l'interface web.

Rejets RADIUS

Vue d'ensemble

OmniTWAG suit les sources RADIUS dont les paquets ont été rejetés. Cela permet à un opérateur de repérer d'un coup d'œil un point d'accès inconnu ou mal configuré. La fonctionnalité est le miroir de Secrets Partagés RADIUS : le magasin de crédentiels enregistre les sources que OmniTWAG **accepte**, et le traqueur de rejets enregistre les sources que OmniTWAG **rejette**.

Le traqueur conserve ses données en mémoire. Les données ne sont pas persistantes. Le traqueur se réinitialise au redémarrage. C'est intentionnel. Voir Dépannage.

Table des Matières

- Concepts
- Comment un paquet atteint le traqueur
- Raisons de rejet
- Modèle d'entrée
- Comportement du magasin
- API REST
- Métriques
- Dépannage

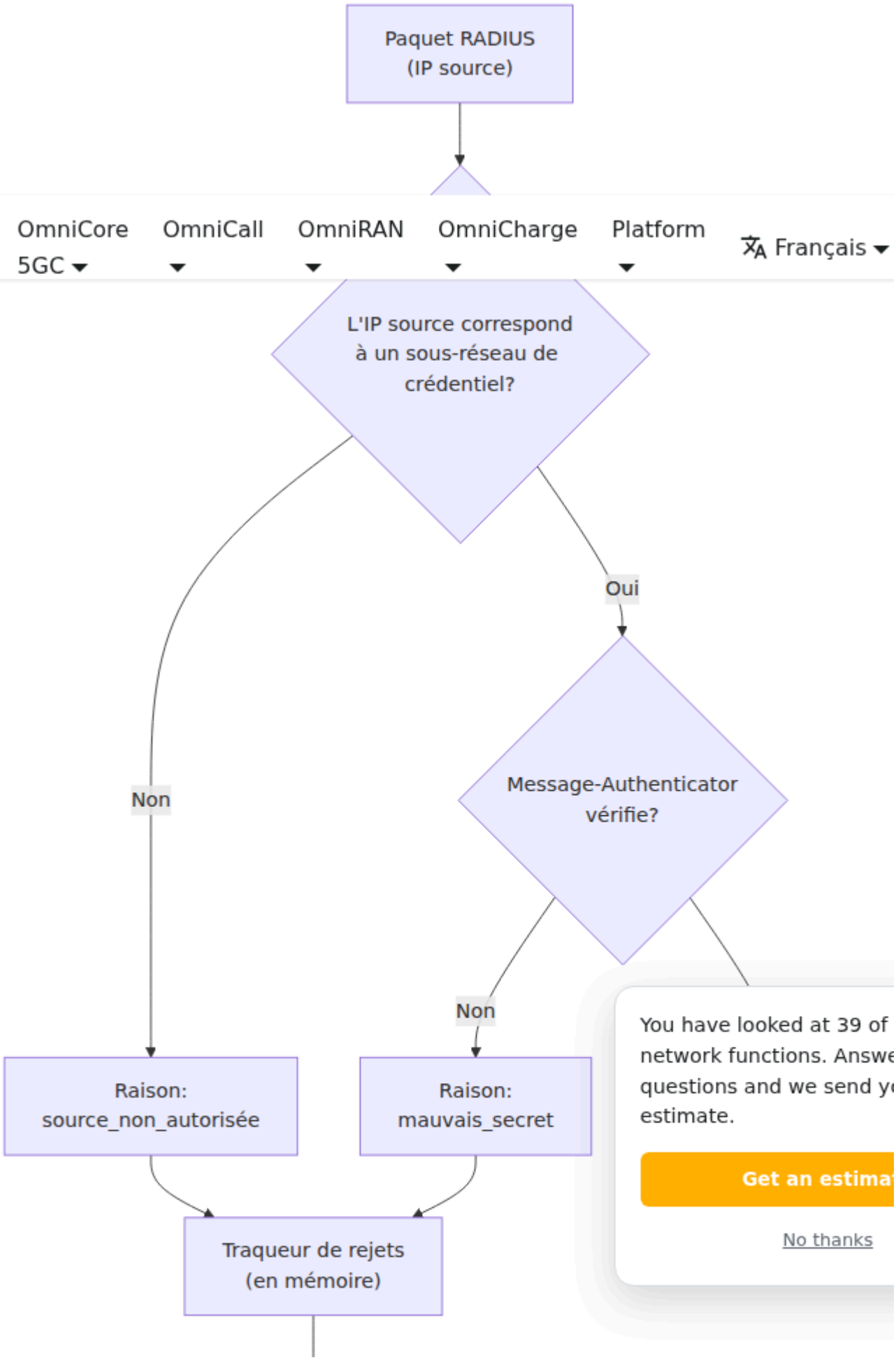
Concepts

Une **source rejetée** est une adresse IP qui a envoyé un paquet RADIUS que OmniTWAG n'a pas accepté. Le traqueur enregistre une entrée par paire unique de **source** et **raison**. Si la même source est rejetée plusieurs fois pour la même raison, OmniTWAG augmente un compteur sur l'entrée existante. OmniTWAG n'ajoute pas une nouvelle ligne.

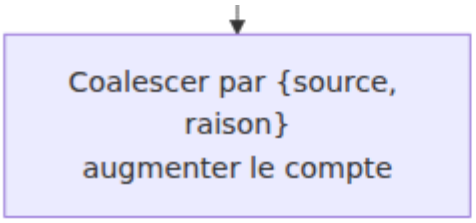
L'**IP source** est l'adresse IP qui a envoyé le paquet RADIUS. OmniTWAG lit l'IP source à partir du paquet UDP reçu. OmniTWAG n'utilise pas l'attribut `NAS-IP-Address`, car un attaquant peut définir cet attribut à n'importe quelle valeur.

Comment un paquet atteint le traqueur

OmniTWAG compare chaque paquet RADIUS entrant avec les sous-réseaux de crédentiels. Un paquet est rejeté lorsqu'il échoue à l'un des deux contrôles. Le traqueur enregistre la source et la raison pour chaque paquet rejeté.



You have looked at 39 of
network functions. Answe
questions and we send y
estimate.
Get an estima
No thanks



Raisons de rejet

Une source est rejetée pour l'une des deux raisons.

Raison	Signification
`source_non_autorisée`	L'IP source ne correspond à aucun sous-réseau de crédentiel. OmniTWAG a rejeté le paquet. C'est le cas du point d'accès inconnu.
`mauvais_secret`	Un sous-réseau de crédentiel a correspondu, mais le `Message-Authenticator` du paquet n'a pas vérifié. Le point d'accès détient un secret partagé différent de celui enregistré.

Pour une entrée `mauvais_secret`, le traqueur enregistre également le crédentiel qui a correspondu. Cela indique à l'opérateur quel crédentiel vérifier.

Modèle d'entrée

Chaque entrée a ces champs.

Champ	Type	Description
`source`	Chaîne	IP source du paquet rejeté. Cette valeur est l'identifiant de l'entrée.
`raison`	Chaîne	Raison du rejet : `source_non_autorisée` ou `mauvais_secret`.
`count`	Entier	Nombre de fois qu'OmniTWAG a rejeté cette source pour cette raison.
`first_seen`	Entier	Temps auquel la source a été rejetée pour la première fois, en millisecondes depuis l'époque Unix.
`last_seen`	Entier	Temps auquel la source a été rejetée pour la dernière fois, en millisecondes depuis l'époque Unix.
`credential_id`	Chaîne	`id` du crédentiel correspondant. Défini uniquement lorsque la `raison` est `mauvais_secret`.
`credential_name`	Chaîne	`nom` du crédentiel correspondant. Défini uniquement lorsque la `raison` est `mauvais_secret`.

Les champs `credential_id` et `credential_name` sont vides pour une entrée `source_non_autorisée`, car aucun crédentiel n'a correspondu.

Comportement du magasin

Le traqueur est un magasin en mémoire. Il a ces propriétés.

- **Non-persistant.** Le magasin se réinitialise au redémarrage. Il ne survit pas à un redémarrage du processus.

- **Coalescé.** OmniTWAG enregistre une entrée par paire de `source` et `raison`. Un rejet répété augmente le `count` et met à jour le `last_seen`. Il n'ajoute pas de ligne.
- **Limité.** Le magasin conserve un nombre maximum de sources. Le maximum par défaut est 256.
- **Éviction par le moins récemment vu.** Lorsque le magasin est plein, OmniTWAG supprime l'entrée avec la plus ancienne valeur de `last_seen` pour faire de la place pour une nouvelle source. Le magasin est limité car une IP source peut être usurpée. Un flot d'IP sources usurpées ne peut pas faire croître le magasin sans limite.

API REST

L'API REST d'OmniTWAG dessert ces points de terminaison sur le port API configuré (`8444` par défaut). Le chemin de base est `/radius_rejections`.

Méthode	Chemin	Action	Description
GET	`/radius_rejections`	index	Lister les sources rejetées. La liste est paginée. Rechercher par `source` ou `raison`.
DELETE	`/radius_rejections/{id}`	delete	Oublier une source. `{id}` est l'IP source.

Utilisez `DELETE` pour oublier une source après l'avoir résolue, par exemple après avoir ajouté un crédentiel qui couvre l'IP source.

Réponses

Statut	Signification
`200 OK`	La demande a réussi.
`404 Not Found`	Aucune entrée n'a l'IP source donnée.

Exemple : lister les sources rejetées

Demande :

```
GET /radius_rejections
```

Réponse :

```
{
  "count": 2,
  "radius_rejections": [
    {
      "source": "203.0.113.7",
      "reason": "source_non_autorisée",
      "count": 42,
      "first_seen": 1734512400000,
      "last_seen": 1734512999000,
      "credential_id": null,
      "credential_name": null
    },
    {
      "source": "10.10.0.15",
      "reason": "mauvais_secret",
      "count": 3,
      "first_seen": 1734512500000,
      "last_seen": 1734512890000,
      "credential_id": "0d1e2f3a-4b5c-6d7e-8f90-1a2b3c4d5e6f",
      "credential_name": "APs du CBD de Melbourne"
    }
  ]
}
```


Exemple : oublier une source

Demande :

```
DELETE /radius_rejections/203.0.113.7
```

Réponse :


```
{
  "deleted": "203.0.113.7"
}
```

Métriques

Le traqueur de rejets complète deux compteurs Prometheus que la fonctionnalité de crédentiels documente. Chaque compteur augmente en même temps qu'OmniTWAG enregistre un rejet.

Compteur	Raison de rejet
`radius_auth_unauthorized_source_count`	`source_non_autorisée`
`radius_auth_bad_secret_count`	`mauvais_secret`

Les compteurs donnent des totaux dans le temps. Le traqueur donne une vue en direct, par source. Voir RADIUS Secrets Partagés pour les définitions des compteurs, et la Référence des Métriques pour le point de terminaison Prometheus et des exemples de requêtes.

Dépannage

Un point d'accès inconnu apparaît dans la liste

Symptômes : Une source apparaît avec la `raison` définie sur `source_non_autorisée`. Le compteur `radius_auth_unauthorized_source_count` augmente pour cette source.

Causes possibles :

- Aucun sous-réseau de crédentiel ne contient l'IP source.
- Le point d'accès est derrière un NAT. L'IP source RADIUS que voit OmniTWAG diffère de l'adresse du point d'accès lui-même.

Résolution :

1. Lisez le champ `source` de l'entrée.
2. Confirmez qu'il s'agit d'un point d'accès connu.

3. Ajoutez un sous-réseau de crédentiel qui couvre l'IP source que voit réellement OmniTWAG. Pour un cas NAT, utilisez l'adresse traduite, pas l'adresse du point d'accès. Voir Secrets Partagés RADIUS.
4. Supprimez l'entrée avec `DELETE /radius_rejections/{id}` pour oublier la source.

Un point d'accès connu apparaît comme mauvais_secret

Symptômes : Une source connue apparaît avec la `raison` définie sur `mauvais_secret`. Le compteur `radius_auth_bad_secret_count` augmente pour cette source.

Causes possibles :

- Le point d'accès détient un secret partagé différent de celui du crédentiel enregistré.

Résolution :

1. Lisez les champs `credential_id` et `credential_name` de l'entrée. Ceux-ci nomment le crédentiel correspondant.
2. Comparez le secret sur le point d'accès avec le secret dans ce crédentiel.
3. Corrigez le secret sur le point d'accès, ou mettez à jour le crédentiel pour qu'il corresponde. Voir Secrets Partagés RADIUS.
4. Supprimez l'entr��e avec `DELETE /radius_rejections/{id}` pour oublier la source.

La liste est vide après un redémarrage

Symptômes : La liste des rejets est vide après un redémarrage, même si des sources étaient listées avant le redémarrage.

Cause :

- C'est un comportement attendu. Le traqueur conserve ses données en mémoire. Les données ne sont pas persistantes. Le magasin se réinitialise au redémarrage.

Résolution :

- Aucune action n'est nécessaire. De nouveaux rejets remplissent à nouveau la liste. Utilisez les compteurs `radius_auth_unauthorized_source_count` et `radius_auth_bad_secret_count` pour un historique qui survit à un redémarrage.

Vue d'ensemble

S2a / Interface GTP-C au PGW

Vue d'ensemble

L'interface S2a connecte l'OmniTWAG au Gateway de Réseau de Données Packet (PGW). Elle transporte la connectivité PDN basée sur GTP pour un Réseau d'Accès WLAN de Confiance (TWAN). L'OmniTWAG agit en tant que point de terminaison GTP-C du TWAN. Ce rôle est similaire à celui d'un ePDG sur l'interface S2b, mais il utilise des types d'interface S2a et un Identifiant TWAN.

L'interface S2a a deux plans :

- **GTP-C (plan de contrôle)** : crée, modifie et supprime des sessions PDN. Elle gère également la gestion des chemins. Ce document couvre GTP-C.
- **GTP-U (plan utilisateur)** : transporte les données UE dans des tunnels GTP. Voir le guide Plan Utilisateur.

GTP-C sur S2a suit 3GPP TS 29.274 (GTPv2-C). Le TWAG utilise les types de messages GTPv2-C et les éléments d'information (IEs) avec des types d'interface spécifiques à S2a.

Table des Matières

- Où S2a s'intègre : Les Trois Modes de Détachement
- Rôle dans le Cycle de Vie de la Session
- Types d'Interface S2a
- Établissement de la Session
  - Déclencheur
  - Demande de Création de Session
  - Support par Défaut
  - Réponse à la Création de Session
  - Sessions d'Urgence

- Gestion des Supports
- Démontage de la Session
- Gestion des Chemins
- Configuration
- Tables de Référence
- Observabilité
- Dépannage
- Références 3GPP

Où S2a s'intègre : Les Trois Modes de Détachement

L'OmniTWAG résout un mode de plan de données par point d'accès (AP), à partir des informations d'identification RADIUS qui correspondent à l'AP. La valeur `breakout_type` sur l'identification sélectionne l'un des trois modes. Voir Informations d'identification RADIUS pour les paramètres `breakout_type`, `address_source` et `ip_pool_id`.

Mode (`breakout_type`)	Qui attribue l'IP UE	Plan de données	S2a / GTP utilisé
`tunneled`	PGW (dans l'IE d'Allocation d'Adresse PDN)	GRE de l'AP au TWAG, puis GTP-U sur S2a vers le PGW, puis vers Internet par le PGW	Oui. Ce document.
`nswo`	TWAG, à partir d'un pool IP	GRE de l'AP au TWAG, puis détachement local vers Internet (NAT) au TWAG	Non. Pas de session PGW.
`local`	Le réseau local à l'AP	L'AP possède le plan de données. Le TWAG ne transporte aucun trafic.	Non. Pas de session PGW, pas d'IP.

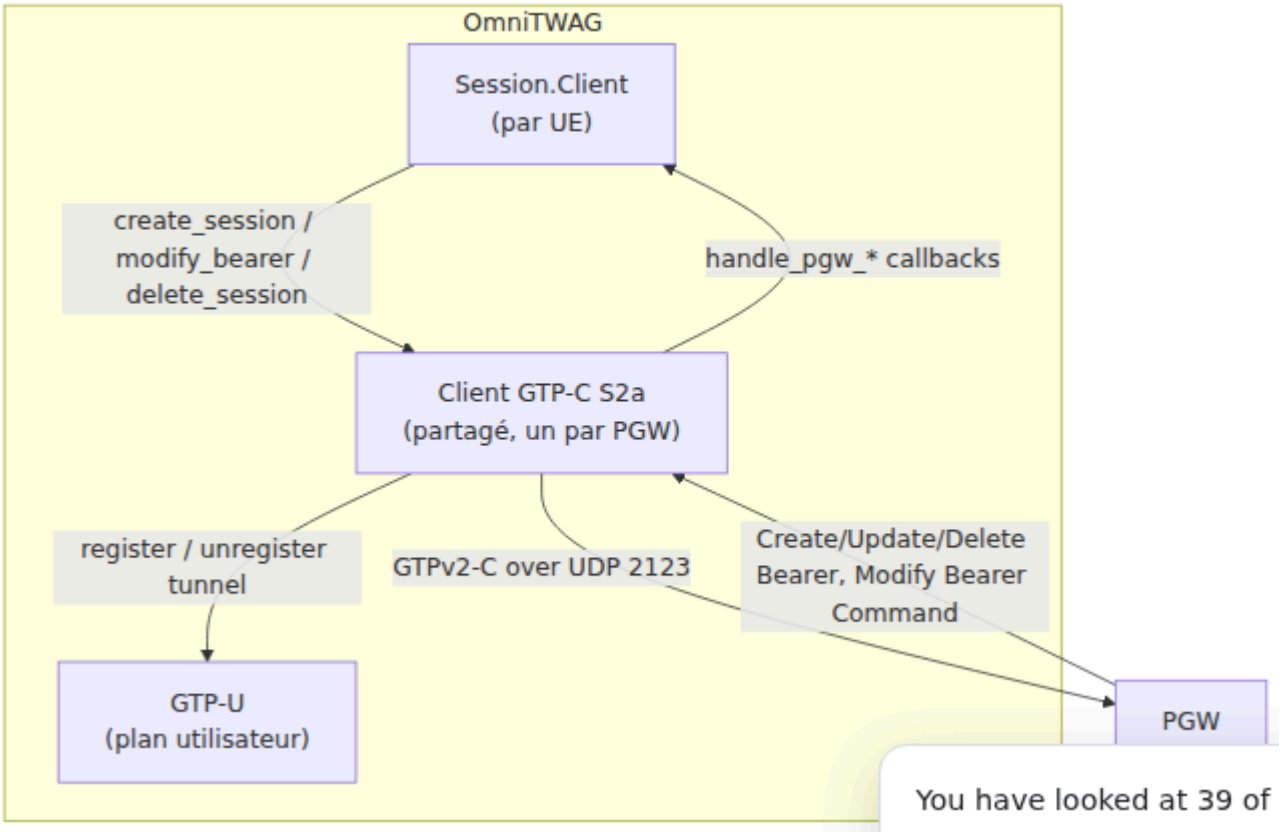
GTP-C S2a s'applique uniquement au mode `tunneled`. Dans le mode `tunneled`, le PGW attribue l'adresse IP UE et applique la politique et la facturation. Dans les modes `nswo` et `local`, le TWAG ne crée pas de session PDN, donc aucun message S2a n'est envoyé.

La facturation dans le mode `tunneled` est effectuée par le PGW, pas par le TWAG. La facturation en ligne Gy du côté TWAG dans Facturation en Ligne s'applique aux modes de détachement local.

> Remarque : Un gestionnaire global séparé de Non-Détachement WLAN sans Seamless (NSWO) (`Twag.NSWO.Handler`, activé avec `nswo.enabled: true`) fournit un détachement local pour chaque session lorsque GTP n'est pas utilisé. Voir la section Détachement Local du guide du Plan Utilisateur. Le type de détachement `nswo` par AP décrit ci-dessus est le modèle basé sur les informations d'identification et est préféré.

Rôle dans le Cycle de Vie de la Session

Le client GTP-C est un processus unique et partagé (`Twag.Gtp.GtpcS2a`). Il gère toutes les sessions contre un PGW configuré. Chaque session UE `tunneled` authentifiée entraîne un échange de Création de Session. Le processus `Session.Client` par session appelle le client GTP-C. Il n'envoie pas lui-même de messages GTPv2-C.



Types d'Interface S2a

Le TWAG utilise des types d'interface S2a F-TEID. Ceux-ci sont différents de S2b (ePDG), qui utilise les types 30 et 31. Les valeurs suivent TS 29.274 Table 8.22-1.

Type d'Interface	Valeur	Direction	Utilisation
`s2a_twan_gtp_c`	35	Contrôle TWAG	F-TEID TWAG dans la Demande de Création de Session
`s2a_twan_gtp_u`	34	Utilisateur TWAG	F-TEID de support TWAG (GTP-U)
`s2a_pgw_gtp_c`	36	Contrôle PGW	F-TEID PGW dans la Réponse à la Création de Session
`s2a_pgw_gtp_u`	37	Utilisateur PGW	F-TEID de support PGW (GTP-U)

Le type RAT est toujours WLAN (valeur 3).

Établissement de la Session

Déclencheur

Une authentification EAP réussie démarre la session. L'AP envoie un Accounting-Start, ce qui déclenche le traitement post-acceptation. Le `Session.Client` résout le mode de détachement de l'AP. Pour le mode `tunneled`, il appelle le client GTP-C pour créer la session PDN. Si le paramètre `gtp.enabled` est `false`, le TWAG saute S2a. Si la session n'a pas d'IMSI, le TWAG ne crée pas la session. Pour les modes `nswo` et `local`, le TWAG n'envoie aucun message S2a. Voir Sessions pour le cycle de vie complet de la session.

Demande de Création de Session

Le TWAG envoie une Demande de Création de Session (type de message selon TS 29.274 §7.2.1) au PGW. Le TWAG génère un TEID de plan de contrôle nouveau et non prévisible et un TEID de plan utilisateur pour la demande. Il construit la demande avec ces IEs :

Élément d'Information	Source / Valeur	Remarques
IMSI	Abonnement UE	Omis s'il n'est pas présent.
Réseau Servant (MCC/MNC)	`gtp.mcc` / `gtp.mnc`, ou premiers chiffres de l'IMSI	Omis si l'une des valeurs est manquante.
Type RAT	WLAN (3)	Fixe pour S2a.
F-TEID (contrôle)	Type d'interface 35, TEID de contrôle TWAG, IP locale TWAG	Identifie le point de terminaison GTP-C du TWAG.
Nom de Point d'Accès (APN)	`gtp.apn`, par défaut `internet`. APN d'urgence si demandé.	Voir Sessions d'Urgence.
Mode de Sélection	2 (fourni par le réseau, abonnement non vérifié)	L'APN est toujours fourni par le réseau dans le TWAN.
Allocation d'Adresse PDN (PAA)	Type demandé avec une adresse zéro	Le TWAG demande une adresse. Le PGW l'attribue.
Débit Maximum Agrégé (APN-AMBR)	Par défaut 256000 en amont / 256000 en aval	Remplacé si un débit est fourni.
Contexte de Support à Créer	EBI par défaut 5, F-TEID de support (type 34), QoS	Voir Support par Défaut.
Restriction APN	2 (normal), ou 0 pour urgence	

Élément d'Information	Source / Valeur	Remarques
Identité ME (IMEISV)	Identité de l'équipement UE	Omis s'il n'est pas présent.
Récupération	Compteur de récupération TWAG	Utilisé pour la détection de redémarrage.
Caractéristiques de Facturation	`0x0800`	Valeur fixe.
Identifiant TWAN	SSID de `twan_ssid`, BSSID optionnel de `twan_bssid`	Identifie le WLAN.
Fuseau Horaire UE	Dérivé du décalage horaire local de l'hôte	
Indication	Drapeau de transfert	Le drapeau de transfert est défini pour un transfert entrant.

Le type de PDN par défaut est IPv4. Il peut être défini sur `:ipv4`, `:ipv6`, ou `:ipv4v6` via le paramètre `gtp.pdn_type`. L'adresse PAA demandée est toujours zéro, donc le PGW attribue l'adresse réelle.

> Remarque : L'IE des Options de Configuration de Protocole Supplémentaires (APCO) n'est pas inclus. Le TWAG ne demande pas d'adresses DNS ou P-CSCF via APCO pour le moment. Il lit le DNS à partir de l'IE PCO ou PCO Supplémentaire dans la réponse si le PGW en fournit un.

Support par Défaut

Le TWAG crée un support par défaut par session. L'EBI par défaut est 5. Le support n'a pas de filtres TFT, donc il transporte tout le trafic. La QoS au niveau du support dans la demande utilise ces valeurs fixes :

Paramètre QoS	Valeur
QCI	9
Niveau de Priorité	10
Capacité de Préemption	désactivée
Vulnérabilité à la Préemption	activée
GBR / MBR (amont et aval)	0

Le TWAG ne prend pas en charge les supports dédiés. Voir Gestion des Supports.

Réponse à la Création de Session

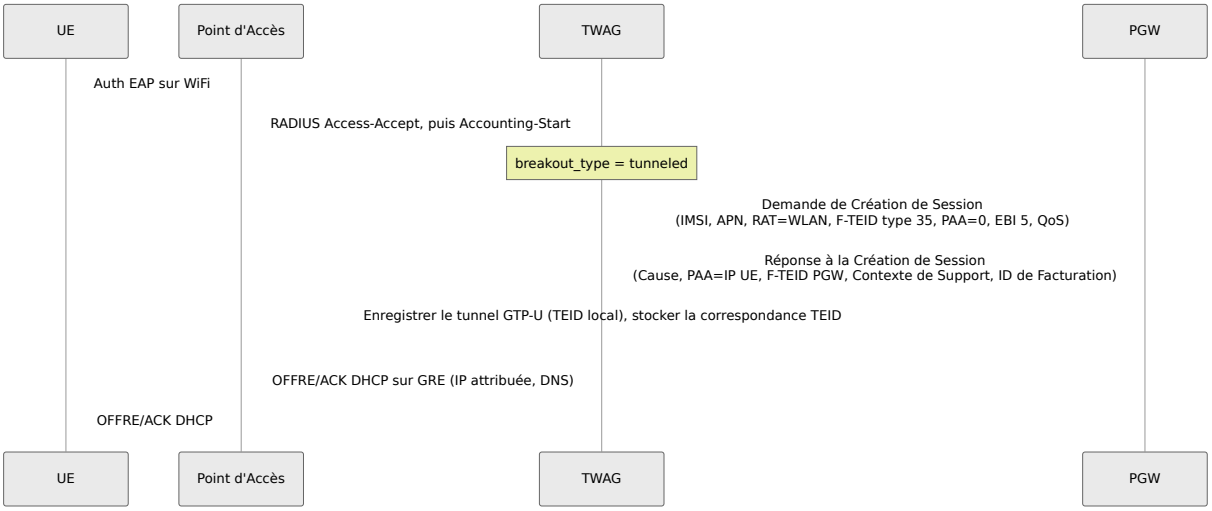
Le TWAG associe la réponse à la demande en attente par numéro de séquence. Il vérifie l'IE Cause. La demande est un succès si la cause est `request_accepted`, `new_pdp_type_due_to_network_preference`, ou `new_pdp_type_due_to_single_address_bearer_only`. Toute autre cause renvoie une `gtpc_error` avec cette valeur de cause.

En cas de succès, le TWAG lit ces valeurs de la réponse :

Élément de Réponse	Valeur Extraite	Utilisation
Allocation d'Adresse PDN (PAA)	Adresse IP UE attribuée	Livrée à l'UE par DHCP sur GRE. Voir Gestion des Adresses IP.
F-TEID (contrôle PGW)	TEID de contrôle PGW et IP PGW	Utilisé comme destination pour les messages de contrôle ultérieurs.
Contexte de Support	F-TEID de support (TEID et IP GTP-U PGW), EBI, ID de Facturation	Utilisé pour enregistrer le tunnel GTP-U.
APN-AMBR	Limites de débit de session	Stocké sur la session.
PCO / PCO Supplémentaire	Adresses de serveur DNS	Livrées à l'UE par DHCP.

Le TWAG stocke une correspondance de son TEID de contrôle local au TEID de contrôle PGW et à l'identité UE. Cette correspondance achemine les messages initiés par le PGW ultérieurs vers la session correcte. Si aucun F-TEID de support n'est présent, le TWAG revient au F-TEID PGW pour le tunnel de plan utilisateur.

Après un succès, le `Session.Client` enregistre le tunnel GTP-U (ce qui génère le TEID de plan utilisateur local côté TWAG), pré-alloue le bail DHCP avec l'IP et le DNS attribués, et enregistre l'UE pour le routage descendant GRE. Voir Plan Utilisateur pour le tunnel et le chemin descendant.



Sessions d'Urgence

Si la session demande un service d'urgence, le TWAG utilise l'APN d'urgence. Le paramètre `emergency_apn` définit cet APN. La valeur par défaut est `sos`. Pour une session d'urgence, la restriction APN est définie sur 0 (pas de restriction).

Gestion des Supports

Support par Défaut Seulement

L'OmniTWAG ne prend en charge que le support par défaut. Il ne prend pas en charge les supports dédiés. Cela suit le profil de Mode de Connexion Unique Transparent (TSCM) du TWAN. Le gestionnaire de support peut suivre une table de support indexée par EBI, mais S2a établit uniquement le support par défaut.

Modifier le Support (Initié par TWAG)

Le TWAG peut envoyer une Demande de Modification de Support (TS 29.274 §7.2.7). Cette demande transporte un contexte de support avec l'EBI par défaut, le F-TEID de plan utilisateur TWAG (type 34), et l'IP locale TWAG. Le TEID de destination est le TEID de contrôle PGW. Le TWAG associe la Réponse de Modification de Support par numéro de séquence et vérifie l'IE Cause pour le succès. Un transfert entrant réutilise le chemin de Création de Session avec le drapeau de transfert défini, pas une Modification de Support séparée.

Procédures Initiées par PGW

Le PGW peut démarrer des procédures de support. Le TWAG les reçoit sur son socket GTP-C. Le TEID de destination dans le message est le TEID de contrôle local TWAG. Le TWAG utilise la correspondance TEID stockée pour trouver le TEID de contrôle PGW et l'identité UE. Il notifie ensuite le processus `Session.Client` correspondant.

Message PGW	Réponse TWAG	Comportement
Demande de Création de Support	Réponse de Création de Support, cause `request_rejected`	Le TWAG rejette tous les supports dédiés (TSCM). Aucun support n'est créé.
Commande de Modification de Support	Demande de Modification de Support renvoyée au PGW	Le TWAG accuse réception des nouveaux paramètres (APN-AMBR, F-TEID). Si l'envoi échoue, il envoie une Indication d'Échec de Modification de Support avec la cause `system_failure`.
Demande de Mise à Jour de Support	Réponse de Mise à Jour de Support, cause `request_accepted` par support	Le TWAG accepte la mise à jour de la QoS et notifie la session.
Demande de Suppression de Support	Réponse de Suppression de Support, cause `request_accepted`	Voir Démontage de la Session.

Pour une Commande de Modification de Support, le TWAG lit l'APN-AMBR et le contexte de support (EBI, nouveau F-TEID, QoS). Il notifie le processus de session, puis envoie une Demande de Modification de Support au PGW pour accusé réception. Si le contexte de support transporte un nouveau F-TEID GTP-

U, la session réenregistre le tunnel GTP-U avec les nouvelles valeurs distantes. Pour une Demande de Mise à Jour de Support, le TWAG lit l'APN-AMBR et chaque contexte de support (EBI, QoS), notifie la session, puis accepte chaque support dans la réponse.

Démontage de la Session

Suppression de Session (Initiée par TWAG)

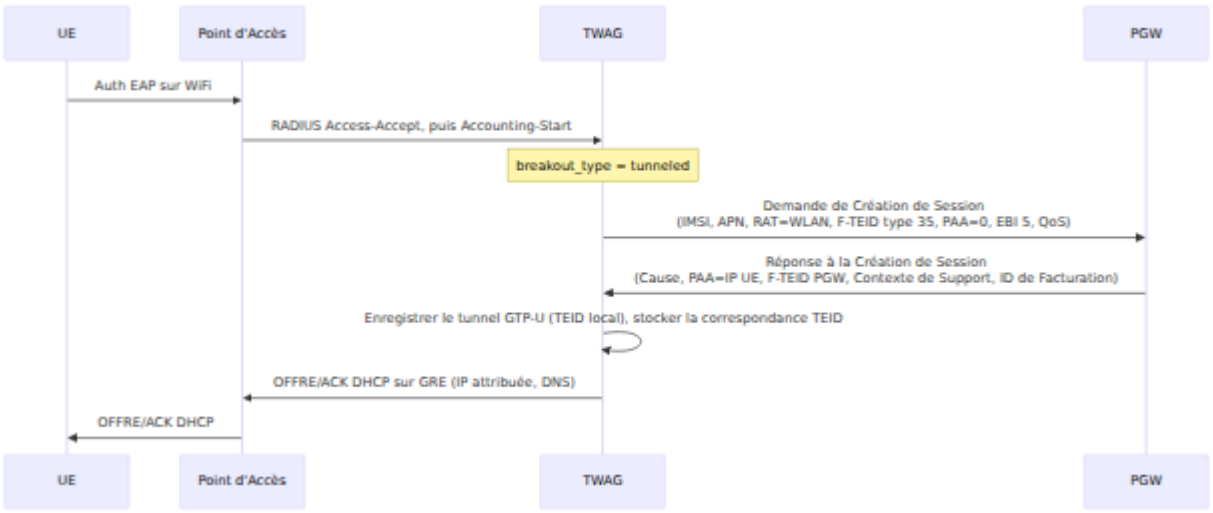
Le TWAG supprime la session PDN lorsque l'UE se déconnecte. Le démontage `Session.Client` appelle le client GTP-C. Le TWAG envoie une Demande de Suppression de Session (TS 29.274 §7.2.9) au TEID de contrôle PGW. La demande transporte :

- ID de Support EPS (l'EBI par défaut, par défaut 5)
- Cause (par défaut `local_detach`)

Le TWAG associe la Réponse de Suppression de Session par numéro de séquence et vérifie l'IE Cause. Il supprime ensuite les entrées de correspondance TEID pour ce TEID de contrôle PGW. Si la session n'a pas de TEID de contrôle PGW (par exemple une session `nswo` ou `local`), le TWAG saute la suppression GTP-C. Après la suppression, le `Session.Client` désenregistre le tunnel GTP-U, libère le bail DHCP et toute adresse de pool IP, désenregistre l'UE de GRE, et nettoie la session.

Suppression de Support (Initiée par PGW)

Le PGW peut démarrer le démontage avec une Demande de Suppression de Support. Le TWAG lit l'EBI lié, la liste des EBI, et la cause. Il notifie le processus `Session.Client` correspondant, qui effectue un nettoyage local uniquement. Le TWAG n'envoie pas de Demande de Suppression de Session dans ce cas, car le PGW a déjà supprimé le support. Le TWAG envoie ensuite une Réponse de Suppression de Support avec la cause `request_accepted`. Si l'EBI lié est le support par défaut, toute la connexion PDN est détruite.



Gestion des Chemins

Écho

Le TWAG exécute la gestion des chemins GTPv2-C avec Demande d'Écho et Réponse d'Écho (TS 29.274 §6). Le paramètre `echo_enabled` l'active. Il est activé par défaut. Le TWAG envoie une Demande d'Écho au PGW toutes les 60 secondes. Chaque Demande d'Écho transporte le numéro de séquence du TWAG. Chaque Réponse d'Écho transporte un compteur de Récupération.

Lorsque le PGW envoie une Demande d'Écho, le TWAG répond avec une Réponse d'Écho qui transporte le compteur de Récupération du TWAG.

État du Chemin

Le TWAG suit l'état du chemin vers le PGW. Une Réponse d'Écho réussie marque le chemin comme UP et réinitialise le compteur d'échec. Les échecs d'Écho consécutifs incrémentent un compteur. Après 3 échecs consécutifs, le TWAG marque le chemin comme DOWN.

Détection de Redémarrage PGW

Le TWAG stocke le compteur de Récupération PGW de chaque Réponse d'Écho. Si le compteur change, le TWAG détecte un redémarrage du PGW. Il nettoie

ensuite toutes les sessions affectées. Il notifie chaque processus de session avec une cause `pgw_restart` et efface la correspondance TEID.

Délai d'Attente de Demande et Réémission

Le TWAG utilise un minuteur T3-RESPONSE et des réémissions N3-REQUESTS pour les demandes de contrôle.

Minuteur / Compteur	Valeur	Signification
T3-RESPONSE	3000 ms	Temps d'attente pour une réponse avant une réémission.
N3-REQUESTS	3	Nombre maximum de réémissions avant que la demande échoue.

Si une demande dépasse le délai d'attente, le TWAG réémet le même message avec le même numéro de séquence. Il redémarre le minuteur. Après 3 réémissions sans réponse, le TWAG échoue la demande. Un échec de Création de Session renvoie `timeout` à l'appelant. Les réémissions d'Écho suivent les mêmes limites de minuteur et de réémission. Un échec d'Écho compte vers le seuil de chemin-down.

Le TWAG ouvre le socket UDP GTP-C au démarrage. Si le socket ne peut pas s'ouvrir, le TWAG réessaie toutes les 5 secondes.

Configuration

S2a est configuré sous la carte `gtp`. Cette carte fait partie de la configuration de l'application `omnitwag` dans `runtime.exs`.

```
config :omnitwag,
  gtp: %{
    # Activer l'interface S2a (nécessaire pour le détachement
tunnélisé)
    enabled: true,

    # Point de terminaison GTP-C PGW
    pgw_ip: "10.5.198.1",
    pgw_port: 2123,

    # IP locale TWAG pour les F-TEID GTP-C et GTP-U
    local_ip: "10.5.198.200",

    # Port de plan utilisateur GTP-U
    gtpu_port: 2152,

    # Gestion des chemins (Écho)
    echo_enabled: true,

    # Valeurs par défaut de la Demande de Création de Session
    apn: "internet",
    pdn_type: :ipv4
  }
```

Paramètres GTP

Paramètre	Type	Requis	Par défaut	Description
`enabled`	Boolean	Non	`false`	Activer l'interface S2a. Définir sur `true` pour le détachement tunnélisé. Lorsque `false`, le TWAG ne démarre pas le client GTP-C et saute S2a pour chaque session.
`pgw_ip`	String	Oui (si activé)	`127.0.0.1`	Adresse IP GTP-C PGW. Le TWAG envoie tous les messages GTPv2-C à cette adresse.
`pgw_port`	Integer	Non	`2123`	Port UDP GTP-C PGW. Le port GTP-C standard est 2123.
`local_ip`	String	Non	Détecté automatiquement	IP locale TWAG. Utilisé dans les F-TEIDs et comme source pour GTP-C et GTP-U. Si non défini, le TWAG détecte automatiquement l'IP sortante.

Paramètre	Type	Requis	Par défaut	Description
`gtpu_port`	Integer	Non	`2152`	Port UDP GTP-U pour le plan utilisateur. Voir Plan Utilisateur.
`echo_enabled`	Boolean	Non	`true`	Activer la gestion des chemins Echo GTPv2-C vers le PGW.
`apn`	String	Non	`internet`	Nom du Point d'Accès pour la session PDN.
`pdn_type`	Atom	Non	`:ipv4`	Type de PDN : `:ipv4`, `:ipv6`, ou `:ipv4v6`.
`mcc`	String	Non	Premiers 3 chiffres IMSI	Code de Pays Mobile pour l'IE Réseau Servant.
`mnc`	String	Non	Chiffres IMSI 4-5	Code de Réseau Mobile pour l'IE Réseau Servant.

Paramètres d'Urgence et TWAN

Les paramètres suivants sont lus à partir de l'environnement de l'application `omnitwag`, pas de la carte `gtp`.

Paramètre	Type	Par défaut	Description
`emergency_apn`	String	`sos`	APN utilisé pour une session d'urgence. La restriction APN est définie sur 0 pour ces sessions.
`twan_ssid`	String	`TWAG`	SSID placé dans l'IE Identifiant TWAN.
`twan_bssid`	Binaire	(non défini)	BSSID optionnel de 6 octets pour l'IE Identifiant TWAN. Omis s'il ne s'agit pas d'une valeur valide de 6 octets.

> Remarque : Le client GTP-C se connecte à un seul PGW. Il n'y a pas de sélection par APN ou multi-PGW pour le moment. Pour changer le PGW, mettez à jour `pgw_ip`.

Tables de Référence

Types de Messages GTPv2-C (S2a)

Message	Direction	Objectif
Demande / Réponse d'Écho	Les Deux	Gestion des chemins.
Demande / Réponse de Création de Session	TWAG vers PGW	Établir la session PDN.
Demande / Réponse de Modification de Support	TWAG vers PGW	Mettre à jour le F-TEID de support ou la QoS.
Demande / Réponse de Suppression de Session	TWAG vers PGW	Détruire la session PDN.
Demande / Réponse de Création de Support	PGW vers TWAG	Support dédié. Rejeté par le TWAG.
Demande / Réponse de Mise à Jour de Support	PGW vers TWAG	Mettre à jour la QoS. Accepté par le TWAG.
Commande de Modification de Support	PGW vers TWAG	PGW demande au TWAG de mettre à jour les paramètres de support.
Indication d'Échec de Modification de Support	TWAG vers PGW	Envoyé lorsqu'une Commande de Modification de Support ne peut pas être honorée.

Message	Direction	Objectif
Demande / Réponse de Suppression de Support	PGW vers TWAG	Démontage initié par PGW.

Codes de Cause GTPv2-C

Le TWAG utilise ces valeurs de cause. Les codes suivent TS 29.274 §8.4.

Code	Nom	Signification
2	Détachement Local	Cause par défaut pour une Suppression de Session initiée par TWAG.
13	Échec Réseau	Échec de réseau signalé par le pair.
16	Demande Acceptée	Succès.
64	Contexte Non Trouvé	Le pair n'a pas de contexte correspondant.
72	Échec Système	Utilisé dans une Indication d'Échec de Modification de Support.
92	Échec d'Authentification Utilisateur	Authentification rejetée.
93	Accès APN Refusé	L'APN n'est pas autorisé.
94	Demande Rejetée	Utilisé lorsque le TWAG rejette une Demande de Création de Support.
96	IMSI/IMEI Inconnu	L'abonné n'est pas connu.

Observabilité

Le TWAG émet des événements de télémétrie pour l'activité de support et de démontage. Ces événements alimentent les métriques dans la Référence des Métriques.

Événement	Quand
`[:twag, :bearer, :modify_command]`	Le TWAG reçoit une Commande de Modification de Support PGW.
`[:twag, :bearer, :update]`	Le TWAG reçoit une Demande de Mise à Jour de Support PGW.
`[:twag, :detach, :pgw_delete]`	Le TWAG reçoit une Demande de Suppression de Support PGW.
`[:twag, :session, :teardown]`	La session est détruite.
`[:twag, :detach, :cleanup_start]`	Le démontage de la session commence.
`[:twag, :detach, :cleanup_complete]`	Le démontage de la session est terminé.

État du Client

Le client GTP-C rapporte un instantané de statut. Ce statut montre la santé du chemin S2a.

Champ	Signification
`connected`	Le socket UDP GTP-C est ouvert.
`pgw_ip` / `pgw_port`	Le point de terminaison PGW configuré.
`pending_requests`	Compte des demandes GTP-C en cours.
`path_up`	L'état du chemin Echo vers le PGW.
`echo_failures`	Compte des échecs d'Écho consécutifs.
`recovery_counter`	Le compteur de Récupération du TWAG.

Journalisation

Le client GTP-C journalise chaque procédure. Les lignes de journal clés comprennent :

- Création de Session, Suppression de Session et envois et réponses de Modification de Support.
- Changements d'état du chemin : `Le chemin vers le PGW ... est UP` et `Le chemin vers le PGW ... est DOWN`.
- Détection de redémarrage du PGW avec les anciens et nouveaux compteurs de Récupération.
- Messages de Création, Mise à Jour, Modification et Suppression de Support initiés par PGW.

Dépannage

Échec de Création de Session

Symptômes : Les sessions ne s'établissent pas. Le journal montre `Échec de Création de Session GTP`.

Causes possibles :

- Le PGW rejette la demande avec une cause non réussie.
- Le chemin vers le PGW est down.
- L'APN n'est pas provisionné sur le PGW.
- L'IMSI n'est pas connu du cœur.

Résolution :

1. Vérifiez le code de cause retourné par rapport à la table Codes de Cause.
2. Vérifiez l'état du chemin avec l'état du client (`path_up`).
3. Confirmez que l'APN est provisionné sur le PGW.
4. Confirmez que l'abonné est provisionné dans le HSS.

Le chemin vers le PGW est Down

Symptômes : Le journal montre `Le chemin vers le PGW ... est DOWN`. Les nouvelles sessions échouent.

Causes possibles :

- Le PGW est injoignable sur UDP 2123.
- Un pare-feu bloque GTP-C.
- Le PGW n'est pas en cours d'exécution.

Résolution :

1. Confirmez que les valeurs `pgw_ip` et `pgw_port` sont correctes.
2. Confirmez que le chemin réseau permet UDP 2123 dans les deux sens.
3. Confirmez que le service GTP-C PGW est en cours d'exécution.
4. Attendez le prochain Écho. Une Réponse d'Écho réussie marque le chemin comme UP.

Les sessions se terminent après un redémarrage du PGW

Symptômes : Toutes les sessions se terminent ensemble. Le journal montre `Redémarrage du PGW détecté`.

Causes possibles :

- Le PGW a redémarré et a changé son compteur de Récupération.

Résolution :

1. C'est un comportement attendu. Le TWAG nettoie les sessions obsolètes après un redémarrage du PGW.
2. Confirmez que le PGW est stable. Les UEs se réauthentifient et rétablissent les sessions.

Les sessions s'établissent mais aucun flux de données

Symptômes : L'UE obtient une adresse IP, mais le trafic ne passe pas.

Causes possibles :

- Le tunnel GTP-U ou le chemin GRE n'est pas opérationnel.

Résolution :

1. Confirmez que `gtp.enabled` et `gre.enabled` sont tous deux `true` pour le détachement tunnélisé.
2. Vérifiez les statistiques GTP-U et GRE dans le guide Plan Utilisateur.

Références 3GPP

Spécification	Titre
TS 29.274	Protocole de Tunneling GPRS Évolué pour le plan de contrôle (GTPv2-C).
TS 23.402	Améliorations de l'architecture pour les accès non-3GPP (TWAN, S2a).
TS 24.302	Accès à l'EPC via des réseaux d'accès non-3GPP.

Vue d'ensemble

Sessions

Vue d'ensemble

Une **session** représente un abonné authentifié sur OmniTWAG. OmniTWAG associe une session à l'identité de l'abonné (l'IMSI ou le nom d'utilisateur). OmniTWAG conserve les sessions en mémoire. L'API Sessions sert l'état de session en direct.

La vue des Sessions transporte également le trafic de l'abonné. OmniTWAG fusionne le registre de comptabilité RADIUS dans la session par `Acct-Session-Id`. Une session est donc la source unique à la fois pour l'état d'un abonné et le trafic de l'abonné. Il n'y a pas de point de terminaison de comptabilité séparé. La comptabilité fait partie de la session.

Table des matières

- Concepts
- Cycle de vie de l'état d'authentification
- Modèle de session
- Référence de l'état d'authentification
- API REST
- Actions par session
- Exemples JSON
- Interface Web
- Dépannage

Concepts

Chaque abonné authentifié a une session. OmniTWAG crée la session lorsqu'une demande d'accès RADIUS commence une authentification EAP. OmniTWAG met à jour la session au fur et à mesure que l'authentification progresse et que le chemin de données est établi.

Une session suit l'identité de l'abonné, le point d'accès, l'adresse MAC du client, l'adresse IP UE allouée, la méthode et la phase EAP, ainsi que l'état GTP et DNS. Lorsque la comptabilité RADIUS arrive pour l'abonné, OmniTWAG fusionne les compteurs de trafic dans la session.

La forme de la session est ouverte. Les clés présentes dépendent de la méthode d'authentification et de l'avancement de la session. Une session qui vient de commencer l'EAP a moins de clés qu'une session qui a atteint l'Accept RADIUS et établi un chemin de données GTP et DHCP.

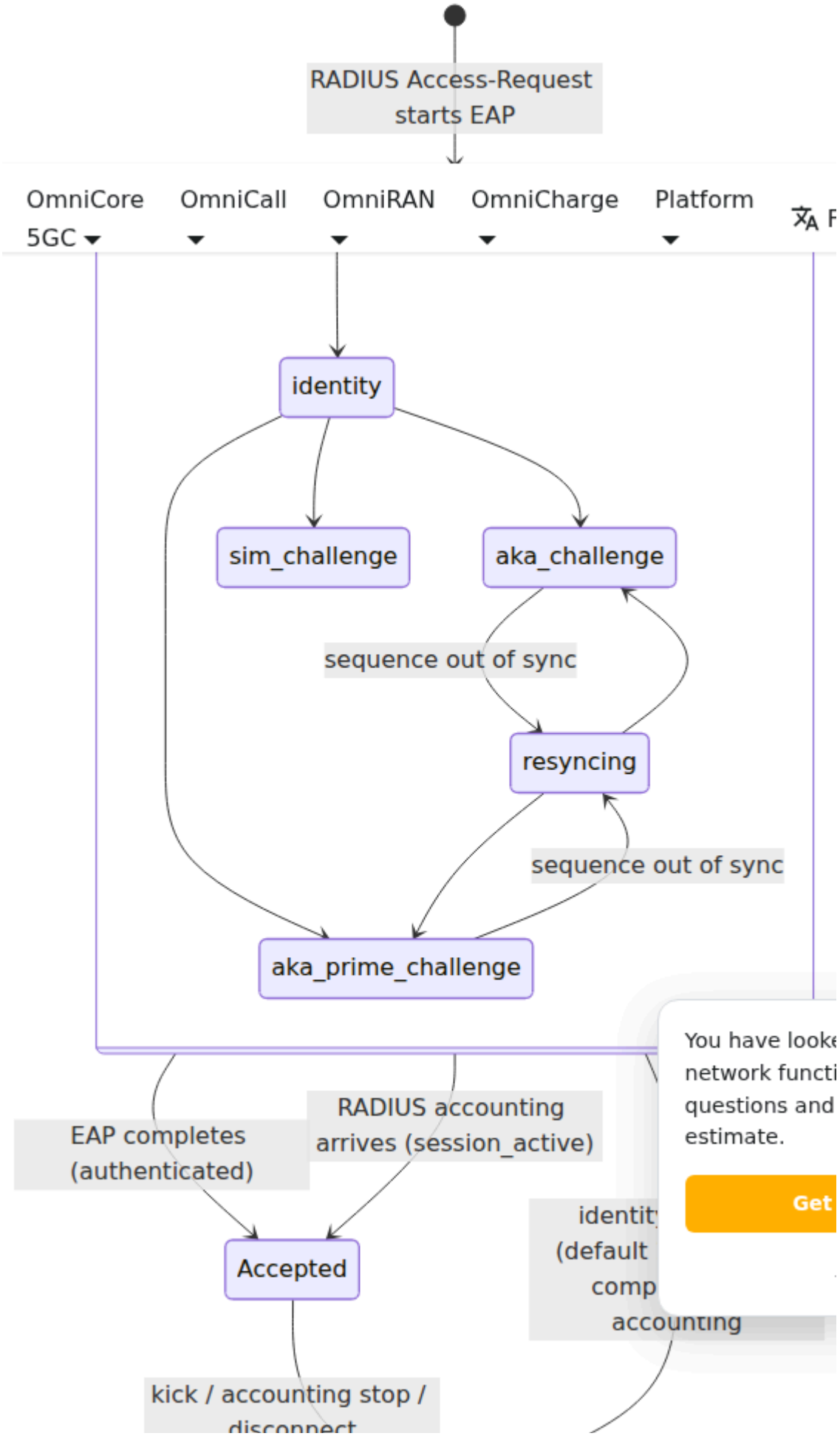
Cycle de vie de l'état d'authentification

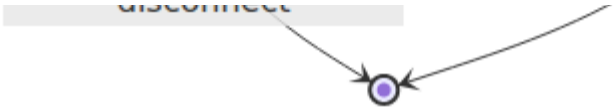
Une session a un état d'authentification. L'état est dérivé de la phase EAP de la session (`eap_phase`).

- Une session est **Authentifiant** tant que `eap_phase` est une phase d'authentification en cours. Les phases en cours sont `identity`, `aka_challenge`, `aka_prime_challenge`, `sim_challenge`, et `resyncing`.
- Une session est **Acceptée** une fois qu'elle est authentifiée, ou une fois qu'elle est active (`session_active`).

Une session qui reçoit la comptabilité RADIUS est promue à active. La comptabilité ne s'applique qu'à un abonné déjà authentifié. L'arrivée de la comptabilité est donc la preuve que l'abonné est actif, donc OmniTWAG marque la session comme Acceptée.

Une session qui reste dans une phase d'authentification en cours est supprimée. Si une session a commencé l'authentification, n'a jamais été complétée, et n'a jamais reçu de comptabilité, OmniTWAG détruit la session après un délai d'attente. Cela empêche une tentative d'authentification inachevée de demeurer en tant qu'Authentifiant indéfiniment. Le délai d'attente par défaut est de 10 minutes.


RADIUS Access-Request starts EAP
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
identity
sim_challenge
aka_challenge
sequence out of sync
resyncing
sequence out of sync
aka_prime_challenge
EAP completes (authenticated)
RADIUS accounting arrives (session_active)
Accepted
kick / accounting stop / disconnect



Modèle de session

Une session a les champs ci-dessous. La forme est ouverte, donc une session peut également porter d'autres clés GTP ou de facturation. Chaque champ ci-dessous est rendu sous forme de chaîne imprimable, d'entier, ou de liste. Les valeurs binaires sont rendues sous forme de chaînes imprimables ou en hexadécimal. Les tuples IP sont rendus sous forme de chaînes à quatre points.

Champ	Type	Description
`identity`	String	Identité de la session. C'est l'IMSI ou le nom d'utilisateur. OmniTWAG associe la session à cette valeur.
`imsi`	String	IMSI de l'abonné.
`mac_address`	String	Adresse MAC de l'UE.
`ue_ip`	String	Adresse IP de l'UE allouée, sous forme de chaîne à quatre points.
`nas_ip`	String	IP source de l'AP (RADIUS `NAS-IP-Address`).
`called_station_id`	String	Identifiant du point d'accès (RADIUS `Called-Station-Id`).
`calling_station_id`	String	MAC du client (RADIUS `Calling-Station-Id`).
`acct_session_id`	String	ID de session de comptabilité RADIUS. OmniTWAG fusionne le registre de comptabilité par cette clé.
`eap_phase`	String	Phase EAP actuelle. Cette valeur détermine l'état d'authentification. Voir Référence de l'état d'authentification.
`eap_type`	String	Méthode EAP utilisée, par exemple `aka` ou `aka_prime`.
`pgw_ip`	String	Adresse IP du PGW, sous forme de chaîne à quatre points.
`bearer_id`	Integer	ID de porteur EPS.

Champ	Type	Description
`dns_servers`	List of String	Serveurs DNS assignés à l'UE.
`created_at`	Integer	Heure de création de la session, en millisecondes depuis l'époque Unix.
`last_activity`	Integer	Heure de la dernière activité, en millisecondes depuis l'époque Unix.
`established_at`	Integer	Heure d'établissement de la session, en millisecondes depuis l'époque Unix.
`acct_input_octets`	Integer	Octets reçus de l'UE. Fusionnés à partir de la comptabilité RADIUS.
`acct_output_octets`	Integer	Octets envoyés à l'UE. Fusionnés à partir de la comptabilité RADIUS.
`acct_input_packets`	Integer	Paquets reçus de l'UE. Fusionnés à partir de la comptabilité RADIUS.
`acct_output_packets`	Integer	Paquets envoyés à l'UE. Fusionnés à partir de la comptabilité RADIUS.
`acct_session_time`	Integer	Durée de la session de comptabilité, en secondes. Fusionnée à partir de la comptabilité RADIUS.
`acct_status_type`	Integer	Dernier état de comptabilité. `1` est Démarrer, `2` est Arrêter, `3` est Intermédiaire.
`acct_last_update`	Integer	Heure de la dernière mise à jour de la comptabilité, en millisecondes depuis l'époque Unix.

Les champs `acct_*` ne sont présents qu'après que le registre de comptabilité de l'abonné a été fusionné dans la session.

Référence de l'état d'authentification

OmniTWAG associe la valeur `eap_phase` à un état affiché. Le tableau ci-dessous liste l'association.

`eap_phase`	État affiché	État d'authentification
`identity`	Identité	Authentifiant
`aka_challenge`	Défi AKA	Authentifiant
`aka_prime_challenge`	Défi AKA'	Authentifiant
`sim_challenge`	Défi SIM	Authentifiant
`resyncing`	Resynchronisation	Authentifiant
`authenticated`	Accepté	Accepté
`session_active`	Accepté	Accepté
`accepted`	Accepté	Accepté
`rejected`	Rejeté	Rejeté

Toute autre valeur `eap_phase` compte comme Authentifiant.

API REST

L'API REST OmniTWAG sert ces points de terminaison via HTTPS sur le port API configuré (`8444` par défaut). Le chemin de base est `/sessions`.

Méthode	Chemin	Action	Description
GET	`/sessions`	index	Lister les sessions authentifiées. La réponse est paginée et consultable.
GET	`/sessions/{id}`	show	Obtenir une session par `identity`.
DELETE	`/sessions/{id}`	delete	Supprimer la session et déconnecter l'abonné.
PATCH	`/sessions/{id}`	action	Effectuer une action sur la session. L'action est dans le corps JSON.
PUT	`/sessions/{id}`	action	Effectuer une action sur la session. L'action est dans le corps JSON.

Le paramètre de chemin `{id}` est l'`identity` de la session (l'IMSI ou le nom d'utilisateur).

Recherche et pagination

Le point de terminaison d'index accepte des paramètres de pagination et de recherche.

Paramètre	Type	Description
`offset`	Integer	Nombre de sessions à ignorer avant la page.
`limit`	Integer	Nombre maximum de sessions dans la page.
`q`	String	Terme de recherche. OmniTWAG fait correspondre le terme avec les champs de recherche ci-dessous.

Le terme `q` est comparé à ces champs de session.

Champ de recherche	Correspondances
`identity`	Identité de la session (IMSI ou nom d'utilisateur).
`imsi`	IMSI de l'abonné.
`nas_ip`	IP source de l'AP (`NAS-IP-Address`).
`called_station_id`	Identifiant du point d'accès.
`calling_station_id`	MAC du client.
`mac_address`	Adresse MAC de l'UE.
`ue_ip`	Adresse IP de l'UE allouée.

Corps de la requête d'action

`PATCH` et `PUT` acceptent un corps JSON avec un champ.

Champ	Type	Requis	Description
`action`	String	Oui	L'action à effectuer. Doit être `kick`, `start_accounting`, ou `stop_accounting`.

Voir Actions par session pour ce que chaque action fait.

Réponses

Statut	Signification
`200 OK`	La requête a réussi (index, show, delete, action).
`404 Not Found`	Aucune session n'a l'`identity` donné.
`422 Unprocessable Entity`	L'action n'est pas l'une de `kick`, `start_accounting`, ou `stop_accounting`.

Une action réussie renvoie un corps avec un champ `status` et l'`identity`. Une suppression renvoie `{"status": "deleted", "identity": ...}`.

Actions par session

Une action exécute une opération administrative contre une session en direct. Envoyez l'action dans le corps JSON d'une requête `PATCH` ou `PUT` à `/sessions/{id}`.

Action	Effet	Résultat `status`
`kick`	Envoyer un CoA-Disconnect RADIUS au point d'accès et détruire la session.	`kicked`
`start_accounting`	Marquer la session comme active et commencer la comptabilité. Cela promeut la session à Acceptée.	`accounting_started`
`stop_accounting`	Terminer la comptabilité, comme si le point d'accès avait envoyé Accounting-Stop. Cela détruit la session.	`accounting_stopped`

Les actions `kick` et `stop_accounting` détruisent la session. Utilisez `kick` pour déconnecter activement l'abonné au point d'accès. Utilisez `stop_accounting` pour terminer la session comme le ferait le réseau.

Exemples JSON

Une session avec trafic

Requête:

```
GET /sessions/001010000000001
```

Réponse:

```
{
  "identity": "001010000000001",
  "imsi": "001010000000001",
  "mac_address": "AA:BB:CC:DD:EE:FF",
  "ue_ip": "10.45.0.2",
  "nas_ip": "192.0.2.10",
  "called_station_id": "AA-BB-CC-DD-EE-FF:WiFi-Calling",
  "calling_station_id": "11-22-33-44-55-66",
  "acct_session_id": "5A2E1F0C",
  "eap_phase": "session_active",
  "eap_type": "aka_prime",
  "pgw_ip": "192.0.2.20",
  "bearer_id": 5,
  "dns_servers": ["8.8.8.8", "8.8.4.4"],
  "created_at": 1712000000000,
  "last_activity": 1712000600000,
  "established_at": 1712000004000,
  "acct_input_octets": 1048576,
  "acct_output_octets": 5242880,
  "acct_input_packets": 1200,
  "acct_output_packets": 3400,
  "acct_session_time": 600,
  "acct_status_type": 3,
  "acct_last_update": 1712000600000
}
```


Une action de kick

Requête:

```
PATCH /sessions/001010000000001
```


```
{
  "action": "kick"
}
```


Réponse (`200 OK`):

```
{
  "status": "kicked",
  "identity": "001010000000001"
}
```

Une action inconnue

Requête:

```
{
  "action": "reboot"
}
```

Réponse (`422 Unprocessable Entity`):

```
{
  "error": "Unknown action \"reboot\" (expected
kick|start_accounting|stop_accounting)",
  "identity": "001010000000001"
}
```

Interface Web

Le panneau de contrôle a une page **Clients RADIUS**. La page liste chaque session avec son état d'authentification. La page montre le nombre de sessions Acceptées, Authentifiant, et Rejetées. Utilisez la page pour examiner les sessions en direct d'un coup d'œil.

Dépannage

Une session est bloquée "Authentifiant"

Symptômes : Une session affiche l'état Authentifiant. L'état ne change pas en Accepté. La session reste dans une phase EAP telle que `identity`,

`aka_challenge`, `aka_prime_challenge`, `sim_challenge`, ou `resyncing`.

Causes possibles :

- L'abonné a commencé l'authentification EAP mais ne l'a jamais complétée.
- Le point d'accès a cessé d'envoyer des réponses EAP.
- L'abonné a échoué à l'authentification et n'a pas réessayé.

Comment cela se résout :

1. OmniTWAG arme un délai d'attente lorsque la session entre dans une phase d'authentification en cours.
2. Si la session ne termine jamais l'authentification et ne reçoit jamais de comptabilité, OmniTWAG détruit la session lorsque le délai d'attente expire. Le délai d'attente par défaut est de 10 minutes.
3. Une session qui termine l'authentification passe à Acceptée. Une session qui reçoit la comptabilité RADIUS est promue à active et passe également à Acceptée.

Pour supprimer une session bloquée immédiatement, envoyez une action `kick` ou une requête `DELETE` à `/sessions/{id}`.

Une session ne montre pas de trafic

Symptômes : Une session est Acceptée, mais les champs `acct_*` sont absents ou à zéro.

Causes possibles :

- Aucun registre de comptabilité RADIUS n'a encore été reçu pour l'abonné.
- Le registre de comptabilité a un `Acct-Session-Id` différent de celui de la session.

Résolution :

1. Confirmez que le point d'accès envoie la comptabilité RADIUS à OmniTWAG.
2. Confirmez que le `Acct-Session-Id` de la comptabilité correspond à l'`acct_session_id` de la session. OmniTWAG fusionne le trafic par cette

clé.

3. Attendez la prochaine mise à jour intermédiaire de la comptabilité, puis relisez la session.

Vue d'ensemble

Plan Utilisateur et Chemin de Données

Vue d'ensemble

OmniTWAG sépare le plan de contrôle du plan utilisateur. Le plan de contrôle exécute l'authentification et la signalisation de session. Le plan utilisateur transporte le véritable trafic UE. Cette page décrit le plan utilisateur. Elle explique comment OmniTWAG déplace les paquets UE, comment il construit et détruit le tunnel GTP-U, et comment il renvoie les paquets de liaison descendante à l'UE.

Pour la signalisation de session GTP-C qui établit le chemin de données tunnelisé, voir S2a / GTP-C. Pour une vue complète du système, voir Architecture.

Table des Matières

- Plan de Contrôle et Plan Utilisateur
- Modes de Chemin de Données
- Chemin de Données GRE
- Tunneling GTP-U
- Transfert de Liaison Descendante
- Supports et Modèles de Flux de Trafic
- Gestion des Erreurs
- MTU et Fragmentation
- Configuration
- Observabilité

Plan de Contrôle et Plan Utilisateur

Le plan de contrôle et le plan utilisateur fonctionnent en tant que processus séparés.

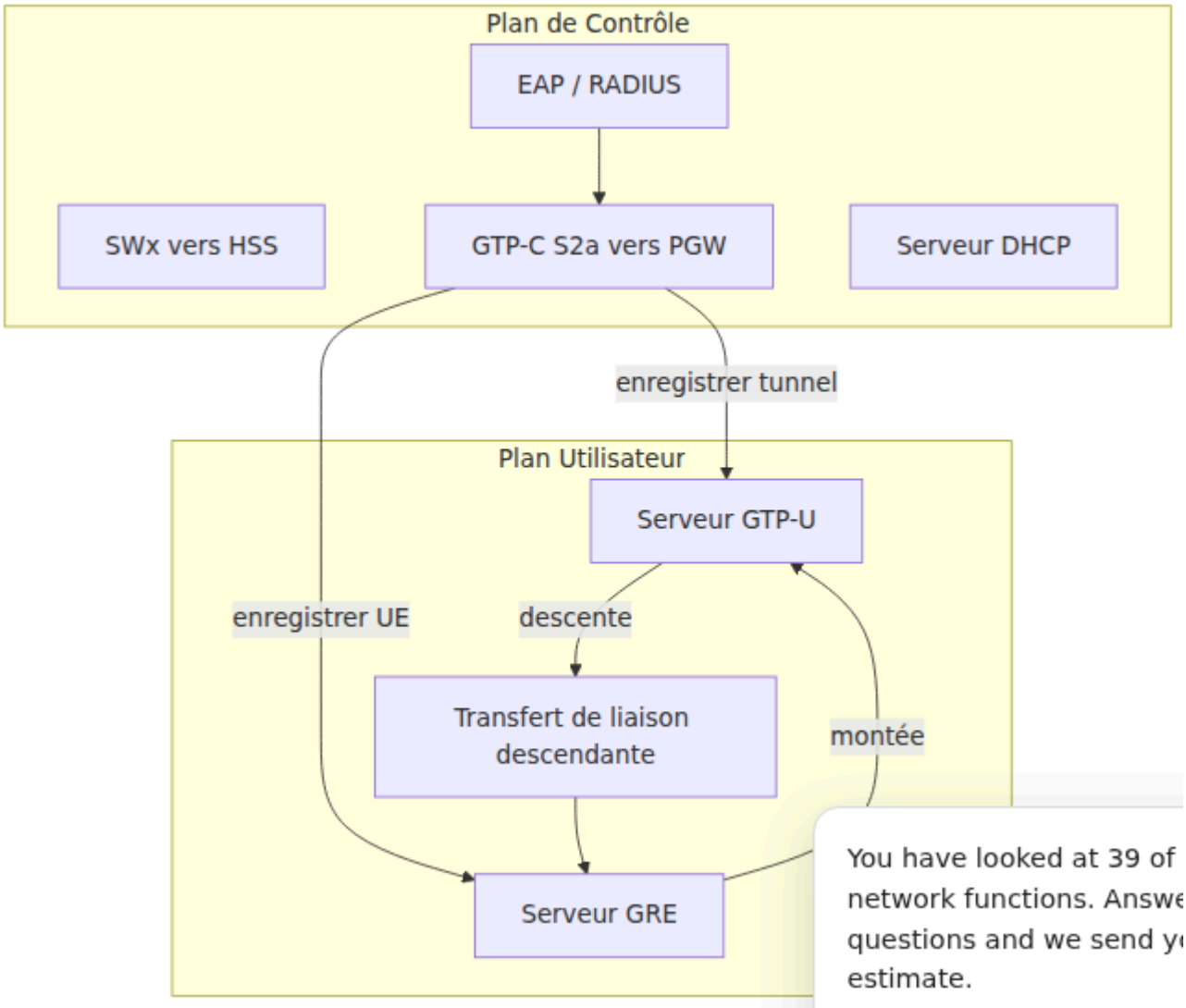
Le **plan de contrôle** effectue les travaux suivants :

- Authentification EAP via RADIUS avec le point d'accès.
- Récupération du vecteur d'authentification depuis le HSS via Diameter SWx.
- Signalisation de session GTP-C vers le PGW sur l'interface S2a. Cela crée et supprime la connexion PDN et les supports. Voir S2a / GTP-C.
- Livraison d'adresse DHCP à l'UE. Voir Gestion des Adresses IP.

Le **plan utilisateur** effectue les travaux suivants :

- Tunneling GRE vers le point d'accès.
- Tunneling GTP-U vers le PGW sur l'interface S2a-U, selon 3GPP TS 29.281 (mode tunnelisé uniquement).
- Transfert de paquets de liaison descendante vers l'UE.
- Classification des paquets de liaison montante en supports.

Le plan de contrôle configure l'état du plan utilisateur. Après authentification, le processus de session crée la session GTP-C (en mode tunnelisé), enregistre le tunnel GTP-U et enregistre l'UE avec le serveur GRE. Ensuite, le plan utilisateur transporte le trafic.



Modes de Chemin de Données

OmniTWAG résout un mode de plan de données par point d'accès (AP) à partir des informations d'identification RADIUS qui correspondent à l'AP. La valeur `breakout_type` sur les informations d'identification sélectionne l'un des trois modes. Voir Informations d'identification RADIUS pour les paramètres `breakout_type`, `address_source` et `ip_pool_id`.

Mode (`breakout_type`)	IP UE attribuée par	TWAG transporte le trafic	Chemin
`tunnelisé`	PGW	Oui	GRE de l'AP au TWAG, puis GTP-U sur S2a vers le PGW.
`nswo`	TWAG, depuis un pool IP	Oui	GRE de l'AP au TWAG, puis sortie locale vers Internet (NAT) au TWAG.
`local`	Réseau local à l'AP	Non	L'AP possède le plan de données. Le TWAG ne transporte aucun paquet et n'attribue aucune IP.

Le chemin de données GRE s'applique à la fois aux sorties `tunnelisées` et `nswo`. Dans les deux modes, l'AP envoie des trames UE au TWAG en GRE, et le TWAG renvoie des paquets de liaison descendante à l'AP en GRE. La différence réside dans la liaison montante après GRE :

- **`tunnelisé`** : le TWAG encapsule le paquet de liaison montante en GTP-U et l'envoie au PGW. Le PGW attribue l'IP UE et applique la politique et la facturation.
- **`nswo`** : le TWAG sort la liaison montante localement vers Internet avec une règle de masquerade NAT. Le TWAG attribue l'IP UE à partir d'un pool IP. Il n'y a pas de PGW ni de session GTP-C. Le Gy côté TWAG Facturation en ligne s'applique.

En mode `local`, le TWAG n'attribue ni IP ni ne transporte de trafic, donc ni le chemin GRE ni le chemin GTP-U n'est utilisé pour cet AP.

Mode Tunnelisé (GTP-U vers PGW)

En mode tunnelisé, OmniTWAG tunnelise le trafic UE vers le PGW. Le PGW alloue l'adresse IP de l'UE, applique la politique et la facturation, et fournit l'accès à Internet. C'est l'interface S2a selon 3GPP TS 23.402 Section 16.

Le chemin de liaison montante est :

```
UE --WiFi--> AP --GRE--> TWAG --GTP-U S2a--> PGW --> Internet
```

Le chemin de liaison descendante est l'inverse :

```
Internet --> PGW --GTP-U S2a--> TWAG --GRE--> AP --WiFi--> UE
```

Pour utiliser ce mode, définissez `gtp.enabled` sur `true` et `gre.enabled` sur `true`, et définissez l'information d'identification de l'AP `breakout_type` sur `tunnelisé`. Voir Configuration.

Sortie Locale (NSWO)

Le Non-Seamless WLAN Offload (NSWO) est une sortie locale. Le trafic UE sort directement du TWAG vers Internet. Le trafic ne passe pas par le PGW. Le TWAG attribue l'IP UE et applique la masquerade NAT. Cela est conforme à 3GPP TS 23.402 Section 16.2.

OmniTWAG a deux façons d'atteindre le NSWO :

1. **Sortie par AP (préférée)** : définissez l'information d'identification de l'AP `breakout_type` sur `nswo`. L'adresse UE provient du pool IP nommé par `ip_pool_id`. Le `address_source` de l'information d'identification sélectionne comment :
    - `pool` pré-alloue une adresse fixe du pool et enregistre l'UE pour la liaison descendante GRE immédiatement.
    - `dhcp` lie l'UE MAC au pool et permet au serveur DHCP du TWAG de distribuer une adresse lors de la découverte DHCP. La liaison descendante GRE est enregistrée lorsque l'adresse est connue.

2. **Gestionnaire NSWO global** : le `Twag.NSWO.Handler` fournit une sortie locale pour chaque session lorsque le GTP n'est pas utilisé. Il est sélectionné avec `nswo.enabled: true` (et s'applique lorsque `gtp.enabled: false`). Le gestionnaire gère un seul pool IP local et configure une règle de masquerade NAT `iptables`. Lors de l'établissement de session, le TWAG alloue une adresse du pool local, la livre à l'UE via DHCP, et enregistre l'UE pour la liaison descendante GRE, sans session PGW / GTP-C. L'allocation est libérée lors de la destruction de la session.

Pour les deux chemins, la liaison montante est masquée NAT vers Internet et la liaison descendante revient par le chemin GRE. Voir Gestion des Adresses IP et Pools IP pour l'adressage.

Le gestionnaire NSWO global lit ces paramètres à partir de la carte `nswo` :

Paramètre	Type	Par défaut	Description
`enabled`	Booléen	`false`	Démarrer le gestionnaire NSWO global et acheminer les sessions via une sortie locale.
`local_pool`	Chaîne (CIDR)	`10.100.0.0/16`	Pool à partir duquel l'adresse UE est allouée.
`nat_interface`	Chaîne	`eth0`	Interface de sortie pour la règle de masquerade NAT.
`dns_servers`	Liste	`["8.8.8.8", "8.8.4.4"]`	Serveurs DNS annoncés à l'UE. La sortie locale n'a pas de PCO PGW pour les transporter.

Chemin de Données GRE

Le serveur GRE (`Twag.Gre.Server`) est le point de terminaison du plan utilisateur vers le point d'accès. Le point d'accès envoie les trames UE au TWAG en GRE, conformément à RFC 2784 et RFC 2890. Le serveur GRE envoie également les paquets de liaison descendante et les réponses DHCP au point d'accès. GRE est le chemin orienté vers l'AP pour les sorties `tunnelisées` et `nswo`.

Le serveur GRE utilise un socket brut pour le protocole IP 47 (GRE), ouvert avec `:socket.open/3`. L'hôte doit accorder la capacité `CAP_NET_RAW`. Si le socket ne s'ouvre pas, le serveur enregistre une erreur et réessaie toutes les 10 secondes. GRE est désactivé par défaut. Pour l'activer, définissez `gre.enabled` sur `true`.

Classification GRE Entrante

Le serveur GRE lit l'en-tête IP extérieur, le retire et décode l'en-tête GRE. Le serveur prend en charge deux types de charge utile GRE :

Protocole GRE	Signification	Gestion
`0x6558`	Pont Ethernet Transparent	Extraire la trame Ethernet intérieure, puis classifier par ethertype.
`0x0800`	IPv4	Traiter comme un paquet IPv4 de liaison montante.

Pour une trame de Pont Ethernet Transparent, le serveur lit l'ethertype intérieur :

Ethertype	Signification	Gestion
`0x0800`	IPv4	Si le paquet est DHCP, le transmettre au serveur DHCP. Sinon, le transférer vers le chemin de liaison montante.
`0x0806`	ARP	Gérer localement. Ne pas transférer.
`0x86DD`	IPv6	Transférer vers le chemin de liaison montante.

Le serveur conserve une carte de l'adresse MAC de l'UE à l'adresse IP du point d'accès. Le serveur apprend cette carte à partir de l'adresse MAC source de chaque trame entrante. Cette carte permet au serveur de router les trames de liaison descendante vers le bon point d'accès.

Gestion de la Liaison Montante

Pour un paquet IP de liaison montante, le serveur GRE lit l'adresse IP source. C'est l'IP UE. Le serveur recherche l'IP UE dans la table des UE enregistrés pour trouver l'identité de la session. Il trouve le processus de session dans le registre et lit le TEID local de la session. Il envoie ensuite le paquet de manière asynchrone au serveur GTP-U avec ce TEID local (en mode tunnelisé). L'envoi asynchrone évite un blocage entre les processus GRE et GTP-U.

S'il n'y a pas de session pour l'IP UE, ou si la session n'a pas encore de tunnel GTP-U, le serveur abandonne le paquet. Un paquet IP de petite taille (moins de 20 octets) est également abandonné.

> Note : Pour la sortie `nswo`, le pool IP local et la masquerade NAT transportent la liaison montante vers Internet. Le serveur GRE enregistre l'UE pour le routage de liaison descendante de la même manière qu'en mode tunnelisé.

Enregistrement de l'UE pour la Liaison Descendante

Lorsque la session devient active, le processus de session enregistre l'UE avec le serveur GRE. L'enregistrement contient l'IP UE, la MAC UE, l'IP du point d'accès et l'identité de la session. Le serveur GRE utilise ces données pour router les paquets de liaison descendante. Lors de la destruction de la session, le processus de session désenregistre l'UE.

Tunneling GTP-U

Le serveur GTP-U (`Twag.Gtp.GtpU`) est le point de terminaison du plan utilisateur vers le PGW en mode tunnelisé. Il exécute un seul socket UDP sur le port `2152` et maintient une table de tunnels. Le serveur implémente GTPv1-U selon 3GPP TS 29.281.

Gestion du TEID

Chaque tunnel a un TEID local et un TEID distant. Le TEID distant est le TEID du PGW. Le TWAG utilise le TEID distant dans l'en-tête des paquets qu'il envoie au PGW. Le PGW utilise le TEID local dans l'en-tête des paquets qu'il envoie au TWAG. Le TWAG recherche le TEID local dans la table des tunnels pour trouver le tunnel.

Lorsque le processus de session reçoit la réponse de création de session du PGW, il enregistre le tunnel. Le serveur GTP-U génère le TEID local. Le TWAG utilise une valeur aléatoire cryptographiquement pour le TEID local, et il n'utilise jamais zéro. Cela suit l'exigence de TEID non prédictible dans 3GPP TS 29.274.

Un tunnel enregistré contient ces champs :

Champ	Description
`local_teid`	Le TEID côté TWAG. Le PGW envoie des paquets de liaison descendante avec cette valeur.
`remote_teid`	Le TEID côté PGW. Le TWAG envoie des paquets de liaison montante avec cette valeur.
`remote_ip`	L'adresse IP GTP-U du PGW.
`remote_port`	Le port UDP GTP-U du PGW. Par défaut `2152`.
`session_id`	L'identité de la session pour le tunnel.

Encapsulation de Liaison Montante

Pour chaque paquet UE de liaison montante, le serveur GTP-U construit un en-tête GTPv1-U G-PDU et place l'en-tête devant le paquet IP intérieur. L'en-tête contient la version GTP (1), le type de protocole (GTP), le type de message (G-PDU, 255), la longueur de la charge utile et le TEID distant. Le serveur envoie le paquet à l'adresse IP et au port du PGW via le socket UDP. Le serveur GRE appelle le serveur GTP-U sur le chemin d'envoi asynchrone, ce qui évite un blocage entre les deux processus.

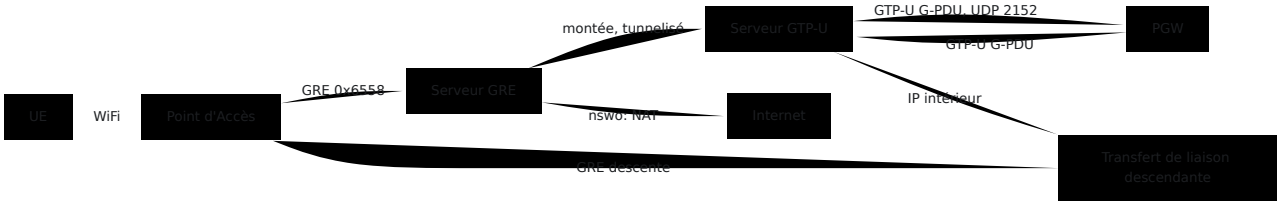
Décapsulation de Liaison Descendante

Pour chaque paquet qui arrive sur le socket GTP-U, le serveur lit l'en-tête GTP-U. Le serveur agit en fonction du type de message :

Message	Type	Action
G-PDU	255	Rechercher le tunnel par le TEID local, puis transférer le paquet IP intérieur au transfert de liaison descendante.
Echo Request	1	Répondre avec un Echo Response.
Echo Response	2	Accepter et ignorer.
Error Indication	26	Accepter et ignorer.
End Marker	254	Accepter et ignorer.

Pour un G-PDU, le serveur retire l'en-tête GTP-U et tous les champs d'en-tête optionnels (numéro de séquence, numéro N-PDU et type d'en-tête d'extension suivant). Ensuite, le serveur passe le paquet IP intérieur à `Twag.Downlink`. Voir Transfert de Liaison Descendante.

Si un G-PDU arrive pour un TEID qui n'a pas de tunnel, le serveur abandonne le paquet et envoie une indication d'erreur GTP-U au source. Voir Gestion des Erreurs.



Transfert de Liaison Descendante

Le transfert de liaison descendante (`Twag.Downlink`) reçoit les paquets IP intérieurs de la décapsulation GTP-U. Le transfert envoie chaque paquet vers l'UE. Le transfert a deux chemins et préfère GRE.

Chemin GRE (Préféré)

Si le serveur GRE fonctionne, le transfert envoie le paquet via GRE. Le transfert lit l'IP de destination du paquet intérieur. C'est l'IP UE. Le transfert appelle le serveur GRE. Le serveur GRE recherche l'IP UE dans la table des UE enregistrés, enveloppe le paquet dans une trame GRE de Pont Ethernet Transparent, et l'envoie à l'IP du point d'accès. Le point d'accès livre la trame à l'UE via WiFi.

Pour la trame Ethernet de liaison descendante, le serveur GRE utilise la MAC UE comme MAC de destination. Le serveur utilise une MAC administrée localement fixe (`02:00:00:00:00:01`) comme MAC source.

Chemin de Socket Brut (Fallback)

Si le serveur GRE ne fonctionne pas, ou si l'envoi GRE échoue, le transfert utilise un socket brut. Le transfert injecte le paquet IP brut. Le noyau route ensuite le paquet vers l'UE à partir de la table de routage locale.

Le transfert ouvre deux sockets bruts : un pour IPv4 et un pour IPv6. Le transfert définit l'option `IP_HDRINCL` (et `IPV6_HDRINCL` pour IPv6). Cette option signifie que le transfert fournit l'en-tête IP complet. Sur certains noyaux, le socket fonctionne toujours sans cette option, donc un échec à définir l'option n'est pas fatal.

L'hôte doit accorder la capacité `CAP_NET_RAW` pour les sockets bruts. Si un socket ne s'ouvre pas, le transfert enregistre un avertissement et désactive cette famille d'adresses. Si ni GRE ni un socket brut ne sont disponibles, le transfert compte et abandonne les paquets, et enregistre un avertissement une seule fois.

Sélection de Chemin

Le paramètre `prefer_gre` définit l'ordre de sélection du chemin.

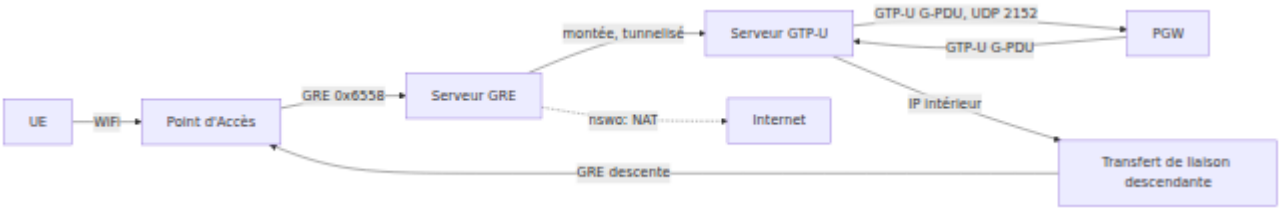
- `prefer_gre: true` (par défaut) : le transfert essaie d'abord le tunnel GRE et revient au socket brut si GRE n'est pas disponible ou si l'envoi GRE échoue.

- `prefer_gre: false` : le transfert essaie d'abord le socket brut pour la famille d'adresses du paquet et utilise GRE uniquement lorsque ce socket brut n'est pas disponible.

Vérifications de Paquet

Le transfert abandonne un paquet dans ces cas :

- Le transfert est désactivé par configuration.
- Aucun socket n'est disponible, et GRE n'est pas disponible.
- Le paquet est un paquet de petite taille (moins de 20 octets).
- Le transfert ne peut pas lire l'IP de destination.



Supports et Modèles de Flux de Trafic

Le gestionnaire de supports (`Twag.Bearer.Manager`) conserve les supports pour une session. Chaque support est mappé à une Identité de Support EPS (EBI). Un support contient les TEID GTP-U, les filtres de paquets du Modèle de Flux de Trafic (TFT), et la classe QoS (QCI). Un support est le support par défaut. Cela suit 3GPP TS 23.402 Section 16.1.6.1.

Supports Par Défaut et Dédiés

Type de Support	Filtres TFT	Objectif
Par Défaut	Aucun	Transporte tout le trafic qui ne correspond à aucun support dédié.
Dédié	Un ou plusieurs	Transporte le trafic qui correspond à ses filtres de paquets TFT.

Si le support par défaut est supprimé, le gestionnaire de supports signale une destruction de session complète. La suppression d'un support dédié ne supprime que ce support.

Filtres de Paquets TFT

Un TFT est un Modèle de Flux de Trafic selon 3GPP TS 24.008 Section 10.5.6.12. Un TFT contient une opération et une liste de filtres de paquets. Chaque filtre de paquet contient une direction, une priorité, et une liste de composants. Le parseur TFT prend en charge ces types de composants :

Composant	Correspondance
Adresse et masque distant IPv4	L'adresse de destination du paquet.
Adresse et masque local IPv4	L'adresse source du paquet.
Adresse et préfixe distant/local IPv6	L'adresse IPv6.
Identifiant de protocole	Le numéro de protocole IP.
Port local unique / plage de ports locaux	Le port source de l'UE.
Port distant unique / plage de ports distants	Le port de destination.
Type de service / classe de trafic	L'octet TOS, avec un masque.
Indice de paramètre de sécurité	Non correspond à la liaison montante TWAN.
Étiquette de flux	Non correspond (IPv6 uniquement).

Le parseur prend en charge les opérations TFT : créer, supprimer, ajouter des filtres, remplacer des filtres, supprimer des filtres, et aucune opération. Le gestionnaire de supports applique ces op��rations lorsque le PGW envoie une Demande de Mise à Jour de Support.

Classification de Liaison Montante

Le classificateur de liaison montante lie chaque paquet de liaison montante à un support. Les règles sont :

1. Lire l'IP source, l'IP de destination, le protocole, les ports, et le TOS du paquet.
2. Tester le paquet contre les filtres TFT des supports dédiés.

3. Un filtre correspond uniquement si tous ses composants correspondent. C'est une logique ET.
4. Si plus d'un filtre correspond, sélectionner le filtre avec le numéro de priorité le plus bas. Un numéro de priorité plus bas est une priorité plus élevée.
5. Si aucun support dédié ne correspond, sélectionner le support par défaut.

Un filtre vide (un filtre sans composants) correspond à tous les paquets. C'est un joker.

> **Remarque :** Le gestionnaire de supports et le classificateur TFT sont présents pour le support des supports dédiés. Le chemin GRE de liaison montante transfère le trafic sur le TEID local de session. Une séparation complète de liaison montante multi-supports dépend du PGW pour créer des supports dédiés pour la session, que le TWAG rejette sous TSCM (voir S2a / GTP-C).

Gestion des Erreurs

Echo GTP-U

Le PGW et le TWAG utilisent la Demande Echo GTP-U et la Réponse Echo pour tester le chemin du tunnel. Lorsque le serveur GTP-U reçoit une Demande Echo, il répond avec une Réponse Echo. Le processus GTP-C S2a exécute son propre écho lorsque `gtp.echo_enabled` est `true`. Voir S2a / GTP-C.

Indication d'Erreur GTP-U

Si un G-PDU arrive pour un TEID qui n'a pas de tunnel, le serveur GTP-U envoie une Indication d'Erreur GTP-U au source. L'Indication d'Erreur contient le TEID inconnu et l'adresse du pair GTP-U du TWAG. Cela indique au PGW d'arrêter d'envoyer du trafic sur le tunnel mort. Le serveur abandonne également le paquet et augmente le compteur d'erreurs.

Marqueur de Fin GTP-U

Le serveur GTP-U peut envoyer un Marqueur de Fin sur un tunnel avant de détruire le tunnel. Cela est pour un transfert. Le Marqueur de Fin indique au pair qu'aucun paquet supplémentaire ne suivra sur l'ancien chemin.

Erreurs de Décodage

Si le serveur GTP-U ne peut pas décoder un paquet, il enregistre un avertissement et augmente le compteur d'erreurs. Si le serveur GRE ne peut pas décoder un paquet, ou obtient un protocole ou un ethertype inconnu, il compte l'événement et abandonne le paquet.

MTU et Fragmentation

Le serveur GRE construit l'en-tête IPv4 extérieur pour chaque paquet GRE de liaison descendante. Le numéro de protocole GRE est 47.

L'encapsulation GTP-U et GRE ajoute chacune des octets d'en-tête au paquet UE. Pour éviter la fragmentation, le point d'accès doit réduire le MSS TCP et le MTU du tunnel. La configuration de référence du point d'accès Cisco définit `mss 1410` et `mtu 1500` sur le tunnel GRE. Voir Architecture pour la configuration du point d'accès.

Configuration

Les processus du plan utilisateur lisent la configuration de l'application `:omnitwag`. Les blocs pertinents sont `gtp`, `gre`, et `downlink`. Le mode de sortie AP provient des informations d'identification RADIUS, et non de ces blocs. Voir Informations d'identification RADIUS.

```
config :omnitwag,
  # GTP-U et GTP-C vers le PGW (S2a) - mode tunnelisé
  gtp: %{
    enabled: false,
    pgw_ip: "10.5.198.1",
    pgw_port: 2123,
    local_ip: "10.5.198.200",
    gtpu_port: 2152,
    echo_enabled: true
  },

  # GRE vers le point d'accès
  gre: %{
    enabled: false,
    local_ip: "10.5.198.200"
  },

  # Transfert de liaison descendante
  downlink: %{
    enabled: true,
    ip_hdr_incl: true,
    prefer_gre: true
  }
```

Paramètres GTP

Paramètre	Type	Requis	Par défaut	Description
`enabled`	Booléen	Non	`false`	Démarrer les serveurs GTP-C et GTP-U. Définir `true` pour le mode tunnelisé.
`pgw_ip`	Chaîne	Non	`"10.5.198.1"`	L'adresse IP du PGW pour GTP-C et GTP-U.
`pgw_port`	Entier	Non	`2123`	Le port UDP GTP-C du PGW.
`local_ip`	Chaîne	Non	Détecté automatiquement	L'IP locale du TWAG pour GTP. Utilisé comme source pour les paquets GTP-U et pour l'adresse du pair d'Indication d'Erreur.
`gtpu_port`	Entier	Non	`2152`	Le port UDP local pour le serveur GTP-U. C'est le port GTP-U standard.
`echo_enabled`	Booléen	Non	`true`	Exécuter l'écho GTP-C vers le

Paramètre	Type	Requis	Par défaut	Description
				PGW. Voir S2a / GTP-C.

Paramètres GRE

Paramètre	Type	Requis	Par défaut	Description
`enabled`	Booléen	Non	`false`	Démarrer le serveur GRE. Définir `true` pour recevoir des trames UE du point d'accès via GRE. Nécessaire pour les sorties `tunnelisées` et `nswo`.
`local_ip`	Chaîne	Non	`"0.0.0.0"`	L'IP locale pour lier le socket brut GRE. C'est l'IP de destination à laquelle le point d'accès envoie le trafic GRE.

Paramètres de Liaison Descendante

Paramètre	Type	Requis	Par défaut	Description
`enabled`	Booléen	Non	`true`	Démarrer le transfert de liaison descendante. Lorsque `false`, le transfert abandonne tous les paquets de liaison descendante.
`ip_hdr_incl`	Booléen	Non	`true`	Définir les options `IP_HDRINCL` et `IPV6_HDRINCL` sur les sockets bruts afin que le transfert fournisse l'en-tête IP complet. Lorsque `false`, l'option n'est pas définie. Laissez `true` à moins que le noyau ne rejette `IP_HDRINCL`.
`prefer_gre`	Booléen	Non	`true`	Ordre de sélection du chemin. Lorsque `true`, le transfert essaie GRE en premier et revient au socket brut. Lorsque `false`, il essaie d'abord le socket brut pour la famille d'adresses du paquet et utilise GRE uniquement lorsque le socket brut n'est pas disponible.

Capacité de l'Hôte

Le serveur GRE et le chemin de socket brut de liaison descendante utilisent tous deux des sockets bruts. L'hôte doit accorder la capacité `CAP_NET_RAW`. Sans cette capacité, les sockets bruts ne s'ouvrent pas. Le serveur GRE réessaie le socket toutes les 10 secondes. Le transfert de liaison descendante revient en arrière ou abandonne les paquets, comme décrit dans Transfert de Liaison Descendante.

Observabilité

Les processus du plan utilisateur conservent des statistiques internes. Chaque processus rapporte son propre état, couvrant le point de terminaison du tunnel GTP-U, le serveur GRE, et le transfert de liaison descendante. Pour les métriques Prometheus que OmniTWAG exporte, voir la Référence des Métriques.

État GTP-U

L'état GTP-U rapporte `socket_open`, `local_ip`, `tunnel_count`, et une carte `stats` imbriquée :

Statistique	Description
`packets_in`	Le nombre de paquets reçus sur le socket GTP-U.
`packets_out`	Le nombre de paquets envoyés sur le socket GTP-U.
`bytes_in`	Le total des octets reçus.
`bytes_out`	Le total des octets envoyés.
`errors`	Le nombre d'erreurs de décodage et d'événements de tunnel inconnu.
`tunnel_count`	Le nombre de tunnels enregistrés (champ de premier niveau).

État GRE

L'état GRE rapporte `running`, `socket_open`, `local_ip`, `ap_tunnels`, `registered_ues`, et une carte `stats` imbriquée :

Statistique	Description
`packets_in`	Le nombre de paquets GRE reçus.
`packets_out`	Le nombre de paquets GRE envoyés.
`bytes_in` / `bytes_out`	Le total des octets entrants et sortants.
`dhcp_in`	Le nombre de paquets DHCP reçus du point d'accès.
`uplink_forwarded`	Le nombre de paquets de liaison montante transférés au GTP-U.
`downlink_forwarded`	Le nombre de paquets de liaison descendante envoyés au point d'accès.
`unknown_protocol`	Le nombre de paquets avec un protocole ou un ethertype inconnu.
`errors`	Le nombre d'erreurs de décodage et d'envoi.
`registered_ues`	Le nombre d'UE enregistrés (champ de premier niveau).

Statistiques de Liaison Descendante

Statistique	Description
`forwarded`	Le nombre de paquets IPv4 transférés (GRE et socket brut IPv4).
`forwarded_v6`	Le nombre de paquets IPv6 transférés.
`dropped`	Le nombre de paquets abandonnés.
`errors`	Le nombre d'erreurs d'envoi.
`bytes`	Le total des octets transférés.
`enabled`	L'état activé du transfert.
`socket_open`	L'état du socket brut IPv4.

Vue d'ensemble