

OmniTWAG

概要

OmniTWAG は無線 LAN 接続 (TWAG) 用の 3GPP TS 23.402 § 16 の無線 LAN 接続 (EPC) の実装です。

機能

- 無線 LAN 接続
- AP 接続
- 無線 LAN 接続
- 無線 LAN 接続
- AP 接続 (GRE 接続)
- DHCP 接続
- 無線 LAN 接続
- 無線 LAN 接続 (無線 LAN)
- Diameter 接続
- 無線 LAN 接続
- 無線 LAN 接続
- 無線 LAN 接続

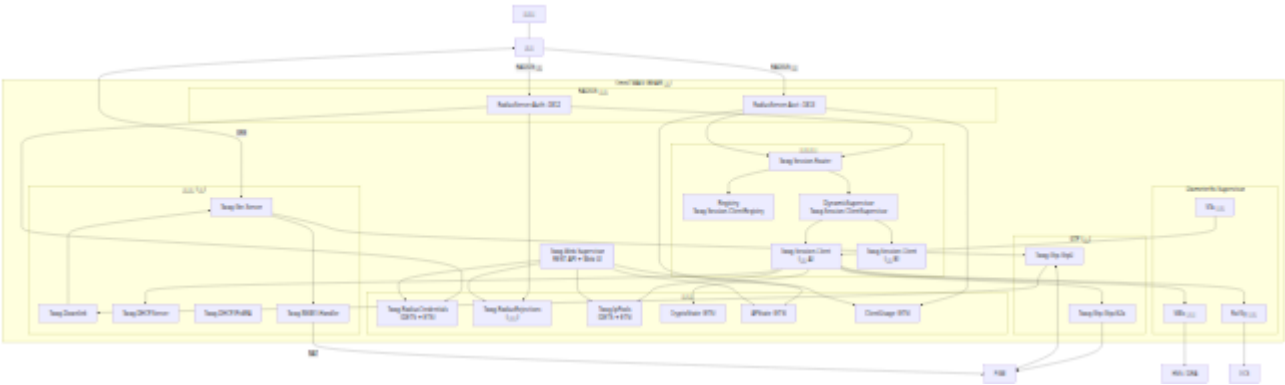
アーキテクチャ

OmniTWAG は BEAM (Erlang VM) の実装です。無線 LAN 接続の GenServer を実装します。

構成

- Web** は **OAM** REST API の Web 接続 (`Twag.Web.Supervisor`)
- 無線 LAN 接続は `Twag.Session.Client` GenServer、`Registry`、`DynamicSupervisor`、`Twag.Session.Router` を実装します。

- Twag.RadiusCredentials Twag.RadiusRejections
Twag.IpPools ETS CryptoState APState ClientUsage
- **RADIUS** RadiusServer.Auth 1812 RadiusServer.Acct 1813
- **Diameter** DiameterEx.Supervisor SWx STa Ro/Gy
- **GTP** Twag.Gtp.GtpcS2a Twag.Gtp.GtpU gtp.enabled
- Twag.Gre.Server Twag.Downlink Twag.DHCP.Server Twag.DHCP.IPv6RA Twag.NSW0.Handler



:one_for_one 60 10 RADIUS IP



*

□□□□

□□	□□	□□□□
Twag.Web.Supervisor	□□□	□□
DiameterEx.Supervisor	□□□	□□
Twag.Session.ClientRegistry	□□□	□□
Twag.Session.ClientSupervisor	DynamicSupervisor	□□
Twag.Session.Router	□□	□□
Twag.Session.Client	GenServer	□□□□
Twag.RadiusCredentials	GenServer	□□
Twag.RadiusRejections	GenServer	□□
Twag.IpPools	GenServer	□□

項目	種別	説明
RadiusServer.Auth	GenServer	認証
RadiusServer.Acct	GenServer	課金
CryptoState	GenServer (ETS)	暗号化状態
APState	GenServer (ETS)	アクセスポイント状態
ClientUsage	GenServer (ETS)	クライアント使用状況
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
Twag.DHCP.IPv6RA	GenServer	ipv6_ra.enabled

名前	種別	設定
Twag.NSW0.Handler	GenServer	nsw0.enabled

インストール

OmniTWAG をインストールする。AP をインストールする **RADIUS** をインストール `gtp / nsw0` をインストール

インストール (GTP 環境)

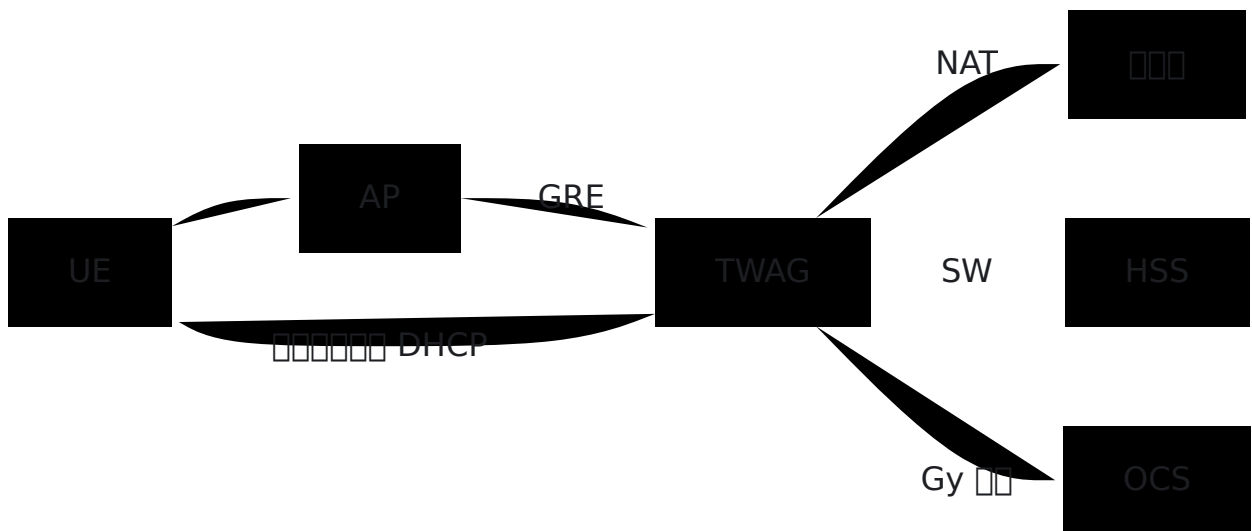
インストール GTP-C/GTP-U S2a をインストール PGW PGW UE IP をインストール/インストール



- **UE IP** PGW をインストール (PAA IE)
- **IP** TWAG を GRE UE UE DHCP PGW IP
- AP を GRE UE TWAG GTP-U PGW
- PGW GTP-U TWAG GRE AP UE
- PGW (PCEF) Gy OCS
- `gtp.enabled: true` `gre.enabled: true`

インストール (NSWO)

インストール TWAG をインストール PGW TWAG NAT/インストール Gy をインストール



- **UE IP** TWAG IP (Twag.IpPools)
- **IP** TWAG GRE UE DHCP
- AP GRE TWAG NAT
- TWAG NAT GRE AP UE
- TWAG PCEF Gy OCS
- nsw.enabled: true gtp.enabled: false

nsw.enabled: true gtp.enabled: false

breakout_type AP NSWO PGW NAT/

AP (GRE)

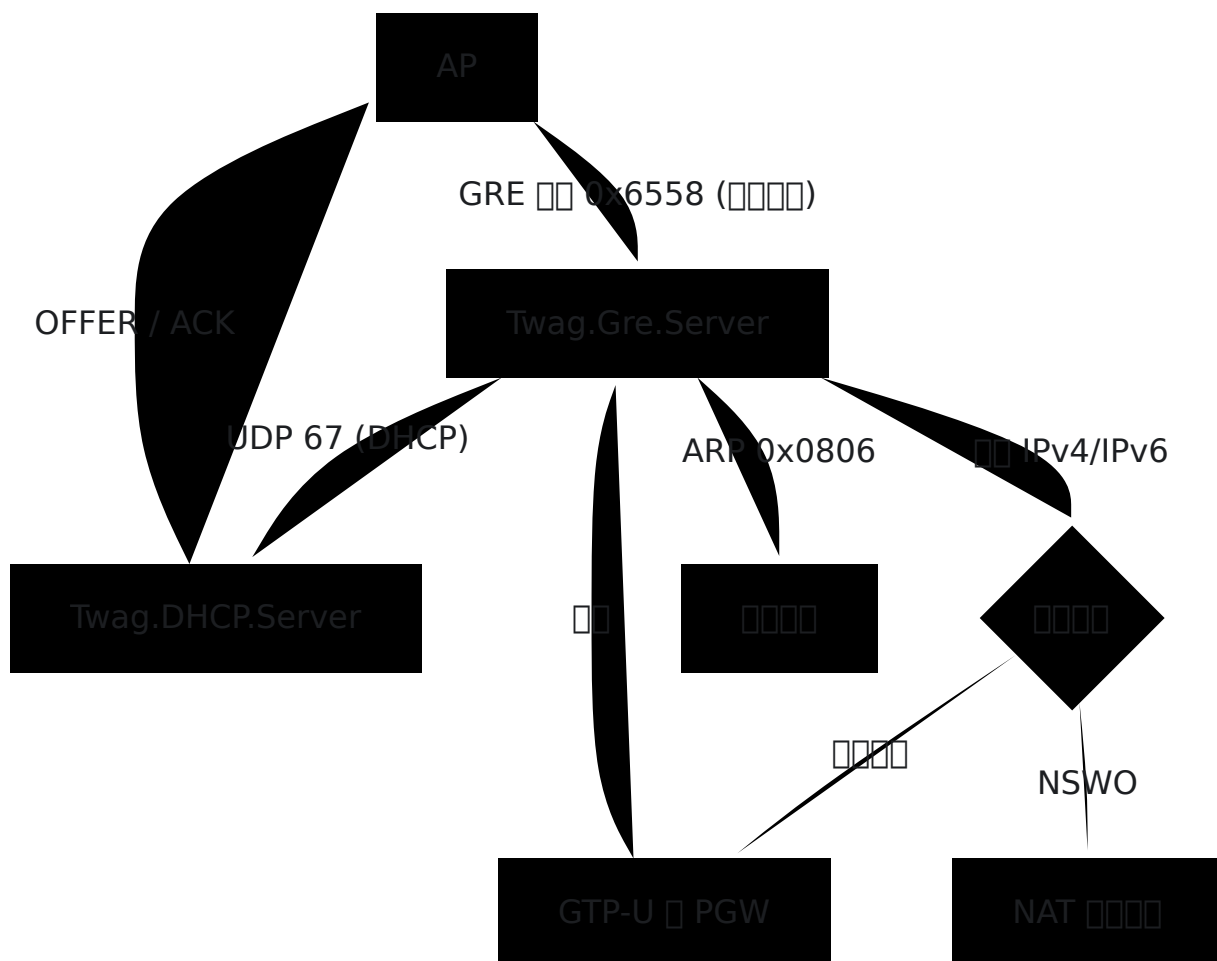
AP GRE IP 47 UE L2 TWAG TS 23.402 16.1.2

Cisco AP

```
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
```

GRE



DHCP

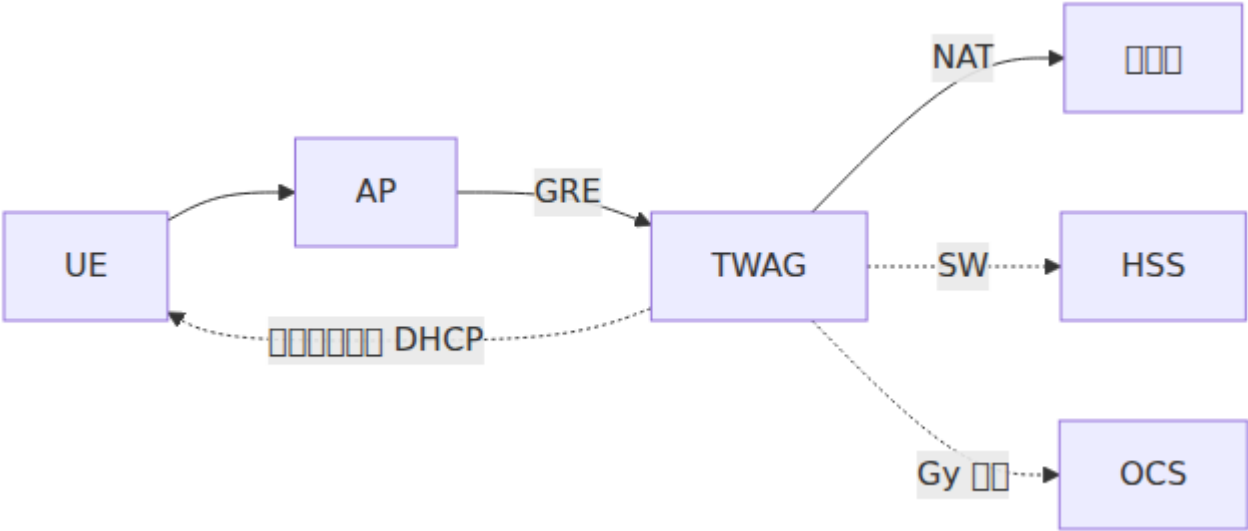
	IP	DHCP	
	PGW (PAA)	TWAG PGW IP GRE	PGW PCO DNS
	TWAG (Twag.IpPools)	TWAG DISCOVER	DNS

TWAG DHCP

- GRE UE DHCP DISCOVER
- OFFER PGW IP MAC
- REQUEST/ACK
- GRE DHCP AP

OmniTWAG TWAG 3GPP AAA EAP-AKA Diameter SWx HSS EAP Twag.Session.Client

网络



网络中配置了RADIUS 服务器 Twag.RadiusCredentials 配置 IP 地址为 10.10.10.10 的 RADIUS 服务器，配置了 unauthorized_source 和 bad_secret 参数，配置了 RADIUS 服务器。

配置

名称	配置	描述	EAP 类型
:swx	auth_mode: :swx	配置 SWx MAR/MAA 与 HSS	配置 (TWAG)
:local	auth_mode: :local	配置本地 (Milenage)	配置 (TWAG)

EAP 配置

配置 eap_method 为 :eap_aka，配置 UE 认证方式为 TWAG，配置 UE 认证方式为 EAP-SIM 或 EAP 认证 (ERP)。

□□□□□□ (□□□□)

EAP 成功 -> Access-
Accept (MSK へ AP)

AP へ Accounting-Start

Twag.Session.Client へ
へ

GTP-C 成功 -> PGW

PGW へ IP へ

へ GTP-U へ (へ + へ
TEID)

へ DHCP へ UE MAC へ
PGW IP

UE DHCP Discover ->
OFFER へ PGW IP

UE -> GRE -> GTP-U -> PGW ->

GTP-C GTP-U + DHCP

twag.session.teardown accounting_stop admin_disconnect network_disconnect STa ASR identity_timeout liveness_timeout 3 10

Diameter

	ID		
SWx	16777265	TWAG -> HSS	(MAR/MAA)
STa	16777250	TWAG	RARASRSTR
Ro/Gy	4	TWAG -> OCS	NSWO

DRA

DRA SWx HSS

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

```
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  
}
```

項目	値	説明
auth_mode	:swx	:swx の HSS 認証モード :l Milenage 認証モード
eap_method	:eap_aka	AAA EAP 認証 :eap_aka :eap_aka_prime 認証
identity_timeout_ms	600000	AAA 認証タイムアウト時間 identity_timeout
gtp.enabled	true	GTP-C/GTP-U 有効
gtp.pgw_ip	127.0.0.1	GTP-C/GTP-U PGW IP
gtp.local_ip		GTP 宛先 TWAG IP
gtp.pgw_port	2123	PGW GTP-C ポート
gtp.gtpu_port	2152	GTP-U ポート
gtp.echo_enabled	true	PGW GTP エコー有効
gre.enabled	false	AP GRE 有効
gre.local_ip	0.0.0.0	GRE 宛先 IP
downlink.enabled	true	ダウンリンク有効
dhcp.enabled	false	DHCPv4 有効
dhcp.interface		DHCPv4 インタフェース "gre"
ipv6_ra.enabled	false	IPv6 RA DHCPv6 有効
nsw.enabled	false	NSWO 有効

配置名	値	説明
<code>nswo.local_pool</code>		ローカル IP アドレスの CIDR
<code>nswo.nat_interface</code>		NAT/ファイアウォール NIC
<code>online_charging.mode</code>	<code>:disabled</code>	Gy 設定 設定: <code>:disabled</code> <code>:auth</code> 設定: <code>:online_charging</code>
<code>rate_limit.max_failures</code>	10	接続失敗の最大回数 (NAS からの接続失敗) 単位: 回
<code>rate_limit.window_ms</code>	60000	接続失敗の監視ウィンドウ (ms)
<code>prometheus.port</code>	9568	Prometheus 監視ポート
<code>radius_config.acct_interim_interval</code>	300	AP からの接続失敗の監視 Accept 回数

概要

EAP-AKA' (TS 33.402 A.2 RFC 5448)

```
HSS 送信: RAND, AUTN, XRES, CK, IK
TWAG 受信: CK' = KDF(CK || IK, FC=0x20, network_name, SQN(+)AK)
          IK' = (KDF 受信)
          MK  = PRF'(IK' || CK', "EAP-AKA'" || Identity)
          MSK, EMSK, K_aut = MK
```

EAP-AKA (RFC 4187)

```
HSS 送信: RAND, AUTN, XRES, CK, IK
TWAG 受信: MK = SHA1(Identity || IK || CK)
          K_encr, K_aut, MSK, EMSK = PRF(MK)
```

TS 24.302 WLAN:mnc{MNC}.mcc{MCC}.3gppnetwork.org

TS 24.302

- TS 24.302
- RADIUS TS 24.302
- RADIUS TS 24.302
- IP TS 24.302
- TS 24.302
- TS 24.302 AP TS 24.302
- TS 24.302 TS 24.302
- TS 24.302
- TS 24.302

TS 24.302

無線LAN (EAP-AKA / EAP-AKA' / EAP-SIM)

概要

OmniTWAG は RADIUS 経由で TWAG 経由で EAP 認証を行う無線LAN TWAG として 3GPP AAA 規格に準拠した EAP 認証を行う Diameter SWx と HSS と接続し、無線LAN TWAG として (MSK) と MPPE による RADIUS Access-Accept による 802.11 認証を行う。

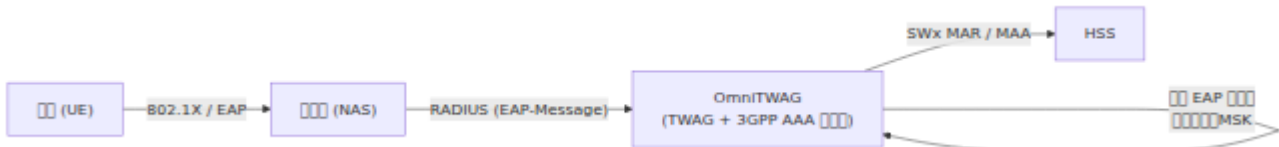
無線LAN TWAG は RADIUS 経由で RADIUS 認証を行う無線LAN TWAG として 3GPP AAA 規格に準拠した EAP 認証を行う Diameter SWx と HSS と接続し、無線LAN TWAG として (MSK) と MPPE による RADIUS Access-Accept による 802.11 認証を行う。

機能

- 無線LAN
- EAP 認証
- EAP 認証
- 無線LAN
- 無線LAN
- 無線LAN NAI 認証
- 無線LAN
- 無線LAN
- 無線LAN (ERP)
- 無線LAN
- 無線LAN
- 無線LAN

1. IEEE 802.1X EAP 2. EAP 3. RADIUS 4. EAP-Message 5. TWAG 6. TWAG 7. EAP 8. RADIUS

	RADIUS	
	Access-Challenge	EAP
	Access-Accept	MSK MPPE
	Access-Reject	EAP-Failure



TWAG 1. EAP 2. (XRES) 3. (K_{aut}) 4. RAND 5. MSK 6. MSK (EMSK) 7. TWAG 8. MPPE 9. TWAG

EAP

TWAG 1. EAP 2. TLV 3. EAP

阶段 (eap_phase)	描述
identity	客户端向TWAG发送 EAP-Identity 消息，TWAG 返回 AKA-Identity 消息。EAP-SIM/Start 消息中携带 identity。
aka_challenge	TWAG 向客户端发送 EAP-AKA 消息，包含 AKA-Challenge。
aka_prime_challenge	TWAG 向客户端发送 EAP-AKA' 消息，包含 AKA-Challenge。
sim_challenge	TWAG 向客户端发送 EAP-SIM 消息，包含 SIM/Challenge。
authenticated	EAP 消息中携带 TWAG 返回的 MSK，客户端返回 S2a。
session_active	S2a 向客户端返回 RADIUS 消息，表示会话激活。

客户端状态机

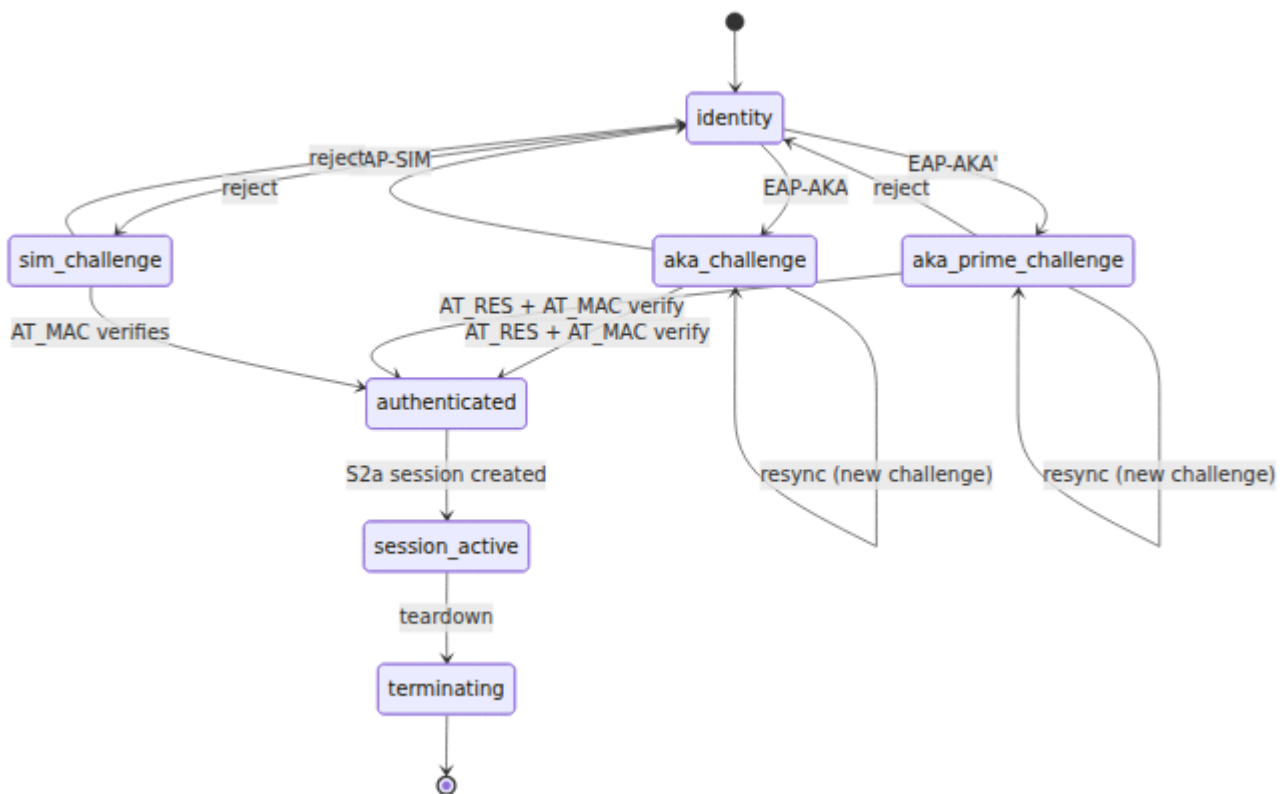
- resyncing 客户端向TWAG发送 EAP-Resync 消息，TWAG 返回 EAP-Resync 消息。客户端在收到 EAP-Resync 消息后，将状态重置为 identity。
- terminating 客户端向TWAG发送 EAP-Terminate 消息，TWAG 返回 EAP-Terminate 消息。客户端在收到 EAP-Terminate 消息后，将状态重置为 identity。

客户端在收到 EAP-Resync 消息后，将状态重置为 identity。客户端在收到 EAP-Terminate 消息后，将状态重置为 identity。客户端在收到 EAP-Resync 消息后，将状态重置为 identity。客户端在收到 EAP-Terminate 消息后，将状态重置为 identity。

客户端在收到 EAP-Resync 消息后，将状态重置为 identity。客户端在收到 EAP-Terminate 消息后，将状态重置为 identity。



EAP-AKA' EAP-AKA EAP-SIM Start
 AKA-Identity AT_RES



□□□□

1. □□□□□□□□□□ (NAI) □ EAP-Identity □□□
2. □□ EAP-AKA' □ EAP-AKA □ TWAG □□□□ AT_PERMANENT_ID_REQ □ AKA-Identity □□□□□□□□□□□□
3. □□ EAP-SIM □ TWAG □□□□ AT_VERSION_LIST (□□□ 1) □ EAP-SIM/Start □□□
4. □□□□□□□□ (AKA-Identity □ AT_IDENTITY) □□ EAP-SIM □ Start □□□ AT_NONCE_MT □

□□□□

1. TWAG □□□□□□ IMSI □□□□□□□□□□□□ □□□□□□□□
2. TWAG □□□□□□□□ MSK □□□□ □□□□□□□□
3. TWAG □□□□□□□□ EAP-AKA' □□□□□□ AT RAND □ AT AUTN □ AT_KDF_INPUT (□□□□) □ AT_KDF □ AT_BIDDING □ AT_RESULT_IND □ AT_MAC □□□□ EAP-AKA □□□□□□ AT RAND □ AT AUTN □ AT_MAC □□□□ EAP-SIM □□□□□□ AT RAND (2 □ 3 □ RAND) □ AT_MAC □
4. □□□□□□ (AUTN) □ MAC □□□□□□□□□□
5. TWAG □□□□□□□□ □□□□□□

6. TWAG EAP-Success MSKTWAG EAP-Failure

TWAG EAP-AKA EAP-AKA'

- 1. K_aut XRES TWAG
- 2. AT_RES AT_MAC TWAG
- 3. MAC TWAG AT_MAC K_aut MAC EAP-AKA HMAC-SHA1 EAP-AKA' HMAC-SHA-256 16
- 4. TWAG AT_RES XRES TWAG
- 5. MAC TWAG TWAG

EAP-SIM AT_RES AT_MAC MAC TWAG

TWAG auth_mode

	auth_mode		
SWx	:swx	HSS Diameter SWx (MAR/MAA)	
	:swx (:local)	Milenage	

TWAG EAP :swx auth_mode Milenage

STa Twag.Auth.StaAuth Diameter STa (DER/DEA) EAP 3GPP AAA AAA EAP-AKA MSK RADIUS :swx :local TWAG EAP

SWx → HSS

→ : SWx → TWAG → SWx → HSS → (MAR) (3GPP TS 29.273 → ID 16777265) → MAR →

- User-Name: IMSI
- SIP-Auth-Data-Item → SIP-Authentication-Scheme → EAP-AKA → SIP-Authorization → RAND → AUTS
- SIP-Number-Auth-Items: 1
- ANID: WLAN
- RAT-Type: 0
- Auth-Session-State: 1 (NO_STATE_MAINTAINED)
- Vendor-Specific-Application-Id → SWx Auth-Application-Id 16777265 → Vendor-Id 10415

HSS → (MAA) → Result-Code → 2001 (→) → TWAG → SIP-Auth-Data-Item →

- SIP-Authenticate = RAND (16 →) → AUTN (16 →)
- SIP-Authorization = XRES
- Confidentiality-Key = CK
- Integrity-Key = IK

→ Result-Code → 2001 → TWAG → → EAP-Failure

SWx → (Experimental-Result)

SWx → Experimental-Result AVP (Vendor-Id 10415) → Result-Code → MAA → Result-Code → TWAG → Experimental-Result-Code → →

→ DIAMETER_ERROR_USER_UNKNOWN (5001) → DIAMETER_ERROR_USER_NO_NON_3GPP_SUBSCRIPTION (5450) → SWx → 3GPP → S6a →

DRA → SWx → HSS → DRA →

▣▣ Milenage

```
□ :local □□□TWAG □□□□□□□□ (test_keys) □□ Milenage □□□□□ (3GPP TS  
35.206)□□□□ ki□opc□amf □ sqn□TWAG □□□□□ RAND□AUTN□XRES□CK □ IK□□□  
□□□□□□□□□□
```

EAP-SIM ☐☐☐

EAP-SIM 2 3 GSM TWAG Milenage TWAG
RAND XRES 4 SRES c3 CK IK GSM Kc (3GPP TS
33.102 C.3) Kc = CK[0..7] XOR CK[8..15] XOR IK[0..7] XOR IK[8..15]

000 NAI 00

```
000000000000 (NAI) NAI 0000 {username}@{realm} 0000
0{IMSI}@wlan.mnc{MNC}.mcc{MCC}.3gppnetwork.org 0
```

IMSI

TWAG 00000000 NAI 000 IMSI0

1. TWAG 電話番号 @ 電話番号
2. TWAG 電話番号 EAP-AKA/AKA' 6 番号 IMSI 番号
IMSI

TWAG 00 14 0 15 00 IMSI 0000000000000000 SWx MAR 00 User-Name 00 TWAG 0000 IMSI 0000000000000000
 GTP-C 0000 MCC 0 MNC 000000 MCC 00 3 00 MNC 000000 2 00

□□□□ (EAP-AKA')

EAP-AKA' AT_KDF_INPUT CK' / IK' network_name TWAG 3GPP TS 24.302 Diameter

- TWAG 通过 epc. 连接
- TWAG 通过 WLAN: 连接

□□□□ epc.mnc{MNC}.mcc{MCC}.3gppnetwork.org □□□□□

WLAN:mnc{MNC}.mcc{MCC}.3gppnetwork.org □□□□□□ epc. □□□TWAG □□ WLAN□

□□□□□□□□

□□□□□ 64 □□□□□□□□ (MSK) □ 64 □□□□□ MSK (EMSK)□TWAG □ MSK □□□□□□□
TWAG □□ EMSK □□□□□□□□□□□□

EAP-AKA' □□□□□□

□□ RFC 5448 □ 3GPP TS 33.402 □□ A□

1. $\text{Key} = \text{CK} \parallel \text{IK}$ □
2. $S = \text{FC}(0x20) \parallel \text{network_name} \parallel \text{len}(\text{network_name}) \parallel (\text{SQN XOR AK}) \parallel \text{len}(\text{SQN XOR AK})$ □
3. $\text{CK}' \parallel \text{IK}' = \text{HMAC-SHA-256}(\text{Key}, S)$ □ CK' □□ 16 □□□ IK' □□□ 16 □□□
4. $\text{MK} = \text{PRF}'(\text{IK}' \parallel \text{CK}', \text{"EAP-AKA'"} \parallel \square)$ □ 208 □□□ PRF' □□ HMAC-SHA-256 □
5. TWAG □ MK □□□□ K_encr (16)□ K_aut (32)□ K_re (32)□ MSK (64)□ EMSK (64)□

EAP-AKA □□□□□□

□□ RFC 4187□

1. $\text{MK} = \text{SHA1}(\square \parallel \text{IK} \parallel \text{CK})$ □
2. TWAG □□ FIPS 186-2 PRF □ MK □□□ 160 □□□
3. TWAG □□□□□ K_encr (16)□ K_aut (16)□ MSK (64)□ EMSK (64)□

EAP-SIM □□□□□□

□□ RFC 4186□

1. $\text{MK} = \text{SHA1}(\square \parallel \text{Kc1} \parallel \text{Kc2} \parallel [\text{Kc3}] \parallel \text{NONCE_MT} \parallel \square \parallel \square \parallel \square \parallel \square)$ □
2. TWAG □□ FIPS 186-2 PRF □ MK □□□ 160 □□□
3. TWAG □□□□□ K_encr (16)□ K_aut (16)□ MSK (64)□ EMSK (64)□

MSK と MPPE の関係

TWAG が 64 ビット MSK を生成し RADIUS Access-Accept から TWAG が MSK を生成し Microsoft MPPE を行う (RFC 2548, RFC 3748)

- MS-MPPE-Recv-Key = MSK[0..31] (32 ビット)
- MS-MPPE-Send-Key = MSK[32..63] (32 ビット)

TWAG が RADIUS から RFC 2548 から TWAG が 2 ビット TWAG が Access-Accept から TWAG が RADIUS から

MPPE を MSK を 802.11 を

Access-Accept が 64 ビット MSK を MSK を TWAG が Access-Reject

AT_AUTS

(SQN) AT_AUTS AKA-Synchronization-Failure (0x04) AT_AUTS 14 ビット SQN 8 ビット MAC-S

TWAG が

1. TWAG が RAND を 14 ビット AUTS
2. TWAG が Milenage f5* SQN MAC-S
3. TWAG が SQN Access-Challenge
4. TWAG が EAP-Failure

TWAG が EAP-AKA EAP-AKA' Milenage AT_AUTS 14 ビット TWAG が

拡張認証プロトコル (ERP)

TWAG が RFC 6696 EAP (ERP) ERP EAP TWAG が EMSK

TWAG が

- $rRK = \text{EMSK}$
- $rIK = rRK$

ERP HMAC-SHA-256

EAP-Initiate/Re-auth (EAP 41) TWAG

1. TWAG rRK rIK TWAG AKA-Identity
2. TWAG TWAG rIK
3. TWAG rRK MSK ($rMSK$) EAP-Finish/Re-auth (EAP 42) $rMSK$ MPPE MSK
4. TWAG

SWx Diameter

EAP

SWx	TWAG
SWx MAR	TWAG EAP-Failure
AT_MAC	Access-Reject EAP-Failure
AT_RES XRES	Access-Reject EAP-Failure
K_aut XRES	Access-Reject EAP-Failure
AT_RES AT_MAC	Access-Reject EAP-Failure
AKA-Authentication-Reject (0x02)	Access-Reject
AKA-Client-Error (0x0E)	Access-Reject
EAP-NAK	Access-Reject
MSK 64	Access-Reject
EAP	Access-Reject EAP-Failure

SWx Diameter

Result-Code Experimental-Result AVP Result-Code SWx

コード	エラー名	説明
2001	DIAMETER_SUCCESS	MAA 正常に 完了
5001	DIAMETER_ERROR_USER_UNKNOWN	HSS から IMSI が見つ かれない 3GPP/SWx セッション の S6a/EPC セッション
5004	DIAMETER_ERROR_NOT_SUPPORTED_USER_DATA	サポートされ ないデータ
5420	DIAMETER_ERROR_UNKNOWN_EPS_SUBSCRIPTION	未知の EPS サブスクリ プション
5450	DIAMETER_ERROR_USER_NO_NON_3GPP_SUBSCRIPTION	3GPP 以外の サブスクリ プションが ない
その他	(Diameter エラー)	上記以外の エラー

参照 RFC 6733 の 7.1 節 3GPP TS 29.273 3GPP TS 29.229

ネットワーク (STa)

3GPP AAA サーバから STa ネットワークの TWAG ネットワーク (3GPP TS 29.273) への ID (16777250) の
TWAG からの Result-Code (2001) の応答

消息	消息	TWAG 消息
ASR (Abort-Session-Request)	AAA 消息 消息	TWAG 消息 ASA
STR (Session-Termination-Request)	HSS 消息	TWAG 消息 STA
RAR (Re-Auth-Request)	AAA 消息 消息	TWAG 消息 RAA 消息 RAR 消息

消息 Diameter 消息 Origin-Host 消息 Origin-Realm 消息

消息

消息 :omnitwag 消息 Diameter 消息 (host 消息 realm) 消息 :diameter_ex
消息 TWAG 消息 SWx 消息 STa 消息 Origin-Host 消息 Origin-Realm

```
config :omnitwag,  
  # :swx ( ) :local ( )  
  auth_mode: :swx,  
  
  # EAP :eap_aka_prime ( ) :eap_aka :eap_sim  
  eap_method: :eap_aka_prime,  
  
  # EAP-AKA' Diameter  
  network_name: "WLAN:mnc001.mcc001.3gppnetwork.org",  
  
  # 600000 (10 )  
  identity_timeout_ms: 600_000,  
  
  # TWAN STa DER/AAR  
  twan_ssid: "ons",  
  twan_bssid: "00:11:22:33:44:55",  
  
  # auth_mode: :local  
  test_keys: %{\n    ki: "11111111111111111111111111111111",\n    opc: "22222222222222222222222222222222",\n    amf: "8000",\n    sqn: 1200\n  }
```

項目	型別	必須	デフォルト値	説明
auth_mode	Atom	○	:local	認証モード。:swx は HSS 認証、:local は test_keys で定義された鍵を使用した認証。:swx はデフォルト。
eap_method	Atom	○	:eap_aka_prime	EAP 方法。EAP-NA は EAP-NA'、EAP-AKA は EAP-AKA'、EAP-AKA' は EAP-AKA'。:eap_aka_prime は EAP-AKA' の別名。EAP-NA は EAP-NA' の別名。
network_name	String	○		EAP-AKA' のネットワーク名。TWAG の TS 番号。
identity_timeout_ms	Integer	○	600000	アイデンティティタイムアウト時間（ミリ秒）。
twan ssid	String	○	-	WLAN SSID。AAR が TWAN を使用する。
twan_bssid	String	○	-	BSSID。twan ssid と同じ。
test_keys	Map	○		Milenage 鍵。:local は test_keys で定義された鍵。

設定例

TWAG の :local test_keys は test_keys で定義された鍵。

10

RADIUS ☐☐☐☐

項目	説明
radius_access_challenge_count	Access-Challenge 回数 (EAP 挑戦回数)
radius_access_accept_count	Access-Accept 回数 (接続成功回数)
radius_access_reject_count	Access-Reject 回数 (接続拒否回数)

STa ☐ ☐ ☐ ☐

```
STa [][:sta, :asr, :received, :count] [][:sta,  
:str, :received, :count] [] [:sta, :rar, :received, :count] []  
disconnect [] reauth []
```

11

TWAG identity imsi

--	--	--	--

項目	説明	備考
不明なエラーコード DIAMETER_ERROR_USER_UNKNOWN (5001)	HSS から IMSI から 3GPP/SWx から 不明なエラー S6a/EPC から 不明な 3GPP から 不明	HSS から 不明
不明なエラーコード DIAMETER_ERROR_USER_NO_NON_3GPP_SUBSCRIPTION (5450)	不明なエラー 3GPP から	HSS から
不明なエラーコード MAA から Result-Code=nil	SWx から Experimental- Result AVP から 不明なエラー Result-Code	不明なエラー Experimental- Code
不明なエラーコード	不明なエラー 不明なエラー	不明な identi
不明な AKA-Authentication-Reject	UE から (AUTN から)	不明な :1 HS 不明
不明な AKA-Synchronization-Failure	SQN から SQN	TWAG 不明な 不明
Access-Reject から EAP-Success	MSK から 64 から	不明な MAA

☐ EAP
 ☐ RADIUS
 ☐ RADIUS



OmniTWAGUEWiFiAPTWAG
HSSSWxPGWS2aOCSGy

S2a GTP-C IP IP RADIUS



- 0000000
- 0000 - 000000
- 00 - NSW00IP
- 00 - NSW00DHCP00
- 00 - 000000AP000000
- 0000 - 000000SWx00
- 00000000
- 000 000000000000
- 00 - 00000000
- 00 - 0000API000
- 00 - 0000
- 0000000

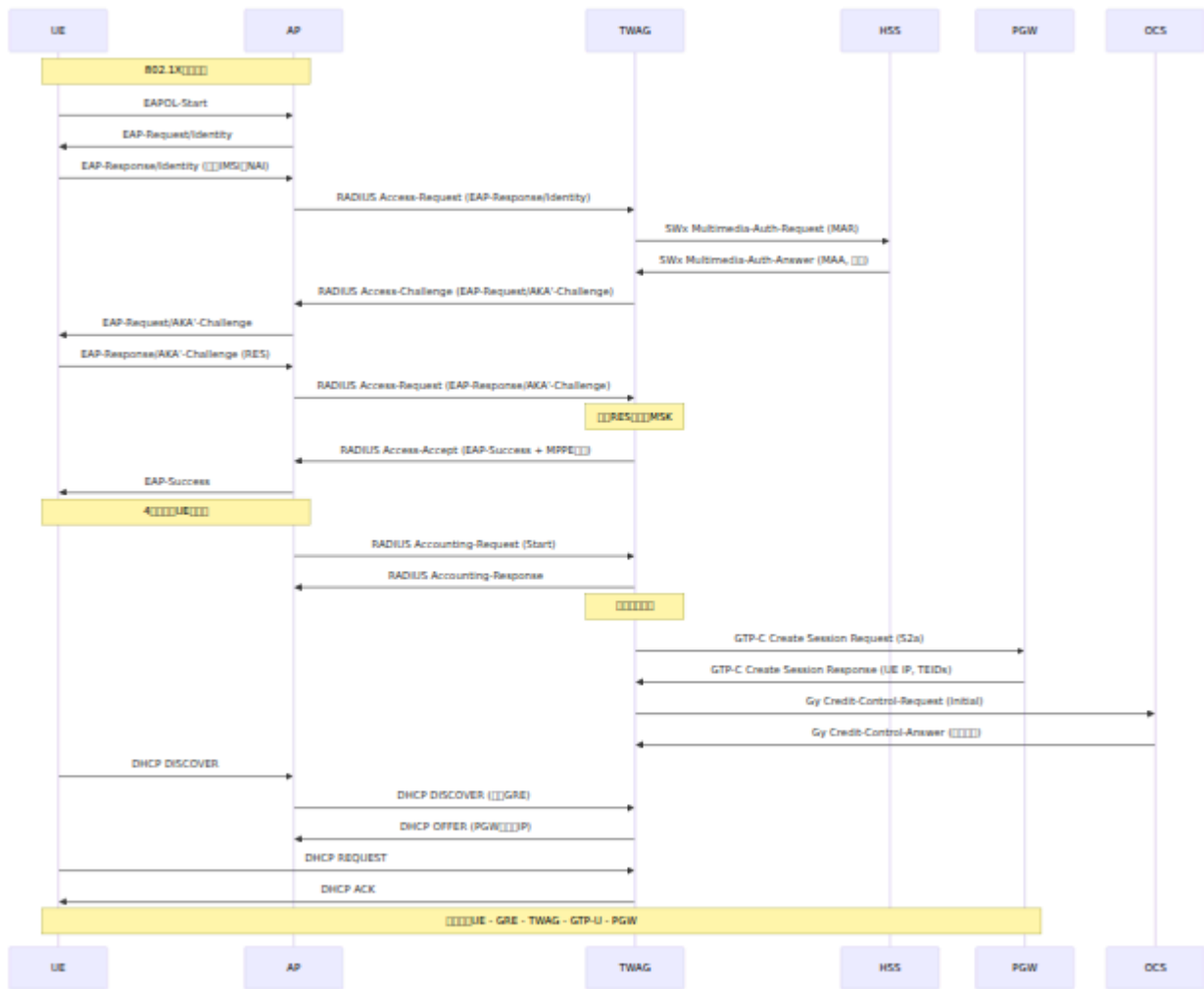
Network Elements

Network Element	Interface	Protocol
UE	Cellular SIM	802.11 / EAPOL
AP	WiFi	RADIUS
TWAG	WiFi	3GPP AAA
HSS	Cellular	Diameter SWx
PGW	Cellular	GTP-C/GTP-U
OCS	Cellular	Diameter Gy

Network Element: AP Interface: RADIUS

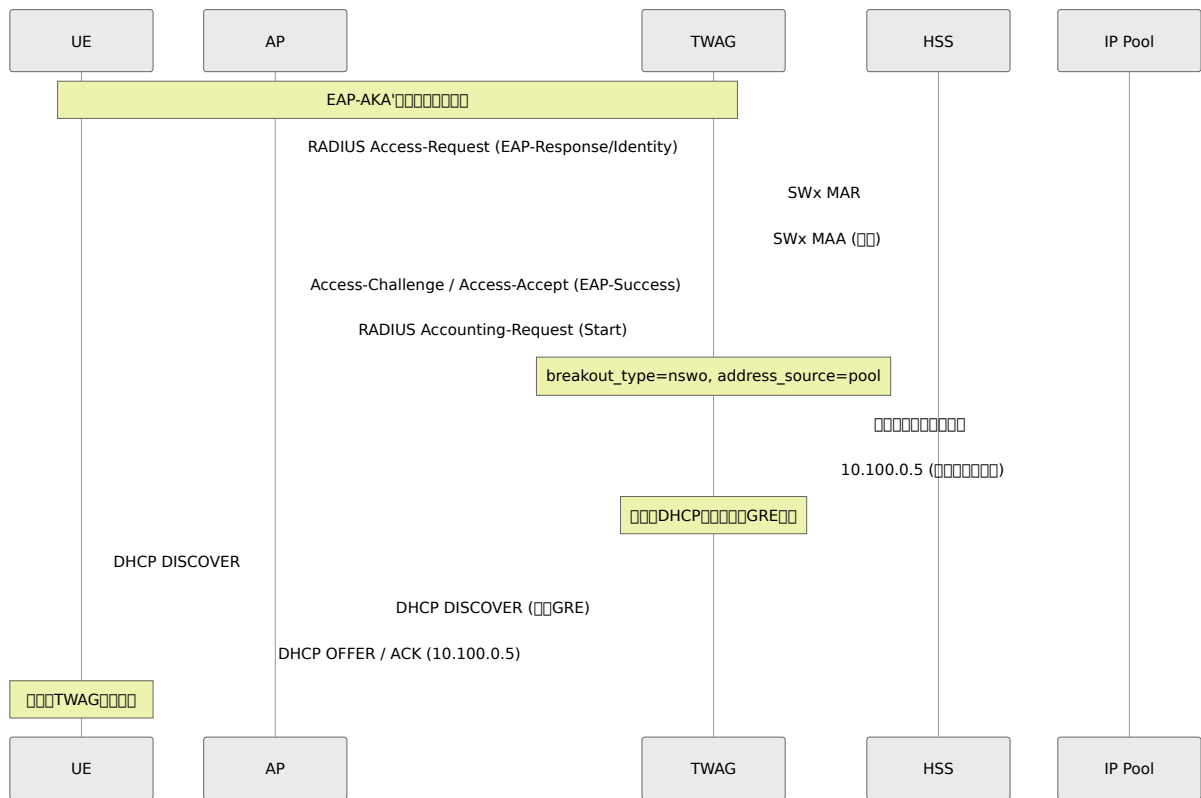
Network Element - Network Element

breakout_type: tunneled charging_mode: charged Network Element: TWAG EAP-AKA' Network Element: PGW GTP Network Element: DHCP Network Element: PGW IP Network Element: Gy



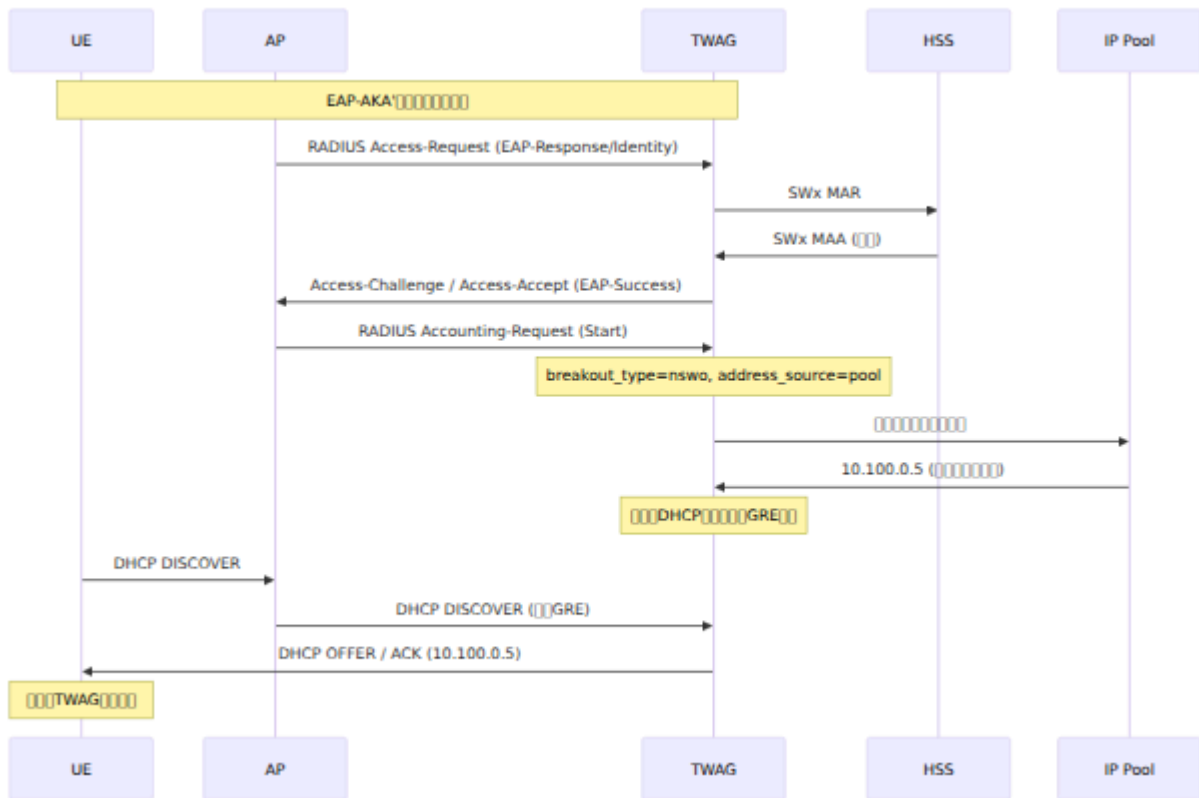
- NSWOPIP

breakout_type: nsw address_source: pool TWAG PGW TWAG IP DHCP



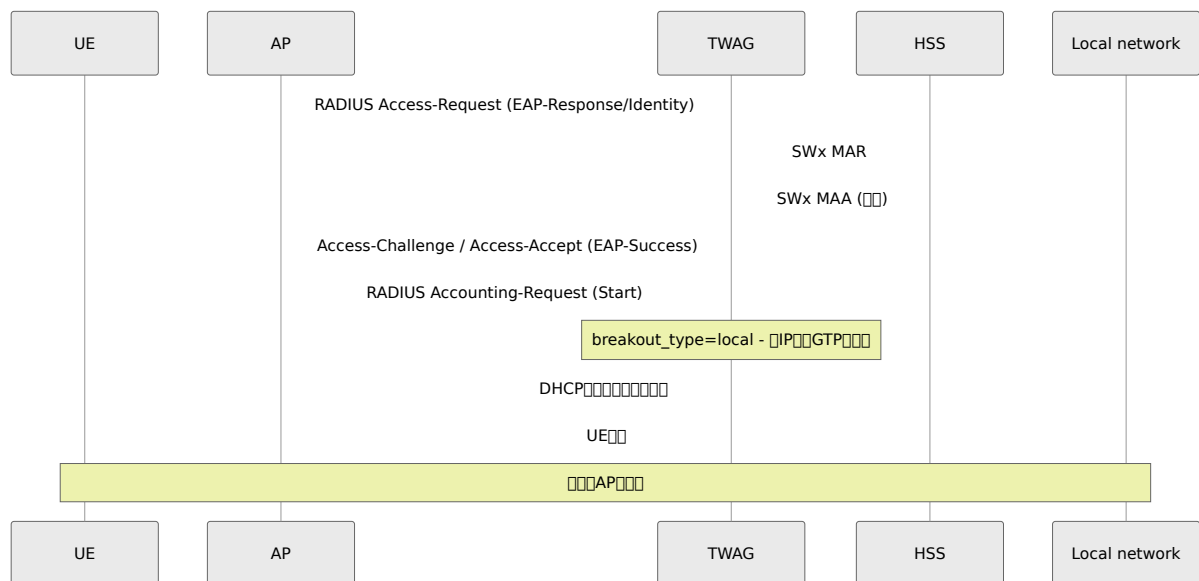
- NSWO DHCP

breakout_type: nswo address_source: dhcp TWAG MAC UE
 DISCOVER DHCP IP



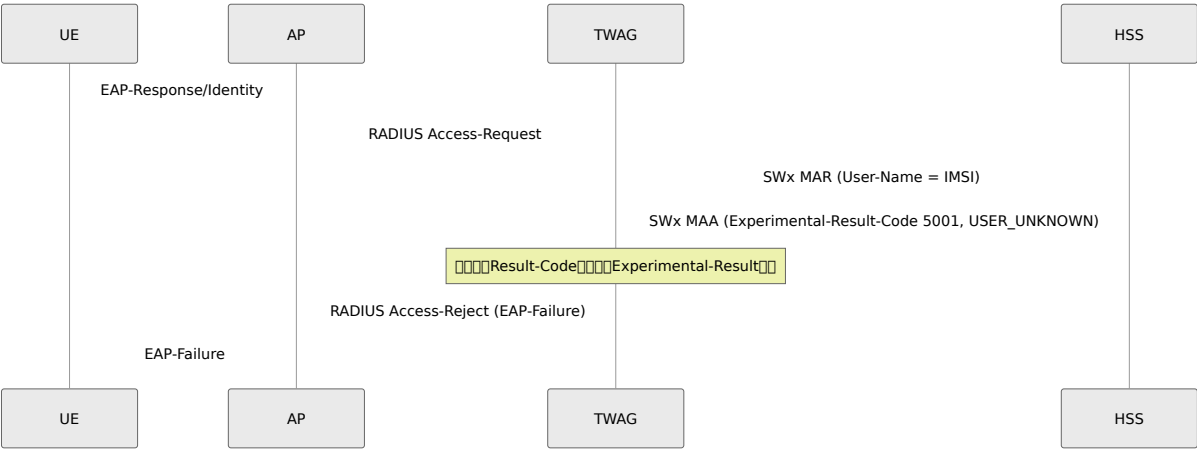
- AP

breakout_type: local AP TWAG IP UE TWAG GTP DHCP

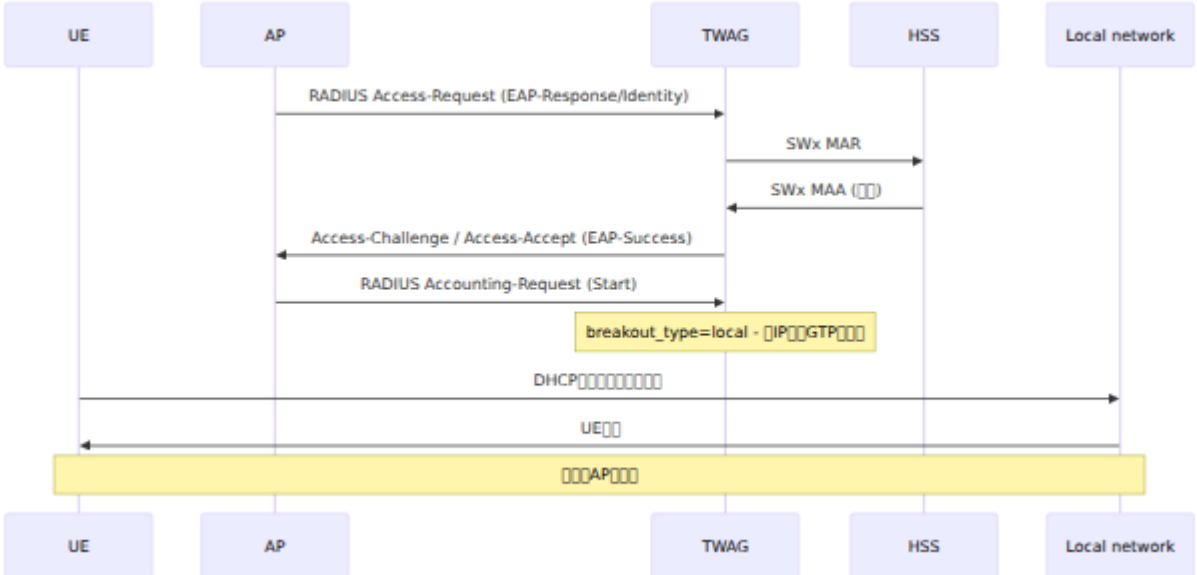


SWx

HSS Experimental-Result AVP SWx Result-Code S6a
HSS IMSI 3GPP SWx TWAG

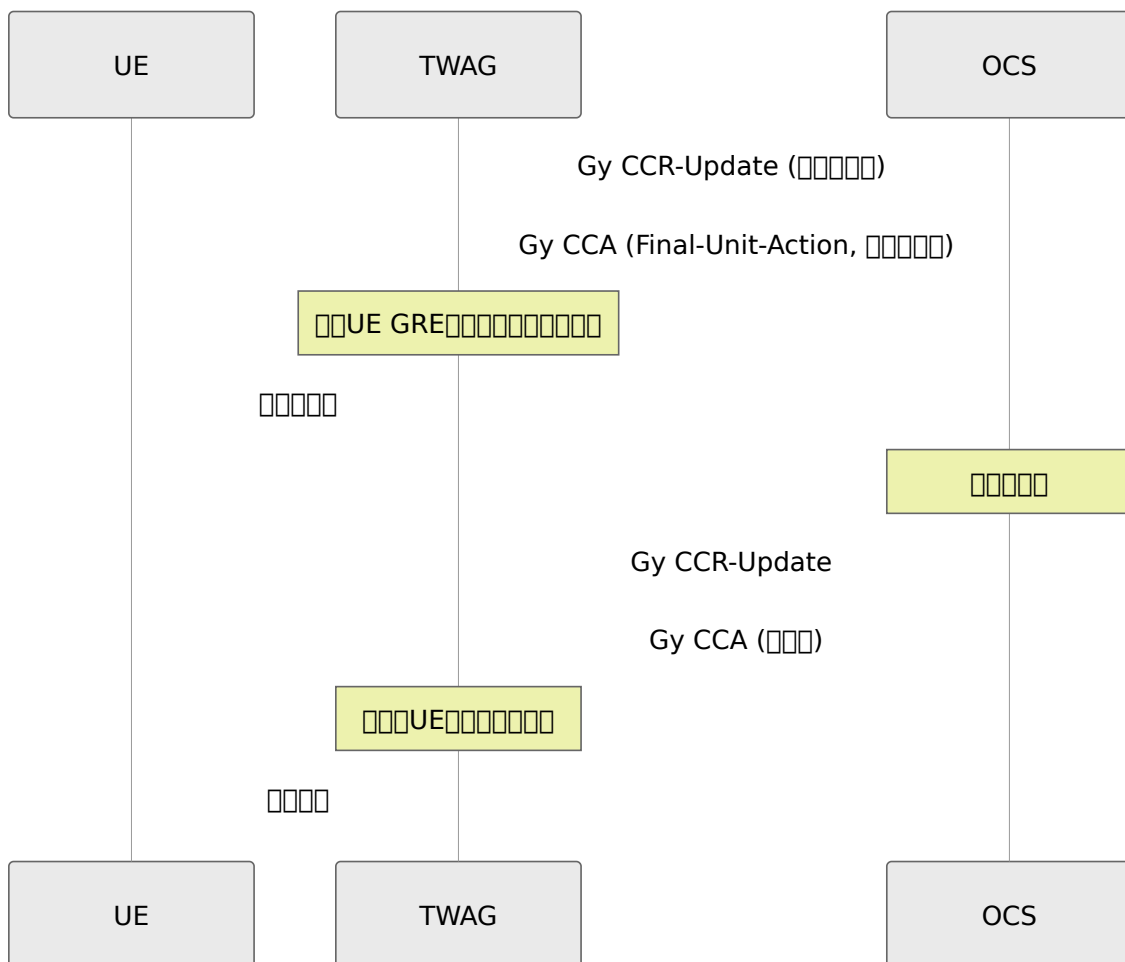


AP RADIUS TWAG charged
OCS



1. 3GPP TS 29.274

OCS (HSS) TWAG UE TWAG (HSS) UE
 (HSS)

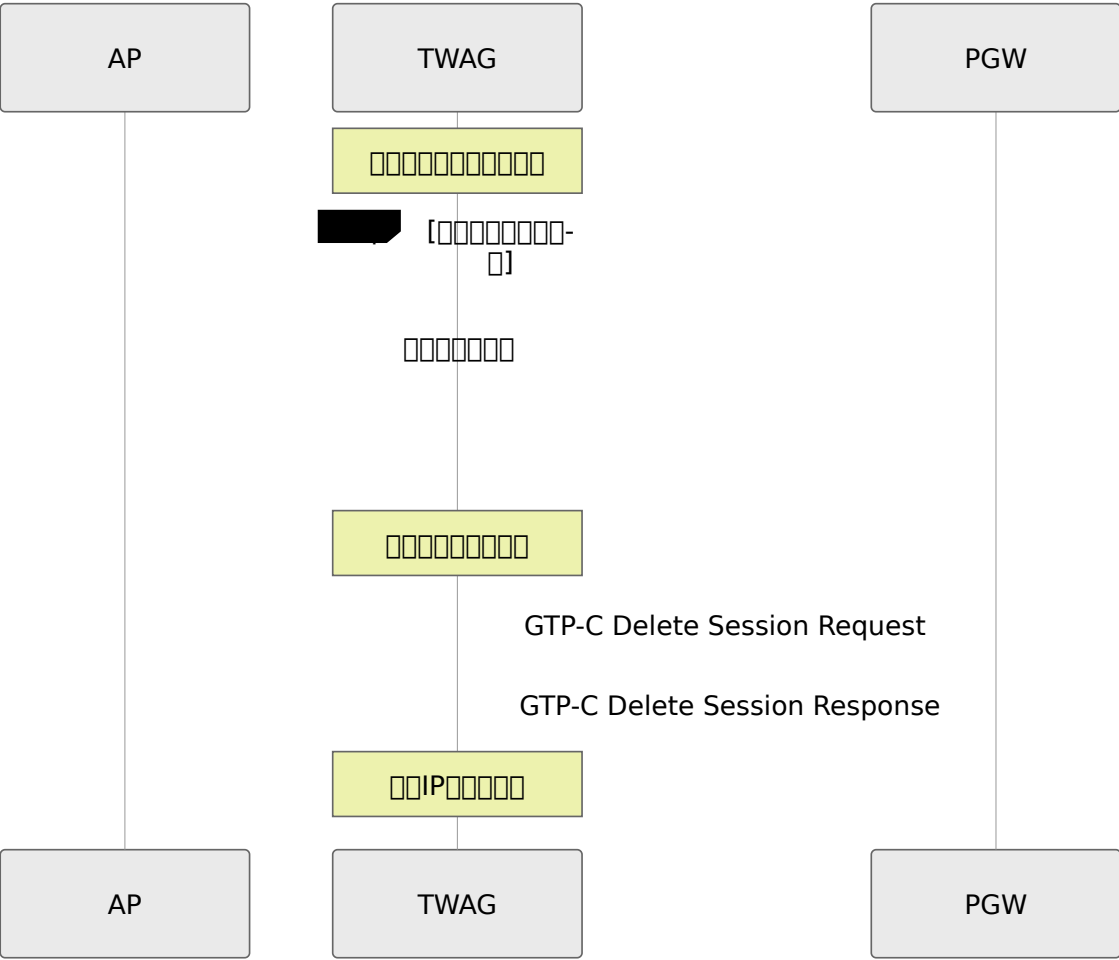


2. 3GPP TS 29.274

UE (HSS) AP Accounting-Stop TWAG (HSS) GTP (HSS) Gy CCR-Terminate (HSS)

TWAG AP TWAG

???

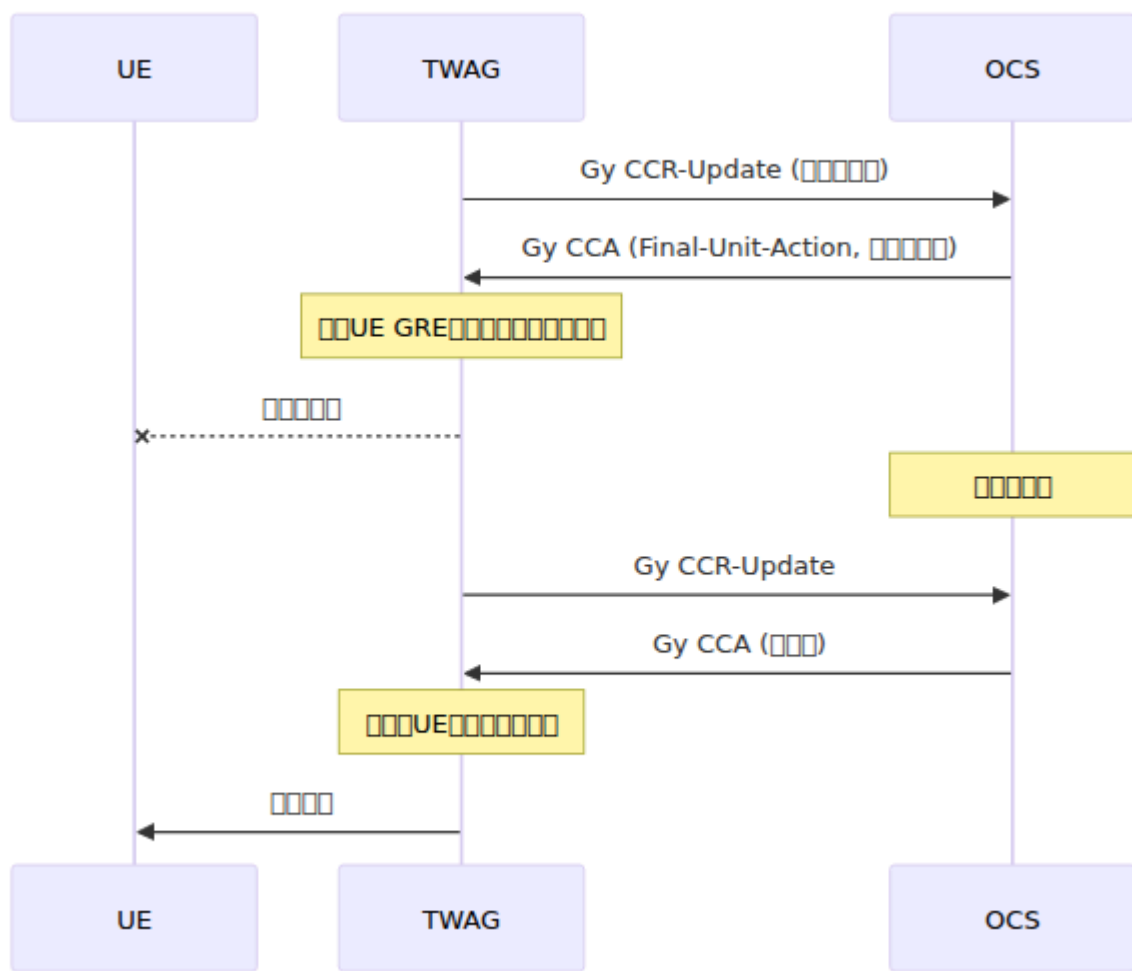


charging_mode

AP charging_mode RADIUS

charging_mode		Gy
charged	breakout_type	CCR-Initial / Update / Terminate
uncharged	breakout_type	Gy
auth_only		Gy

uncharged NSWO TWAG Gy auth_only Access-Accounting-Start TWAG



□□

WLCP

22

[illegible]

TWAG 3GPP TS 23.402 16 WLAN S2a

- 無線 LAN 標準
- IEEE 802.11 EAP-AKA / EAP-AKA' 認証方式
- S2a GTP-C 無線 LAN 標準
- IEEE 802.11 無線 LAN 標準
- IEEE 802.11 GRE GTP-U 無線 LAN 標準
- RADIUS 無線 LAN 標準 AP 無線 LAN 標準

11

- 〇〇〇〇〇〇〇〇
- 〇〇〇〇
- 〇〇〇〇〇〇〇〇 (TSCM)
- 〇〇〇〇〇〇 (SCM)
- 〇〇〇〇〇〇 (MCM) 〇 WLCF
- 〇〇〇〇〇〇〇〇
- 〇〇
- 〇〇〇〇〇
- 〇〇〇〇〇

--	--	--	--	--	--	--

TWAG ــ WLAN ــــــــ EPC ــــــــ UE ــــــــــــــــــــــــــــــــ UE ــــــــــــــــ IP ــــــــUE ــــــــــــــــ
 APNــــــــ UE ــــــــــــــــــــــــ PDN ــــــــ

項目	TSCM	SCM	MCM
モード	0	1	2
プロトコル	PPP	PPP	PPP
UE の PDN	なし	なし	なし
IP アドレス	DHCP	EAP	EAP / WLCP
UE の APN	なし	なし	PPP
IP アドレス	なし	なし	なし
モード	なし	なし	なし
UE のモード	なし	なし	WLCP (TS 24.244)
UE の S2a	なし	なし	なし

TSCM モードは TSCM モード Release-12 の UE の TWAG の APN の PDN の DHCP の PGW の IP の

モード 0 1 2 は 3GPP TS 29.273 の TWAN-Connection-Mode AVP

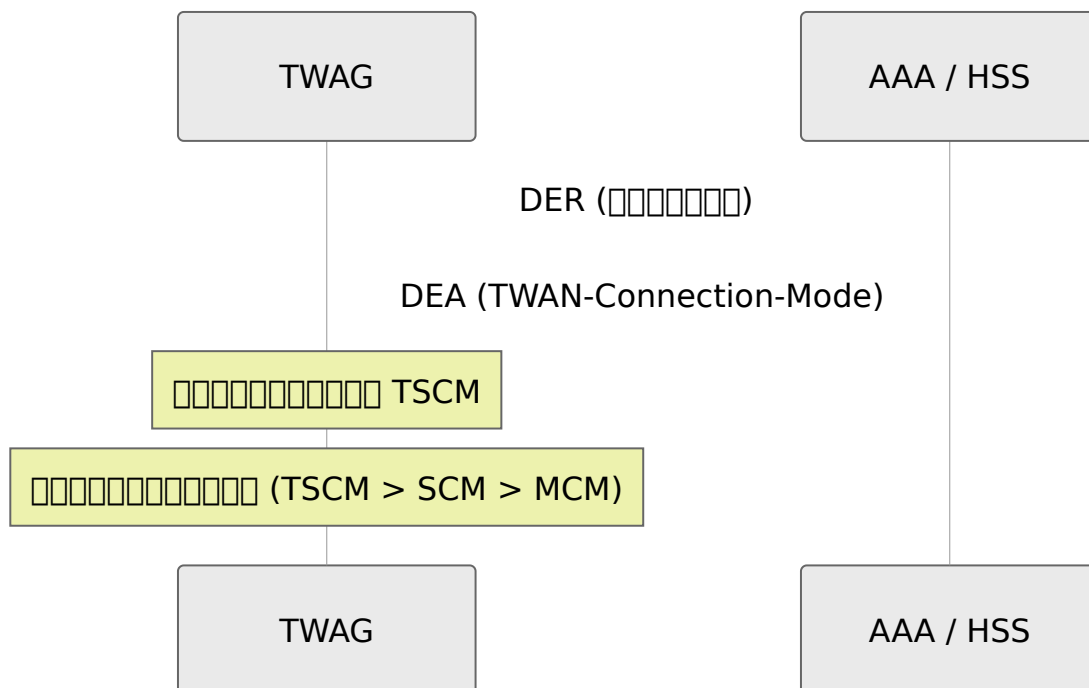


モード

モードは EAP の STa の EAP のモード

1. TWAG の AAA の connection_modes
2. AAA の STa DEA の TWAN-Connection-Mode AVP
3. AAA の TWAG の TSCM

AAA TWAG TWAG TSCM SCM MCM TWAG



TWAG AAA twan_connection_mode TWAN-Connection-Mode connection_mode Connection-Mode (0/1/2) (:tscm/:scm/:mcm) TSCM

STa STa DER TWAN-Connection-Mode AVP TWAG STa DEA DEA TSCM

STa (StaClient) DER DEA STa RADIUS store_connection_mode TSCM DHCP IP WLCPU APN TSCM

TSCM

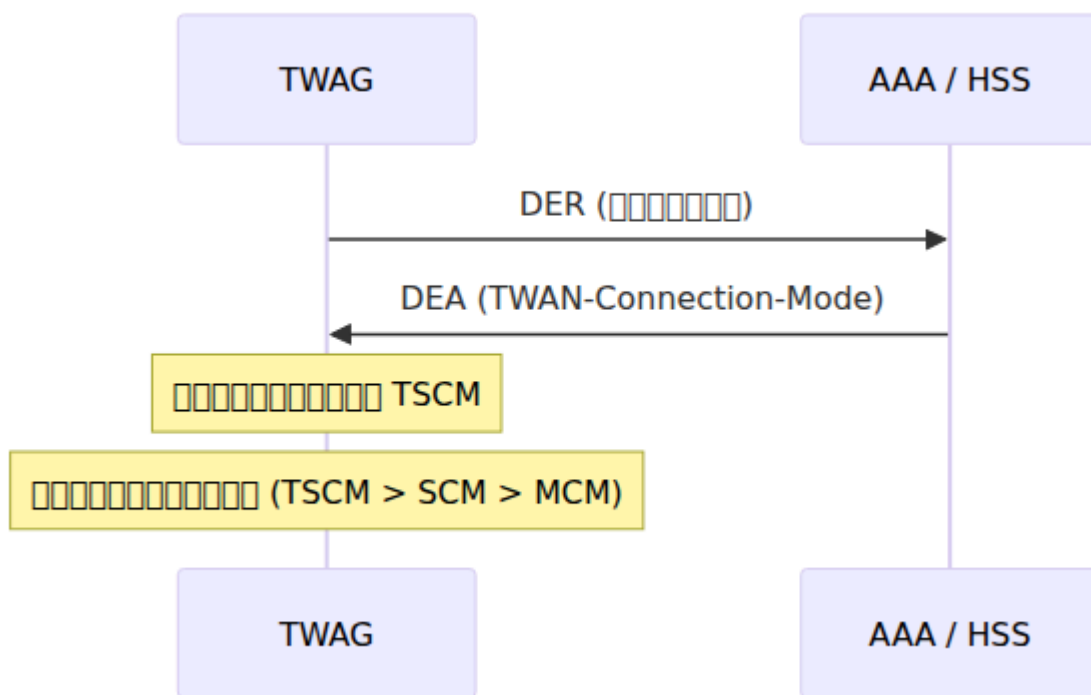
TSCM UE TSCM

TSCM

- UE PDN

- TWAG 向 APN UE 向 APN
- UE 向 DHCP 向 IP 向 TWAG 向 GRE 向 向 向
- 向 WLCP 向
- IP 向 UE 向 APN 向

TWAG 向 S2a 向 PGW 向 Handover Indication 向
 DHCP 向 UE MAC 向 PGW 向 IP 向 S2a GTP-C 向
 向 Gy 向 (Gy)



向 (SCM)

SCM 向 UE 向 PDN 向 UE 向 S2a 向 SCM 向 TSCM 向

- UE 向 EAP 向 IP 向 DHCP
- UE 向 APN
- UE 向 3GPP 向 WLAN 向 3GPP 向 IP 向
- PGW 向 向

向 SCM 向 SCM 向 EAP 向 IP 向 APN 向

SCM AAA UE SCM TSCM PDN DHCP IP EAP IP UE APN

MCM WLCP

MCM UE PDN UE WLCP 3GPP TS 24.244 TWAG PDN UE WLCP PDN TWAG PDN PGW S2a GTP-C

WLCP

TWAG UDP 2156 WLCP WLCP UE WLCP UE UE IP

WLCP TS 23.402 16.1.4A.3.1 DTLS UDP DTLS

WLCP

WLCP IE

(1) | (2) | ID (1) | IEs...

IE

IE (1) | IE (2) | IE ()

4

WLCP 프로토콜

코드	발신자	수신자	설명
0x01	PDN	UE ↔ TWAG	PDN 연결 설정
0x02	PDN	TWAG ↔ UE	IP 주소 / DNS / QoS 정보 교환
0x03	PDN	TWAG ↔ UE	연결 상태 확인
0x04	PDN	UE ↔ TWAG	PDN 연결 해제
0x05	PDN	TWAG ↔ UE	연결 재설정
0x06	관리자	TWAG ↔ UE	UE 관리 메시지

WLCP

	IE	
0x01	APN	
0x02	PDN	1 IPv4, 2 IPv6, 3 IPv4v6
0x03	PCO	
0x04	PDN	UE IP (IPv4IPv6 IPv4v6)
0x05	QoS	QCI MBR / GBR
0x06	TFT	
0x07		
0x08	ID	EPS (EBI)
0x09	DNS	DNS

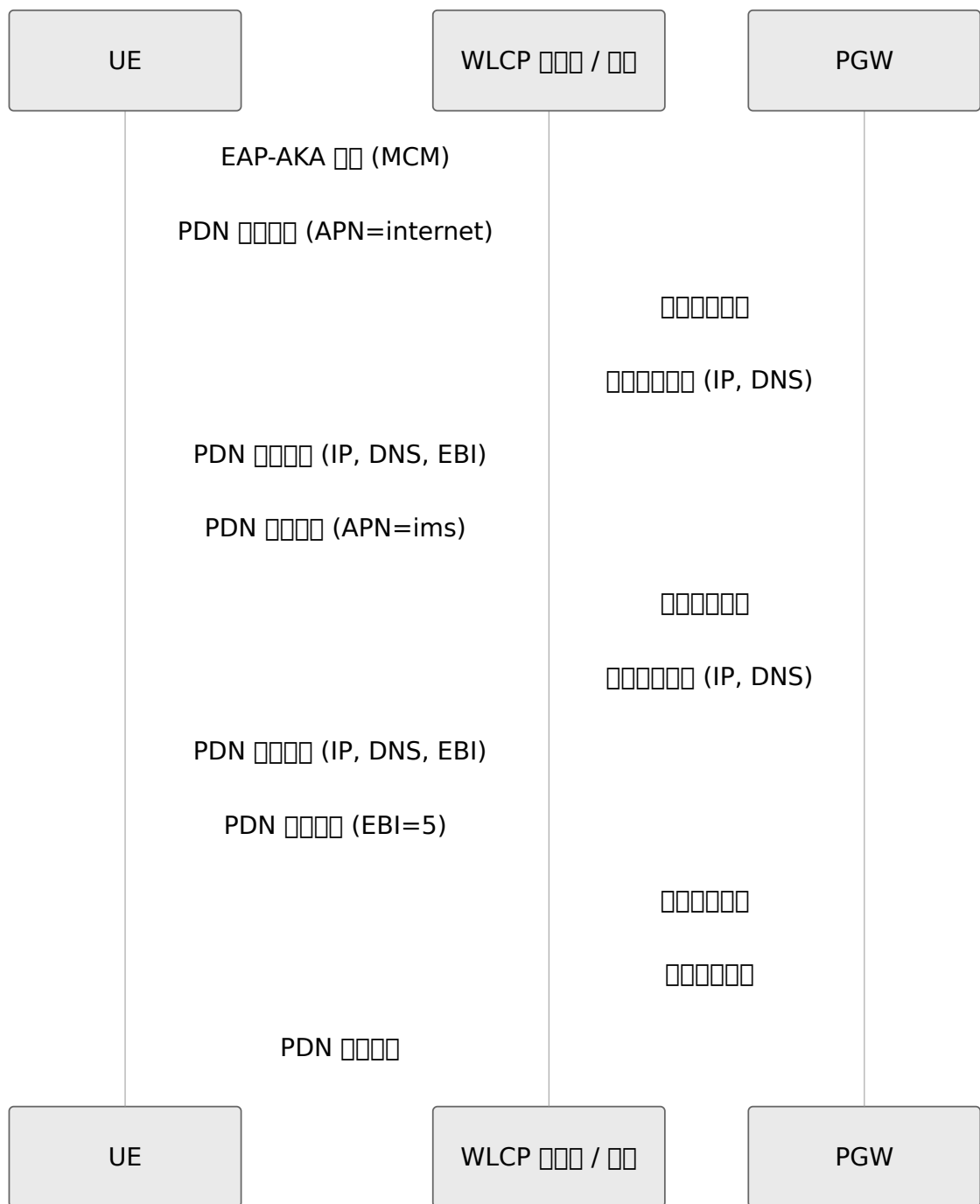
WLCP 消息

消息	消息	消息
0x10	request_accepted	消息成功
0x20	apn_not_allowed	UE 请求 APN
0x21	no_resources	资源不足
0x22	unknown_apn	APN 未知
0x23	user_auth_failed	用户认证失败
0x24	network_failure	网络失败 S2a 消息
0x25	pdn_type_not_supported	PDN 类型不支持
0x30	request_rejected	请求被拒绝

PDN 消息

PDN 消息包括 WLCP 消息和 PDN 消息

1. 消息成功 APN PDN 消息 PCO 消息 APN 消息 internet PDN 消息 IPv4
2. 消息 APN 消息 allowed_apns
3. 消息 UE 消息 IMSI 消息 user_auth_failed 消息
4. 消息 APN 消息 PGW 消息 S2a GTP-C 消息 RAT 消息 3 WLAN 消息 Handover Indication
5. 消息 GTP-U 消息 PDN 消息 EBI 消息 PDN 消息 UE IP 消息 DNS 消息 QoS 消息 EBI
6. 消息 S2a 消息 PDN 消息 network_failure



PDN

PDN connection establishment EBI PDN connection EBI PDN connection
 request_rejected EBI PDN connection S2a GTP-C PDN connection GTP-U PDN connection
 PDN connection UE PDN connection WLCP PDN connection

PGW PDN connection PDN connection GTP-U PDN connection PDN connection
 PGW PDN connection

WLCP

WLCP K_WLCP WLCP EAP (MSK) 3GPP TS 33.402 6.2 A

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

FC = 0x21
P0 = TWAG IP (IPv4 4 IPv6 16 )
L0 = P0 (2 )
```

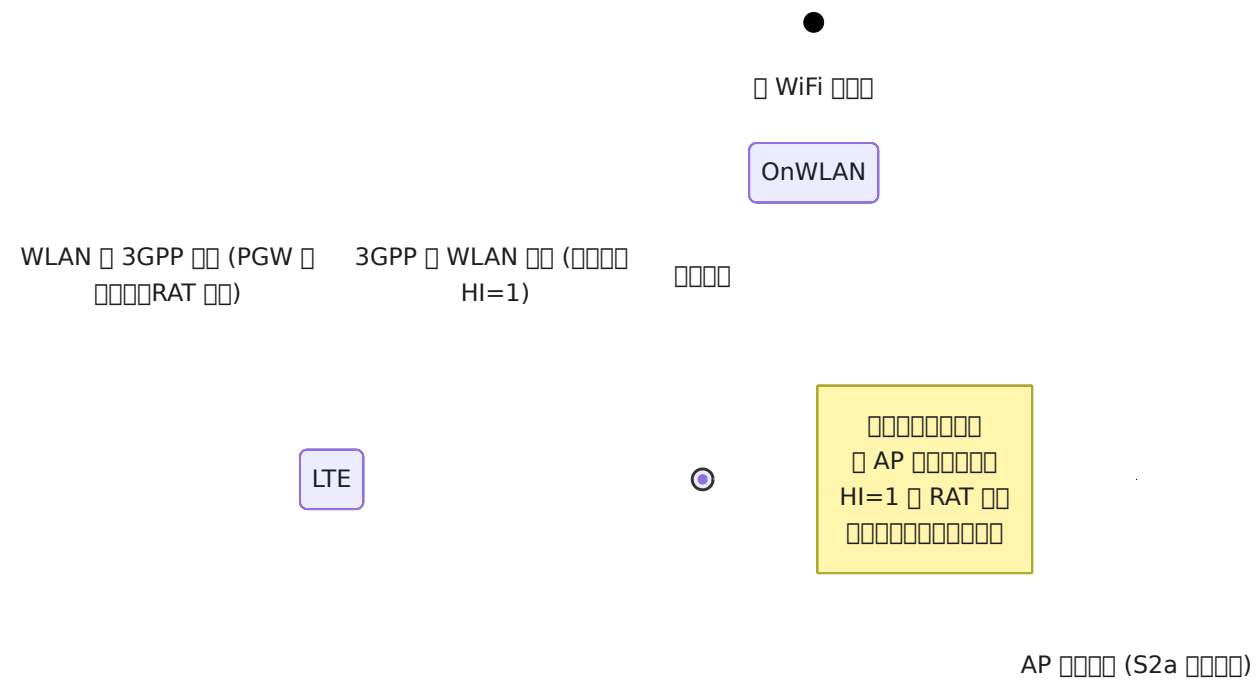
32 256 TWAG P0 IP TWAG IP MSK

MCM WLCP WLCP UDP UE

- WLCP DTLS UDP
- MCM QoS TFT IEs PGW
- MCM
-

MCM UE

TWAG TS 23.402 AP



3GPP 网络 WLAN 网络

当 UE 在 LTE 网络 WiFi 网络 TWAG 网络 IP 网络 TWAG 网络 Handover Indication (HI) 网络 PGW 网络 IP 网络 IE 网络 S2a GTP-C 网络

S2a 网络 Handover Indication 网络 UE 网络

网络 Handover Indication 网络 3GPP 网络 WLAN 网络 IP 网络

WLAN 网络 3GPP 网络

当 UE 在 WiFi 网络 LTE 网络 PGW 网络 RAT 网络 TWAG 网络 RAT 网络

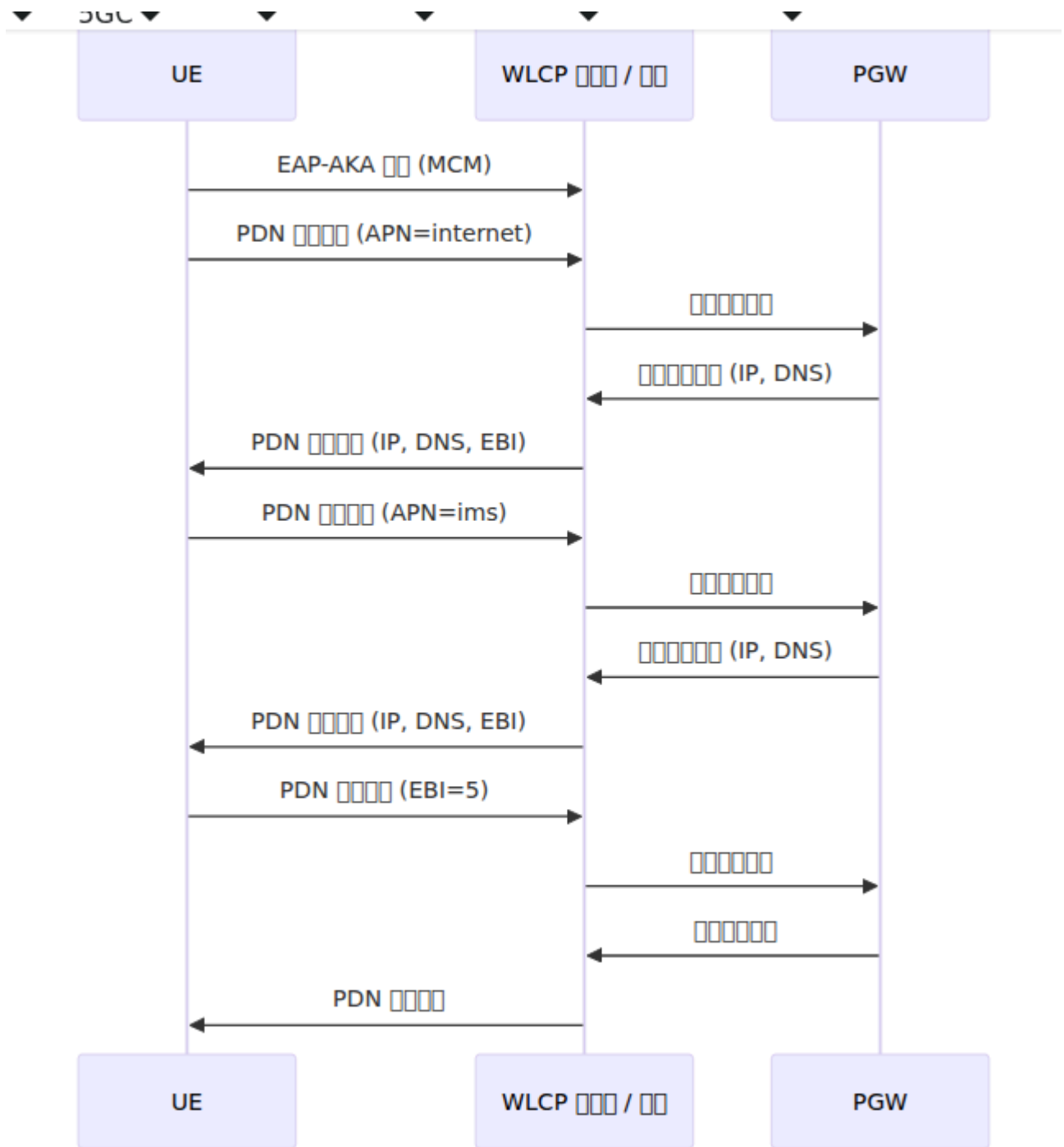
順序	項目
4	RAT 3GPP 3GPP
5	ISR
10	3GPP 3GPP

RAT TWAG WiFi GTP-C PGW UE
LTE IP

UE PGW
RAT

AP

UE TWAN WiFi AP S2a GTP UE
NAS-IP-Address Called-Station-Id TWAG UE AP
UE IP AP



AP UE NAS-IP Called-Station-Id
 AP AP AP

connection_modes TWAG [:tscm]

```
config :omnitwag,
  # TWAG AAA
  connection_modes: [:tscm, :scm]
```

項目	項目	項目	項目
connection_modes		[:tscm]	TWAG AAA :tscm :scm :mcm

STa DER RADIUS SCM MCM [:tscm]

WLCP (MCM)

WLCP :wlcp WLCP MCM

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

項目	項目名	項目名	項目名	項目名
enabled	項目名	項目名	false	項目名 WLCB 項目名 MCM項目名 false
port	項目名	項目名	2156	WLCB 項目名 UDP 項目名 2156 項目名 WLCB 項目名
allowed_apns	項目名	項目名	:all	WLCB PDN 項目名 APN 項目名 :all 項目名 項目名 APN項目名 APN 項目名 ["internet", "ims"] 項目名 APN 項目名 apn_not_allowed

項目名 WLCB 項目名 enabled 項目名

項目名

MCM PDN 項目名 S2a 項目名 GTP 項目名 gtp 項目名 pgw_ip
local_ip 項目名 mcc 項目名 mnc

項目名

WLCB 項目名

이벤트	설명
[:wlcplcp, :packet, :received]	WLCPLCP 패킷이 수신되었습니다. bytes
[:wlcplcp, :packet, :malformed]	WLCPLCP 패킷이 잘못되었습니다.
[:wlcplcp, :packet, :unregistered]	WLCPLCP 패킷이 등록되지 않았습니다.
[:wlcplcp, :pdn, :request]	PDN 요청이 들어왔습니다.
[:wlcplcp, :pdn, :accept]	PDN 요청이 승인되었습니다.
[:wlcplcp, :pdn, :reject]	PDN 요청이 거부되었습니다.
[:wlcplcp, :pdn, :disconnect_request]	PDN 연결 해제 요청이 들어왔습니다.
[:wlcplcp, :pdn, :disconnect_accept]	PDN 연결 해제 요청이 승인되었습니다.
[:twag, :handover, :in]	3GPP 또는 WLAN에서 이동이 시작되었습니다. source
[:twag, :handover, :out]	WLAN 또는 3GPP에서 이동이 종료되었습니다. cause
[:twag, :handover, :ap_change]	AP가 변경되었습니다.

WLCPLCP는 PDN을 관리하고 UE가 APN을 EBI를 통해 RAT에 연결할 수 있도록 합니다. AP는 UE가

이벤트

이벤트는 WLCPLCP와 TWAG 간의 상호작용을 나타냅니다.

項目	説明
TSCMがPDNにDHCP IPを	取得
AP がS2a に	接続
S2a Handover Indication が	通知される
STa DER が	通知される TWAN-Connection-Mode が SSID / BSSID が
STa DEA が	通知される StaClient が TSCM が
	store_connection_mode が
SCM がEAP IP APN が	通知される TSCM が
WLCP が IEs が	通知される
WLCP UDP が UE が	通知される
WLCP が K_WLCP が FC=0x21 が	通知される
MCM が PDN が	通知される
WLCP が DTLS が	通知される UDP が
MCM が	通知される
WLCP が	通知される

UE	Network
MCM 3GPP	3GPP
3GPP 3GPP WLAN IP	3GPP HI
WLAN 3GPP RAT	3GPP

IP 割当 (DHCP / DHCPv6 / IPv6 RA / PCO)

概要

OmniTWAG から UE へ IP 割当機能 (DHCPv4/DHCPv6/IPv6 SLAAC) と DNS と P-CSCF 割当機能を提供する。また、AP から AP へ

- ・ IP 割当機能は PGW から S2a GTP 経由で OmniTWAG から UE へ提供 (3GPP TS 23.402 § 16.2 参照)
- ・ **NSWO** 割当機能は PGW から OmniTWAG から **IP** を割当する **IP**

割当機能は RADIUS から AP へ提供。AP から breakout_type、address_source、ip_pool_id を RADIUS から

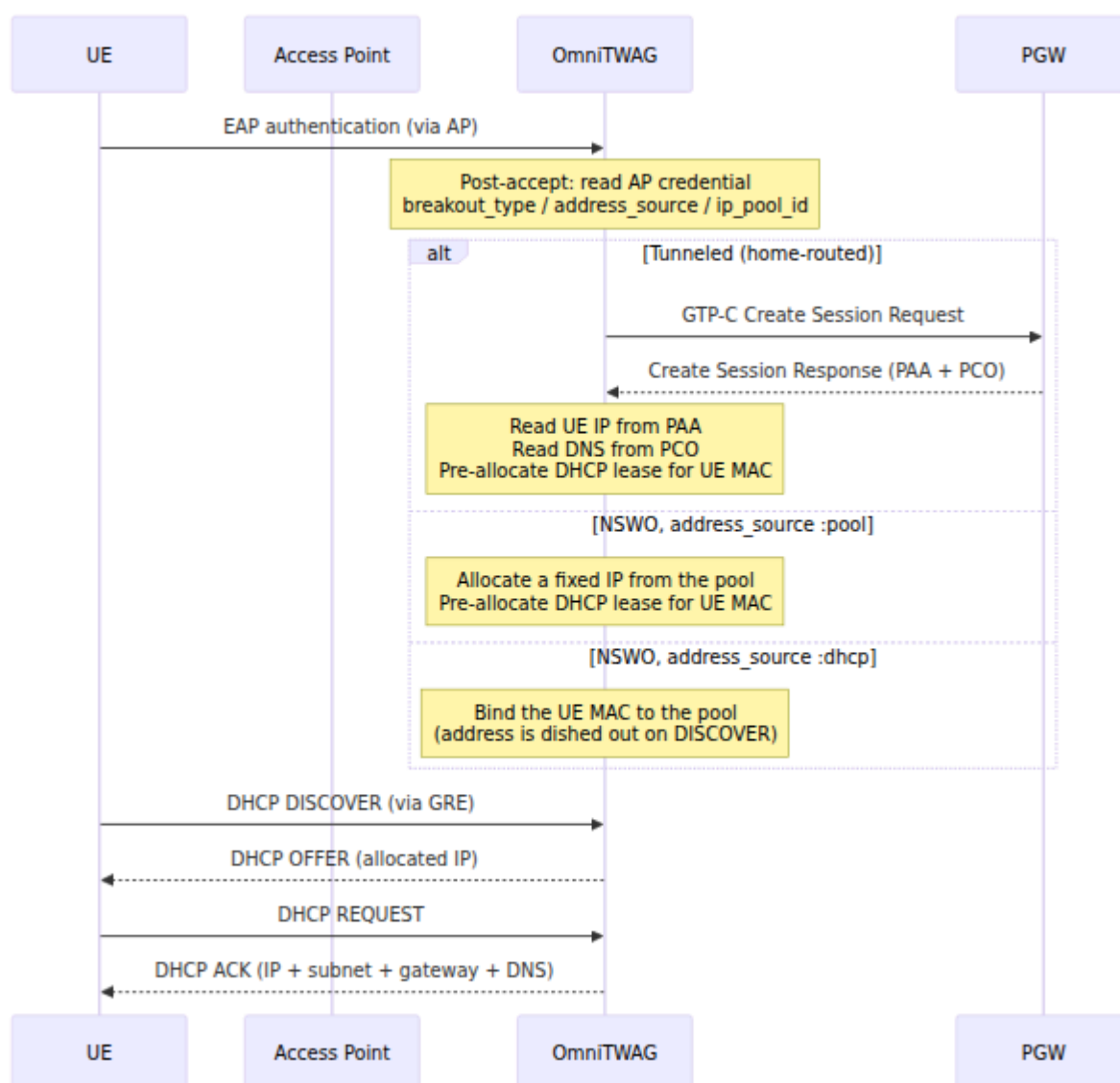
割当機能は GRE 経由で IP 割当機能 PAA から PCO から GTP-C 経由で S2a GTP-C 経由で UE へ提供

機能

- ・ IP 割当機能
- ・ DNS
- ・ PGW から
- ・ NSWO から DHCP
- ・ DHCPv4
- ・ DHCPv6
- ・ IPv6 SLAAC
- ・ DNS から PCO
- ・ IP 割当
- ・ IP
- ・ IP

- | | | | | | | | |
|--|--|--|--|--|--|--|--|
| | | | | | | | |
|--|--|--|--|--|--|--|--|

UE Access-Accept UE AP



AP

- breakout_type: :tunneled :nsw :local
- address_source: :pool :dhcp breakout_type :nsw

		UE	UE
	breakout_type: :tunneled	PGW GTP-C PAA	DHCPv4 DHCPv6 IPv6 RA
NSWO	breakout_type: :nsw, address_source: :pool	OmniTWAG ip_pool_id	DHCPv4
NSWO	breakout_type: :nsw, address_source: :dhcp	OmniTWAG ip_pool_id DHCP DISCOVER	DISCOVER DHCPv4
AP-	breakout_type: :local	AP	OmniTWAG

RADIUS RADIUS IP

OmniTWAG GTP-C PGW S2a PGW

- PDN (PAA) UE IP
- (PCO) DNS P-CSCF MTU

OmniTWAG は PAA と PCO 間の OmniTWAG と UE MAC 間の DHCP 間の UE と DHCP 間の OmniTWAG 間の UE と PGW 間の S2a GTP-C

PAA と OmniTWAG

PAA	UE
ipv4	PAA IPv4
ipv6	PAA IPv6
ipv4v6	PAA IPv4

PAA OmniTWAG UE

NSWO DHCP

NSWO PGW OmniTWAG ip_pool_id IP IPv4 CIDR DNS IP

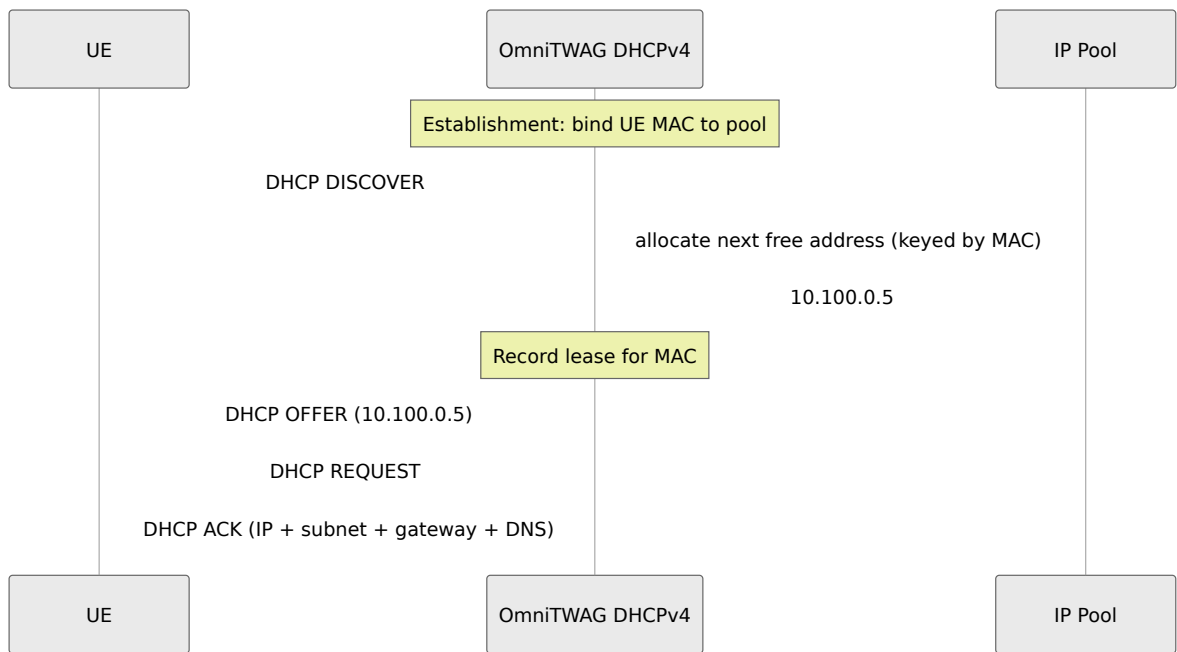
address_source UE

address_source: :pool ()

OmniTWAG UE MAC DHCPv4 DNS CIDR GRE UE UE DISCOVER

address_source: :dhcp (DISCOVER)

OmniTWAG UE MAC MAC DHCP DISCOVER DHCPv4 DISCOVER



OmniTWAG 的 DHCP 消息中，AP 的 `ip_pool_id` 是 OmniTWAG 的 UE 的 IP 池。

DHCPv4

DHCPv4 使用 UDP 的 67 端口，UE 的 MAC 地址。

- PGW 的 OmniTWAG 的 NSWO `address_source: :pool`
- DISCOVER** 消息中，NSWO `address_source: :dhcp`，UE 的 MAC 地址，MAC 地址，DISCOVER 消息。

UE 的

MAC 地址，DISCOVER 消息。

Parse error on line 2: ...te (PGW PAA or pool :pool) [*] --> B -----^
 Expecting 'SPACE', 'NL', 'HIDE_EMPTY', 'scale', 'COMPOSIT_STATE',
 'STRUCT_STOP', 'STATE_DESCR', 'ID', 'FORK', 'JOIN', 'CHOICE', 'CONCURRENT',
 'note', 'acc_title', 'acc_descr', 'acc_descr_multiline_value', 'CLICK', 'classDef',

'style', 'class', 'direction_tb', 'direction_bt', 'direction_rl', 'direction_lr', 'EDGE_STATE', got 'DESCR'



OmniTWAG 网络 DHCP 网络

消息	次数	OmniTWAG 消息
DISCOVER	1	手机 MAC 地址向OmniTWAG 发送 OFFER消息，手机 MAC 地址向OmniTWAG 发送消息，OmniTWAG 发送 OFFER消息，OmniTWAG 发送消息
REQUEST	3	手机发送 IP (地址 50) 消息，OmniTWAG 发送 ACK消息，IP 地址向OmniTWAG 发送 NAK消息，OmniTWAG 发送 NAK消息
RELEASE	7	OmniTWAG 发送 released消息，OmniTWAG 发送 PDN 消息，TS 23.402 §16.1.5.2 NOTE 2
DECLINE	4	OmniTWAG 发送OmniTWAG 消息
INFORM	8	OmniTWAG 发送 ACK消息，ACK 消息，DNS消息 (yiaddr 地址 0.0.0.0)

手机 INFORM消息OmniTWAG 发送 default_gateway 地址 default_subnet_mask地址
OmniTWAG 发送 INFORM 消息 8.8.8.8 地址 DNS 地址

网络

手机 UE IP地址DNS 地址 PGW 地址 UE IP 地址 DNS 地址
手机 DHCP 地址 NSWO 地址 IP 地址 IP地址 CIDR 地址
手机DNS 地址 IP 地址 地址

allocated_at expires_at expires_at allocated_at
OmniTWAG 60 expires_at
expires_at

DHCP

OmniTWAG DHCP OFFER ACK

	53	OFFER ACK
	54	DHCP server_ip
	1	
	3	
	6	DNS
IP	51	
	28	IP UE 55

GRE DHCP

GRE GRE UE DHCP GRE DHCP
DHCPv4 OmniTWAG AP OmniTWAG DHCP
UDP/IP GRE AP 10.0.0.1:67 68

UDP DHCP GRE UDP 68
OmniTWAG OmniTWAG RFC 2131
ciaddr ciaddr

DHCPv6

DHCPv6 使用 IPv6 使用 UDP 547 和 546 端口进行通信

- 客户端 消息 (消息 11) 和服务器 消息 (消息 12) 包含 DNS 信息，参考 RFC 3736 和 RFC 3646 中 UE 和 RA 消息格式 (O) 部分
- 客户端 消息 (消息 1) 和服务器 消息 (消息 2) 包含消息 (消息 3) 和消息 (消息 7) 中的 IA_NA 消息，参考 RFC 3315

客户端使用 DUID 标识设备，服务器使用 GTP 标识设备，DUID 格式

DHCPv6 消息

消息	消息	OmniTWAG 消息
消息 消息	11	OmniTWAG 客户端消息 DUID 和 ID 和 DNS 消息
消息	1	OmniTWAG 服务器消息   DUID 和 IA_NA 消息，包含 NoAddrsAvail 消息 2 次
消息	3	OmniTWAG 客户端消息 DUID 和 IA_NA 消息
消息 消息	-	OmniTWAG 服务器消息

客户端消息 ID 参考 RFC 3315 §15 部分，DHCPv6 消息 T1 和 T2 消息

消息 DUID

OmniTWAG 使用 DUID-EN 格式，DUID 参考 3GPP 消息 10415 部分

DHCPv6

	1	ID
	2	DUID
(IA_NA)	3	IPv6 NoAddrsAvail
DNS	23	DNS
	24	dhcpcv6_domain_list

IPv6 (SLAAC)

IPv6 RA (SLAAC) UE PGW IPv6 RFC 4861 ICMPv6

GTP

- 30
- RA

(ICMPv6 133)

項目	値
ホップ数	hop_limit 64
ルータの生存時間	router_lifetime 1800
M O	0 RA
MAC 1	MAC 6
MTU 5	MTU
3	L A
DNS RDNSS 25	DNS RFC 8106 DNS

86400 14400 A UE IPv6 RDNSS

DNS PCO

OmniTWAG DNS P-CSCF PGW PCO

PGW PCO

PCO 3GPP TS 24.008

項目	PCO 項目 ID	説明
dns_ipv4	0x000D	IPv4 DNS 設定項目
dns_ipv6	0x0003	IPv6 DNS 設定項目
pcscf_ipv4	0x000C	IPv4 P-CSCF 設定項目
pcscf_ipv6	0x0001	IPv6 P-CSCF 設定項目
mtu	0x0010	IPv4 最大 MTU
ip_alloc_nas	0x000A	PGW から NAS へ IP 割り当て true

raw 項目は PCO 項目の raw 属性を指定する。

OmniTWAG 項目は PCO 項目の OmniTWAG 項目 PGW 項目の IPv4 DNS 項目 MTU

DNS 項目 UE

DNS 項目 UE 項目 UE

項目	DNS 項目	説明
DHCPv4 (項目 6)	項目 PCO 項目 IPv4 DNS 項目	項目 PCO 項目 IPv4 DNS 項目 OmniTWAG 項目 8.8.8.8
DHCPv6 (項目 23)	DHCPv6 DNS 項目	項目 PGW PCO 項目 dhcpv6_dns_servers 項目 Google 項目 DNS IPv6
IPv6 RA (RDNSS項目 25)	項目 IPv6 DNS 項目	

DHCPv4 項目 PCO dns_ipv4 項目 IPv4 DNS 項目 OmniTWAG 項目 8.8.8.8

00 **P-CSCF** 0000 PCO 0000 PGW 00 P-CSCF 000000 DHCP/DHCPv6 0 RA 000
 0000 P-CSCF DHCP 0000 RA 000P-CSCF 0000000000000000000000000000000000
 UE0

--	--	--	--	--	--

□ □ □ □ □ □ □ □ □ □

- **NSWO** PGW OmniTWAG IP /30 IPv4 CIDR pool_exhausted address_source: :pool UE address_source: :dhcp DISCOVER OmniTWAG DHCP IP

Twag.Cidr 0000000000000000 IPv4 0 IPv6 CIDR 0000000000000000RADIUS 00
0000000000 IP 000IP 0000000000000000000000000000

11

```
IP 192.168.1.1 :omnitwg 192.168.1.1 dhcp 192.168.1.1 ipv6_ra DHCPv6 192.168.1.1
```

```
config :omnitwig,  
  # 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"  
  },  
  
  # IPv6 ☐☐☐☐  
  ipv6_ra: %{  
    enabled: true,  
    interface: "wlan0",  
    ra_interval: 30,  
    hop_limit: 64,  
    router_lifetime: 1800  
  },  
  
  # DHCPv6☐☐☐☐☐  
  dhcpv6_dns_servers: ["2001:4860:4860::8888",  
"2001:4860:4860::8844"],  
  dhcpv6_domain_list: ["example.com"]
```

DHCPv4 参数 (dhcp)

参数	数据类型	默认值	说明
enabled	boolean	true	是否启用 DHCPv4 服务 false 则不启用
interface	string	wlan0	DHCP 服务的接口名称 gre 则使用 gre 接口
server_ip	string	10.0.0.1	DHCP 服务器的 IP 地址 OmniWAG 的 DHCP 服务器 IP 地址为 54 的 GRE 接口
default_lease_time	integer	3600	默认租约时间 OmniWAG 的默认租约时间为 3600 秒
default_subnet_mask	string	255.255.255.0	默认子网掩码 RFC 2131 标准
default_gateway	string	10.0.0.1	默认网关 DHCP 服务的默认网关

IPv6 RA 設定 (ipv6_ra)

項目	型別	単位	初期値	説明
enabled	boolean		true	IPv6 RA を有効にするかどうか。false の場合は無効。
interface	string		wlan0	RA を送信するインターフェース。
ra_interval	integer	秒	30	RA を送信する間隔。
hop_limit	integer		64	RA の Hop Limit の値。
router_lifetime	integer	秒	1800	RA の有効期間。

DHCPv6 設定

項目	型別	単位	説明	備考
dhcpcv6_dns_servers	list of string		Google DNS IPv6 (2001:4860:4860::8888, 2001:4860:4860::8844)	DHCPv6 による IPv6 DNS 情報の取得。PGW PCO による提供も可能。
dhcpcv6_domain_list	list of string		ドメインリスト	DHCPv6 による DNS 情報の取得。24時間以内の OmniTWAG による提供も可能。

Web ☐ ☐

REST API

メソッド	パス	説明
GET	/dhcp (GET)	DHCPv4 情報を取得
GET	/dhcp/{id} (GET)	指定した MAC アドレスの DHCPv4 情報を取得

□ □ □ □ □ □ □ □ □ □ □ □

필드	설명
mac	UE MAC 주소
ip	할당된 IPv4 주소
subnet_mask	서브넷 마스크
gateway	기본 게이트웨이
dns_servers	DNS 서버 주소
lease_time	라이스 기간 (초)
allocated_at	할당된 Unix 타임스탬프
expires_at	만료 Unix 타임스탬프
status	active (활성) 또는 expired (만료)
time_remaining	남은 시간 (0이면 만료)

예시

IP 할당 이벤트를 Prometheus에 수집하는 방법 [여기](#)

関数	プロトコル	説明
<code>[:dhcp, :lease, :allocate]</code>	DHCPv4	クライアントの MAC アドレスを登録
<code>[:dhcp, :lease, :deallocate]</code>	DHCPv4	クライアントの MAC アドレスを削除
<code>[:dhcp, :lease, :reap]</code>	DHCPv4	リース期限切れのクライアントを削除
<code>[:dhcp, :discover]</code>	DHCPv4	クライアントが DISCOVER を送信し、サーバーが OFFER を返す
<code>[:dhcp, :discover, :pool_allocated]</code>	DHCPv4	クライアントの MAC アドレスが DISCOVER を送信し、サーバーが OFFER を返す。NSWO :dhcp を返す
<code>[:dhcp, :discover, :no_allocation]</code>	DHCPv4	クライアントの MAC アドレスが DISCOVER を送信し、サーバーが OFFER を返す。DISCOVER を返す
<code>[:dhcp, :ack]</code>	DHCPv4	クライアントが REQUEST を送信し、サーバーが ACK を返す
<code>[:dhcp, :nak]</code>	DHCPv4	クライアントが REQUEST を送信し、サーバーが NAK を返す
<code>[:dhcp, :release]</code>	DHCPv4	クライアントが RELEASE を送信し、サーバーが ACK を返す
<code>[:dhcp, :decline]</code>	DHCPv4	クライアントが DECLINE を送信し、サーバーが ACK を返す
<code>[:dhcp, :inform]</code>	DHCPv4	クライアントが INFORM を送信し、サーバーが ACK を返す
<code>[:dhcp, :packet, :malformed]</code>	DHCPv4	クライアントが不正な DHCP パケットを送信
<code>[:dhcpv6, :info_request, :received]</code>	DHCPv6	クライアントが INFO_REQUEST を送信し、サーバーが ACK を返す
<code>[:dhcpv6, :solicit, :received]</code>	DHCPv6	クライアントが SOLICIT を送信し、サーバーが ACK を返す

操作	プロトコル	説明
<code>[:dhcpv6, :request, :received]</code>	DHCPv6	クライアントからのリクエストを受信
<code>[:dhcpv6, :info_reply, :sent]</code>	DHCPv6	クライアントへの情報応答を送信
<code>[:dhcpv6, :advertise, :sent]</code>	DHCPv6	クライアントへのアドバタイズメントを送信
<code>[:dhcpv6, :reply, :sent]</code>	DHCPv6	クライアントへの応答を送信
<code>[:ipv6_ra, :prefix, :add]</code>	IPv6 RA	IPv6 前缀をルーターに追加
<code>[:ipv6_ra, :prefix, :remove]</code>	IPv6 RA	IPv6 前缀をルーターから削除
<code>[:ipv6_ra, :advertisement, :sent]</code>	IPv6 RA	IPv6 ルーターアドバタイズメントを送信

操作

UE 側の IP 操作

UE 側で IPv4 側の DHCP 操作を行う UE MAC 側で `[ :dhcp, :discover, :no_allocation]` 操作

操作

- GTP 側の PAA 側の OmniTWAG 側の操作
- GTP-C 側の操作

操作 **NSWO**

- AP 側の `ip_pool_id` 側の OmniTWAG 側の操作

- 0000000000000000
- 00 address_source: :dhcp 00DISCOVER 00 MAC 00000000000000000000000000000000 MAC

□□□□□

- DHCPv4 □□□□□□□□

5/5

1. 〇〇 OmniTWAG 〇〇〇〇〇〇〇〇〇〇〇〇 NSW〇〇 UE IP〇
2. 〇〇〇〇〇〇〇〇〇 PGW 〇 APN 〇 PAA〇
3. 〇〇 NSW〇 〇〇〇〇 AP 〇〇〇〇〇〇〇〇〇〇 `ip_pool_id`〇〇〇〇〇〇〇〇〇〇〇〇〇〇〇 IP 〇 RADIUS 〇〇〇
4. 〇〇 DHCP 〇〇〇〇 DHCPv4 〇〇〇〇〇〇〇〇〇

UE □□□□□□ DNS

UE IPv4 addresses

□ □ □ □ □ □

- PGW PCO IPv4 DNS OmniTWAG 8.8.8.8
- UE PGW DNS

□□□□□

1. DHCP 서버에서 REST API 통해 DNS 서버
2. PGW PCO 통해 DNS 서버
3. DNS 서버에서 UE에 전달

UE **NAK**

```
0000 UE 00 REQUEST 0000 NAK0000 [:dhcp, :nak] 0000
```

□ □ □ □ □ □

- UE 接收 IP 地址 50 个字节
- 接收数据

□□□□

1. □□ UE MAC □□□□
2. □□ UE □□□ IP □ PGW □□□□□ IP □□□
3. □□□□□□□□□ UE □□□□□□□□□ DHCP □□□

□□□□

- □□
- IP □
- RADIUS □□
- □□
- S2a GTP-C
- □□□□
- □□□□ (Gy)

□□

IP

概要

OmniTWAG は **IP** を NSWO WLAN として IPv4 の NSWO TWAG として OmniTWAG の IP の IPv4 CIDR の DNS の REST API の Web の

OmniTWAG の DETS の OmniTWAG の

機能

-
-
-
-
- REST API
- Web
-

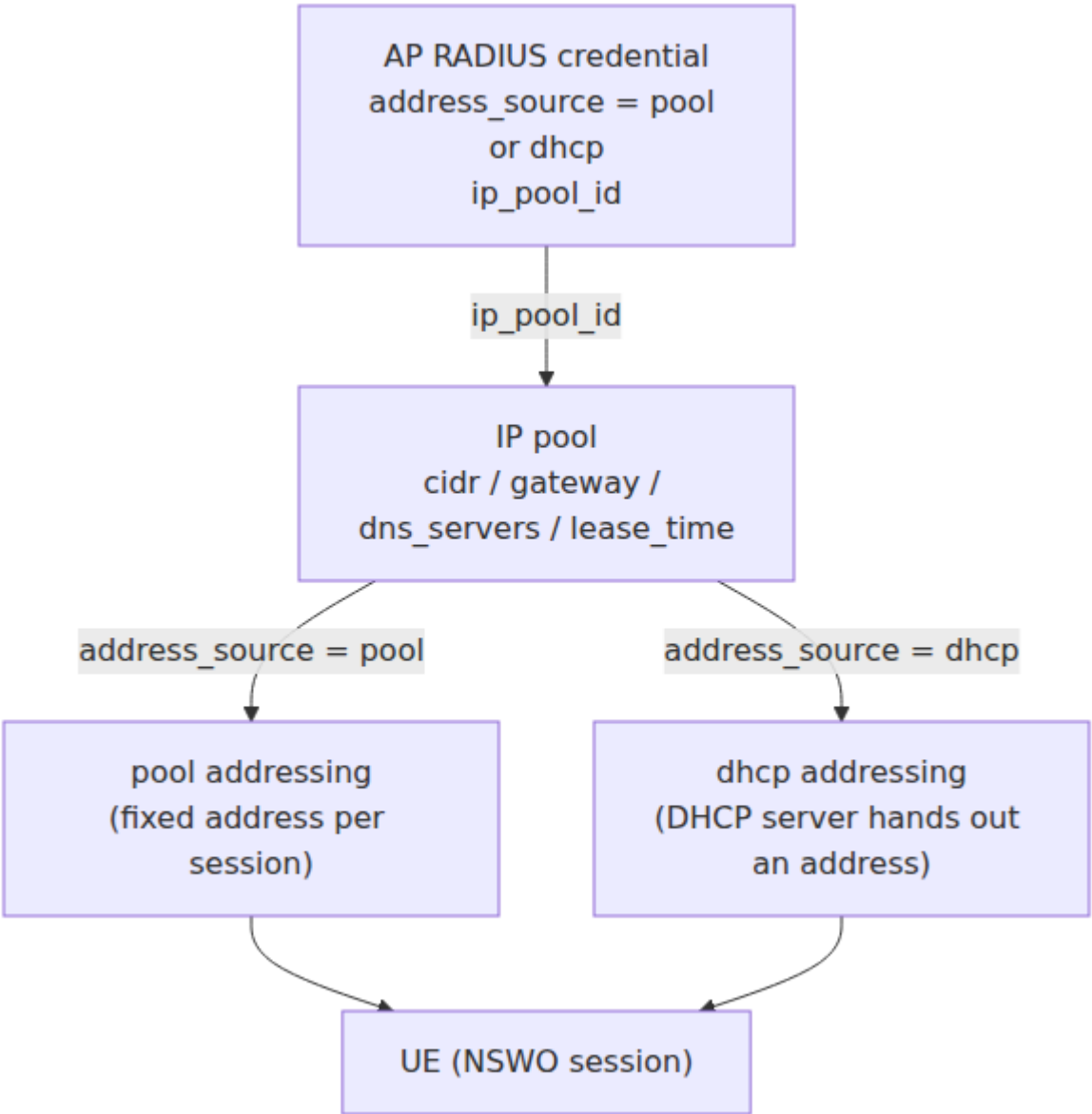
インストール

IPv4 CIDR の DNS の

IP の AP の RADIUS の `ip_pool_id` の `id` の RADIUS の NSWO の

`address_source` の OmniTWAG の UE の

項目	説明
pool	OmniTWAG 設定ファイルに定義された IP アドレスプール
dhcp	OmniTWAG DHCP サーバが UE に DHCP DISCOVER を応答する

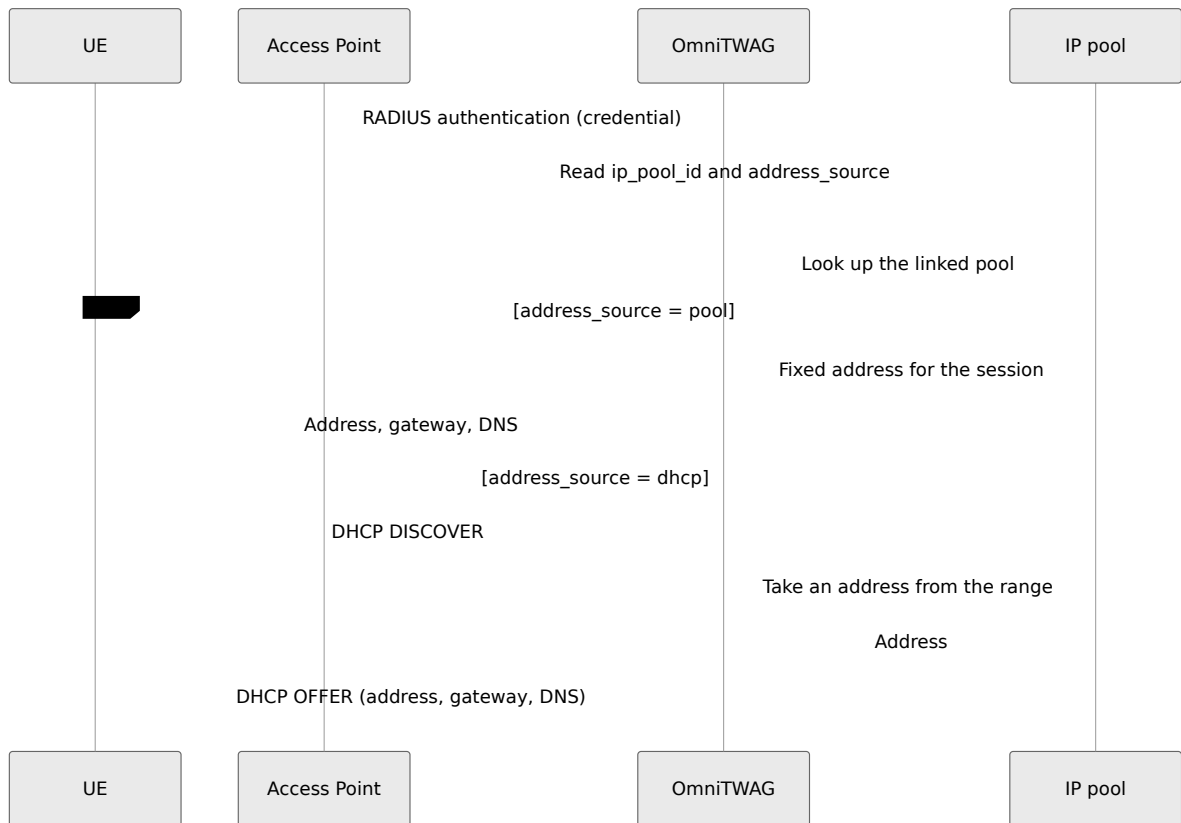


設定手順

OmniTWAG 設定 AP 設定 NSWO 設定

1. RADIUS 設定

2. OmniTWAG 取得 `ip_pool_id` 情報
3. OmniTWAG 取得 `address_source` 情報
4. 取得した `address_source` が `pool` の場合 OmniTWAG 関連の IP 情報を取得
5. 取得した `address_source` が `dhcp` の場合 OmniTWAG が UE に DHCP DISCOVER による DHCP 情報を取得
6. OmniTWAG が IP 情報を取得したら DNS 情報も UE に提供



IP 管理

OmniTWAG は CIDR 形式で IP 範囲を管理します。

1 CIDR 範囲は `/30` の範囲で、2 の範囲は `/30` の範囲で 2 の範囲で管理します。
OmniTWAG は IP 範囲を管理します。

IP 範囲が OmniTWAG の `pool_exhausted` 状態になると、IP 範囲が枯渇します。

IP 範囲が `allocated` 状態になると、OmniTWAG が IP 範囲を管理します。

id		32bit UUID
name		Melbourne guest WiFi
cidr		IPv4 CIDR 10.60.0.0/24 /30
gateway		UE IPv4 CIDR
dns_servers		UE IPv4 DNS
lease_time		3600
default		true DHCP
allocated		
inserted_at		Unix
updated_at		Unix

name cidr gateway dns_servers lease_time id allocated inserted_at updated_at

REST API

OmniTWAG REST API 透過 API 端點 8444 透過 HTTPS 與機器通訊
/ip_pools

動作	URL	動作名稱	說明
GET	/ip_pools	index	取得所有 IP Pool 清單 (name, cidr)
GET	/ip_pools/{id}	show	取得 ID 的 IP Pool
POST	/ip_pools	create	新增 IP Pool
PUT	/ip_pools/{id}	update	更新 IP Pool
DELETE	/ip_pools/{id}	delete	刪除 IP Pool

參數

POST 或 PUT 透過 JSON 傳送參數

參數名稱	類型	是否必填	說明
name	字串	否	IP Pool 名稱
cidr	字串	否	IPv4 CIDR 格式 (例如 /30) 指定
gateway	字串	否	IPv4 網關 IP 地址 (CIDR 格式)
dns_servers	陣列	否	IPv4 DNS 伺服器 IP 地址
lease_time	數字	否	租約時間 (秒) (預設 3600)
default	字串	否	預設 DHCP 租約時間 (秒)

PUT 创建IP池

返回

状态码	描述
200 OK	创建成功
201 Created	创建成功
404 Not Found	不存在id
422 Unprocessable Entity	cidr 不是IPv4 CIDR 格式

返回

返回

GET /ip_pools

返回

```
{
  "count": 1,
  "ip_pools": [
    {
      "id": "0d1e2f3a-4b5c-6d7e-8f90-1a2b3c4d5e6f",
      "name": "Melbourne guest WiFi",
      "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
    }
  ]
}
```

□□□□□□□□

□□□

POST /ip_pools

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

□□□□□□ 201 Created □□□□□□□□ id □□□□□ gateway □□□□ CIDR □□□□□□□□□□□□□□
□ 10.60.0.1 □□

□□□ **CIDR** □□

□□□

```
{
  "name": "Broken pool",
  "cidr": "10.60.0.0/31"
}
```

422 Unprocessable Entity

```
{
  "error": "cidr too small: 10.60.0.0/31 has no usable hosts"
}
```

DHCP DHCP
DHCP API NSW0

default OmniTWAG

IP PGW (tunneled) NSW0 pool default pool

Web

IP CIDR

DHCP IP

NSWO OmniTWAG pool_exhausted

1. address_source UE
2. allocated
3. ip_pool_id

TWAG 開箱

簡介

TWAG 預設 **9568** 埠 `prometheus.port` 與 Prometheus 預設 9090 埠不同，5 分鐘快速上手

功能

- 支援
- RADIUS 認證
- RADIUS 計費
- RADIUS 認證
- EAP 認證
- 支援 AP 認證
- 支援
- TWAG 認證
- Gy 認證
- STa Diameter 認證
- Diameter 認證
- NSWO 認證
- DHCP 認證
- DHCPv6 與 IPv6 RA 認證
- WLCP 認證
- Erlang VM 認證
- 支援 Grafana 認證
- 支援
- 支援

简介

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

Prometheus 通过以下配置可以获取到以下指标
[:radius, :access, :request, :count] 指标 radius_access_request_count

RADIUS 简介

Radius 是 RFC 2865 定义的 RADIUS Access-Request/Response 协议

指标	单位	类型	描述
radius_access_request_count	每秒	计数器	每秒 RADIUS Access-Request 请求数
radius_access_accept_count	每秒	计数器	每秒 RADIUS Access-Accept 响应数
radius_access_reject_count	每秒	计数器	每秒 RADIUS Access-Reject 响应数
radius_access_challenge_count	每秒	计数器	每秒 RADIUS Access-Challenge 响应数 EAP 挑战

更多

```
# 5m 5m
rate(radius_access_request_count[5m])

# EAP 5m
rate(radius_access_challenge_count[5m])

# 5m
sum(rate(radius_access_accept_count[5m])) /
  (sum(rate(radius_access_accept_count[5m])) +
   sum(rate(radius_access_reject_count[5m])))
```

RADIUS 5m 5m

5m 5m RADIUS 5m 5m RADIUS 5m 5m

5m	5m	5m	
radius_auth_unauthorized_source_count	5m	source	5m 5m AP 5m 5m
radius_auth_bad_secret_count	5m	source, credential_id	5m 5m 5m 5m
radius_access_rate_limited_count	5m	nas_ip	5m NAS 5m rate_lim: 5m 5m Acc

5m 5m

```
# 增加 AP 未授权源计数
increase(radius_auth_unauthorized_source_count[1h])

# 增加 AP 坏秘密计数
sum by (source) (increase(radius_auth_bad_secret_count[1h]))

# 增加 NAS 速率受限计数
sum by (nas_ip) (increase(radius_access_rate_limited_count[15m]))
```

RADIUS 认证

认证 RFC 2866 的 RADIUS 认证

radius_accounting_request_count

认证 认证

认证	认证
status_type	认证认证认证认证

认证认证 start, stop, interim_update, accounting_on, accounting_off

认证认证

```
# 认证认证
rate(radius_accounting_request_count{status_type="start"}[5m]) * 60

# 认证认证
rate(radius_accounting_request_count{status_type="stop"}[5m]) * 60
```

EAP 认证

认证认证 RFC 4187 的 EAP-AKA 认证 EAP-SIM 认证 EAP 认证 ERP 认证 EAP 认证

項目	単位	説明
eap_aka_identity_count	回 分	EAP-AKA 識別子要求 1回
eap_aka_challenge_count	回 分	EAP-AKA チャレンジ要求 2回
eap_aka_sync_failure_count	回 分	EAP-AKA 同期失敗 SQL 実行回数
eap_aka_auth_success_count	回 分	成功した EAP-AKA 認証
eap_aka_auth_reject_count	回 分	失敗した EAP-AKA 認証 (UE からの EAP / AUTN 要求)
eap_aka_client_error_count	回 分	UE からの EAP-AKA 要求エラー
eap_auth_rejected_count	回 分	HSS USER_UNKNOWN による EAP 認証拒否
eap_process_crash_count	回 分	EAP 処理中に発生した UE への EAP 要求エラー
eap_sim_start_count	回 分	EAP-SIM 開始
eap_sim_auth_success_count	回 分	成功した EAP-SIM 認証
eap_sim_auth_reject_count	回 分	失敗した EAP-SIM 認証
eap_erp_initiate_count	回 分	EAP ERP 開始

項目	単位	説明
<code>eap_erp_success_count</code>	回 分	成功した EAP ERP の回数
<code>eap_erp_mac_failure_count</code>	回 分	EAP ERP MAC の失敗回数
<code>eap_erp_fallback_count</code>	回 分	EAP ERP 失敗理由 <code>reason</code>

例

```
# 成功した EAP ERP の回数が増加したかどうかを確認
increase(eap_erp_success_count[1h])

# EAP ERP AKA の失敗率を計算
rate(eap_erp_auth_success_count[5m]) * 60

# EAP 失敗理由を確認
increase(eap_process_crash_count[1h])
```

AP 監視

5 分間隔で監視

項目	単位	項目	項目
radius_active_clients_count	個 個	個	クライアントの総数 Twag.Session.Client 個 個
radius_access_points_count	個 個	個	アクセスポイントの総数 APState 個
radius_clients_per_ap_count	個 個	nas_ip, ssid	クライアントの総数 個
radius_active_sessions_count	個 個	status	セッションの総数 start, stop, interim_update 個
radius_active_sessions_total	個 個	個	セッションの総数 start 個 interim_update 個

クライアント

```
# 5 個の AP  
topk(5, radius_clients_per_ap_count)  
  
# SSID 別  
sum by (ssid) (radius_clients_per_ap_count)  
  
# クラウド監視  
delta(radius_active_sessions_total[5m]) < -10
```

クライアント

クライアント RADIUS クラウド監視

項目	単位	種類	説明
radius_data_bytes_in	バイト	整数	Acct-Input-Octets
radius_data_bytes_out	バイト	整数	Acct-Output-Octets
radius_session_total_time	秒	整数	Acct-Session-Time

計算式

```
# トラフィック量(MB)
(radius_data_bytes_in + radius_data_bytes_out) / 1024 / 1024

# セッション時間
radius_session_total_time / radius_active_sessions_total
```

TWAG 設定

Twag.Session.Client 設定

項目	単位	説明	備考
twag_session_tearardown_count	回 分	reason	セッション終了理由 accounting_stop admin_disconnect network_disconnect identity_timeout liveness_timeout
twag_bearer_modify_command_count	回 分	回	PGW Modify Bearer-Command
twag_bearer_update_count	回 分	回	PGW Update Bearer
twag_detach_cleanup_start_count	回 分	回	
twag_detach_cleanup_complete_count	回 分	回	
twag_detach_pgw_delete_count	回 分	回	PGW Delete Session
twag_detach_liveness_timeout_count	回 分	回	
twag_liveness_arp_probe_success_count	回 分	回	ARP 成功

指标	单位	维度	说明
twag_liveness_arp_probe_failure_count	个		失败 ARP 探测
twag_handover_in_count	个	source	Handover-in 数量
twag_handover_out_count	个	cause	Handover-out 数量
twag_handover_ap_change_count	个		切换 AP 数量

配置

```
# 配置 Prometheus 抓取
sum by (reason) (increase(twag_session_tearardown_count[1h]))

# 配置 Prometheus 抓取
increase(twag_session_tearardown_count{reason="identity_timeout"}
[1h])

# 配置 UE 抓取
increase(twag_detach_liveness_timeout_count[1h])
```

twag_gy_quota_blocked twag_gy_quota_unblocked Twag.Session.Client Prometheus /metrics

Gy

Diameter Ro/Gy NSWO

twag_gy_ccr_sent_count	type	type	OCS type initial, update terminate
twag_gy_ccr_error_count	type	type	CCR
twag_gy_cca_received_count	type, result		OCS

```
# CCR
sum by (type) (rate(twag_gy_ccr_sent_count[5m]))

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

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

STa Diameter

TWAG STa

項目	単位	項目	項目
sta_asr_received_count	個 個 個	個	Abort-Session-Requests
sta_rar_received_count	個 個 個	個	Re-Auth-Requests
sta_str_received_count	個 個 個	個	Session-Termination-Requests
sta_session_disconnect_count	個 個 個	reason	STa 理由
sta_session_reauth_count	個 個 個	re_auth_type	STa 再認証タイプ

Diameter

diameter_message_count

項目 単位

필드	설명
application	Diameter 프로토콜의 swx, sta 필드
command	Diameter 프로토콜의 MAR, MAA 필드
direction	inbound 또는 outbound

예시

```
# Diameter 메시지 수 집계
sum by (application) (rate(diameter_message_count[5m]))
```

NSWO 필드

필드	유형	설명
nswo_allocate_count	uint64	NSWO IP 할당 횟수
nswo_release_count	uint64	NSWO IP 해제 횟수

DHCP 필드

DHCPv4의 discover.pool_allocated 필드는 NSWO의 DHCPv4 pool에 할당된 IP 주소의 수를 나타냅니다.

項目	単位	説明
dhcp_lease_allocate_count	回数 /	リース割当回数
dhcp_lease_deallocate_count	回数 /	リース解放回数
dhcp_discover_count	回数 /	クライアントからの DISCOVER 受信回数
dhcp_discover_no_allocation_count	回数 /	クライアントからの DISCOVER 受信回数 （リース割当失敗時）
dhcp_discover_pool_allocated_count	回数 /	クライアントからの DISCOVER 受信回数 （NSWO 割当成功時）
dhcp_ack_count	回数 /	クライアントからの ACK 受信回数
dhcp_nak_count	回数 /	クライアントからの NAK 受信回数
dhcp_release_count	回数 /	クライアントからの RELEASE 受信回数
dhcp_decline_count	回数 /	クライアントからの DECLINE 受信回数
dhcp_inform_count	回数 /	クライアントからの INFORM 受信回数
dhcp_packet_malformed_count	回数 /	不正な DHCP パケット受信回数

合計

```
# NSWD0 00000000
rate(dhcp_discover_pool_allocated_count[5m])

# DISCOVER
increase(dhcp_discover_no_allocation_count[15m])
```

DHCP 00000000000000000000 dhcp_lease_reap 0000000000 Prometheus 000
00000000 /metrics 0000

DHCPv6 0 IPv6 RA 00

00	00	00
dhcpv6_info_request_received_count	000	0000 DHCPv6 0000000000
dhcpv6_solicit_received_count	000	0000 DHCPv6 00000000
dhcpv6_request_received_count	000	0000 DHCPv6 00000000
dhcpv6_info_reply_sent_count	000	000 DHCPv6 0000000000
dhcpv6_advertise_sent_count	000	000 DHCPv6 00000000
dhcpv6_reply_sent_count	000	000 DHCPv6 00000000
ipv6_ra_prefix_add_count	000	IPv6 RA 000000
ipv6_ra_prefix_remove_count	000	IPv6 RA 000000
ipv6_ra_advertisement_sent_count	000	000 IPv6 00000000

WLCP 00

00000000000000000000 WLAN 000000

項目	単位	型	説明
<code>wlcp_packet_received_count</code>	個	int	受信した WLCP パケットの数
<code>wlcp_packet_received_bytes</code>	バイト	int	受信した WLCP パケットの総バイト数
<code>wlcp_packet_malformed_count</code>	個	int	破損した WLCP パケットの数
<code>wlcp_packet_unregistered_count</code>	個	int	登録されていない WLCP パケットの数
<code>wlcp_pdn_request_count</code>	個	int	PDN 要求の数
<code>wlcp_pdn_accept_count</code>	個	int	PDN 受入れの数
<code>wlcp_pdn_reject_count</code>	個	int	PDN 拒否の数
<code>wlcp_pdn_disconnect_request_count</code>	個	int	PDN 切断要求の数
<code>wlcp_pdn_disconnect_accept_count</code>	個	int	PDN 切断受入れの数

Erlang VM について

BEAM 仮想マシンについて

内存

名称	描述
vm_memory_total	内存总量
vm_memory_processes	Erlang 进程内存
vm_memory_processes_used	进程内存使用量
vm_memory_system	系统内存
vm_memory_atom	原子内存
vm_memory_atom_used	原子内存使用量
vm_memory_binary	二进制内存
vm_memory_code	代码内存
vm_memory_ets	ETS 表内存 AP 内存使用量

系统

名称	描述
vm_system_info_process_count	进程 Erlang 数量
vm_system_info_port_count	端口数量
vm_system_info_atom_count	原子数量
vm_system_info_schedulers	调度器数量
vm_system_info_schedulers_online	在线调度器数量

配置

項目	単位
<code>vm_statistics_run_queue</code>	ランキュー
<code>vm_total_run_queue_lengths_total</code>	ランキューの長さ
<code>vm_total_run_queue_lengths_cpu</code>	CPU のランキューの長さ
<code>vm_total_run_queue_lengths_io</code>	IO のランキューの長さ

設定

```
# メモリMB
vm_memory_total / 1024 / 1024

# ランキューの長さ
avg_over_time(vm_statistics_run_queue[5m])
```

Grafana

設定

```
# 受け取り
sum(rate(radius_access_accept_count[5m]))

# 拒否
sum(rate(radius_access_reject_count[5m]))

# 割合
100 * sum(rate(radius_access_accept_count[5m])) /
  (sum(rate(radius_access_accept_count[5m])) +
   sum(rate(radius_access_reject_count[5m])))
```

■ AP ■■

```
# ■ AP ■■■■Access-Request ■■ nas_ip ■■■■■■■■■■■■  
sum by (nas_ip) (rate(radius_accounting_request_count[5m]))  
  
# ■ AP ■■■■  
radius_clients_per_ap_count
```

■■■■■■■■■■■■

```
# ■■■■■■■■■■■■  
sum(increase(radius_access_reject_count[1h]))  
sum(increase(radius_auth_unauthorized_source_count[1h]))  
sum(increase(radius_auth_bad_secret_count[1h]))  
  
# ■■■■  
sum by (reason) (increase(twag_session_tearardown_count[1h]))
```

■■■■

```
# ■■■■■MB  
(radius_data_bytes_in + radius_data_bytes_out) / 1024 / 1024
```

□□□□

□□□□

```
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: "□□□□□□ (> 10%)"

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

- alert: UnknownRadiusSources
  expr: increase(radius_auth_unauthorized_source_count[15m])
  > 0
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "□□□□□□□□□□ RADIUS □□□"

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

```

- alert: NoActiveClients
  expr: radius_active_clients_count == 0
  for: 15m
  labels:
    severity: warning
  annotations:
    summary: "15 无活跃客户端"

```

配置

```

groups:
- name: twag_resources
  rules:
    - alert: HighMemoryUsage
      expr: vm_memory_total > 1073741824
      for: 10m
      labels:
        severity: warning
      annotations:
        summary: "TWAG 内存 1GB"

    - alert: HighRunQueue
      expr: avg_over_time(vm_statistics_run_queue[5m]) > 10
      for: 5m
      labels:
        severity: warning
      annotations:
        summary: "TWAG CPU 高"

```

配置

- 配置
- RADIUS 配置
- RADIUS 配置
- 配置
- 配置

- **AP**

□□□□ (Gy)

11

OmniTWAG 無線ネットワーク (OCS) 無線ネットワーク OmniTWAG 無線 Diameter Ro (Gy) 無線
無線ネットワーク (CCR) 無線ネットワーク (CCA)無線ネットワーク TWAG 無線 WLAN 無線ネットワーク

TWAG 000000000000 **NSWO**000000 0 TWAG 000000000000TWAG 000000
 00000000 TWAG 0000000000 00000**GTP**000 00PGW 0000000000000000000000
 :disabled00000000000000000000 0000 000

□□□□□□ Gy □□□□□□ UE □□□□

- `online_charging.mode` OmniTWAG 無線充電モード
- **AP** 無線充電 AP モード (`charging_mode`) OmniTWAG 無線充電 UE 無線 Gy 無線
無線 **RADIUS** 無線

□□□□□□□□ CCR-Initial □□□□□□□□ □□ AP ? ? ? □□□□□□

11

- 二二
- 三三三三
- 二二 AP 三三三三
- 三三三三三
- 三三三三三三三三
- 三三三三
- 三三三三三三三
- 二二
- 三三三三三三三三
- 三三三三
- 三三三

概要

Gy 側から Diameter 側へ送信される OCS 側から OmniTWAG へ 3GPP Ro へ
Auth-Application-Id 4 RFC 4006 3GPP TS 32.299 側へ OmniTWAG へ
Service-Context-Id 32251@3gpp.org 側へ 3GPP PS 側へ

OmniTWAG 側から送信される

- **CCR-Initial (CCR-I)**側から OmniTWAG へ OCS 側へ送信される
- **CCR-Update (CCR-U)**側から OmniTWAG 側へ送信される
- **CCR-Termination (CCR-T)**側から OmniTWAG 側へ送信される

OCS 側から (CCA) 側へ CCA 側へ Result-Code 側へ (Granted-Service-Unit) 側へ Validity-Time

OmniTWAG 側から Gy 側へ Gy 側へ Diameter Session-Id 側へ CC-Request-Number 側へ OCS 側から OmniTWAG 側へ CCR-I 側へ送信される

設定

online_charging.mode 側へ 側へ OmniTWAG 側へ
側へ AP 側へ charging_mode 側へ AP 側へ UE 側へ OmniTWAG 側へ Gy 側へ

側	側 CCR-I	側 CCR-U	側 CCR-T
:disabled	側	側	側
:authorize_only	側	側	側
:online_charging	側	側	側

:disabled

OmniTWAG 側へ OCS 側へ GTP 側へ PGW 側へ

:authorize_only

OmniTWAG 在 CCR-I 上注册 CCR-T 的 OmniTWAG 在 CCR-U 上注册
在 CCR-U 上注册 CCR-T 的 OmniTWAG 在 CCR-I 上注册

:online_charging

OmniTWAG 在 CCR-I 上注册 CCR-U 的 OmniTWAG 在 CCR-T 上注册
在 CCR-U 上注册 CCR-T 的 OmniTWAG 在 CCR-I 上注册

AP 注册

AP 注册 charging_mode 注册 online_charging.mode 注册
OmniTWAG 注册 RADIUS 注册 RADIUS 注册

AP 注册 charging_mode	注册	Gy 注册
:charged	AP 注册 注册	OmniTWAG 注册 UE 注册 Gy 注册 (CCR-Initial)
:uncharged	AP 注册 注册	OmniTWAG 注册 Gy 注册 注册 CCR
:auth_only	注册 注册	OmniTWAG 注册 注册 Gy 注册

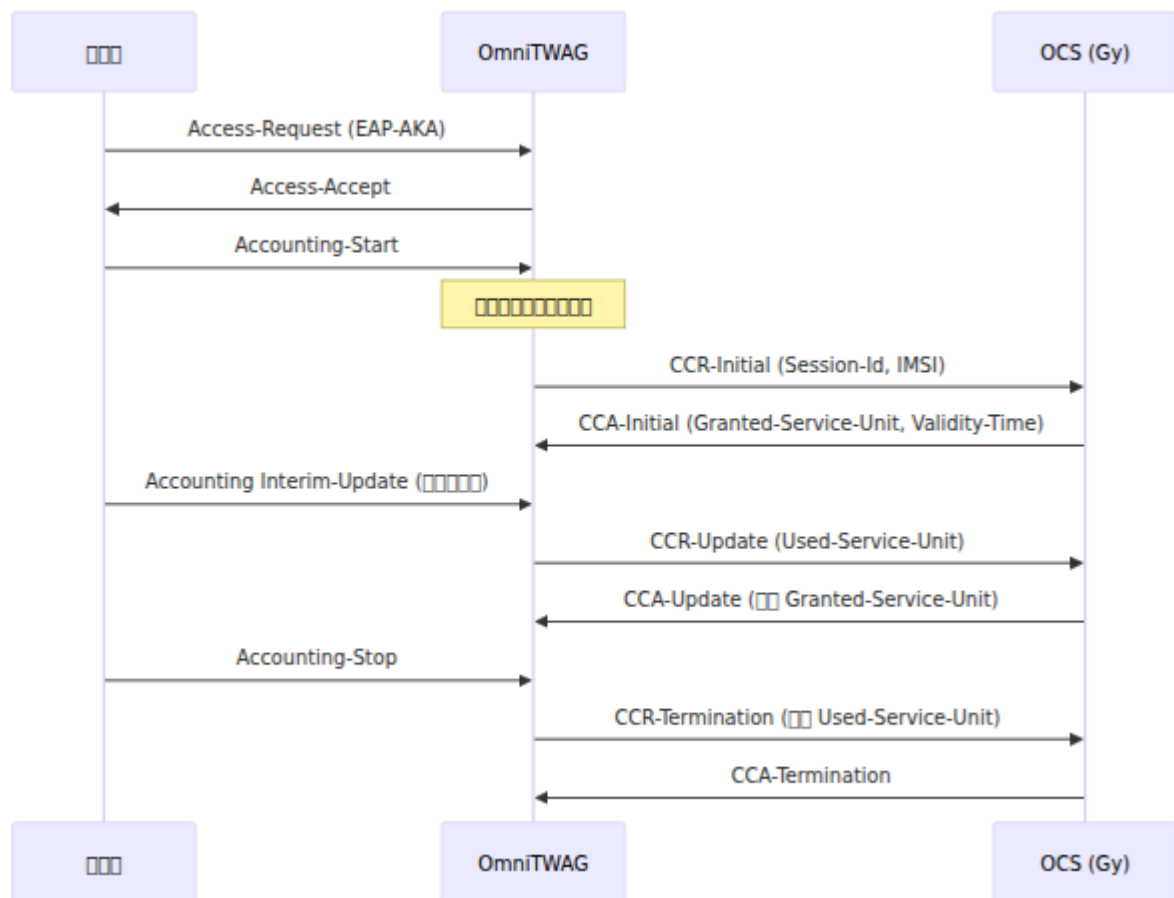
注册 注册 OmniTWAG 注册 CCR-Initial

- 注册 online_charging.mode 注册 :disabled 注册 Gy 注册
- 注册 IMSI
- AP 注册 charging_mode 注册 :charged

注册 OmniTWAG 注册 UE 注册 Gy 注册 CCR-U 注册 CCR-T 注册 AP 注册
注册 Gy 注册 :online_charging 注册 :uncharged 注册 AP 注册 Gy 注册

Sequence Diagram

Sequence Diagram illustrating the interaction between the User Equipment (UE), OmniTWAG, and OCS (Gy) for RADIUS Accounting and Charging Control (CCR).



CCR-Initial

OmniTWAG sends CCR-Initial to OCS (Gy) after receiving Access-Accept and Accounting-Start from the UE. The CCR-Initial message includes the Session-Id, IMSI, and the charging mode (e.g., `charging_mode : charged`).

The CCR-Initial message includes the `Subscription-ID` AVP and the `IMSI` AVP. The `END_USER_IMSI` AVP is set to 1. The `Multiple-Services-Indicator` AVP is set to 1. The `Requested-Service-Unit` AVP is set to the `Rating-Group` and `Multiple-Services-Credit-Control` (MSCC) AVP.

CCR-Update

The UE sends Accounting Interim-Update to OmniTWAG. OmniTWAG then sends CCR-Update to OCS (Gy). The CCR-Update message includes the `Used-Service-Unit` AVP.

- 0.8
- Validity-Time

CCR-U OmniTWAG Used-Service-Unit AVP CCR-U Requested-Service-Unit OCS

CCR-U

OmniTWAG CCR-U RADIUS

RADIUS	Gy
Acct-Input-Octets	CC-Input-Octets
Acct-Output-Octets	CC-Output-Octets
Acct-Session-Time	CC-Time

OmniTWAG CC-Total-Octets

CCR-Termination

OmniTWAG CCR-T RADIUS Accounting-Stop OmniTWAG Used-Service-Unit AVP Termination-Cause 1 DIAMETER_LOGOUT OmniTWAG :authorize_only :online_charging CCR-T

CCA MSCC Granted-Service-Unit OmniTWAG Gy

- CC-Total-Octets CC-Input-Octets CC-Output-Octets

- CC-Time
- Validity-Time

OmniTWAG CCA MSCC Final-Unit-Indication MSCC
 Result-Code Final-Unit-Action TERMINATE REDIRECT
 RESTRICT_ACCESS MSCC Result-Code 4010
 END_USER_SERVICE_DENIED 4012 CREDIT_LIMIT_REACHED
 0000 0000

OmniTWAG CCA-I CCA-U

0000

OmniTWAG Gy RADIUS
 OmniTWAG Used-Service-Unit CCR-U Reporting-Reason
 4 VALIDITY_TIME CCR-T Reporting-Reason 2 FINAL

0000

OmniTWAG

- RADIUS CCR-U
- OmniTWAG Validity-Time
 OmniTWAG CCR-U OCS Validity-Time OmniTWAG
 quota.validity_time

CCA CC-Request-Number 1 RFC 4006

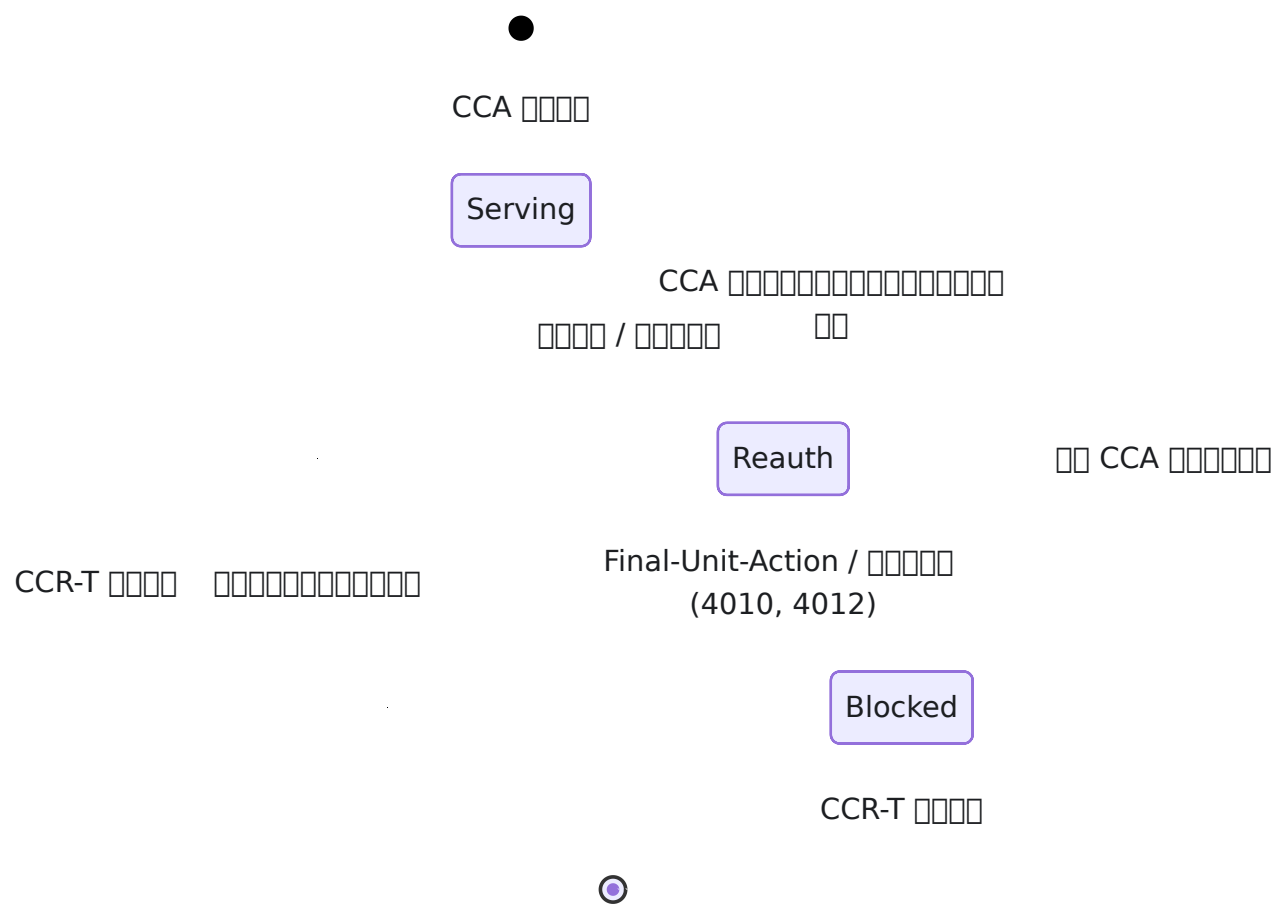
0000

OmniTWAG :online_charging TWAG
 GRE UE UE

Gy CCR-U OCS

- 最終 Final-Unit-Action TERMINATE REDIRECT RESTRUCT_ACCESS
- MSCC Result-Code 4010 4012

OmniTWAG REDIRECT RESTRUCT_ACCESS Final-Unit-Action OmniTWAG UE IP



Diameter CCR-U UE OCS CCA MSCC

rating_group

OmniTWAG rating_group CCR-I MSCC Rating-Group 1 OmniTWAG Requested-Service-Unit OCS



```
:omnitwag ██████████ config/runtime.exs | online_charging |
```

```
config :omnitwag,
  online_charging: %{
    # []:disabled[]:authorize_only [] :online_charging
    mode: :disabled,

    # OCS Diameter []CCR [] Destination-Realm[]
    ocs_realm: "epc.mnc001.mcc001.3gppnetwork.org",

    # []
    rating_group: 1,

    # [] :online_charging []
    quota: %{
      # [] CCR-Update
      update_threshold: 0.8,

      # [] OCS [] Validity-Time []
      validity_time: 3600
    }
  }
}
```

mode			:disabled	:disabled:authorize_on:online_charging
ocs_realm			Diameter	OCS Diameter OmniTWAG CCR Destination-Realm OmniTWAG Diameter
rating_group			1	OmniTWAG CCR-Initial MSCC Rating-Group
quota.update_threshold			0.8	0.0 1.0 CCR-Update
quota.validity_time				OCS Validity-Time

AP charging_mode AP RADIUS

Gy Diameter Origin-Host Origin-Realm :diameter_ex Diameter host realm Destination-Host OmniTWAG CCA Origin-Host Destination-Host AVP CCR-U CCR-T

Diameter Ro

Gy Diameter OCS Diameter :diameter_ex :ro

```
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
      }
    ]
  }
}
```

項目	型	単位	デフォルト値	説明
service_name	文字列	文字列	-	Diameter サーバのサービス名 OmniTWAG の場合は Gy の名前
host	文字列	文字列	-	Diameter サーバのホスト名 OmniTWAG の場合は host realm Gy の Origin Host
realm	文字列	文字列	-	Diameter サーバの realm OmniTWAG の場合は Origin-Realm ocs_realm Destination-Realm
request_timeout	整数	秒	5000	CCA のリクエストのタイムアウト時間
application_name	文字列	文字列	-	Diameter アプリケーション名 :ro Gy
application_dictionary	文字列	文字列	-	Diameter アプリケーション辞書 :diameter_gen_3gpp_r Gy
vendor_specific_application_ids	文字列	文字列	-	ベンダー固有のアプリケーション ID vendor_id: 0 auth_application_id: 4
peers	文字列	文字列	-	Diameter サーバのピア OCS の場合は Gy

配置参数

参数	数据类型	默认值	取值范围	说明
host	字符串	-	-	Diameter 主机 FQDN 或 OCS 主机名
realm	字符串	-	-	Diameter realm OCS realm
ip	字符串	-	-	OCS 的 TCP 的 IP 地址
port	整数	3868	-	OCS Diameter 端口 3868
transport	字符串	:diameter_tcp	:diameter_tcp 或 TCP	传输协议
initiate_connection	布尔值	false	true 或 false	是否由 OmniWAG 发起连接

```
config/runtime.exs 配置 online_charging 配置 ocs_host
service_identififer request_timeout quota initial_request
subsequent_request 配置 mode ocs_realm
rating_group quota.update_threshold quota.validity_time
Diameter 配置 :diameter_ex request_timeout OmniWAG 配置
Requested-Service-Unit 配置 initial_request subsequent_request
配置 CCR 配置
```

--	--	--	--	--	--	--	--	--

□ □ □ □ □ □

OmniTWAG 1 Gy 10000000000000000 Diameter 10000000000000000 OmniTWAG 10000000000000000

- **OCS** 設定済みのCCR の Diameter 設定済みのOmniTWAG 設定済みの CCR-Iの OmniTWAG 設定 Gy 設定済みのCCR-Tの CCR-UのOmniTWAG 設定 Gy の 設定済みの UEの CCR-TのOmniTWAG 設定済みの
- **CCA** 設定済みのOCS 設定済みの Result-Code の 2001のDIAMETER_SUCCESSの OmniTWAG 設定済みのOmniTWAG 設定済みの

0000000000000000 :online_charging 0000OmniTWAG 0000 OCS 00000000 CCA
 MSCC 00000000000000 UE 00000000 00000000000000000000 :authorize_only
 00000000000000000000 OCS 0000

```
Parse error on line 2: ...-> NoCharging: [ ]:disabled [ ] AP charg... -----
-----^ Expecting 'SPACE', 'NL', 'HIDE_EMPTY', 'scale', 'COMPOSIT_STATE',
'STRUCT_STOP', 'STATE_DESCR', 'ID', 'FORK', 'JOIN', 'CHOICE', 'CONCURRENT',
'note', 'acc_title', 'acc_descr', 'acc_descr_multiline_value', 'CLICK', 'classDef',
'style', 'class', 'direction_tb', 'direction_bt', 'direction_rl', 'direction_lr',
'EDGE STATE', got 'DESCR'
```



□□□□□□

OmniTWAG □□□□□□□□□□

- 結果コード OmniTWAG の 2001 年調査の結果コード
調査結果コードは、 調査結果コードの調査結果コード OmniTWAG の調査結果コード

- **MSCC** 的 **Result-Code** 为 MSCC 的 **Result-Code** 为 4010
END_USER_SERVICE_DENIED 4012 CREDIT_LIMIT_REACHED
:online_charging UE 黄色

接口

OmniTWAG 的 type 为 :initial :update :terminate

接口	消息	参数
[:twag, :gy, :ccr, :sent]	OmniTWAG 的 CCR	type
[:twag, :gy, :cca, :received]	OmniTWAG 的 CCA	type result :success :error
[:twag, :gy, :ccr, :error]	CCR 的	type
[:twag, :gy, :quota, :blocked]	UE 的	identity
[:twag, :gy, :quota, :unblocked]	UE 的	identity

接口 的 Gy 的 Prometheus 的 接口 的 Gy 的
接口 的 Gy 的 INFO 的 CCR-I CCR-U CCR-T 的 WARN 的
ERROR 的 Diameter Session-Id 的 CCR-I 的 IMSI 的

☐☐☐

CC-Request-Type ☐

☐	☐☐	☐☐
1	INITIAL_REQUEST	CCR-Initial
2	UPDATE_REQUEST	CCR-Update
3	TERMINATION_REQUEST	CCR-Termination

☐☐ RFC 4006 §8.3 ☐☐☐☐

☐☐☐ **Reporting-Reason** ☐

☐	☐☐	OmniTWAG ☐☐☐☐☐☐
2	FINAL	CCR-Termination
4	VALIDITY_TIME	CCR-Update

Result-Code

Result-Code	Result-Code	Result-Code	OmniTWAG Result-Code	Result-Code
2001	Result-Code	DIAMETER_SUCCESS	Result-Code Gy Result-Code	RFC 6733 §7.1.1
(Result-Code)	Result-Code	-	Result-Code Result-Code Result-Code	-
Result-Code	Result-Code	(Result-Code)	Result-Code Result-Code	RFC 4006 §9
4010	MSCC	END_USER_SERVICE_DENIED	Result-Code Result-Code :online_charging Result-Code UE	RFC 4006 §9
4012	MSCC	CREDIT_LIMIT_REACHED	Result-Code Result-Code :online_charging Result-Code UE	RFC 4006 §9

Final-Unit-Action

MSCC Final-Unit-Indication Result-Code Final-Unit-Action OmniTWAG Result-Code Result-Code REDIRECT RESTRICT_ACCESS Result-Code Final-Unit-Action Result-Code UE

Result-Code	Result-Code	OmniTWAG Result-Code
0	TERMINATE	Result-Code UE Result-Code Result-Code
1	REDIRECT	Result-Code UE Result-Code Result-Code Result-Code
2	RESTRICT_ACCESS	Result-Code UE Result-Code Result-Code Result-Code

Diameter

ID				
4	Ro / Gy	diameter_gen_3gpp_ro	TWAG OCS	RFC 40063GPP TS 32.299

Diameter	RFC 4006
Diameter (Ro)	3GPP TS 32.299
	3GPP TS 32.240
TWAG / 3GPP (S2a, NSW0)	3GPP TS 23.402
Diameter	RFC 6733

-
- RADIUS
- IP
- IP
-
- S2a GTP-C
-

設定

概要

OmniTWAG は RADIUS 認証サーバーと連携して、無線 LAN 接続のクライアントの認証を行います。OmniTWAG は、NAS-IP-Address を RADIUS サーバーの NAS-IP-Address と一致させることで、UBNT の OmniTWAG が RADIUS サーバーの UDP/IP アドレスを認識できるようにします。

設定

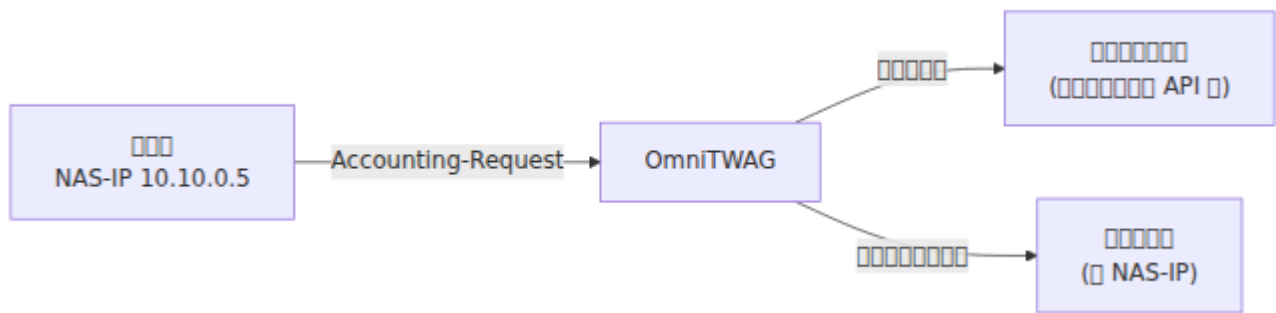
- 設定
- 接続
- 設定
- 接続
- 設定
- 接続
- 設定
- 接続

設定

RADIUS Accounting-Request の Acct-Status-Type を設定します。

- (1): 接続
- (3): 接続
- (2): 接続

OmniTWAG は NAS-IP-Address を OmniTWAG の IP アドレスと一致させることで、無線 LAN 接続のクライアントの認証を行います。



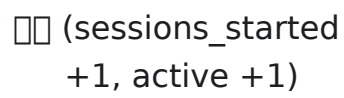
Configuration

RADIUS configuration is done on the OmniTWAG device. The configuration is as follows:

OmniTWAG configuration is done on the OmniTWAG device. The configuration is as follows:

- `acct_sessions_started` 1 `acct_active_sessions` 1
- `acct_sessions_stopped` 1 `acct_active_sessions` 1
- `acct_sessions_stopped` 1 `acct_active_sessions` 1
- `acct_sessions_stopped` 1 `acct_active_sessions` 1

OmniTWAG configuration is done on the OmniTWAG device. The configuration is as follows:



```
    sessions_stopped
+1, active -1, sessions/active
    )
```

□□□□ (□□□)



OmniTWAG [redacted] SSID [Called-Station-Id]

項目	単位	説明
acct_input_octets	バイト	入力されたデータの総量
acct_output_octets	バイト	出力されたデータの総量
acct_input_packets	パケット	入力されたパケットの総数
acct_output_packets	パケット	出力されたパケットの総数
acct_sessions_started	セッション	開始されたセッションの数
acct_sessions_stopped	セッション	終了したセッションの数
acct_active_sessions	セッション	現在アクティブなセッションの数。started と stopped の差。RADIUS クラウド環境では、この値は常に 0 である。これは「セッション」ではなく、単に「セッション」を示す。
last_accounting_at	時刻	最後の会計処理が行われた時刻。Unix 時刻で表される。

この情報は、RADIUS プロトコル (RFC 2866) に基づいて収集される。

クライアント

この情報は、API の client_count によって提供される。acct_active_sessions は、OmniWAG のアクティブセッションを示す。

[[NAS-IP-Address]]

[[RADIUS Accounting-Stop]]

[[client_count]]
[[acct_active_sessions]]

[[]]

[[]]

- **REST API:** GET /access_points GET /access_points/{id} NAS-IP-Address
- **Web** : [[]]

[[]]

[[RADIUS Accounting-Request]]

- [[APState]]
- [[ClientUsage]]

[[API]]

[[Access-Request]]

目標

前提条件

OS: CentOS7

必要なツール:

- CentOS7にOmniTWAGをインストールする
- 設定

手順:

1. CentOS7にOmniTWAGのAccounting-Requestをインストール
2. 設定
3. CentOS7にRADIUSをインストール

インストール

OS: CentOS7にacct_active_sessionsをインストール

必要なツール:

- CentOS7にacct_active_sessionsをインストール

手順:

1. CentOS7にacct_active_sessionsをインストール
2. CentOS7にOmniTWAGをインストール
3. CentOS7にRADIUSをインストール

OS: CentOS7にacct_active_sessionsをインストール、client_count APIをインストール

確認

OS: CentOS7にacct_active_sessionsをインストール

□□□□:

- OmniTWAG □□□□□□□□ IP □□□□□□□□

□□□□:

1. □□ radius_auth_unauthorized_source_count □□□□□ □□□□ □ RADIUS □□□
2. □□□□□□□□□□□□ RADIUS □□□□□

□□□□

- □□
- □□
- RADIUS □□□□
- RADIUS □□
- □□□□
- □□

□□

OmniTWAG 測試結果

測試環境 WLAN 測試環境測試環境測試環境

測試環境

CPU	Intel(R) Core(TM) i5-5250U CPU @ 1.60GHz (2 cores)
Runtime	Elixir 1.17.0 / Erlang OTP 27 (JIT) (erts 15.0.1)
Measurement	1 s warmup + 3 s measure + 1 s memory per op
Commit	main @ 01ec4e8

測試環境 ips測試環境測試環境Avg 測試環境 Median 測試環境測試環境測試環境測試環境
Mem/op 測試環境測試環境

測試環境測試環境測試環境

測試環境測試環境測試環境EAP 測試環境測試環境EAP-AKA' 測試環境測試環境MAC測試環境
測試環境 MAC 測試環境測試環境ips測試環境 測試環境測試環境測試環境//// 測試環境測試環境Milenage測試環境
STa 測試環境 HSS 測試環境 SWx/STa 測試環境

測試環境	測試環境 (ips)	測試環境 (μs)	測試環境 (μs)	Mem/op
測試環境 EAP-AKA' 測試環境STa 測試環境測試環境 HSS測試環境	17,311	57.77	52.90	7,048 B
測試環境 EAP-AKA' 測試環境測試環境測試環境測試環境 測試環境	12,493	80.05	73.31	10,560 B



□□□□□□□□□□SIM □□□□□□□□□□Milenage□□EAP-AKA' □□□□□□CK'/IK'□□□□□□MAC□□
ERP □□□□□□□□□□□□□□□□ 3GPP □□□□□□□□

□□	□□□ (ips)	□□ (μs)	□□□ (μs)	Mem/op
WLCP □□□□□EMSK -> WLCP □□□	262,975	3.80	3.54	72 B
EAP-AKA' □□□□□□□□□□□□	257,991	3.88	0.87	456 B
EAP-AKA' □□ CK'/IK'	225,191	4.44	3.93	328 B
EAP-AKA' □□ MAC□HMAC-SHA-256/16□	223,503	4.47	4.11	72 B
3GPP □□ KDF□HMAC-SHA-256□	198,596	5.04	4.32	248 B
ERP □□ rMSK□□□□□□□□□□	113,269	8.83	8.01	376 B
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
EAP-AKA' derive_keys□MK -> □□□□□□	31,696	31.55	28.93	1,552 B
EAP-AKA' □□□□□□□□□□CK'/IK' + □□□□□	28,380	35.24	32.59	1,808 B

項目	説明
EAP-AKA' のフィールド	AT EAP-Request/AKA'-Challenge AT_RAND AT_AUTN AT_KDF AT_MAC
WLCP のフィールド	EMSK WLCP のフィールド
3GPP の KDF	HMAC-SHA-256 のフィールド
EAP-AKA' の MAC	EAP の HMAC-SHA-256 AT_MAC の 16 ビット
EAP-AKA' の CK'/IK'	CK/IK のフィールド CK'/IK'
ERP の rMSK	EAP の RFC 6696
ERP のフィールド	rRK → rIK → rMSK
Milenage compute_all	AKA の RES CK IK AK MAC-A
EAP-AKA' derive_keys	K_encr K_aut K_re MSK EMSK
EAP-AKA' のフィールド	CK'/IK' のフィールド

フィールド

TS 24.244 EAP WLCP TS 24.244 RADIUS Access-Accept MPPE

項目	項目 (ips)	項目 (μs)	項目 (μs)	Mem/op
WLCP 項目PDN 項目項目	4,229,558	0.24	0.20	448 B
WLCP 項目PDN 項目項目	2,492,441	0.40	0.24	160 B
WLCP 項目 + 項目項目	1,989,407	0.50	0.40	608 B
EAP 項目項目項目	336,816	2.97	2.66	1,136 B
MPPE 項目 VSA項目Send-Key項目RC4 + MD5項目	135,026	7.41	5.32	1,296 B

項目	項目
WLCP 項目	項目項目 WLCP PDN 項目項目
WLCP 項目	項目 WLCP PDN 項目項目項目項目項目
WLCP 項目 + 項目項目 項目	項目 WLCP 項目項目項目項目項目
EAP 項目項目項目	項目 EAP 項目項目NAI 項目項目
MPPE 項目 VSA	項目項目 MSK 項目 MS-MPPE-Send-Key RADIUS VSA項目RC4 + MD5項目

RADIUS 設定ガイド IP 設定

概要

OmniTWAG は RADIUS 設定で NAS 設定と IP 設定を REST API を利用して Web から設定できます。

目次

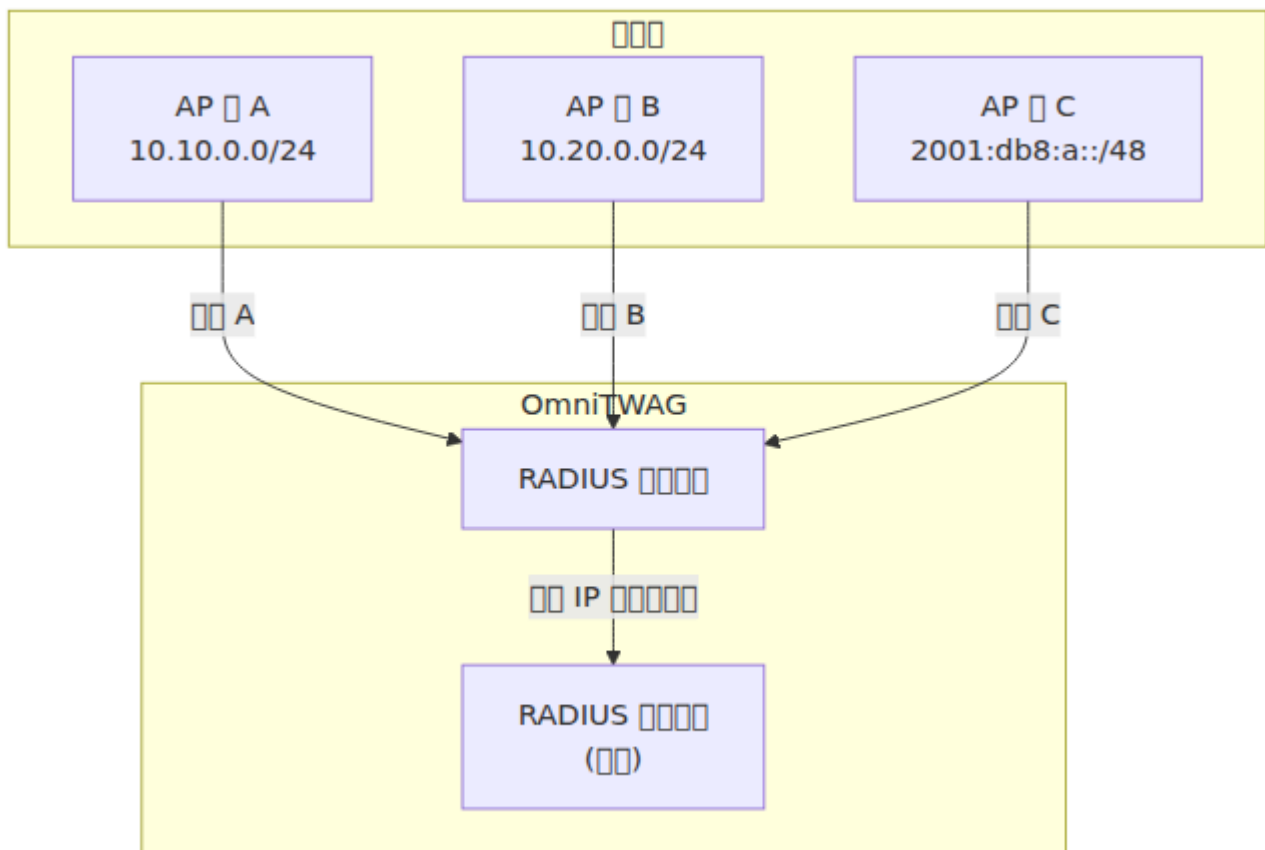
- 概要
- 設定項目一覧
- 設定方法
- AP 設定
- REST API
- Web 設定
- 設定例
- 設定確認
- 設定確認
- 設定確認

設定

設定項目は CIDR 形式の IPv4/IPv6 設定項目です。

IP 設定 RADIUS 設定 IP 設定 OmniTWAG 設定 UDP 設定 IP 設定 OmniTWAG 設定 NAS-IP-Address 設定

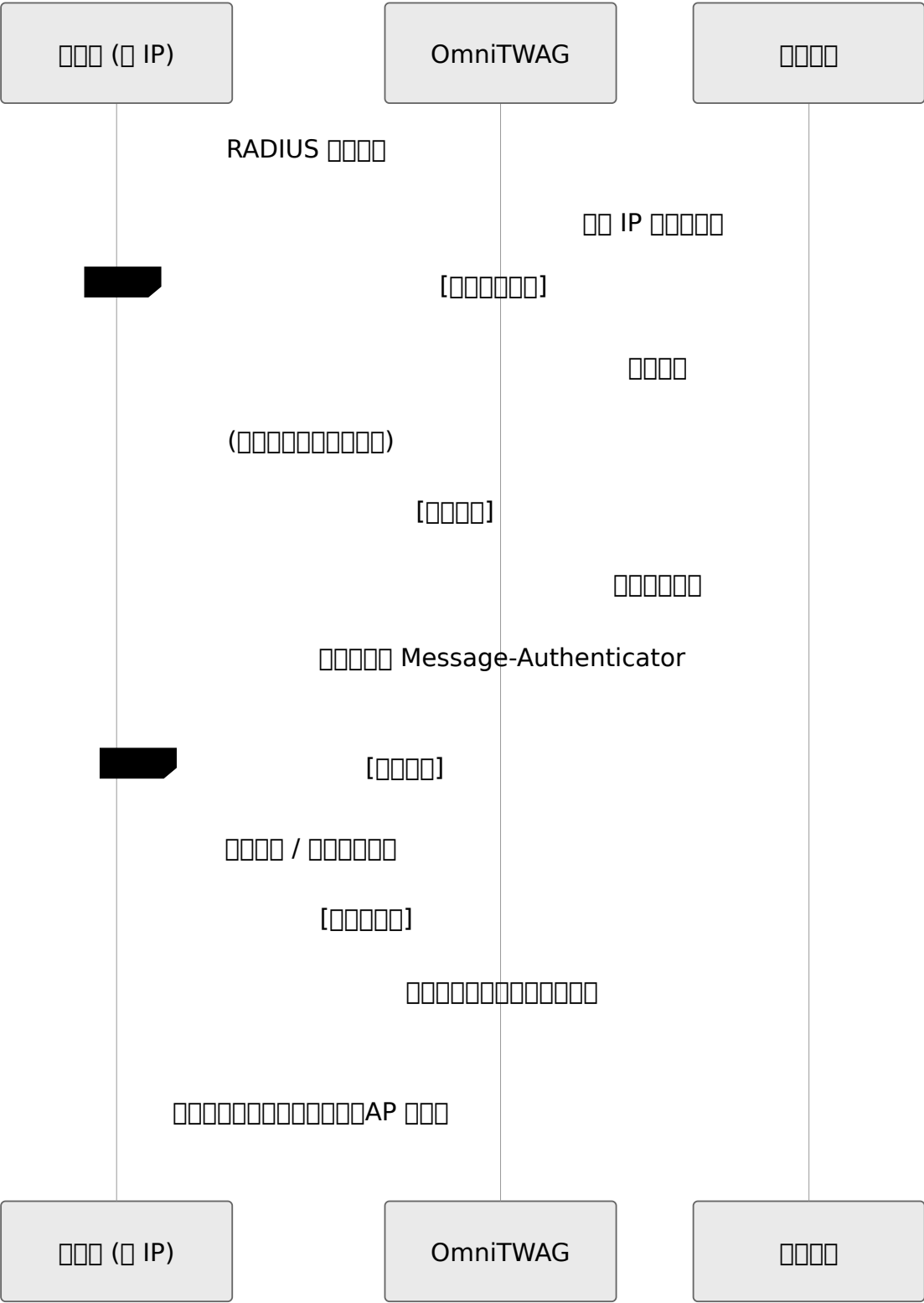
設定項目は IP 設定項目です。IP 設定項目は OmniTWAG 設定項目です。



OmniTWAG RADIUS

1. OmniTWAG IP
2. OmniTWAG IP OmniTWAG
3. OmniTWAG
4. OmniTWAG Message-Authenticator MPPE

Message-Authenticator OmniTWAG
 OmniTWAG



Client (IP)

Server

欄名	型	説明
id	整数	一意なUUIDを生成して付与
name	文字列	名前。例: "CBD APs"
secret	文字列	RADIUS の秘密鍵
subnets	文字列 配列	割り当てられる CIDR。IPv4/IPv6 の両方に対応 ["10.10.0.0/24", "2001:db8:a::/48"]
breakout_type	文字列	接続タイプ。例: tunneled, nsw, local tunneled の場合は AP が必要
address_source	文字列	IP アドレスのソース。例: nsw, UE IP, pool, dhcp pool の場合は AP が必要
charging_mode	文字列	Gy の充電モード。例: charged, uncharged, auth_only charged の場合は AP が必要
ip_pool_id	整数	IP プールの ID。例: nsw, null など
inserted_at	日時	登録日時 (Unix 形式)
updated_at	日時	更新日時 (Unix 形式)

検索条件: name, secret, subnets, breakout_type, address_source, charging_mode, ip_pool_id, id, inserted_at, updated_at

AP 管理

AP は GTP と PGW を接続する

breakout_type

breakout_type 网络侧 TWAG

枚举	描述
tunneled	OmniTWAG 通过 PGW 的 GTP-C/GTP-U 隧道连接到网络侧 通过 S2a GTP-C 隧道
nsw	通过 WLAN 接入 OmniTWAG 连接到网络侧 PGW，OmniTWAG 的 IP 地址由 UE IP 地址映射到 TWAG 地址
local	OmniTWAG 连接到网络侧 IP 地址由 UE 地址映射

address_source

address_source 网络侧 breakout_type 为 nsw 时 UE IP 地址由 IP 地址

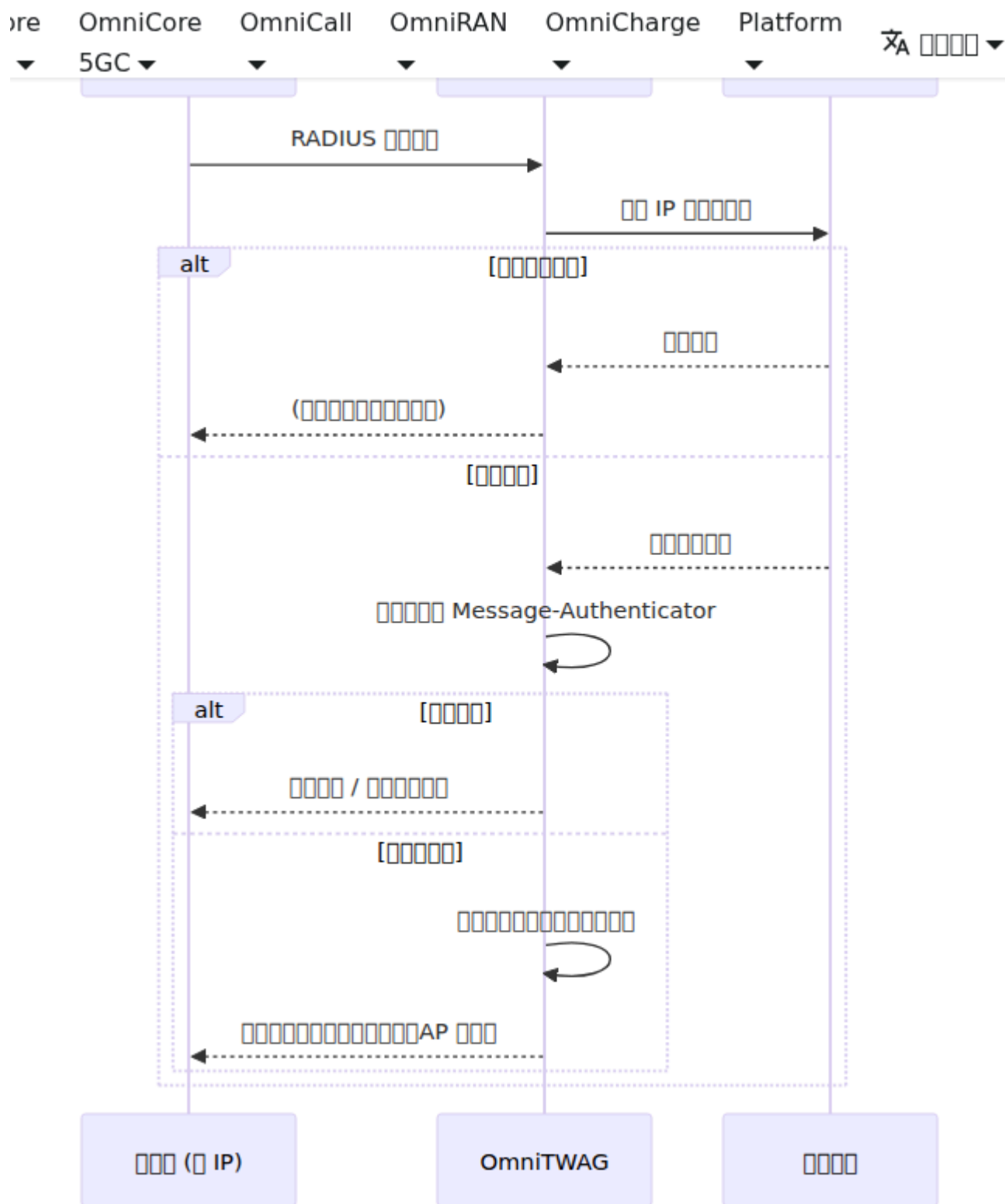
枚举	描述
pool	OmniTWAG 通过 IP 地址连接到网络侧
dhcp	UE 通过 DHCP DISCOVER 请求 OmniTWAG DHCP 地址

charging_mode

charging_mode 网络侧 Gy 计费模式

枚举	描述
charged	OmniTWAG 通过 Gy 计费模式 OCS 计费
uncharged	OmniTWAG 通过 Gy 计费模式
auth_only	OmniTWAG 通过计费模式

```
charging_mode: auth_only  auth_only  OmniTWAG  
breakout_type  address_source
```



REST API

OmniTWAG REST API 通过 HTTPS 访问 `api_ex` 端口 `8444` 访问地址 `/radius_credentials`

方法	URL	操作	说明
GET	<code>/radius_credentials</code>	index	获取列表
GET	<code>/radius_credentials/{id}</code>	show	获取 <code>id</code> 详情
POST	<code>/radius_credentials</code>	create	创建
PUT	<code>/radius_credentials/{id}</code>	update	更新
DELETE	<code>/radius_credentials/{id}</code>	delete	删除

注意

`POST` 和 `PUT` 请求体为 JSON 格式

項目	型	必須	説明
name	文字列	○	名前
secret	文字列	○	RADIUS 認証用パスワード
subnets	文字列 配列	○	割り当てられる CIDR 範囲
breakout_type	文字列	○	tunneled nsw local tunneled AP 接続
address_source	文字列	○	pool dhcp nsw pool
charging_mode	文字列	○	charged uncharged auth_only charged
ip_pool_id	文字列	○	nsw IP の id

PUT 送信

レスポンス

ステータスコード	説明
200 OK	正常に作成されました
201 Created	正常に作成されました
404 Not Found	指定された id は見つかりません
422 Unprocessable Entity	secret CIDR breakout_type / address_source / charging_mode が不正です

GET /radius_credentials

200

GET /radius_credentials

200

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

POST /radius_credentials

201

POST /radius_credentials

```
{
  "name": "CBD APs",
  "secret": "s3cr3t",
  "subnets": ["10.10.0.0/24", "2001:db8:a::/48"]
}
```

201 Created 200 OK id

--	--	--	--	--	--	--

111

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

422 Unprocessable Entity

```
{
  "error": "CIDR 10(s): not-a-subnet"
}
```

Web ☐ ☐

```

##### /radius_credentials ##### RADIUS #####
#####

```

11

RADIUS 00000000 DETS 00000 radius_config 00000000

```
config :omnitwag,
  radius_config: %{
    # !! RADIUS !!
    credentials_dets_path: "priv/radius_credentials.dets",
    # !! "!!" !!
    allowed_source_subnets: [],
    secret: "123456"
  }
end
```

参数	数据类型	是否必填	默认值	说明
credentials_dets_path	字符串	否	priv/radius_credentials.dets	DETS 文件的路径。DETS 文件是 Radius 服务器的配置文件，其中包含 Radius 服务器的配置信息。该文件位于 /priv/radius_credentials.dets 目录下。
secret	字符串	否	123456	Radius 服务器的共享密钥。该密钥用于 Radius 服务器与客户端之间的通信。该密钥的长度至少为 8 个字符，且必须包含大写字母、小写字母、数字和特殊字符。该密钥的默认值为 123456。
allowed_source_subnets	字符串	否	[]	允许访问 Radius 服务器的 IP 地址范围。该范围由一个或多个 IP 地址或 IP 地址范围组成。该范围的默认值为空数组 []。

設定

secret 設定 allowed_source_subnets 設定 OmniTWAG 設定

- secret
- secret allowed_source_subnets

secret 設定 OmniTWAG 設定

allowed_source_subnets 設定 OmniTWAG 設定 OmniTWAG 設定
RADIUS REST API Web

secret allowed_source_subnets 設定 API Web

設定

OmniTWAG Prometheus 設定

radius_auth_unauthorized_source_count

設定
IP 設定 RADIUS

設定	設定
source	IP

radius_auth_bad_secret_count

設定
IP Message-Authenticator 設定 RADIUS

項目	説明
source	送信元 IP
credential_id	認証 ID

設定

基本設定

この機能は RADIUS 認証サーバーの `radius_auth_unauthorized_source_count` 設定値

に基づいて

- 送信元 IP
- 送信元 IP が NAT された

設定

1. OmniTWAG の送信元 IP
2. 送信元 IP
3. 送信元 IP が NAT された

詳細設定

この機能は `radius_auth_bad_secret_count` 設定値に基づいて

設定

- 送信元 IP

設定

1. 送信元 `credential_id` 設定値に基づいて
2. 送信元 IP
3. 送信元 IP が NAT された

□□□□□□

□□ □□□□□□□□ OmniTWAG □□□□□□

□□□□□

- `credentials_dets_path` □□□□□□□□□□□□□□□□□□

□□□□

1. □ `credentials_dets_path` □□□□□□□□□□
2. □□ OmniTWAG □□□□□□□□
3. □□ API □ Web □□□□□□□□

RADIUS 簡介

簡介

OmniTWAG 支援 RADIUS 認證與授權功能，可與 RADIUS 伺服器進行認證與授權。OmniTWAG 支援 RADIUS 認證與授權功能，可與 RADIUS 伺服器進行認證與授權。

OmniTWAG 支援 RADIUS 認證與授權功能，可與 RADIUS 伺服器進行認證與授權。

功能

- 支援 RADIUS 認證與授權
- 支援 RADIUS 認證與授權
- 支援 RADIUS 認證與授權
- 支援 RADIUS 認證與授權
- 支援 RADIUS 認證與授權
- REST API
- 支援 RADIUS 認證與授權
- 支援 RADIUS 認證與授權

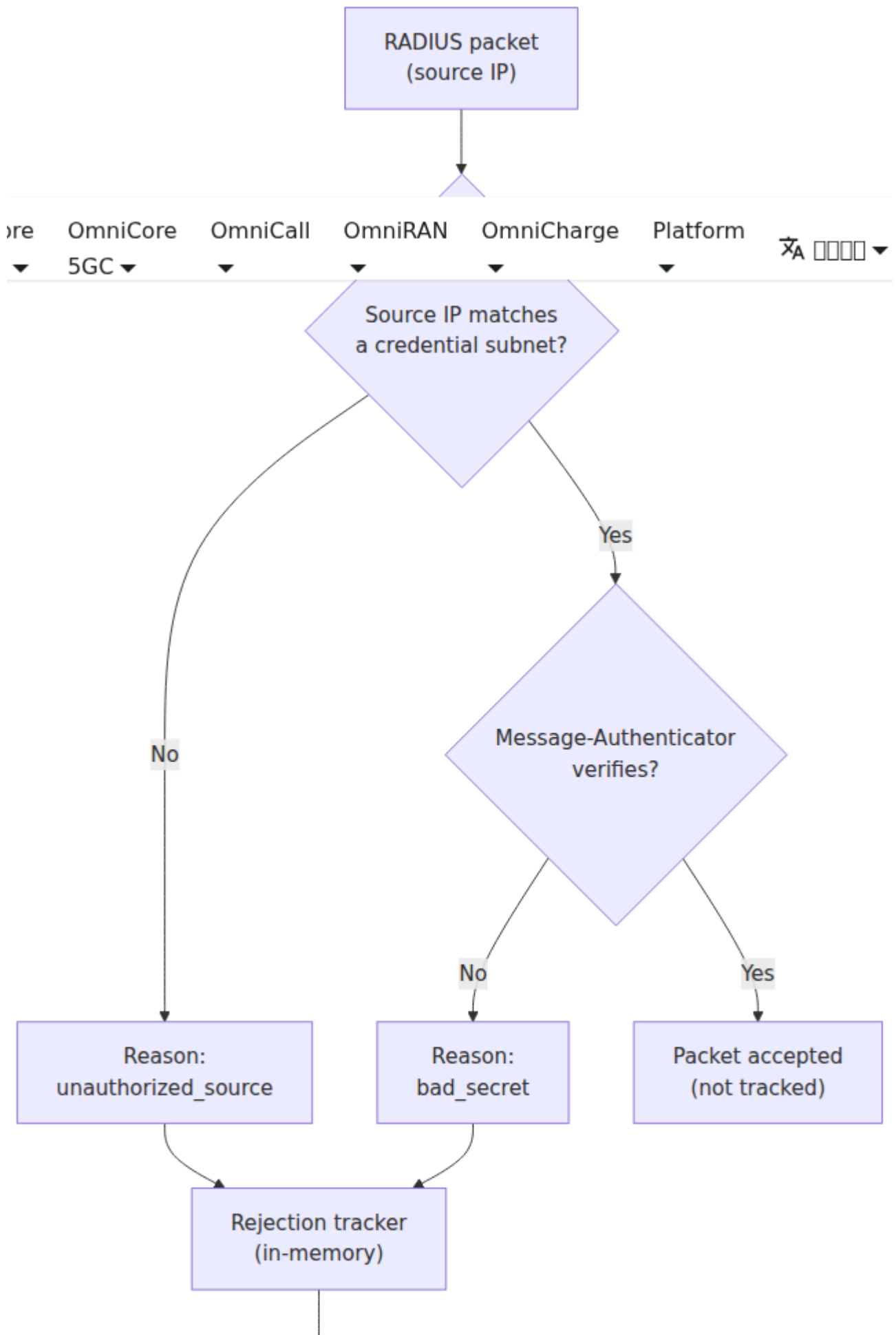
安裝

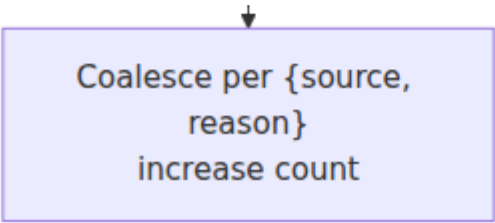
OmniTWAG 支援 RADIUS 認證與授權功能，可與 RADIUS 伺服器進行認證與授權。OmniTWAG 支援 RADIUS 認證與授權功能，可與 RADIUS 伺服器進行認證與授權。

IP 地址 RADIUS 認證與授權 IP 地址 OmniTWAG 支援 UDP 協議，IP 地址 OmniTWAG 支援 NAS-IP-Address 認證與授權。

認證與授權

OmniTWAG 支援 RADIUS 認證與授權功能，可與 RADIUS 伺服器進行認證與授權。OmniTWAG 支援 RADIUS 認證與授權功能，可與 RADIUS 伺服器進行認證與授權。





□□□□

□□□□□□□□□□□□

□□	□□
unauthorized_source	□ IP □□□□□□□□□□OmniTWAG □□□□□□□□□□□□□□□□
bad_secret	□□□□□□□□□□□□ Message-Authenticator □□□□□□□□ □□□□□□□□□□□□□□□□

□□ bad_secret □□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□

□□□□□□□□□□

項目	型	説明
source	文字列	接続元 IP アドレス
reason	文字列	拒否理由: unauthorized_source または bad_secret
count	整数	OmniTWAG での拒否回数
first_seen	日時	初回拒否の日時 (Unix 形式)
last_seen	日時	最終拒否の日時 (Unix 形式)
credential_id	文字列	拒否理由: id または reason または bad_secret
credential_name	文字列	拒否理由: name または reason または bad_secret

拒否理由: unauthorized_source, 拒否理由: credential_id, 拒否理由: credential_name

注意事項

注意事項

- 拒否理由: unauthorized_source
- 拒否理由: OmniTWAG 拒否理由: source または reason 拒否理由: count 拒否理由: last_seen
- 拒否理由: 拒否理由: 256
- 拒否理由: 拒否理由: OmniTWAG 拒否理由: last_seen 拒否理由: IP 拒否理由: IP

REST API

OmniTWAG REST API 拒否理由: API 拒否理由: 8444 拒否理由: /radius_rejections

Method	URI	Action	Description
GET	/radius_rejections	index	Return all radius rejection records source [reason]
DELETE	/radius_rejections/{id}	delete	Delete radius rejection record with ID {id}

DELETE endpoint is used to delete a specific IP address from the radius rejection list.

Response

Status Code	Message
200 OK	Success
404 Not Found	IP address not found

Example Request

Example

```
GET /radius_rejections
```

Example

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

□□□□□□□□

□□□

```
DELETE /radius_rejections/203.0.113.7
```

□□□

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

11

□□□□□□□□□□□□□□ Prometheus □□□□□□□□□□ OmniTWAG □□□□□□□□□□

名前	データ型
radius_auth_unauthorized_source_count	unsigned_source
radius_auth_bad_secret_count	bad_secret

Prometheus **RADIUS** **AAA**

--	--	--	--

[illegible]

```

原因 未授权源
radius_auth_unauthorized_source_count

```

□□□□□

- 配置 IP
- 配置 NAT 和 OmniTWAG 和 RADIUS 和 IP 地址

□ □ □ □ □

1. 配置 `source`
2. 配置认证服务器
3. 配置 OmniTWAG 使用 IP 地址 NAT 转换，并配置 RADIUS 认证
4. 删除 `/radius_rejections/{id}` 接口

S2a / GTP-C 和 PGW

概要

S2a 是 OmniTWAG 网络 (PGW) 和 WLAN 网络 (TWAN) 之间的 GTP 和 PDN 网络。OmniTWAG 和 TWAN GTP-C 网络。S2b 是 ePDG 网络。S2a 是 TWAN 网络。

S2a 网络

- **GTP-C** 网络 PDN 网络 GTP-C
- **GTP-U** 网络 GTP 网络 UE 网络 网络

S2a 是 GTP-C 是 3GPP TS 29.274 GTPv2-C 和 TWAG 是 GTPv2-C 网络。IE 是 S2a 网络。

网络

- S2a 网络
- 网络
- S2a 网络
- 网络
 - 网络
 - 网络
 - 网络
 - 网络
 - 网络
- 网络
- 网络
- 网络
- 网络
- 网络
- 网络
- 网络

- 3GPP

S2a

OmniTWAG AP RADIUS (AP) breakout_type RADIUS breakout_type address_source ip_pool_id

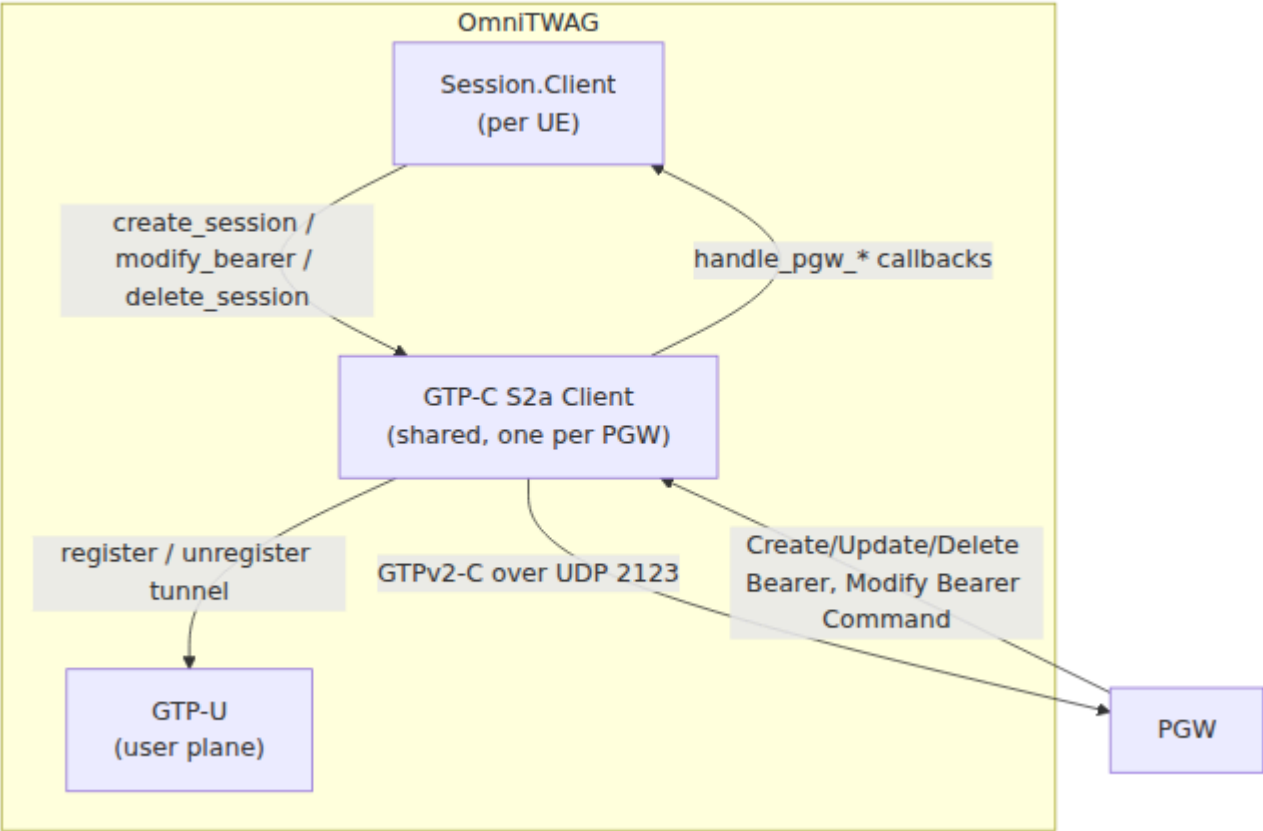
(breakout_type)	UE IP		S2a / GTP
tunneled	PGW PDN IE	AP TWAG GRE S2a GTP-U PGW	
nswo	TWAG IP	AP TWAG GRE TWAG NAT	PGW
local	AP	AP TWAG	PGW IP

S2a GTP-C tunneled tunneled PGW UE IP nswo local TWAG PDN S2a

tunneled PGW TWAG TWAG Gy

```
WLAN (NSWO) (Twag.NSW0.Handler  
nswo.enabled: true) GTP  
AP nswo
```


GTP-C (Twag.Gtp.GtpcS2a) PGW tunneled UE Session.Client GTP-C GTPv2-C



S2a 〇〇〇〇

TWAG 〇〇 S2a F-TEID 〇〇〇〇〇〇〇〇 S2b〇ePDG〇〇〇〇〇〇〇〇〇〇 30 〇 31〇〇〇〇 TS 29.274 〇 8.22-1〇

〇〇〇〇	〇	〇〇	〇〇
s2a_twan_gtp_c	35	TWAG 〇〇	TWAG F-TEID 〇〇〇〇〇〇〇〇
s2a_twan_gtp_u	34	TWAG 〇〇	TWAG 〇〇 F-TEID〇GTP-U〇
s2a_pgw_gtp_c	36	PGW 〇〇	PGW F-TEID 〇〇〇〇〇〇〇〇
s2a_pgw_gtp_u	37	PGW 〇〇	PGW 〇〇 F-TEID〇GTP-U〇

RAT 〇〇〇〇〇 WLAN〇〇 3〇〇

--	--	--	--

11

0000 EAP 0000000000AP 00 Accounting-Start0000000000Session.Client 00 AP 0
 00000000 tunneled 00000000 GTP-C 00000000 PDN 000000 gtp.enabled 000 false 0
 TWAG 000 S2a00000000 IMSI0TWAG 0000000000 nsw0 0 local 000TWAG 00000
 S2a 00000000 00 0000000000000000

□ □ □ □ □ □

TWAG PGW TS 29.274 §7.2.1 TWAG
 TEID TEID

項目	項目 / 値	単位
IMSI	UE 番号	15桁
国コード (MCC/MNC)	<code>gtp.mcc</code> / <code>gtp.mnc</code> 国コード IMSI 桁数	1桁 / 2桁
RAT 種別	WLAN (3)	3 S2a 種別
F-TEID 種別	35桁 TWAG 種別 TEID TWAG 種別 IP	35桁 TWAG GTP-C 種別
APN (APN)	<code>gtp.apn</code> 種別 <code>internet</code> 種別 APN	3桁 4桁
種別	2桁	APN 種別 TWAN 種別
PDN 種別 (PAA)	1桁	TWAG 種別 PGW 種別
APN-AMBR	2桁 256000 桁 / 256000 桁	2桁
種別	2桁 EBI 5桁 F-TEID 種別 34桁 QoS	3桁 4桁
APN 種別	2桁 0桁	
ME 種別 (IMEISV)	UE 種別	15桁
種別	TWAG 種別	1桁
種別	<code>0x0800</code>	1桁
TWAN 種別	種別 <code>twan_ssid</code> 種別 SSID 種別 <code>twan_bssid</code> 種別 BSSID	3桁 WLAN 種別
UE 種別	15桁	

項目	値 / 単位	単位
値	項目	項目項目項目項目

PDN 項目 IPv4項目 gtp.pdn_type 項目 :ipv4 :ipv6 :ipv4v6項目 PAA 項目
項目項目 PGW 項目項目

項目項目項目 (APCO) IE 項目TWAG 項目 APCO 項目 DNS 項目 P-CSCF 項目
PGW 項目 DNS項目 PCO 項目 PCO IE 項目

項目項目

TWAG 項目項目項目 EBI 項目 5項目 TFT 項目項目項目 QoS 項目項目
項目

QoS 項目	項目
QCI	9
項目項目	10
項目項目	項目
項目項目	項目
GBR / MBR項目項目	0

TWAG 項目項目項目 項目項目

項目項目項目

TWAG 項目項目項目 IE項目 request_accepted
new_pdp_type_due_to_network_preference
new_pdp_type_due_to_single_address_bearer_only項目項目項目
gtpc_error

項目TWAG 項目項目項目

項目	説明	備考
PDN 名前 (PAA)	UE IP 範囲	DHCP による GRE による UE への IP 割り当て
F-TEID/PGW 識別子	PGW 識別子 / PGW IP	
	UE F-TEID/PGW GTP-U TEID 識別子 / IP 範囲 / EBI 識別子 / ID	GTP-U 識別子
APN-AMBR		
PCO / 優先度 PCO	DNS 識別子	DHCP による UE への

PGW	TWAG	
Create Bearer Request	Create Bearer Response, cause <code>request_rejected</code>	TWAG TSCM
Modify Bearer Command	Modify Bearer Request back to the PGW	TWAG APN-AMBR F-TEID <code>system_failure</code>
Update Bearer Request	Update Bearer Response, cause <code>request_accepted</code> per bearer	TWAG QoS
Delete Bearer Request	Delete Bearer Response, cause <code>request_accepted</code>	

TWAG APN-AMBR EBI F-TEID QoS PGW
 GTP-U F-TEID GTP-U
 TWAG APN-AMBR EBI QoS

TWAG

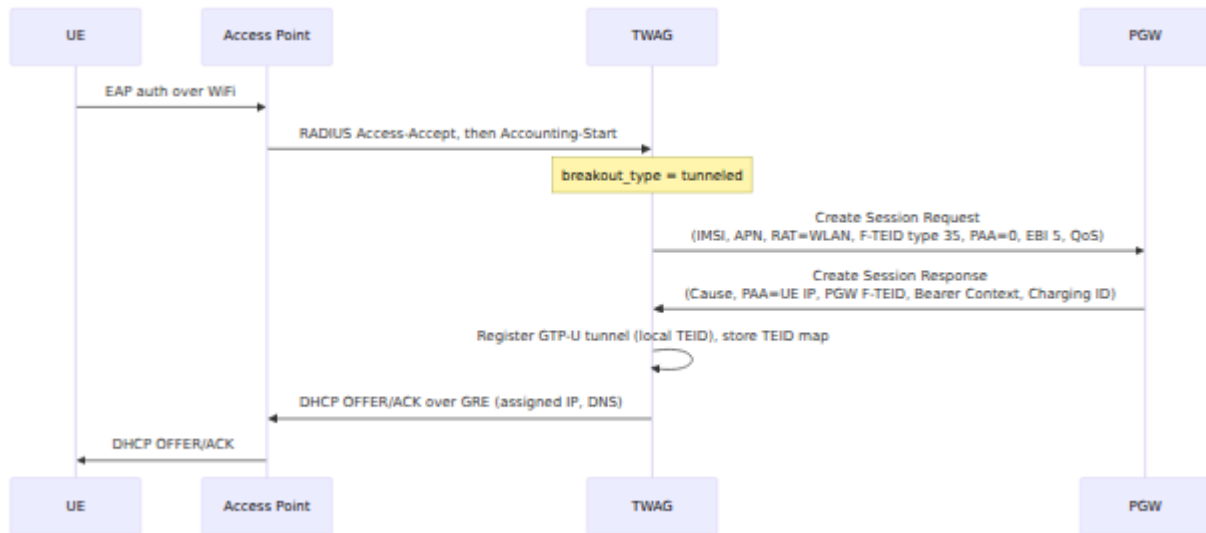
UE PDN `Session.Client` GTP-C TWAG PGW
 TEID TS 29.274 §7.2.9

- EPS ID EBI 5
- `local_detach`

TWAG IE PGW TEID TEID
 PGW TEID `nsw` `local` TWAG GTP-C `Session.Client`
 GTP-U DHCP IP UE GRE

PGW

PGW Session.Client
TWAG PGW
request_accepted EBI PDN



TWAG GTPv2-C TS 29.274 §6 echo_enabled
TWAG 60 PGW TWAG

PGW TWAG TWAG

TWAG PGW UP 3
TWAG DOWN

PGW

TWAG PGW TWAG PGW
pgw_restart TEID

設定項目

TWAG の T3-RESPONSE と N3-REQUESTS の設定

項目 / 単位	値	説明
T3-RESPONSE	3000 ms	応答時間
N3-REQUESTS	3	リクエスト回数

設定した TWAG の応答時間が 3 秒を超えると TWAG の状態が `timeout` となり、`DOWN` 状態になります。

TWAG が GTP-C UDP を受信すると TWAG は 5 秒間

実行

S2a に `gtp` を実行し、`runtime.exe` と `omnitwag` を実行します。

```
config :omnitwag,  
  gtp: %{  
    # S2a  
    enabled: true,  
  
    # PGW GTP-C  
    pgw_ip: "10.5.198.1",  
    pgw_port: 2123,  
  
    # TWAG IP GTP-C GTP-U F-TEIDs  
    local_ip: "10.5.198.200",  
  
    # GTP-U  
    gtpu_port: 2152,  
  
    #  
    echo_enabled: true,  
  
    #  
    apn: "internet",  
    pdn_type: :ipv4  
  }  
}
```

GTP

項目	項目	項目	項目	項目
enabled			false	S2a 接続を true 有効にする false TWAG 側の GTP-C 接続 を S2a
pgw_ip			127.0.0.1	PGW GTP-C IP TWAG 側の GTPv2-C 接続
pgw_port			2123	PGW GTP-C UDP 側の GTP-C 接続 2123
local_ip				TWAG 側の IP F-TEIDs 側の GTP-C 接続 GTP-U 側の TWAG 側の IP
gtpu_port			2152	GTP-U UDP 側の 接続
echo_enabled			true	GTPv2-C 側の PGW
apn			internet	PDN 接続
pdn_type			:ipv4	PDN 接続 :ipv4 :ipv6 :ipv4v6
mcc			IMSI 3 00	IE 接続

項目	項目名	項目値	項目説明
mnc	項目名	項目値	IMSI 項目 4-5 項目 項目 IE 項目項目項目

項目 TWAN 項目

項目項目 omnitwag 項目項目項目項目項目 gtp 項目項目

項目	項目名	項目値	項目説明
emergency_apn	項目項目	sos	項目項目 APN項目項目項目APN 項目項目 0
twan ssid	項目項目	TWAG	項目 TWAN 項目 IE 項目 SSID
twan_bssid	項目項目	項目項目項目	TWAN 項目 IE 項目 6 項目 BSSID項目項目項目 6 項目項目項目

項目GTP-C 項目項目項目 PGW項目項目 APN 項目 PGW 項目項目 PGW項目項目 pgw_ip

□□□

GTPv2-C □□□□□**S2a**□

□□	□□	□□
Echo Request / Response	□□	□□□□□
Create Session Request / Response	TWAG □ PGW	□□ PDN □□□
Modify Bearer Request / Response	TWAG □ PGW	□□□□ F-TEID □ QoS□
Delete Session Request / Response	TWAG □ PGW	□□ PDN □□□
Create Bearer Request / Response	PGW □ TWAG	□□□□□□ TWAG □□□
Update Bearer Request / Response	PGW □ TWAG	□□ QoS□□ TWAG □□□
Modify Bearer Command	PGW □ TWAG	PGW □□ TWAG □□□□□ □□
Modify Bearer Failure Indication	TWAG □ PGW	□□□□□□□□□□□□□□□□
Delete Bearer Request / Response	PGW □ TWAG	PGW □□□□□□□

GTPv2-C □□□□

TWAG □□□□□□□□□□□□□□□□ **TS 29.274 §8.4**□

이벤트	설명
<code>[:twag, :bearer, :modify_command]</code>	TWAG에서 PGW로 명령을 보냅니다.
<code>[:twag, :bearer, :update]</code>	TWAG에서 PGW로 명령을 보냅니다.
<code>[:twag, :detach, :pgw_delete]</code>	TWAG에서 PGW로 명령을 보냅니다.
<code>[:twag, :session, :teardown]</code>	세션 종료
<code>[:twag, :detach, :cleanup_start]</code>	세션 종료
<code>[:twag, :detach, :cleanup_complete]</code>	세션 종료

연결

GTP-C 프로토콜을 사용하여 S2a 인터페이스를 연결합니다.

이벤트	설명
<code>connected</code>	GTP-C UDP 연결이 성공했습니다.
<code>pgw_ip / pgw_port</code>	PGW의 IP 주소와 포트 번호
<code>pending_requests</code>	GTP-C 프로토콜을 사용하여 보낸 요청의 수
<code>path_up</code>	PGW에 대한 경로가 설정되었습니다.
<code>echo_failures</code>	에코 실패 횟수
<code>recovery_counter</code>	TWAG의 복구 카운터

연결

GTP-C 프로토콜을 사용하여 S2a 인터페이스를 연결합니다.

- 000000000000000000000000
- 00000000 Path to PGW ... is UP 0 Path to PGW ... is DOWN 0
- PGW 00000000000000000000
- PGW 00000000000000000000

0000

00000000

00000000000000 GTP Create Session failed 0

0000000

- PGW 000000000000
- 0 PGW 0000 DOWN 0
- APN 00 PGW 0000
- IMSI 00000000

000000

1. 000000000000 0000 00
2. 00000000000000 (path_up) 0
3. 00 APN 0 PGW 000000
4. 000000 HSS 000000

0 **PGW 0000 DOWN**

00000000 Path to PGW ... is DOWN 00000000

0000000

- PGW 0 UDP 2123 000000
- 000000 GTP-C 0
- PGW 0000

000000

1. 設定 `pgw_ip` と `pgw_port` を指定
2. 宛先ポート番号 UDP 2123 を指定
3. 設定 PGW GTP-C を有効にする
4. 設定を適用して PGW を再起動

PGW の再起動

PGW の再起動が検出された場合 `PGW restart detected`

確認事項

- PGW の再起動が完了しているか

確認事項

1. 設定 `twag` と PGW の再起動が完了しているか
2. 設定 PGW の再起動が完了しているか

PGW の再起動

設定 UE の IP を指定

確認事項

- GTP-U と GRE を有効にする

確認事項

1. 設定 `gtp.enabled` と `gre.enabled` を `true` に設定
2. 設定 `gtp.enabled` と `gre.enabled` を `true` に設定

3GPP

TS	TS
TS 29.274	TS GPRS TS (GTPv2-C)
TS 23.402	TS 3GPP TS (TWAN, S2a)
TS 24.302	TS 3GPP TS EPC

TS



OmniTWAG は、OmniTWAG IMSI を OmniTWAG API から取得する。

```
000000000000OmniTWAG 00 Acct-Session-Id 0 RADIUS 0000000000000000  
000000000000000000000000000000000000
```



- 関数
- 変数宣言と変数
- 条件分岐
- 繰り返し処理
- REST API
- 辞書
- JSON 辞書
- Web 辞書
- 辞書



RADIUS Access-Request EAP OmniTWAG
 OmniTWAG

0000000000000000 MAC 0000 UE IP 000EAP 00000000 GTP 0 DNS 0000 RADIUS
 000000000000OmniTWAG 000000000000

☐ EAP
 ☐ RADIUS
 ☐ Accept
 ☐ GTP
 ☐ DHCP

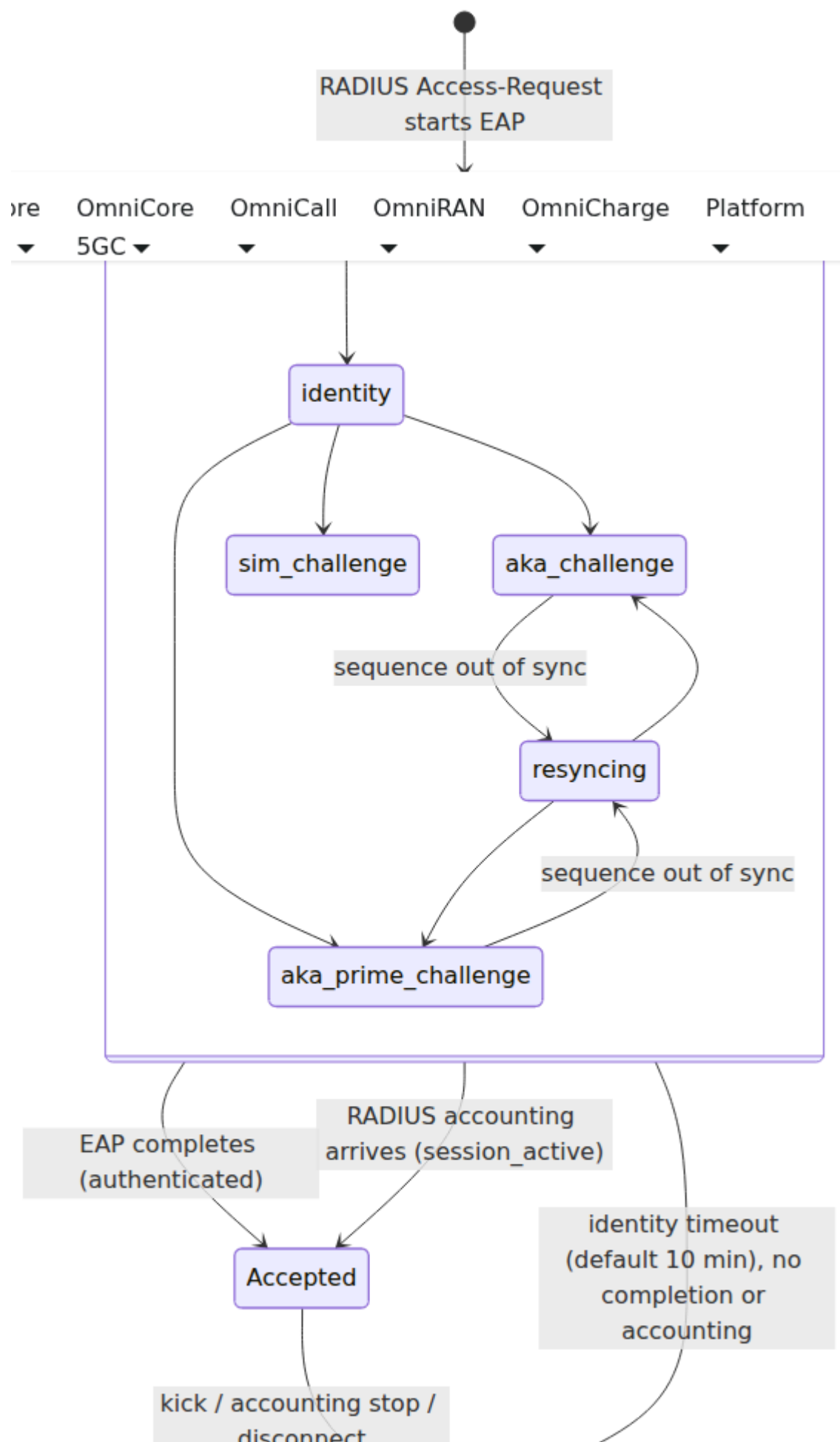
設定

設定項目 EAP (eap_phase)

- eap_phase 設定項目 identity、aka_challenge、aka_prime_challenge、sim_challenge、resyncing
- session_active 設定項目

RADIUS 設定項目 OmniTWAG 設定項目

設定項目 OmniTWAG 設定項目
設定項目 10 項目



disconnect



--

GTP

IP

項目	単位	説明
identity	文字列	文字列形式 IMSI 文字列形式OmniTWAG 文字列形式 文字列
imsi	文字列	文字列 IMSI文字列
mac_address	文字列 文字列	UE MAC 文字列
ue_ip	文字列	文字列 UE IP 文字列形式文字列形式文字列形式
nas_ip	文字列	AP 文字列 IP文字列RADIUS NAS-IP-Address 文字列
called_station_id	文字列	文字列形式文字列RADIUS Called-Station-Id 文字列
calling_station_id	文字列	文字列 MAC文字列RADIUS Calling-Station-Id 文字列
acct_session_id	文字列	RADIUS 文字列 ID文字列OmniTWAG 文字列形式文字列形式 文字列
eap_phase	文字列	文字列 EAP 文字列形式文字列形式文字列形式文字列形式 文字列形式文字列形式
eap_type	文字列	文字列 EAP 文字列形式 aka 文字列 aka_prime 文字列
pgw_ip	文字列	PGW IP 文字列形式文字列形式文字列形式文字列形式
bearer_id	文字列	EPS 文字列 ID文字列
dns_servers	文字列形式	文字列 UE 文字列 DNS 文字列形式
created_at	文字列	文字列形式文字列形式 Unix 文字列形式文字列形式
last_activity	文字列	文字列形式文字列形式 Unix 文字列形式文字列形式
established_at	文字列	文字列形式文字列形式 Unix 文字列形式文字列形式

項目	単位	説明
acct_input_octets	バイト	この UE が RADIUS サーバに送信したバイト数
acct_output_octets	バイト	この UE から RADIUS サーバに送信したバイト数
acct_input_packets	パケット	この UE が RADIUS サーバに送信したパケット数
acct_output_packets	パケット	この UE から RADIUS サーバに送信したパケット数
acct_session_time	秒	RADIUS サーバとのセッション時間
acct_status_type	整数	1: 正常終了 2: 異常終了 3: 保留
acct_last_update	秒	Unix 時刻で最後の更新日時

acct_* 項目は RADIUS サーバから取得されます

その他の項目

OmniTWAG の eap_phase 項目は RADIUS サーバから取得されます

eap_phase	説明	データ型
identity	ID	文字列
aka_challenge	AKA 挑戦	文字列
aka_prime_challenge	AKA' 挑戦	文字列
sim_challenge	SIM 挑戦	文字列
resyncing	同期中	文字列
authenticated	認証済み	文字列
session_active	セッション有効	文字列
accepted	承認済み	文字列
rejected	拒否済み	文字列

説明 eap_phase データ型

REST API

OmniTWAG REST API 詳細 API 詳細 8444 詳細 HTTPS 詳細?? 詳細
/sessions

HTTP Method	URI	Action	Description
GET	/sessions	index	Return a list of sessions
GET	/sessions/{id}	show	Return the session with the given <code>identity</code> IMSI
DELETE	/sessions/{id}	delete	Delete the session with the given <code>identity</code> IMSI
PATCH	/sessions/{id}	action	Update the session with the given <code>identity</code> IMSI with the given JSON body
PUT	/sessions/{id}	action	Create the session with the given <code>identity</code> IMSI with the given JSON body

`{id}` is the `identity` IMSI

Example

Example

Parameter	Type	Description
<code>offset</code>	int	Offset of the first session to return
<code>limit</code>	int	Limit of the number of sessions to return
<code>q</code>	string	Query string for OmniTWAG

`q` is the query string

項目名	説明
identity	ユーザのIMSI
imsi	IMSI
nas_ip	AP の IP アドレス
called_station_id	被呼番号
calling_station_id	呼出番号
mac_address	UE の MAC アドレス
ue_ip	UE の IP アドレス

動作概要

PATCH 又は PUT による JSON 送信

項目名	項目	項目	説明
action	kick start_accounting stop_accounting		

動作概要

00

00	00
200 OK	000000000000000000000000
404 Not Found	000000 identity 0000
422 Unprocessable Entity	0000 kick start_accounting stop_accounting 000

0000000000 status 000 identity 00000000 {"status": "deleted", "identity": ...}

000000

000000000000000000000000 /sessions/{id} PATCH PUT 000 JSON 0000

00	00	00 status
kick	000000 RADIUS CoA-Disconnect 000000	kicked
start_accounting	00000000000000000000000000000000	accounting_started
stop_accounting	000000000000 Accounting-Stop 0000000000	accounting_stopped

kick stop_accounting 0000000000 kick 0000000000000000 stop_accounting 0000000000000000

JSON

00000000

000

```
GET /sessions/0010100000000001
```

000

```
{
  "identity": "0010100000000001",
  "imsi": "0010100000000001",
  "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
}
```

00000000

000

```
PATCH /sessions/0010100000000001
```

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

200 OK

```
{  
  "status": "kicked",  
  "identity": "0010100000000001"  
}
```

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

422 Unprocessable Entity

```
{  
  "error": "Unknown action \"reboot\" (expected  
kick|start_accounting|stop_accounting)",  
  "identity": "0010100000000001"  
}
```

Web

RADIUS is a protocol for authentication, authorization, and accounting (AAA) for network access. It is used by network devices to communicate with a central server. The server can be a dedicated RADIUS server or a network access server (NAS) that handles the requests.

環境構築

環境構築 "環境構築"

環境構築環境構築環境構築環境構築 EAP 環境構築 identity、aka_challenge、aka_prime_challenge、sim_challenge、resyncing

環境構築

- 環境構築 EAP 環境構築環境構築
- 環境構築 EAP 環境構築
- 環境構築環境構築環境構築

環境構築

1. 環境構築環境構築環境構築OmniTWAG 環境構築
2. 環境構築環境構築環境構築環境構築OmniTWAG 環境構築環境構築環境構築 10 環境構築
3. 環境構築環境構築環境構築環境構築 RADIUS 環境構築環境構築環境構築環境構築環境構築

環境構築環境構築 kick 環境構築 /sessions/{id} 環境構築 DELETE 環境構築

環境構築環境構築

環境構築 環境構築 acct_* 環境構築環境構築

環境構築

- 環境構築 RADIUS 環境構築
- 環境構築 Acct-Session-Id 環境構築

環境構築

1. 環境構築 OmniTWAG 環境構築 RADIUS 環境構築
2. 環境構築 Acct-Session-Id 環境構築 acct_session_id 環境構築OmniTWAG 環境構築環境構築
3. 環境構築環境構築環境構築環境構築環境構築

環境構築

Network Architecture

Overview

OmniTWAG connects to the core network and provides services to UE. OmniTWAG connects to UE via GTP-U and to the core network via GTP-C.

The core network consists of GTP-C and S2a / GTP-C interfaces.

Interfaces

- S2a
- GTP-U
- GRE
- GTP-C
- S2b
- S2c
- S2d
- MTU
- S2e
- S2f

Network Elements

Network elements include:

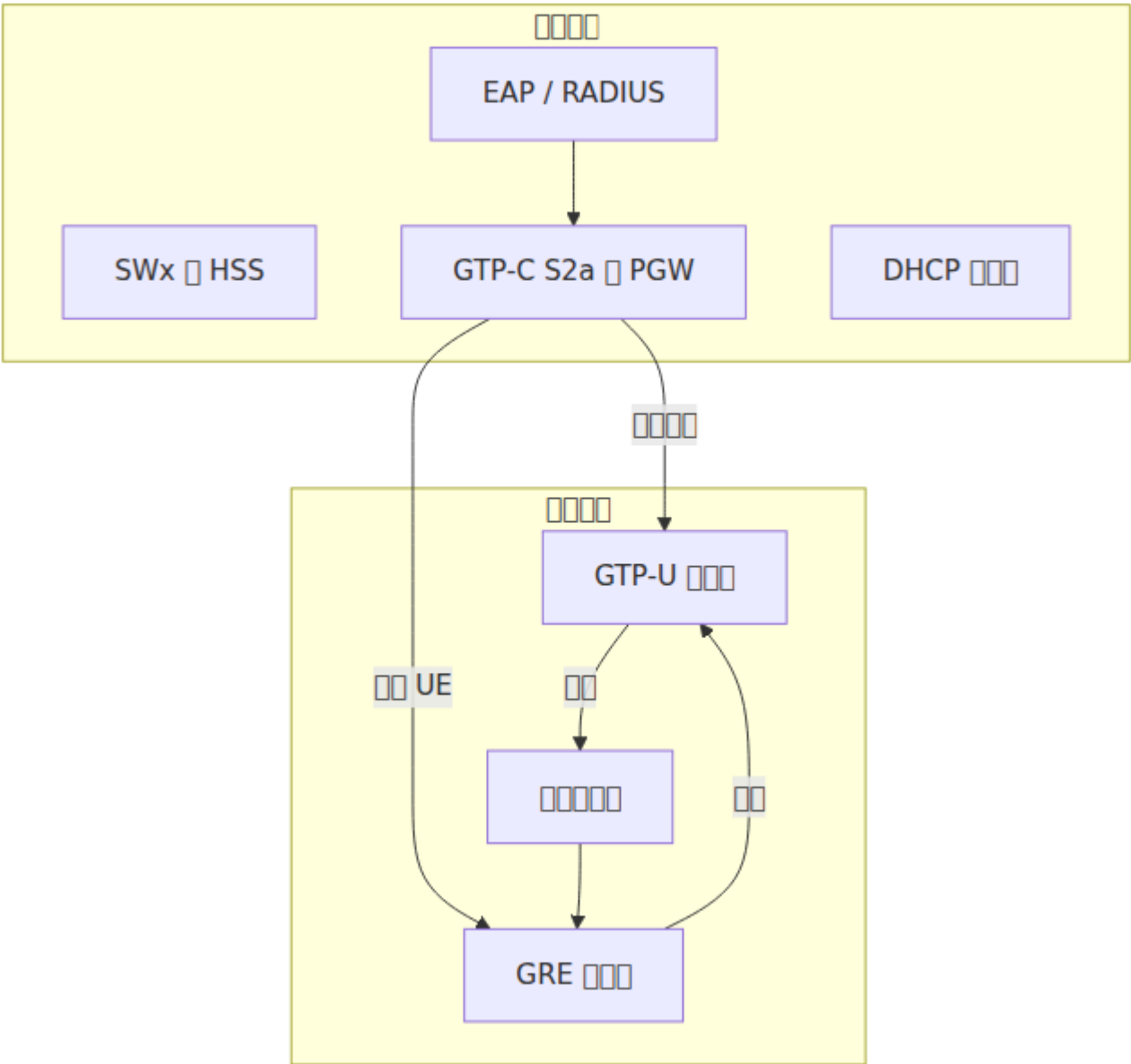
Core network elements:

- RADIUS / EAP
- Diameter SWx / HSS
- S2a / PGW / GTP-C / PDN
- UE / DHCP / IP

Access network elements:

- 3GPP TS 29.281 GRE tunnel
- S2a-U interface PGW and GTP-U interface 3GPP TS 29.281
- UE interface
- Network architecture

Network architecture for GTP-C and GTP-U interfaces. The diagram shows the interaction between the Core Network (CN) and the User Equipment (UE) through the GTP-C and GTP-U interfaces. The Core Network (CN) includes the EAP / RADIUS interface, the GTP-C S2a interface, and the DHCP interface. The User Equipment (UE) includes the GTP-U interface, the UE interface, and the GRE interface.



Network architecture

OmniTWAG interface AP interface RADIUS interface breakout_type interface
 interface breakout_type address_source ip_pool_id interface RADIUS interface

breakout_type	UE IP	TWAG	
tunneled	PGW		AP TWAG GRE S2a PGW GTP-U
nswo	TWAG IP		AP TWAG GRE TWAG NAT
local	AP		AP TWAG IP

GRE tunneled nswo AP UE GRE TWAG TWAG GRE AP GRE

- tunneled TWAG GTP-U PGW PGW UE IP
- nswo TWAG NAT TWAG IP UE IP PGW GTP-C TWAG Gy

local TWAG IP AP GRE GTP-U

GTP-U PGW

OmniTWAG UE PGW PGW UE IP 3GPP TS 23.402 16 S2a

UE --WiFi--> AP --GRE--> TWAG --GTP-U S2a--> PGW --> Internet

Internet --> PGW --GTP-U S2a--> TWAG --GRE--> AP --WiFi--> UE

gtp.enabled true gre.enabled true AP breakout_type tunneled

NSWO

WLAN NSWO UE TWAG PGW TWAG UE IP NAT 3GPP TS 23.402 16.2

OmniTWAG NSWO

1. AP breakout_type nswo UE ip_pool_id IP address_source
 - pool UE GRE
 - dhcp UE MAC TWAG DHCP DHCP DISCOVER GRE
2. NSWO Twag.NSW0.Handler GTPnswo.enabled: true gtp.enabled: false IP iptables NAT TWAG DHCP UE UE GRE PGW / GTP-C

NAT GRE IP IP

NSWO nswo

enabled		false	NSWO
local_pool	CIDR	10.100.0.0/16	UE
nat_interface		eth0	NAT
dns_servers		["8.8.8.8", "8.8.4.4"]	UE DNS PGW PCO

GRE 配置

GRE 配置 Twag.Gre.Server 配置 Rfc 2784 Rfc 2890 UE
GRE 配置 TWAG GRE 配置 DHCP 配置 GRE 配置 tunneled 配置 nsw
配置 AP 配置

GRE 配置 IP 配置 47 GRE 配置 :socket.open/3 配置
CAP_NET_RAW 配置 10 配置 GRE 配置
gre.enabled 配置 true

GRE 配置

GRE 配置 IP 配置 GRE 配置 GRE 配置

GRE 配置	配置	配置
0x6558	配置	配置
0x0800	IPv4	配置 IPv4 配置

配置

配置	配置	配置
0x0800	IPv4	配置 DHCP 配置 DHCP 配置 ? ? 配置
0x0806	ARP	配置
0x86DD	IPv6	配置

配置 UE MAC 配置 IP 配置 MAC 配置

IP

UE IP address is GRE tunnel IP address. UE IP address is GRE tunnel IP address. TEID is GTP-U tunnel ID. TEID is GTP-U tunnel ID. GRE is GTP-U tunnel.

UE IP address is GTP-U tunnel IP address. 20 IP address.

nswo IP address NAT GRE tunnel UE IP address.

UE

UE IP address GRE tunnel UE IP address UE MAC address IP address GRE tunnel UE IP address.

GTP-U

GTP-U tunnel Twag.Gtp.GtpU PGW tunnel 2152 UDP tunnel 3GPP TS 29.281 GTPv1-U.

TEID

TEID tunnel TEID tunnel PGW TEID TWAG tunnel PGW tunnel TEID PGW tunnel TWAG tunnel TEID TWAG tunnel TEID tunnel.

PGW tunnel GTP-U tunnel TEID TWAG tunnel TEID tunnel 32 3GPP TS 29.274 TEID tunnel.

項目	説明
local_teid	TWAG 側 TEID PGW 側から送信される
remote_teid	PGW 側 TEID TWAG 側から送信される
remote_ip	PGW GTP-U IP アドレス
remote_port	PGW GTP-U UDP ポート番号 2152
session_id	セッション ID

動作概要

UE 側から GTP-U 送信される GTPv1-U G-PDU の IP アドレスとポート番号を GTP 送信元アドレスと GTP 送信元ポート番号として、255.255.255.255 TEID の UDP 送信元として PGW IP アドレスと GRE ポート番号を GTP-U 送信元として送信する。

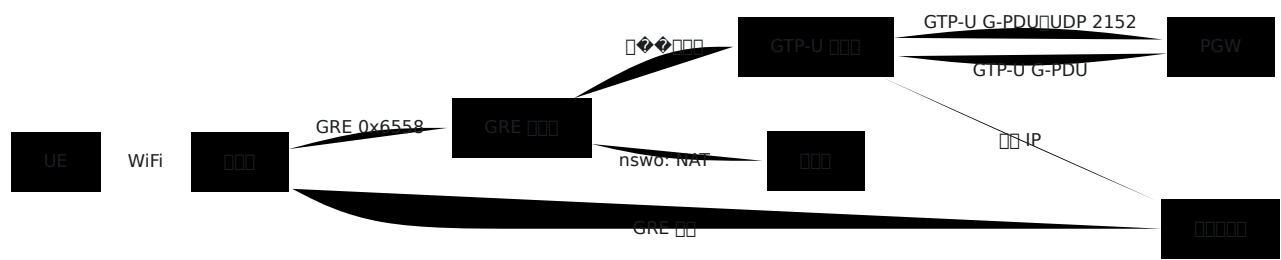
送信データ

GTP-U 送信元として GTP-U 送信元アドレスと GTP-U 送信元ポート番号を送信する。

項目	値	説明
G-PDU	255	UE 側 TEID PGW 側から送信される IP アドレス
Echo Request	1	UE 側 Echo Response
Echo Response	2	PGW 側から
Error Indication	26	PGW 側から
End Marker	254	PGW 側から

G-PDU の GTP-U 送信元として N-PDU の送信元アドレスと IP アドレス Twag.Downlink のポート番号を送信する。

UE G-PDU TEID GTP-U



Twag

Twag.Downlink GTP-U IP UE GRE

GRE

GRE GRE IP UE IP GRE GRE UE UE IP GRE IP WiFi UE

GRE UE MAC MAC MAC 02:00:00:00:00:01 MAC

GRE

GRE GRE IP UE

IPv4 IPv6 IP_HDRINCL IPv6 IPV6_HDRINCL IP

CAP_NET_RAW GRE

prefer_gre

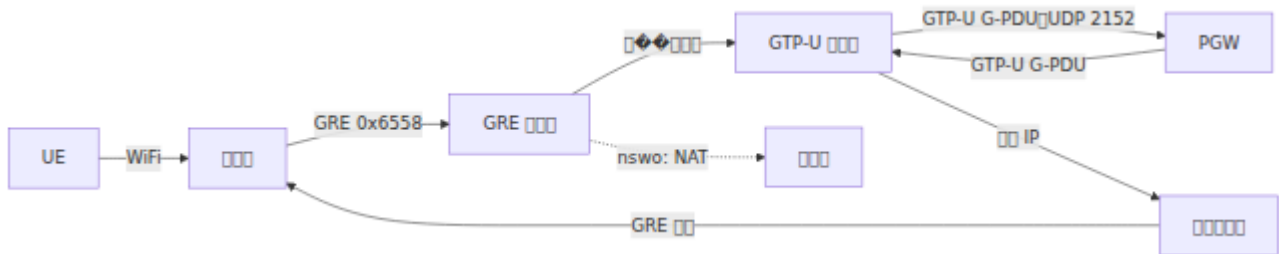
prefer_gre

- `prefer_gre: true` GRE tunnel GRE tunnel GRE tunnel
- `prefer_gre: false` GRE tunnel

Network Diagram

Network topology diagram:

- UE
- GRE tunnel
- 20 tunnels
- IP



Network Configuration

Network configuration details: `Twag.Bearer.Manager` EPS network EBI network GTP-U TEID TFT QoS QCI 3GPP TS 23.402 16.1.6.1

Table

Field	TFT	Value
Field	Value	Value
Field	Value	Value TFT

Network configuration details: 16.1.6.1

TFT 設定

TFT は 3GPP TS 24.008 の 10.5.6.12 で定義されています。TFT は、パケットの送信先を決定するために使用されます。TFT は、パケットの送信先を決定するために使用されます。

項目	説明
IPv4 アドレス	IPv4 アドレス
IPv4 アドレス	IPv4 アドレス
IPv6 アドレス/アドレス範囲	IPv6 アドレス
IP アドレス	IP アドレス
パケット / アドレス	UE アドレス
パケット / アドレス	パケット
パケット / アドレス	TOS アドレス
パケット	IP TWAN アドレス
パケット	パケット IPv6 アドレス

パケット TFT は、パケットの送信先を決定するために使用されます。PGW は、パケットの送信先を決定するために使用されます。

設定

パケットの送信先を決定するために使用されます。

- パケットの送信先を決定するために使用されます。
- パケットの送信先を決定するために使用されます。
- パケットの送信先を決定するために使用されます。
- パケットの送信先を決定するために使用されます。
- パケットの送信先を決定するために使用されます。

配置 GTP-U 的隧道

PGW 和 TWAG 通过 TSCM 配置 GRE 隧道，TEID 由 PGW 和 TWAG 协商，S2a / GTP-C 隧道

配置

GTP-U Echo

PGW 和 TWAG 通过 GTP-U Echo Request 和 Echo Response 消息，GTP-U 隧道 Echo Request 和 Echo Response 消息，gtp.echo_enabled 为 true，GTP-C S2a 隧道 S2a / GTP-C

GTP-U 隧道

PGW 和 TWAG 通过 G-PDU 消息，TEID 由 PGW 和 TWAG 协商，GTP-U 隧道 TEID 由 TWAG 和 PGW 协商，PGW 和 TWAG 协商

GTP-U 隧道

GTP-U 隧道配置

配置

PGW 和 TWAG 通过 GTP-U 隧道，GRE 隧道，GRE 隧道配置

MTU 配置

GRE 隧道配置 GRE 隧道 IPv4 GRE 隧道 47

GTP-U 隧道 GRE 隧道 UE 隧道 TCP MSS 隧道 MTU 隧道 Cisco 隧道 GRE 隧道 mss 1410 隧道 mtu 1500 隧道



omnitwag gtp gre downlink AP RADIUS RADIUS

```
config :omnitwag,
  # GTP-U  GTP-C  PGW-S2a - 
  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 
  gre: %{
    enabled: false,
    local_ip: "10.5.198.200"
  },

  # 
  downlink: %{
    enabled: true,
    ip_hdr_incl: true,
    prefer_gre: true
  }
```

GTP

項目	型別	デフォルト値	説明
<code>enabled</code>	ブール値	<code>false</code>	GTP-C と GTP-U を有効にするかどうか 有効にする場合は <code>true</code>
<code>pgw_ip</code>	IP アドレス	<code>"10.5.198.1"</code>	GTP-C と GTP-U の PGW IP アドレス
<code>pgw_port</code>	ポート番号	<code>2123</code>	PGW GTP-C UDP ポート番号
<code>local_ip</code>	IP アドレス	<code>"0.0.0.0"</code>	GTP と TWAG の IP アドレス GTP-U の 宛先 IP アドレス
<code>gtpu_port</code>	ポート番号	<code>2152</code>	GTP-U の宛先 UDP ポート番号 GTP-U のポート番号
<code>echo_enabled</code>	ブール値	<code>true</code>	PGW の GTP-C 宛てに S2a / GTP-C を送信するかどうか

GRE

項目	型別	デフォルト値	説明
<code>enabled</code>	ブール値	<code>false</code>	GRE を有効にするかどうか <code>true</code> の場合は GRE を有効にする UE の GRE 宛てに <code>tunneled</code> と <code>nsw</code> を送信する
<code>local_ip</code>	IP アドレス	<code>"0.0.0.0"</code>	GRE の宛先 IP アドレス GRE の宛先 IP アドレス

配置

項目	デフォルト値	有効	説明
<code>enabled</code>	<code>true</code>	<code>true</code>	GRE を有効にするかどうか。デフォルトは <code>true</code> 。false に設定すると、GRE は無効になります。
<code>ip_hdr_incl</code>	<code>true</code>	<code>true</code>	IP ヘッダーを含めるかどうか。デフォルトは <code>true</code> 。false に設定すると、IP ヘッダーは含まれません。IP ヘッダーを含めるには、 <code>IP_HDRINCL</code> を <code>true</code> に設定する必要があります。
<code>prefer_gre</code>	<code>true</code>	<code>true</code>	GRE を優先するかどうか。デフォルトは <code>true</code> 。false に設定すると、GRE は優先されず、代わりに IPsec を使用します。

インストール

GRE をインストールするには、`CAP_NET_RAW` をインストールする必要があります。GRE は 10 のポート番号を必要とします。

デバッグ

GTP-U をデバッグするには、GRE をインストールする必要があります。OmniTWAG は Prometheus をサポートしています。

GTP-U

GTP-U は `socket_open`、`local_ip`、`tunnel_count`、`stats` をサポートしています。

項目名	単位
packets_in	1 GTP-U パケット受信回数
packets_out	1 GTP-U パケット送信回数
bytes_in	バイト受信量
bytes_out	バイト送信量
errors	エラー発生回数
tunnel_count	トンネル数

GRE 項目

GRE 項目 running socket_open local_ip ap_tunnels registered_ues
stats

項目名	説明
packets_in	受信 GRE パケット数
packets_out	送信 GRE パケット数
bytes_in / bytes_out	送信/受信バイト数
dhcp_in	受信 DHCP パケット数
uplink_forwarded	送信 GTP-U パケット数
downlink_forwarded	受信 GTP-U パケット数
unknown_protocol	未知プロトコルパケット数
errors	エラーパケット数
registered_ues	登録 UE 数

配置

項目	説明
forwarded	IPv4 の GRE を IPv4 に転送する
forwarded_v6	IPv6 を転送する
dropped	パケットをドロップする
errors	エラーを報告する
bytes	バイト数を報告する
enabled	有効にする
socket_open	IPv4 のソケットを開く

