

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
- 無線 LAN 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

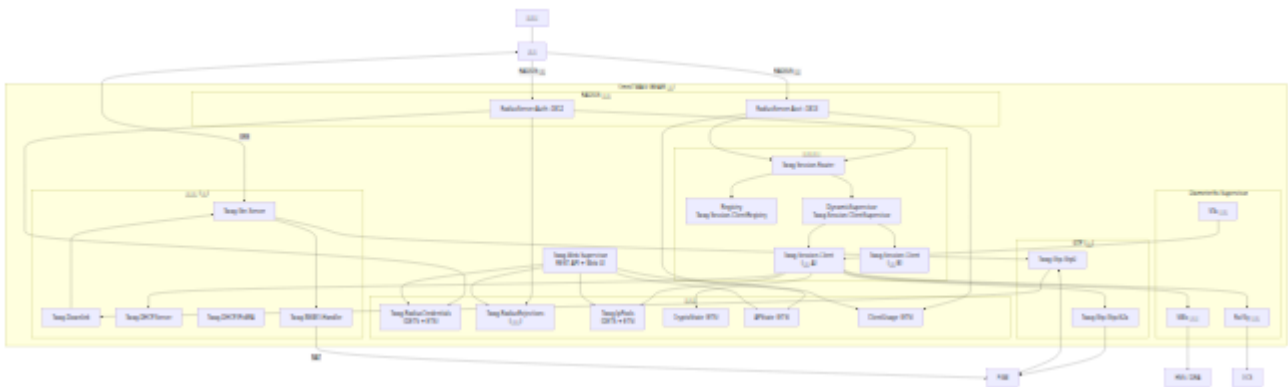
アーキテクチャ

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) 接続
- 無線 LAN 接続 Twag.Session.Client GenServer 接続 Registry 接続 DynamicSupervisor 接続 Twag.Session.Router 接続

- `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	デフォルト値
RadiusServer.Auth	GenServer	true
RadiusServer.Acct	GenServer	true
CryptoState	GenServer (ETS)	true
APState	GenServer (ETS)	true
ClientUsage	GenServer (ETS)	true
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	型	デフォルト値
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 をインストールする。gtp / nsw0 をインストールする。

インストール (GTP 環境)

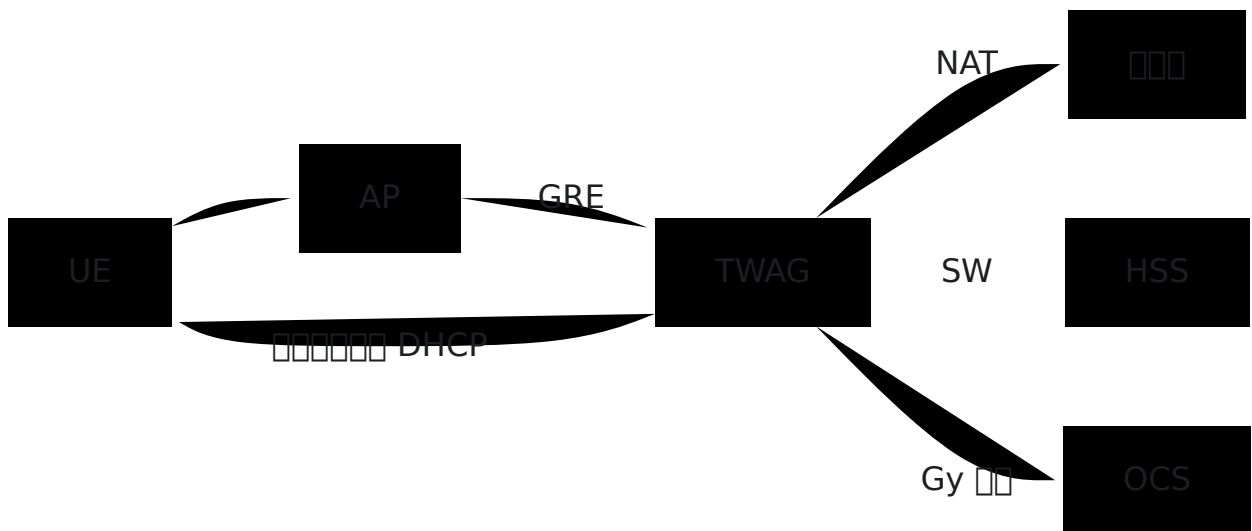
インストール GTP-C/GTP-U S2a をインストールする。PGW PGW を UE IP をインストールする/インストールする。



- **UE IP** PGW をインストールする (PAA IE)
- **IP** TWAG を GRE UE UE DHCP PGW IP
- AP 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 をインストールする PGW TWAG NAT/インストール Gy をインストールする



- **UE IP** TWAG IP (Twag.IpPools) IP
- **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)

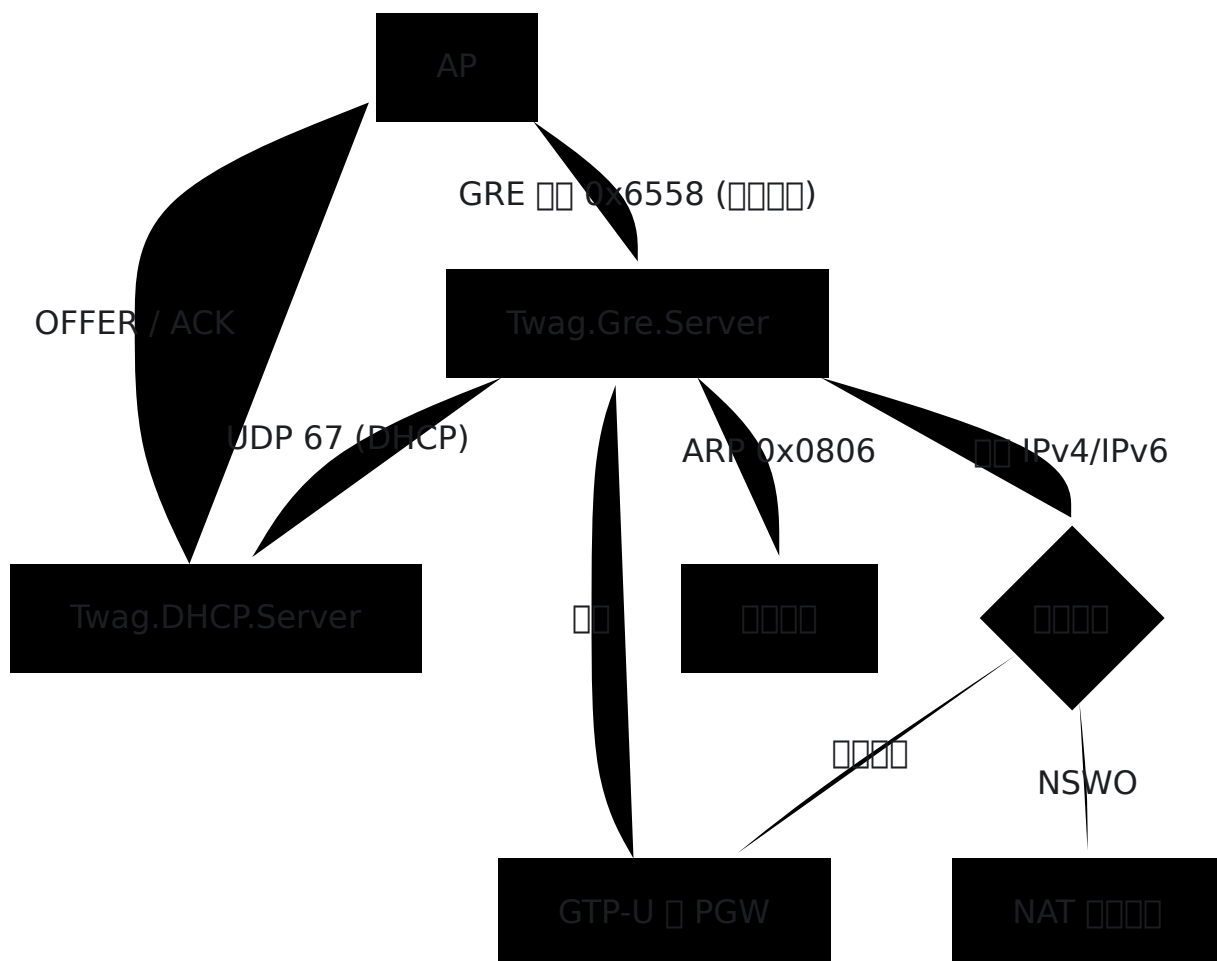
AP GRE IP 47 UE L2 TWAG TS 23.402 16.1.2

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

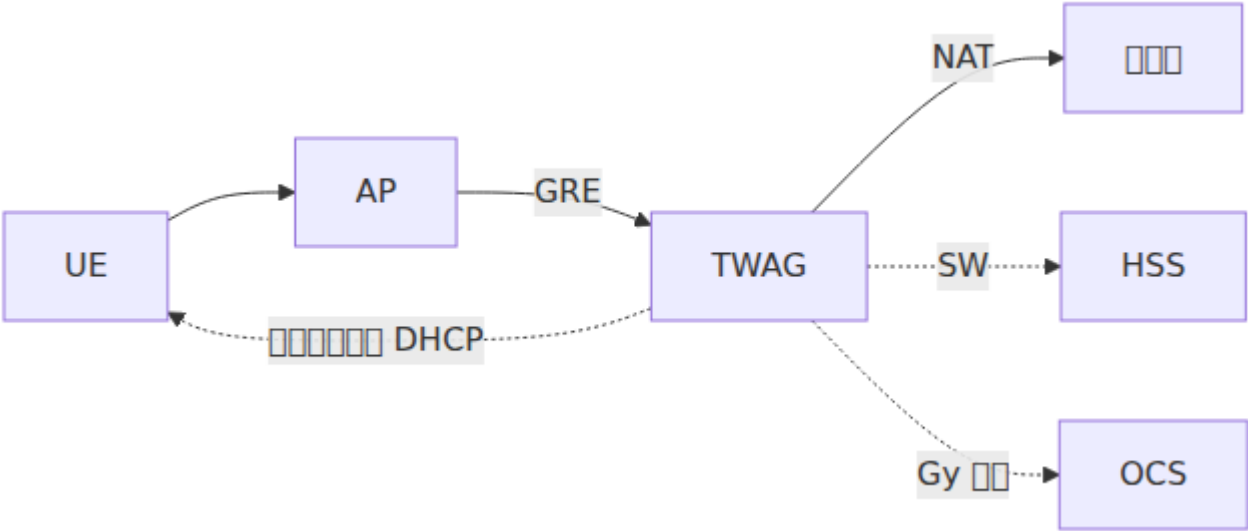
	IP	DHCP	
	PGW (PAA)	TWAG PGW IP GRE	PGW PCO DNS
	TWAG (Twag.IpPools)	TWAG DISCOVER	DNS

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 的服务器
配置了 unauthorized_source 配置了 bad_secret 配置了 RADIUS 服务器

配置

配置	配置	配置	EAP 配置
:swx	auth_mode: :swx	配置 SWx MAR/MAA 与 HSS	配置 (TWAG)
:local	auth_mode: :local	配置 (Milenage)	配置 (TWAG)

EAP 配置

配置 eap_method 配置 :eap_aka 配置 UE 配置 TWAG 配置 UE 配置
配置 EAP-SIM 配置 EAP 配置 (ERP)配置 配置

□□□□□□ (□□□□)

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 1

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 時 HSS からの認証は Milenage 方式
eap_method	:eap_aka	EAP 方式は :eap_aka、:eap_aka_prime
identity_timeout_ms	600000	identity_timeout のデフォルト値
gtp.enabled	true	GTP-C/GTP-U を有効にする
gtp.pgw_ip	127.0.0.1	GTP-C/GTP-U の PGW IP
gtp.local_ip		GTP の TWAG IP
gtp.pgw_port	2123	PGW の GTP-C ポート
gtp.gtpu_port	2152	GTP-U のポート
gtp.echo_enabled	true	PGW の GTP エコーを有効にする
gre.enabled	false	AP の GRE を有効にする
gre.local_ip	0.0.0.0	GRE の宛先 IP
downlink.enabled	true	ダウンリンクを有効にする
dhcp.enabled	false	DHCPv4 を有効にする
dhcp.interface		DHCPv4 のインターフェイス "gre"
ipv6_ra.enabled	false	IPv6 の RA/DHCPv6 を有効にする
nsw.enabled	false	NSWO を有効にする

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 送信: CK' = KDF(CK || IK, FC=0x20, network_name, SQN(+)AK)
          IK' = (KDF 送信)
          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 送信: MK = SHA1(Identity || IK || CK)
          K_encr, K_aut, MSK, EMSK = PRF(MK)
```

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

-
- RADIUS
- RADIUS
- IP
-
- AP
-
-
-

無線LAN (EAP-AKA / EAP-AKA' / EAP-SIM)

概要

OmniTWAG は RADIUS 経由で TWAG 側で 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 側で (MSK) と MPPE による RADIUS Access-Accept による 802.11 認証を行う。

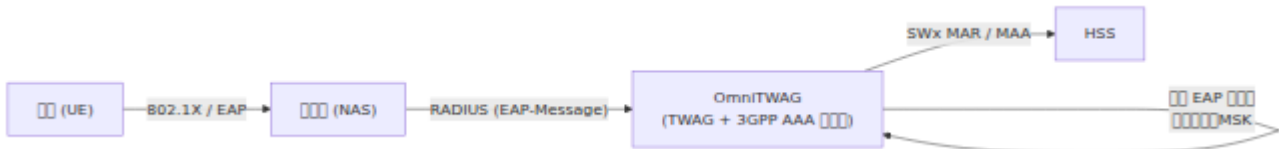
無線LAN TWAG 側で RADIUS 経由で RADIUS 認証を行う無線LAN TWAG 側で S2a GTP-C 認証を行う S2a GTP-C 認証を行う無線LAN TWAG 側で (Gy) 認証を行う。

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

IEEE 802.1X EAP EAP RADIUS EAP-Message TWAGTWAG EAP RADIUS

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



TWAG EAP (XRES)(K_{aut})RANDMSK MSK (EMSK) TWAG MPPE TWAG

EAP

TWAG EAP TLV EAP

Protocol	EAP Code	Reference	Network	K _{aut}	MAC	Key Derivation
EAP-AKA'	50 (0x32)	RFC 5448	UMTS	32 bytes	HMAC-SHA-256 16 bytes	CK'/IK' PRF' (HMAC-SHA-256)
EAP-AKA	23 (0x17)	RFC 4187	UMTS	16 bytes	HMAC-SHA1 16 bytes	MK = SHA1(CK IK) PRF FIPS 186-2 PRF
EAP-SIM	18 (0x12)	RFC 4186	2G/3G GSM	16 bytes	HMAC-SHA1 16 bytes	MK = SHA1(Kc... NONCE_MT) PRF FIPS 186-2 PRF

Configuration

TWAG configuration: `eap_method :eap_aka_prime (EAP-AKA')`

Configuration EAP-NAK (EAP Code 3) NAK configuration EAP TWAG configuration NAK 50 (EAP-AKA') 23 (EAP-AKA) 18 (EAP-SIM)

- NAK configuration TWAG configuration EAP-AKA' EAP-AKA TWAG configuration `AT_PERMANENT_ID_REQ` AKA-Identity configuration EAP-SIM TWAG configuration EAP-SIM/Start configuration
- NAK configuration TWAG configuration

EAP Configuration

TWAG configuration EAP TWAG configuration EAP configuration

阶段 (eap_phase)	描述
identity	客户端向TWAG发送 EAP-Identity请求，TWAG返回 AKA-Identity 响应。EAP-SIM/Start 阶段后，客户端发送 identity 请求。
aka_challenge	TWAG 向客户端发送 EAP-AKA 响应，包含 AKA-Challenge 数据。
aka_prime_challenge	TWAG 向客户端发送 EAP-AKA' 响应，包含 AKA-Challenge 数据。
sim_challenge	TWAG 向客户端发送 EAP-SIM 响应，包含 SIM/Challenge 数据。
authenticated	EAP 认证成功，TWAG 向客户端发送 MSK 数据，S2a 建立。
session_active	S2a 建立成功，客户端向 RADIUS 服务器发送认证请求。

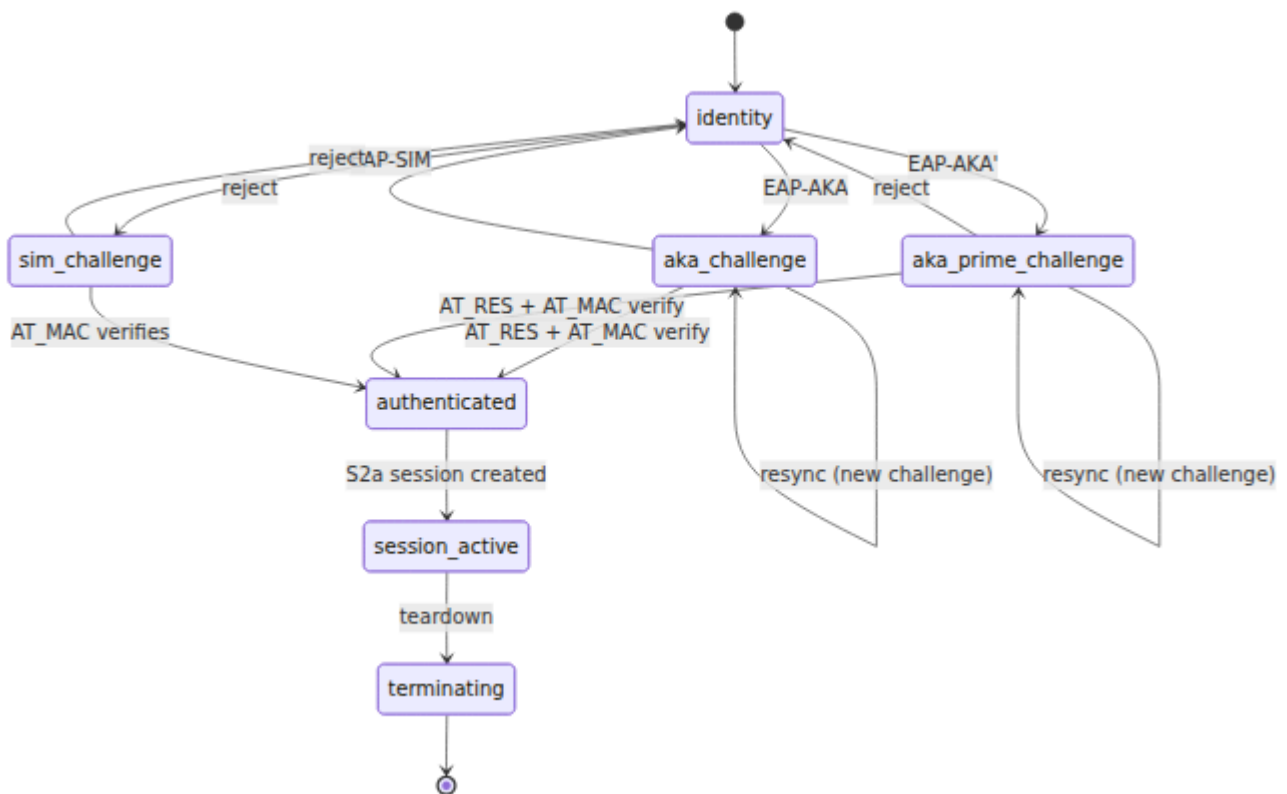
认证失败处理

- resyncing 客户端向TWAG发送 EAP-Resync 请求，TWAG返回 AKA-Challenge 或 AKA-Prime-Challenge 响应，客户端发送 resyncing 请求。
- terminating 认证失败，客户端向TWAG发送 EAP-Terminate 请求。

认证失败处理逻辑：identity 失败时，客户端发送 aka_challenge 或 aka_prime_challenge 请求，TWAG 返回 resyncing 响应，客户端发送 identity_timeout_ms 请求，TWAG 返回 200 响应。

认证失败处理 UI 与 API 交互逻辑





□□□□

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 00000000 @ 00000000
2. TWAG 0000000000000000 00000000EAP-AKA/AKA'0000 6 0000 IMSI0000000000
IMSI0

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. $\text{MK} = \text{PRF}'(\text{IK}' \parallel \text{CK}', \text{"EAP-AKA'"} \parallel \square\square)$ □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. $\text{MK} = \text{SHA1}(\square\square \parallel \text{IK} \parallel \text{CK})$ □
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. $\text{MK} = \text{SHA1}(\square\square \parallel \text{Kc1} \parallel \text{Kc2} \parallel [\text{Kc3}] \parallel \text{NONCE_MT} \parallel \square\square\square\square \parallel \square\square\square\square)$ □
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 (RFC 2548 RFC 3748) を

- 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 Access-Accept から TWAG RADIUS を

MPPE MSK MSK 802.11

Access-Accept が 64 ビット MSK MSK TWAG 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

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

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

拡張認証プロトコル (ERP)

TWAG RFC 6696 EAP (ERP) ERP EAP TWAG EMSK

TWAG

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

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 ネットワークの Result-Code (2001) を返す

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,
  # 配置:swx (本地) :local (本地)
  auth_mode: :swx,

  # EAP 配置:eap_aka_prime (本地):eap_aka :eap_sim
  eap_method: :eap_aka_prime,

  # EAP-AKA' 配置Diameter 配置
  network_name: "WLAN:mnc001.mcc001.3gppnetwork.org",

  # 配置 600000 (10 秒)
  identity_timeout_ms: 600_000,

  # TWAN 配置 STa DER/AAR
  twan_ssid: "ons",
  twan_bssid: "00:11:22:33:44:55",

  # 配置 auth_mode: :local 配置
  test_keys: %{
    ki: "11111111111111111111111111111111",
    opc: "22222222222222222222222222222222",
    amf: "8000",
    sqn: 1200
  }

```

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 からの認証、:local は test_keys から取得した鍵を使用した認証。:swx は HSS からの認証。
eap_method	Atom	○	:eap_aka_prime	EAP 認証方法: :eap_aka_prime は EAP-AKA'、:eap_aka_prime は EAP-AKA。EAP-NA は EAP-None。
network_name	String	○		EAP-AKA' のネットワーク名 (TWAG の TS 23.122 参照)。
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 (TWAN の AAR 参照)。
twan_bssid	String	○	-	BSSID (twan ssid 参照)。
test_keys	Map	○		Milenage 鍵: :local は test_keys から取得した鍵。

設定例

TWAG の :local test_keys から取得した test_keys

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 [][]
```

5555

TWAG identity imsi

--	--	--	--

項目	項目名	
未知ユーザエラー DIAMETER_ERROR_USER_UNKNOWN (5001)	HSS 側 IMSI 側 3GPP/SWx 側 未知ユーザ S6a/EPC 側 3GPP 側	HSS 側 未知ユーザ
3GPP 非サブスクリプションエラー DIAMETER_ERROR_USER_NO_NON_3GPP_SUBSCRIPTION (5450)	3GPP 非サブスクリプション	HSS 側
MAA 側 Result-Code=nil	SWx 側 Experimental-Result AVP 側 Result-Code	Experimental-Result Code
未知ユーザエラー	未知ユーザエラー	未知ユーザ identi
AKA-Authentication-Reject	UE 側 (AUTN 側)	UE 側 :l HS 側
AKA-Synchronization-Failure	SQN 側 SQN	TWAG SQN SQN
Access-Reject 側 EAP-Success 側	MSK 側 64 側	MSK 側 MAA 側

☐ EAP
 ☐ RADIUS
 ☒ RADIUS



OmniTWAG UE WiFi AP TWAG
HSS SWx PGW S2a OCS Gy

00000000000000000000 00S2a GTP-C000000IP000000IP000000RADIUS0000
00



- bbbbbb
- bbbb - bbbb
- bb - NSWbIP
- bb - NSWbDHCPbb
- bb - bbbbAPbbbb
- bbbb - bbbbSWxbb
- bbbbbb
- bbbbbb
- bb - bbbbbb
- bb - bbbbAPIbb
- bb - bbb
- bbbbbb

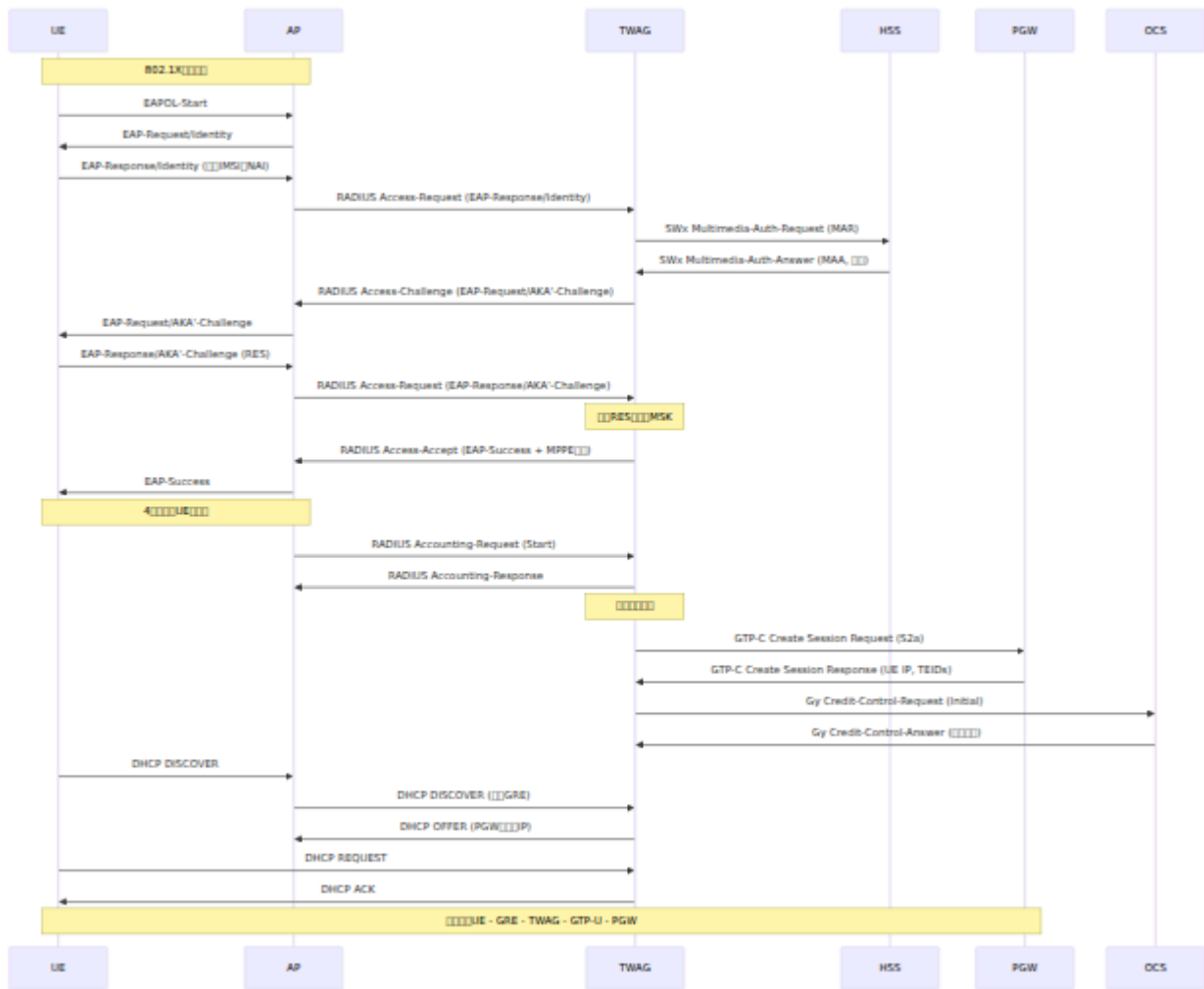
Network Elements

Entity	Interface	Protocol / Notes
UE	Cellular / SIM	802.11 / EAPOL (for WLAN)
AP	WiFi / RADIUS	RADIUS (UDP 1812/1813)
TWAG	WiFi / 3GPP AAA	-
HSS	Core Network	Diameter SWx
PGW	Core Network	GTP-C/GTP-U / S2a
OCS	Core Network	Diameter Gy

Network Elements: AP, RADIUS

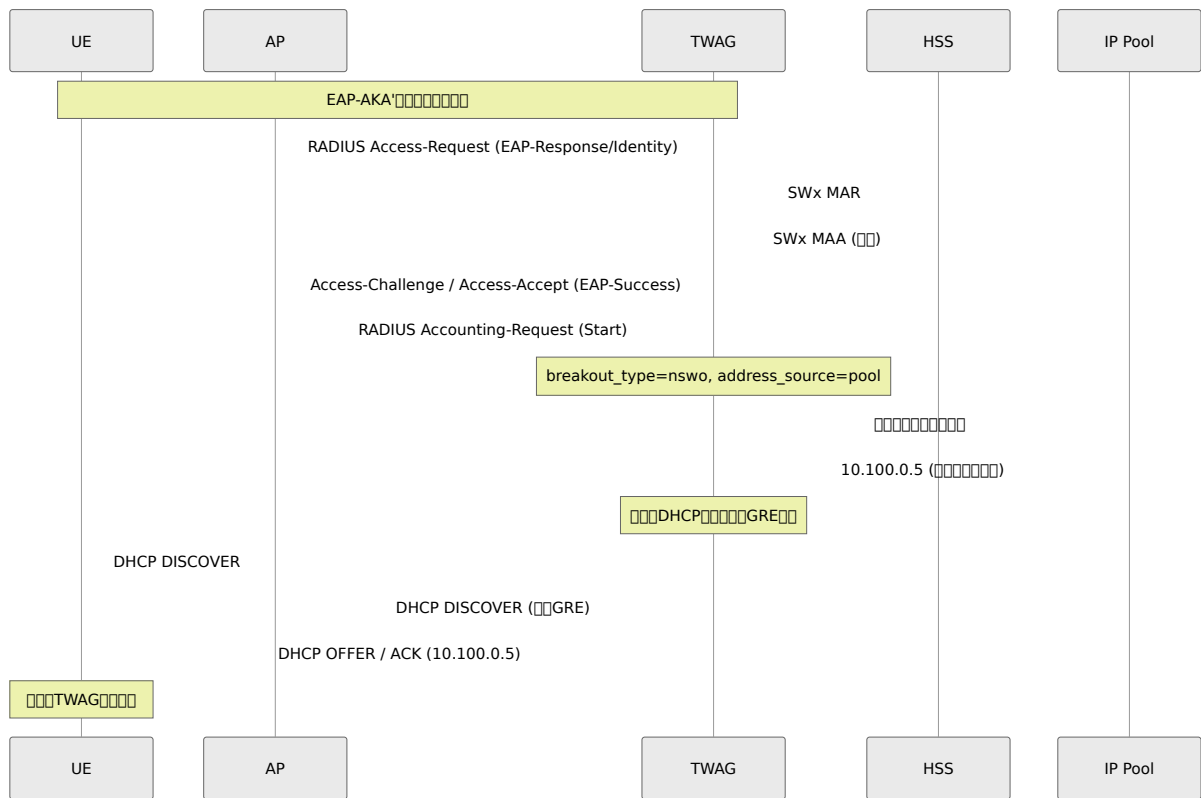
Network Elements - Core Network

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



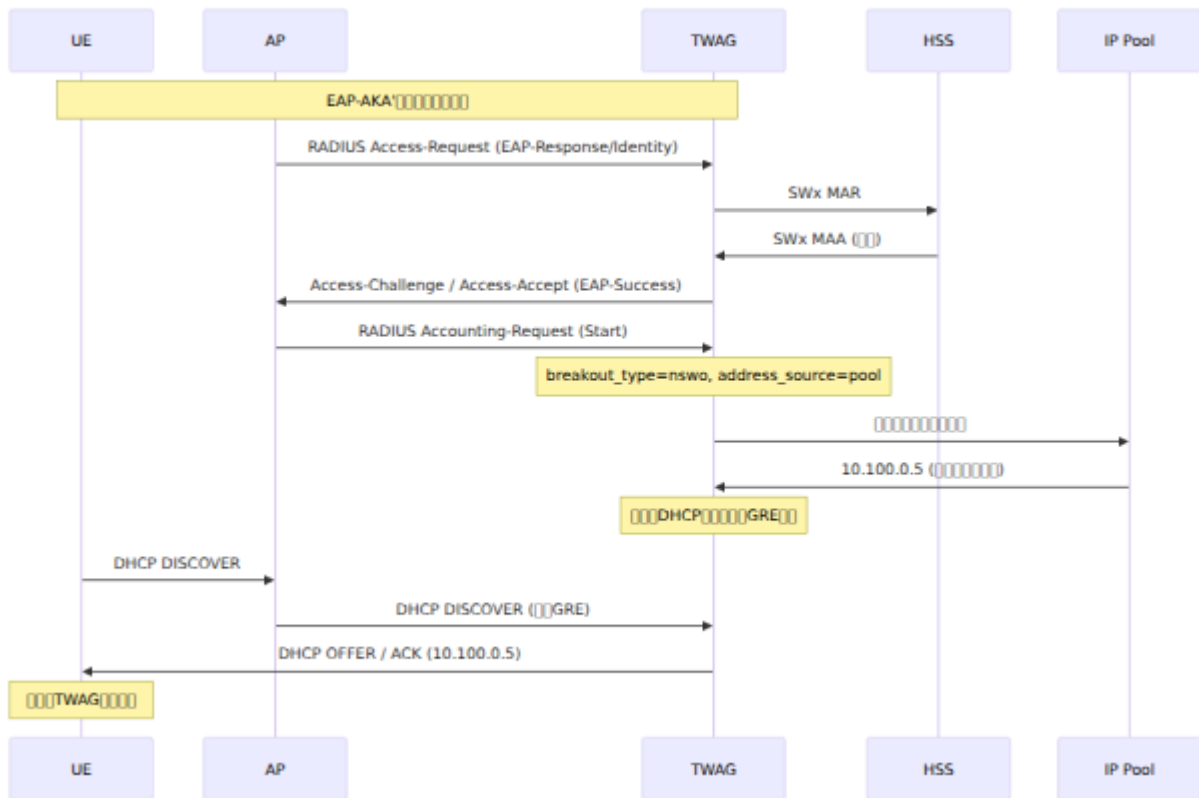
- NSWOPIP

breakout_type: nsw address_source: pool TWAG PGW TWAG
 IP DHCP



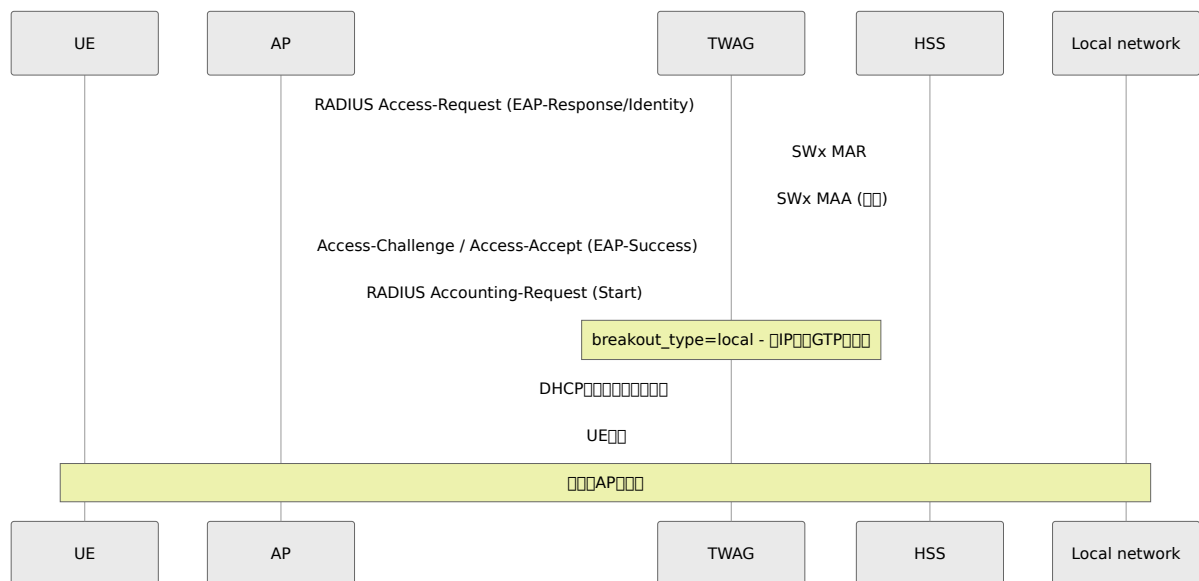
- NSWODHCP

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



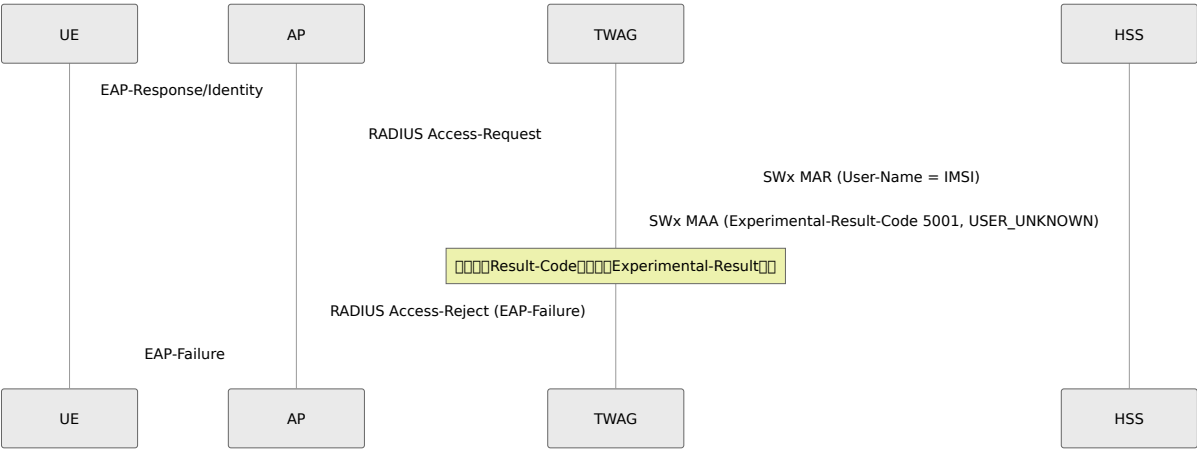
- AP

breakout_type: local AP TWAG IP UE TWAG GTP DHCP

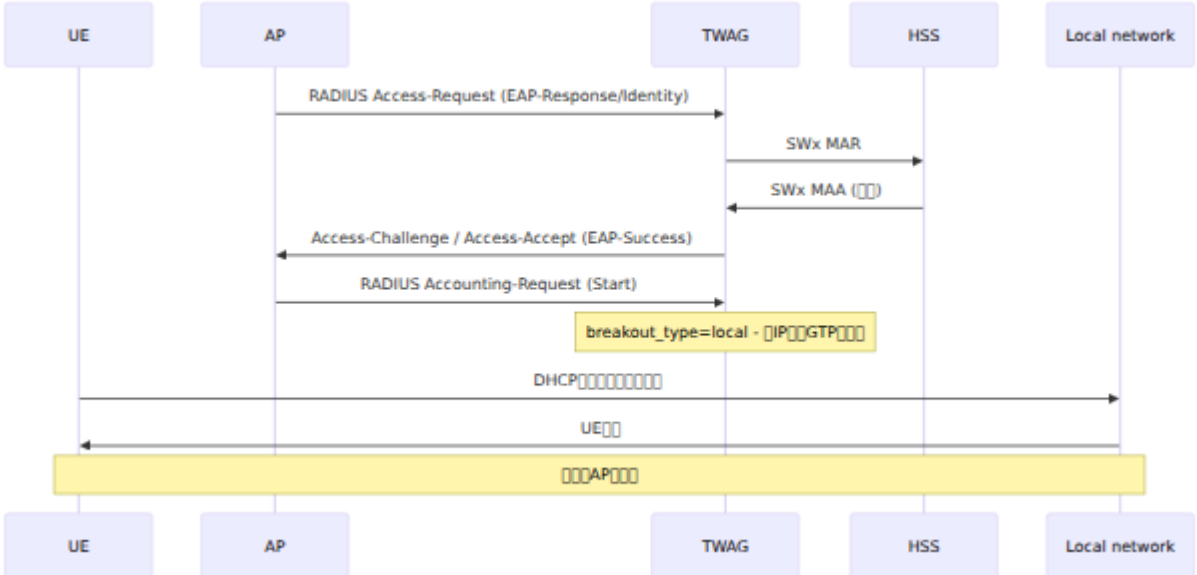


SWx

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

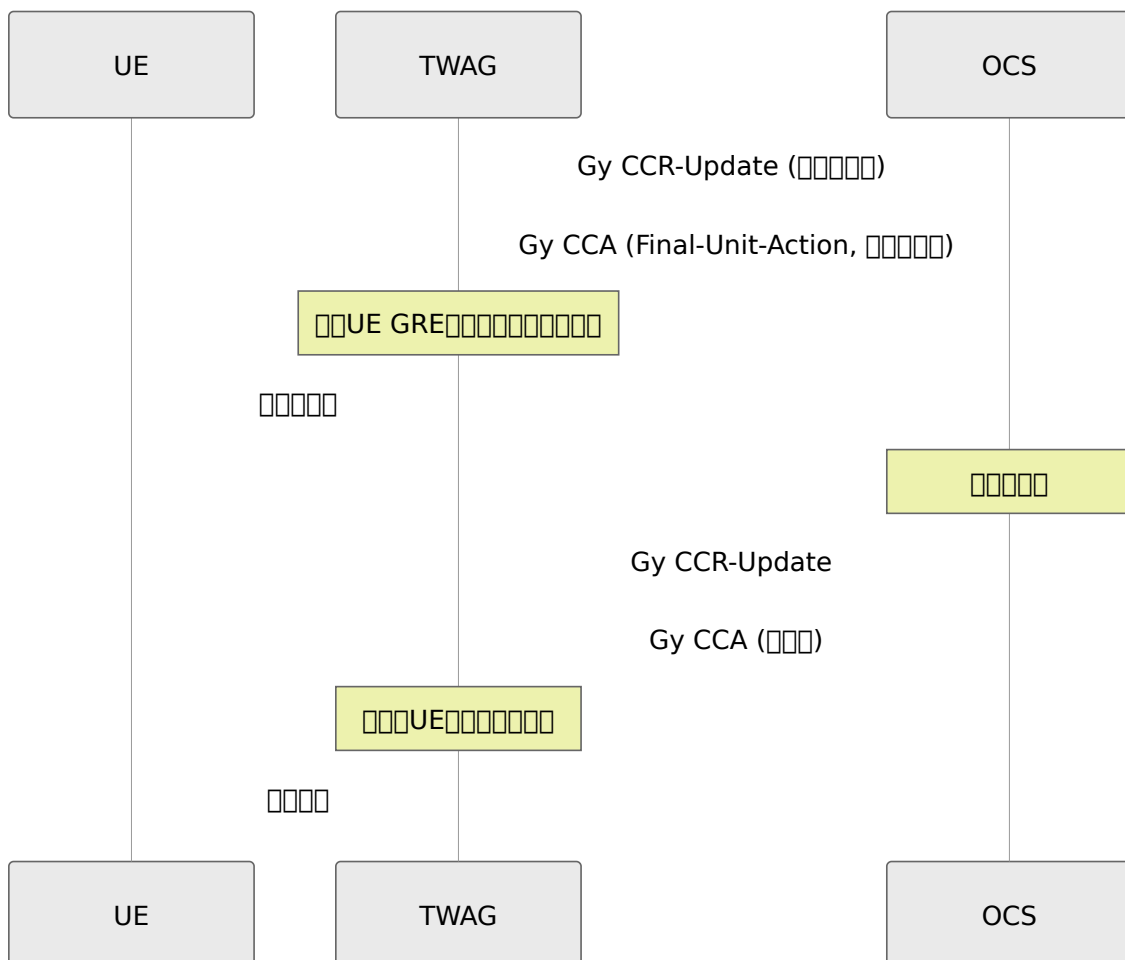


AP RADIUS TWAG charged
OCS



1. 3GPP TS 29.274

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

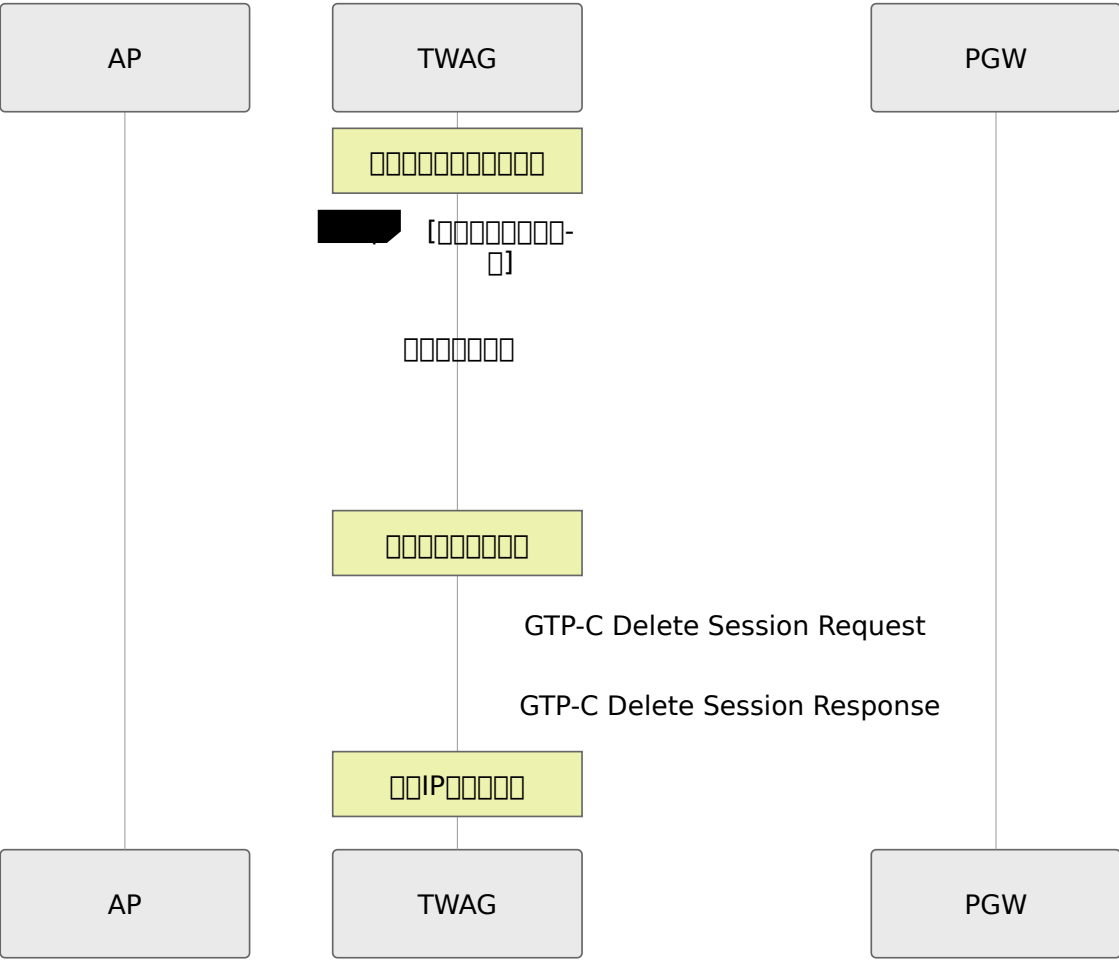


2. 3GPP TS 29.274

UE → AP → Accounting-Stop → TWAG → GTP → Gy CCR-Terminate → OCS

TWAG AP TWAG

AP TWAG

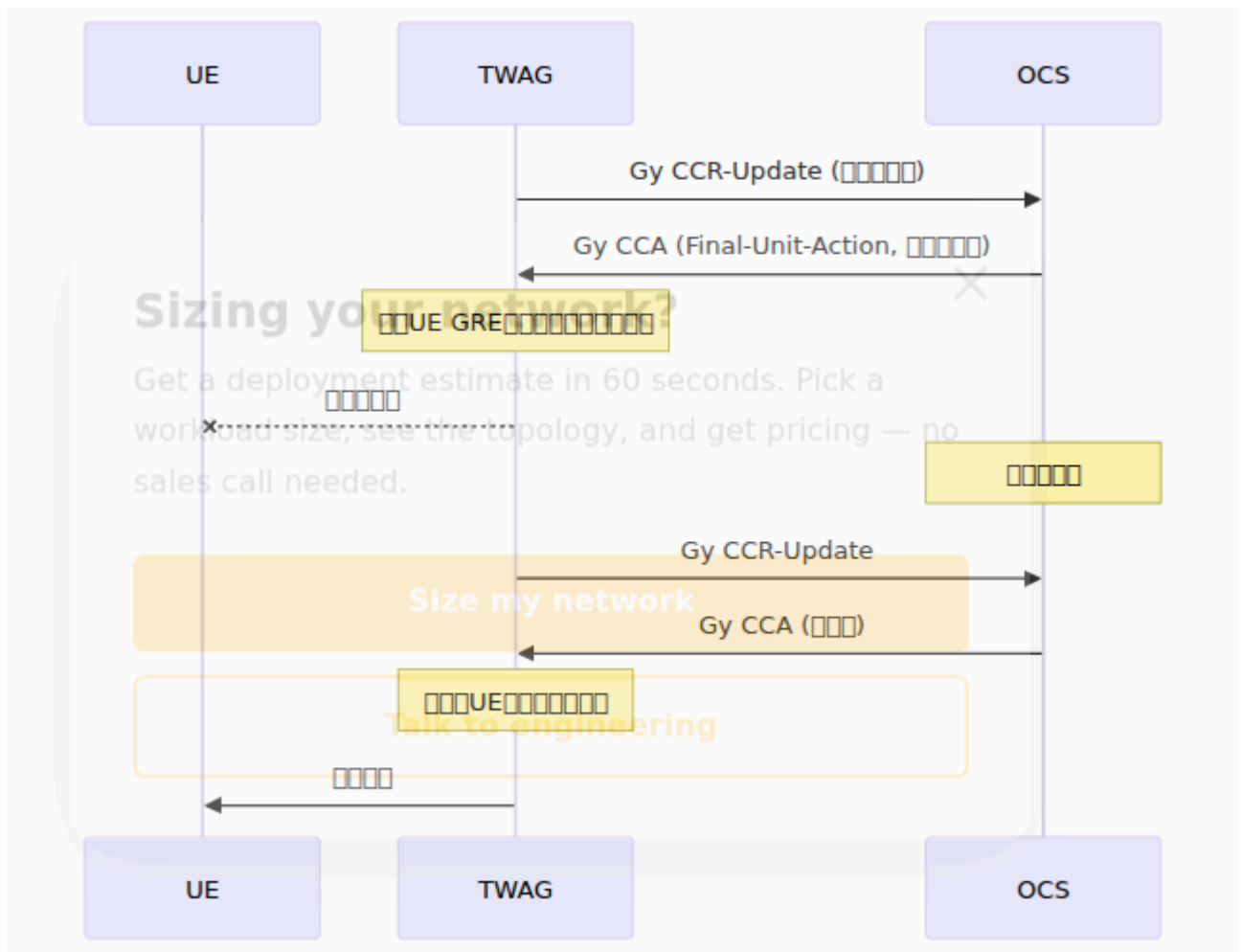


charging_mode

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

11

Österreichische TWAG Luftverkehrs-Gesellschaft mbH (WLCP) Österreichische OmniTWAG Luftverkehrs-Gesellschaft mbH

TWAG 3GPP TS 23.402 16 WLAN S2a

- 無線 LAN 無線 LAN 無線 LAN 無線 LAN
- 無線 LAN EAP-AKA / EAP-AKA' 無線 LAN
- S2a GTP-C 無線 LAN 無線 LAN 無線 LAN
- 無線 LAN 無線 LAN 無線 LAN 無線 LAN
- 無線 LAN GRE GTP-U 無線 LAN DHCP 無線 LAN
- RADIUS 無線 LAN 無線 LAN AP 無線 LAN 無線 LAN 無線 LAN

11

- 00000000
- 0000
- 00000000 (TSCM)
- 00000 (SCM)
- 00000 (MCM) 0 WLCF
- 0000000
- 00
- 0000
- 0000

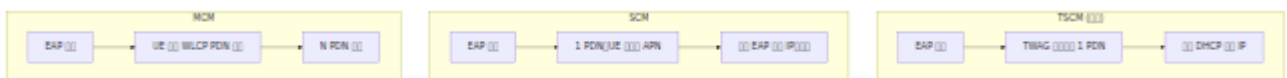
--	--	--	--	--	--	--

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

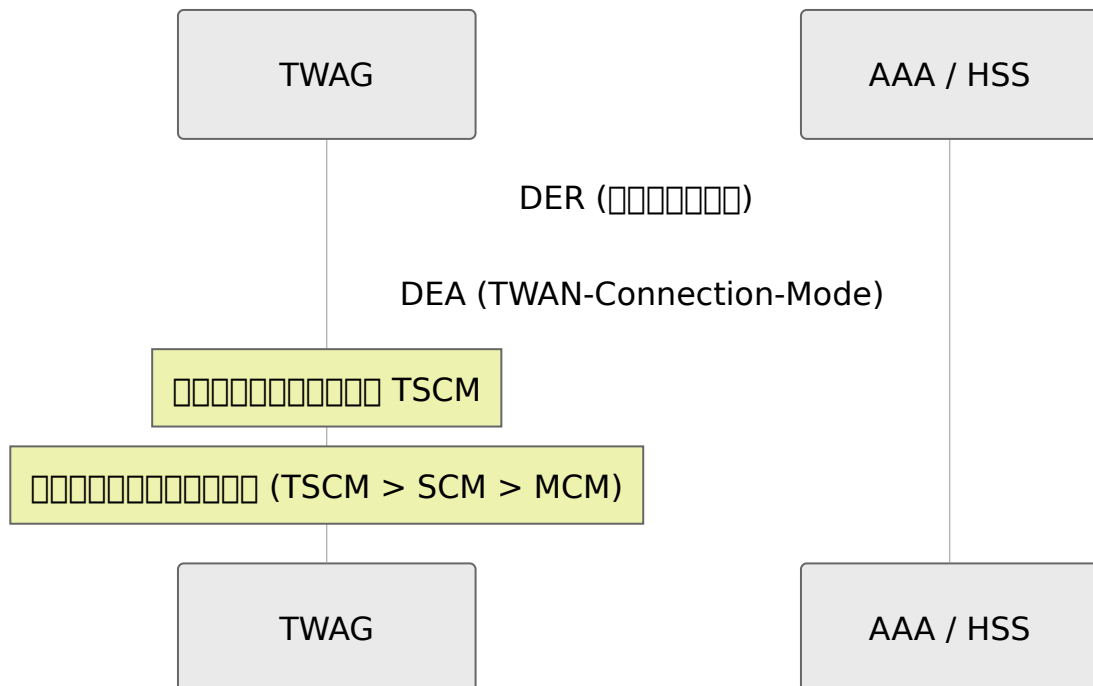


モード

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 0000000000000000 twan_connection_mode TWAN-Connection-Mode connection_mode Connection-Mode 0000000000 (0/1/2) (:tscm/:scm/:mcm) TSCM

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

STa (StaClient) DER DE
 STa RADIUS
 store_connection_mode TSCM
 DHCP IP WLC UE APN TSCM

□□□□□□ (TSCM)

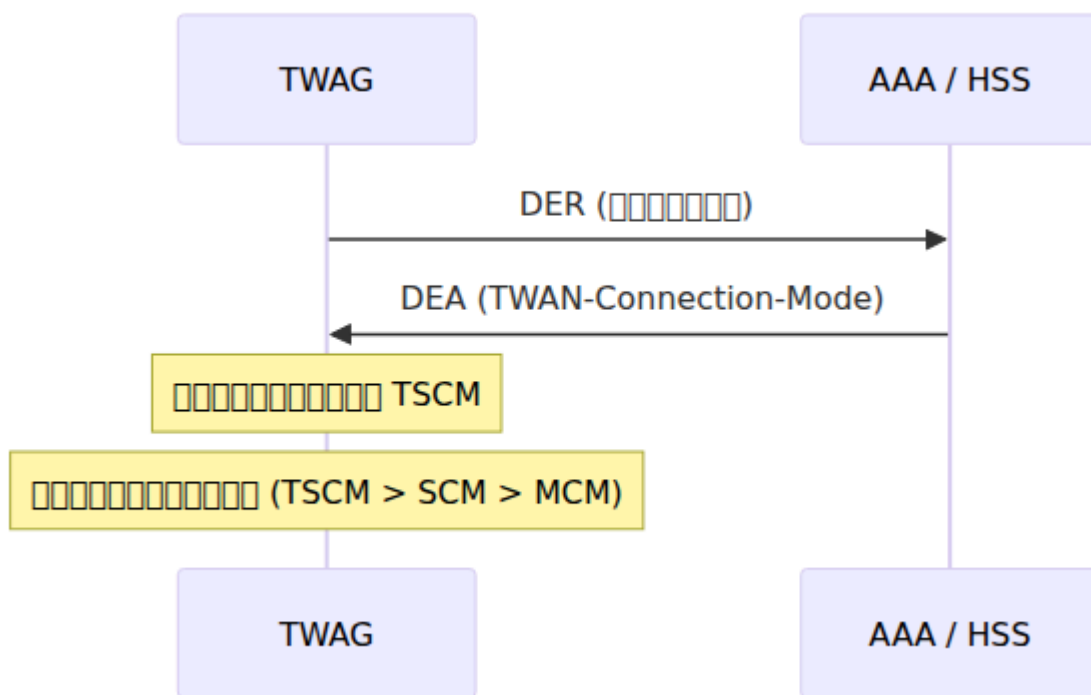
TSCM UE TSCM

□ TSCM □□

- UE 连接到 PDN 网络

- TWAG 向 APN UE 向 APN
- UE 向 DHCP 向 IP 向 TWAG 向 GRE 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 0000

00	IE	00
0x01	APN	00000000
0x02	PDN 00	1 IPv4, 2 IPv6, 3 IPv4v6
0x03	PCO	000000
0x04	PDN 00	000 UE IP 00 (IPv4IPv6 0 IPv4v6)
0x05	00 QoS	QCI 0 MBR / GBR 0
0x06	TFT	00000
0x07	00	00000
0x08	00 ID	EPS 0000 (EBI)
0x09	DNS 00	00000 DNS 00000

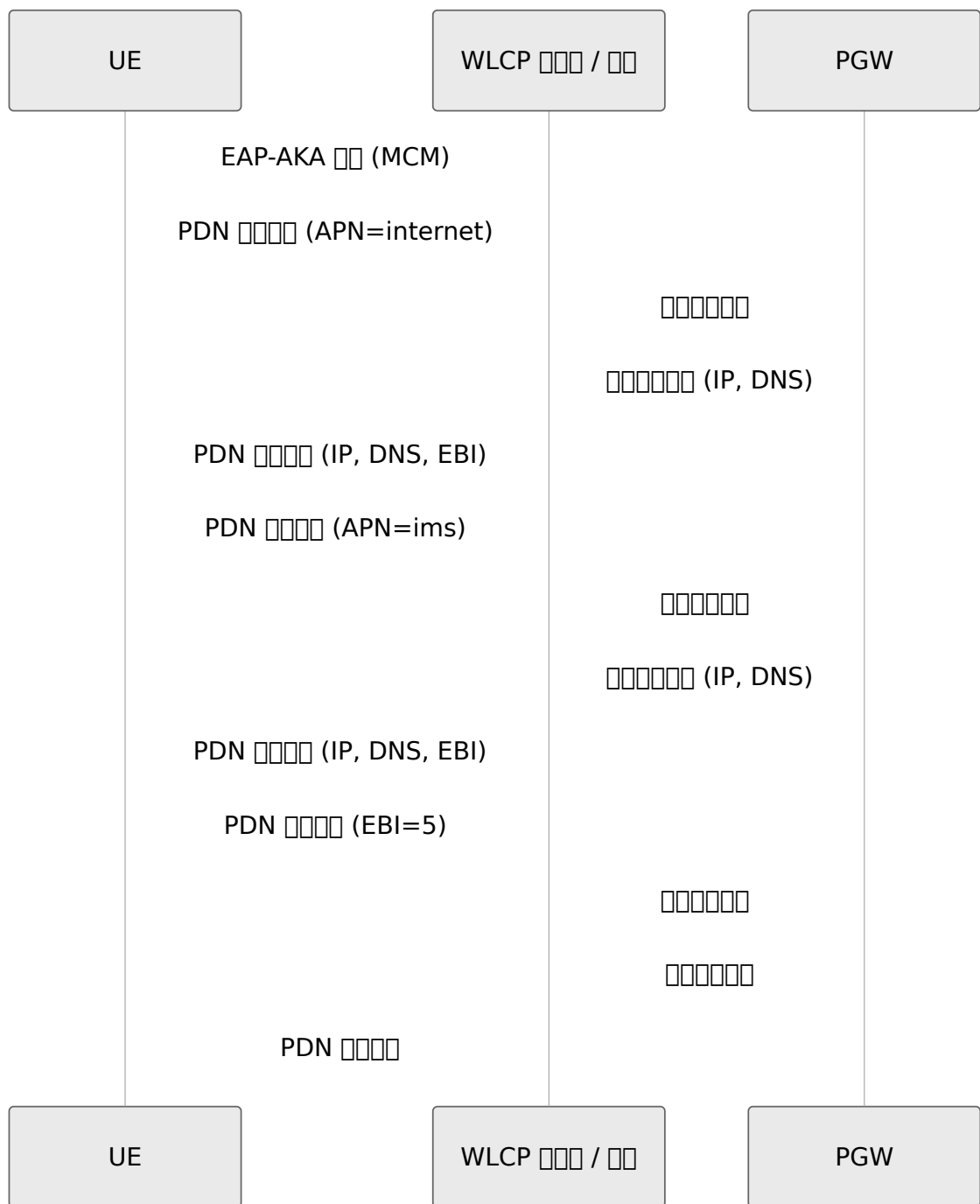
WLCP 消息

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

- 消息成功 APN PDN 消息 PCO 消息 APN 消息 internet PDN 消息 IPv4
- 消息 APN 消息 allowed_apns
- 消息 UE 消息 IMSI 消息 user_auth_failed 消息
- 消息 APN 消息 PGW 消息 S2a GTP-C 消息 RAT 消息 3 WLAN 消息 Handover Indication 消息
- 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
- 消息 S2a 消息 PDN 消息 network_failure



PDN

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

PGW PDN GTP-U PGW

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 )
```

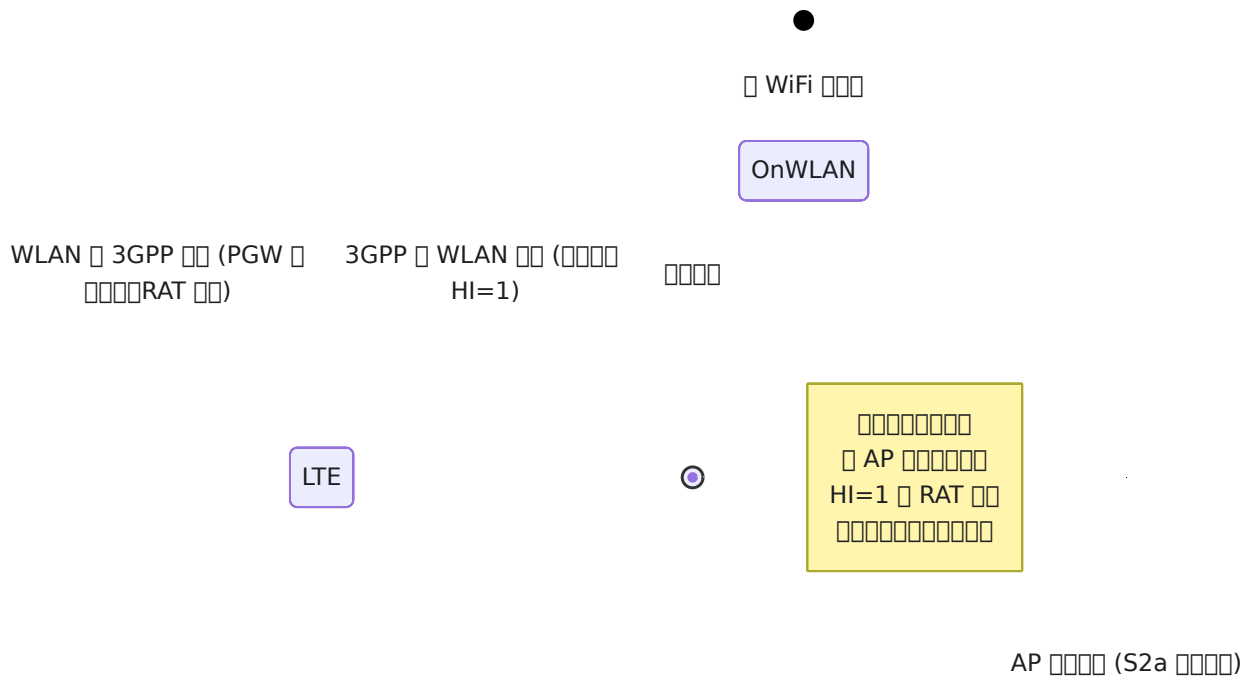
32 256 TWAG P0 IP TWAG IP MSK

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

UE 网络 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

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

UE 网络 WiFi 网络 LTE 网络 PGW 网络 RAT 网络 TWAG 网络 RAT 网络

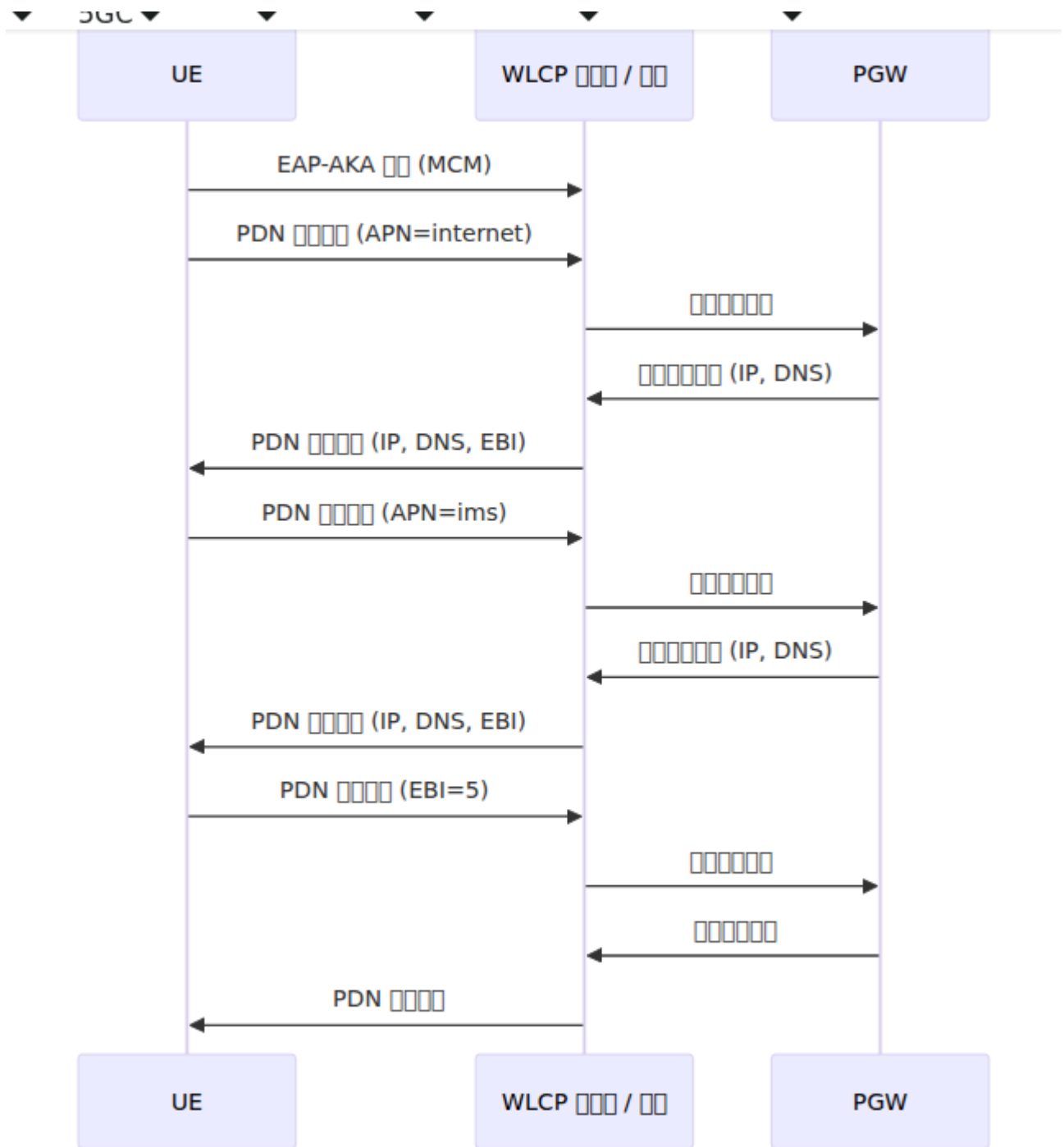
順序	項目
4	RAT 3GPP 3GPP
5	ISR
10	3GPP 3GPP

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]
```

Field	Value	Value	Value	Value
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	説明
enabled	boolean	boolean	false	WLCP を有効にするかどうか MCM を有効にするかどうか false
port	integer	integer	2156	WLCP が UDP を使用するポート番号 WLCP のポート番号
allowed_apns	array	array	:all	WLCP PDN を使用する APN のリスト :all はすべての APN を許可する ["internet", "ims"] はインターネットと IMS の APN を許可する apn_not_allowed は許可しない

WLCP を有効にするには enabled を true に設定する必要があります

設定

MCM PDN を使用する S2a を使用する GTP を使用する gtp を使用する pgw_ip を使用する local_ip を使用する mcc を使用する mnc を使用する

設定

WLCP を有効にするには enabled を true に設定する必要があります

이벤트	설명
<code>[:wlc, :packet, :received]</code>	WLCP 수신된 바이트 수
<code>[:wlc, :packet, :malformed]</code>	오류된 데이터
<code>[:wlc, :packet, :unregistered]</code>	등록되지 않은 데이터
<code>[:wlc, :pdn, :request]</code>	PDN 요청
<code>[:wlc, :pdn, :accept]</code>	PDN 수락
<code>[:wlc, :pdn, :reject]</code>	PDN 거부
<code>[:wlc, :pdn, :disconnect_request]</code>	PDN 연결 해제 요청
<code>[:wlc, :pdn, :disconnect_accept]</code>	PDN 연결 해제 수락
<code>[:twag, :handover, :in]</code>	3GPP → WLAN 핸드오버 원인
<code>[:twag, :handover, :out]</code>	WLAN → 3GPP → RAT 핸드오버 원인
<code>[:twag, :handover, :ap_change]</code>	AP 변경

WLCP 수신된 바이트 수 PDN 요청 UE APN EBI 핸드오버 원인 RAT AP 변경 UE

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
ネットワーク	構成
STa DER が	STa DER が TWAN-Connection-Mode が SSID / BSSID が
STa DEA が	STaClient が TSCM が
ネットワーク	store_connection_mode が
SCM がEAP IP APN が	SCM が TSCM が
WLCP が IEs が	取得
WLCP UDP が UE が	取得
WLCP が K_WLCP が FC=0x21 が	取得
MCM が PDN が	取得
WLCP が DTLS が	取得
MCM が	取得
WLCP が	取得

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

IP 配置 (DHCP / DHCPv6 / IPv6 RA / PCO)

概要

OmniTWAG には UE に対して IP 配置を制御する DHCPv4/DHCPv6/IPv6 配置 (SLAAC) 及び DNS 及び P-CSCF 配置を制御する PCO 機能があります。また、AP にも

- 無線ネットワーク側から PGW 側 S2a GTP 経由で OmniTWAG に対して UE に対して 3GPP TS 23.402 16.2 による
- **NSWO** 無線ネットワーク側から PGW/OmniTWAG に対して **IP** 及び **IP** による

無線ネットワーク側から RADIUS 経由で AP に対して AP に対して breakout_type 及び address_source 及び ip_pool_id による RADIUS による

無線ネットワーク側から GRE 経由で無線ネットワーク側 PAA 及び PCO 及び GTP-C 経由で S2a GTP-C 経由で UE に対して無線ネットワーク側による

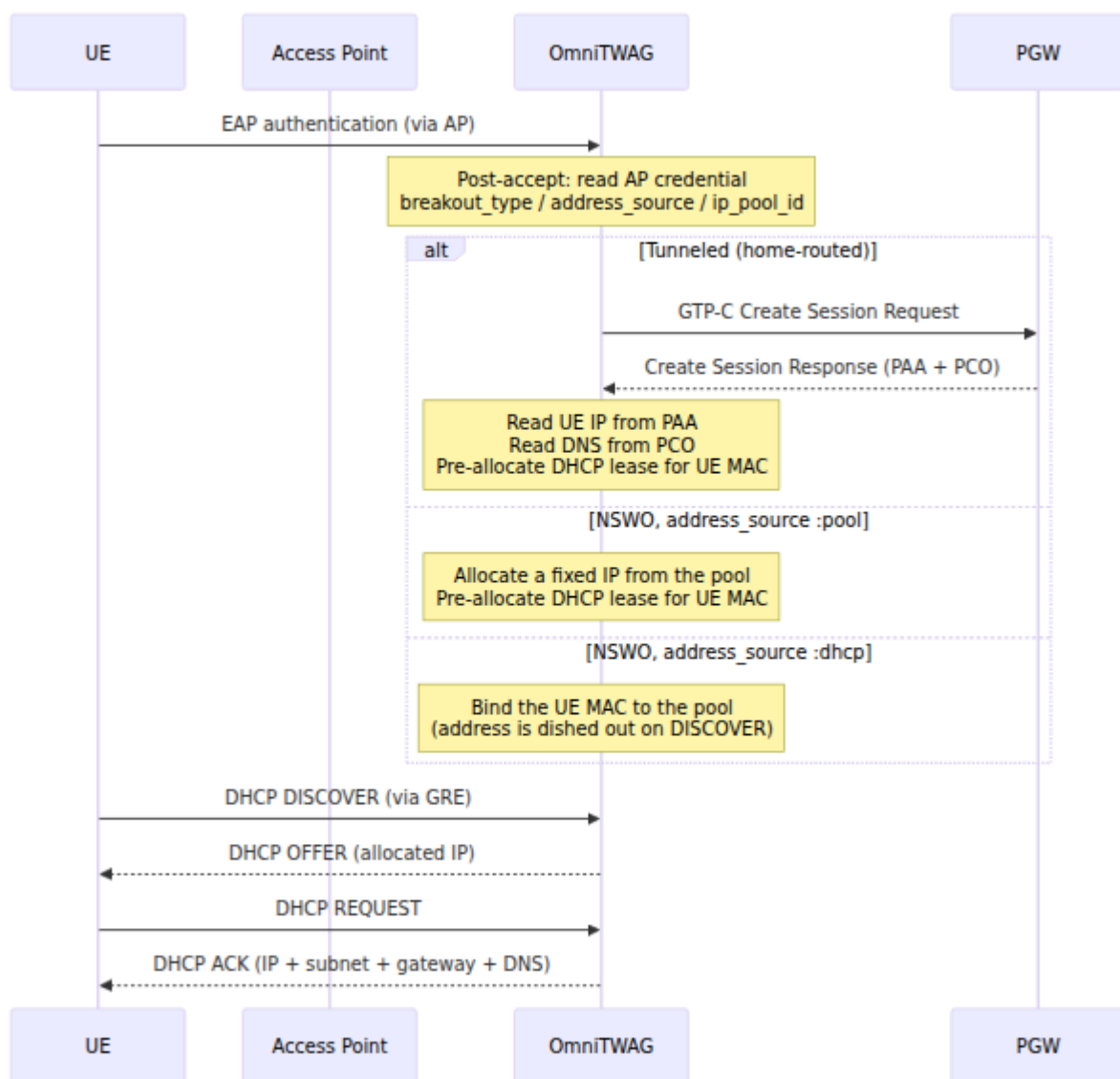
機能

- 無線ネットワーク側
- 無線ネットワーク側
- 無線ネットワーク側 PGW 無線ネットワーク側
- NSWO 無線ネットワーク側 DHCP
- DHCPv4
- DHCPv6
- IPv6 無線ネットワーク側 (SLAAC)
- DNS 及び PCO 無線ネットワーク側
- 無線ネットワーク側
- 無線ネットワーク側
- 無線ネットワーク側

-

OmniTWAG DHCPv4 DHCPv6 IPv6 PGW NSWO DHCPv4 OmniTWAG 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 間の OmniTWAG 間の

PAA	UE
ipv4	PAA IPv4
ipv6	PAA IPv6
ipv4v6	PAA IPv4

OmniTWAG PAA 間の UE 間の

NSWO DHCP

NSWO PGW 間の OmniTWAG 間の ip_pool_id IP IPv4 CIDR DNS IP

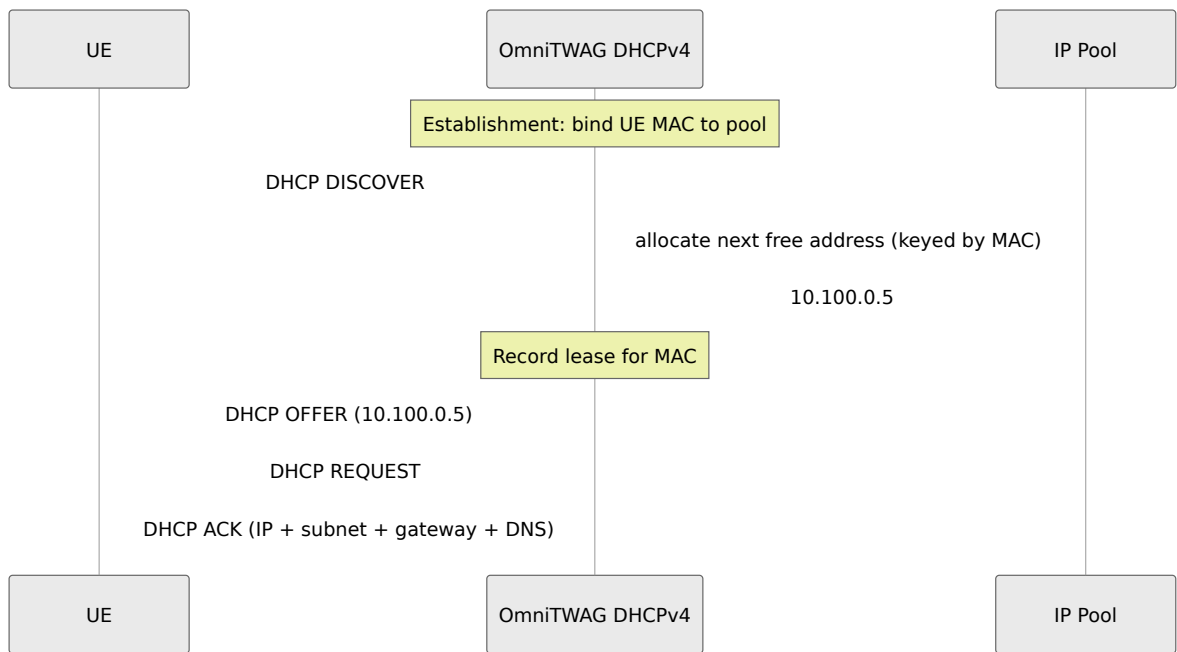
address_source UE

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` 和 `ip_pool_id` 的 OmniTWAG 的 UE 的

DHCPv4

DHCPv4 的 UDP 的 67 的 ETS 的 UE MAC 的

- PGW 的 OmniTWAG 的 NSWO `address_source: :pool` 的
- DISCOVER** 的 NSWO `address_source: :dhcp` 的 UE MAC 的 MAC 的 MAC 的 DISCOVER 的

的 MAC 的 MAC 的 DISCOVER 的

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 如何 DHCP 如何如何

如何	如何 如何 如何 如何	OmniTWAG 如何
DISCOVER	1	如何 MAC 如何如何OmniTWAG 如何 OFFER如何如何如何 MAC 如何如何 OmniTWAG 如何如何如何如何如何如何如何 OFFER如何如何如何如何如何如何 OmniTWAG 如何如何如何如何如何如何
REQUEST	3	如何如何如何如何 IP (如何 50) 如何如何如何如何如何如何 IP如何OmniTWAG 如何 如何 ACK如何如何如何 IP 如何如何OmniTWAG 如何 NAK如何如何如何如何 OmniTWAG 如何 NAK如何
RELEASE	7	OmniTWAG 如何如何 released如何如何如何如何OmniTWAG 如何如何如何 PDN 如何如何如何如何 TS 23.402 §16.1.5.2 NOTE 2如何
DECLINE	4	OmniTWAG 如何如何如何OmniTWAG 如何如何如何如何
INFORM	8	OmniTWAG 如何如何如何 ACK如何ACK 如何如何如何如何 DNS如何如何如何如何 (yiaddr 如何 0.0.0.0)如何

如何 INFORM如何OmniTWAG 如何如何 default_gateway 如何 default_subnet_mask如何
OmniTWAG 如何 INFORM 如何如何 8.8.8.8 如何 DNS 如何如何

如何如何

如何如何 UE IP如何如何如何如何DNS 如何如何如何如何如何如何如何如何 PGW 如何如何 UE IP 如何 DNS 如何
如何如何 DHCP 如何如何如何如何如何如何如何如何 NSWO 如何如何如何如何 IP 如何如何如何如何 IP如何如何 CIDR 如何
如何如何如何如何如何如何如何如何DNS 如何如何如何如何如何 IP 如何 如何 如何

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 548 端口

- 客户端 消息 (消息 11) 和 服务器 消息 (消息 12) 包含 DNS 服务器地址 [RFC 3736](#) 和 [RFC 3646](#) 的 UE 和 RA 消息 (O) 包含 DNS 服务器地址
- 客户端 消息 (消息 1) 和 服务器 消息 (消息 2) 包含 (消息 3) 和 (消息 7) 包含 IA_NA 消息 (消息 4) 和 IA_NA 消息 (消息 5) 包含 IPv6 地址 [RFC 3315](#)

客户端 消息 DUID 和 服务器 消息 GTP 消息 包含 DUID 消息

DHCPv6 消息

消息	消息	OmniTWAG 消息
消息 11	11	OmniTWAG 消息 包含 DUID 消息 ID 和 DNS 消息
消息 1	1	OmniTWAG 消息 包含 DUID 消息 IA_NA 消息 NoAddrsAvail 消息 2 消息
消息 3	3	OmniTWAG 消息 包含 DUID 和 IA_NA 消息
消息 4	-	OmniTWAG 消息

客户端 消息 ID 消息 [RFC 3315 §15](#) 包含 DHCPv6 消息 T1 消息 T2 消息

消息 DUID

OmniTWAG 消息 DUID-EN 消息 3GPP 消息 [10415](#) 消息

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 MTU 設定項目
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 項目

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 項目

• **P-CSCF** 向 PCO 向 PGW 向 P-CSCF 向 DHCP/DHCPv6 向 RA 向
向 P-CSCF DHCP 向 RA 向 P-CSCF 向
UE

向向向向向

向向向向向

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

Twag.Cidr 向向向向向向向向向向向 IPv4 向 IPv6 CIDR 向向向向向向向向向向向 RADIUS 向
向向向向向向向 IP 向 IP 向向向向向向向向向向向向向向向向向向向向向向向

向向

IP 向向向向向 :omnitwag 向向向向向向向向向向向 dhcp 向 ipv6_ra 向 DHCPv6 向向向向向向向

```
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 表示禁用 DHCPv4 服务
interface	string	wlan0	DHCP 服务的接口名称 gre 表示 GRE 隧道
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 54.0.0.1 GRE 隧道
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 服务的默认网关为 300

IPv6 RA 設定 (ipv6_ra)

項目	型別	単位	初期値	説明
enabled	ブール値		true	IPv6 RA を有効にするかどうか。false の場合は無効。
interface	文字列		wlan0	RA を送信するインターフェース。
ra_interval	整数	秒	30	RA を送信する間隔。
hop_limit	整数		64	RA のホップ制限。
router_lifetime	整数	秒	1800	RA の有効期限。

DHCPv6 設定

項目	型別	単位	説明	備考
dhcpcv6_dns_servers	文字列		Google DNS IPv6 (2001:4860:4860::8888, 2001:4860:4860::8844)	DHCPv6 による IPv6 DNS 設定。PGW PCO によって提供される。
dhcpcv6_domain_list	文字列		ドメインリスト	DHCPv6 による DNS 設定。24時間有効。OmniTWAG によって提供される。

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

□ □ □ □ □ □ □ □ □ □ □ □

項目	説明
mac	UE MAC アドレス
ip	IPv4 アドレス
subnet_mask	サブネットマスク
gateway	ゲートウェイ
dns_servers	DNS サーバ
lease_time	リース期間
allocated_at	割り当てられた Unix タイムスタンプ
expires_at	有効期限の Unix タイムスタンプ
status	active または expired
time_remaining	残り時間 (0 から開始)

概要

IP アドレス管理を Prometheus で監視する

関数	プロトコル	説明
<code>[:dhcp, :lease, :allocate]</code>	DHCPv4	指定した MAC アドレスに DHCPv4 レースを割り当てる
<code>[:dhcp, :lease, :deallocate]</code>	DHCPv4	指定した MAC アドレスに DHCPv4 レースを削除する
<code>[:dhcp, :lease, :reap]</code>	DHCPv4	期限切れの DHCPv4 レースを削除する
<code>[:dhcp, :discover]</code>	DHCPv4	DHCPv4 DISCOVER メッセージを送信し、OFFER を受信する
<code>[:dhcp, :discover, :pool_allocated]</code>	DHCPv4	指定した MAC アドレスに DHCPv4 DISCOVER メッセージを送信し、OFFER を受信する。NSWO :dhcp を使用する
<code>[:dhcp, :discover, :no_allocation]</code>	DHCPv4	指定した MAC アドレスに DHCPv4 DISCOVER メッセージを送信する
<code>[:dhcp, :ack]</code>	DHCPv4	DHCPv4 REQUEST メッセージを送信し、ACK を受信する
<code>[:dhcp, :nak]</code>	DHCPv4	DHCPv4 REQUEST メッセージを送信し、NAK を受信する
<code>[:dhcp, :release]</code>	DHCPv4	DHCPv4 RELEASE メッセージを送信する
<code>[:dhcp, :decline]</code>	DHCPv4	DHCPv4 DECLINE メッセージを送信する
<code>[:dhcp, :inform]</code>	DHCPv4	DHCPv4 INFORM メッセージを送信する
<code>[:dhcp, :packet, :malformed]</code>	DHCPv4	不正な DHCPv4 パケットを受信した
<code>[:dhcpv6, :info_request, :received]</code>	DHCPv6	DHCPv6 INFO_REQUEST メッセージを受信した
<code>[:dhcpv6, :solicit, :received]</code>	DHCPv6	DHCPv6 SOLICIT メッセージを受信した

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 前缀の追加
<code>[:ipv6_ra, :prefix, :remove]</code>	IPv6 RA	IPv6 前缀の削除
<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 設定

UE 側で IPv4 側の DHCP 設定を行う。UE MAC 設定は `[ :dhcp, :discover, :no_allocation]` 設定

設定項目

- GTP 設定は PAA 側の OmniTWAG 設定
- GTP-C 設定

設定項目 **NSWO**

- AP 側の `ip_pool_id` 側の OmniTWAG 設定

- 送信元アドレスを指定
- 送信元アドレス: :dhcp 送信元 MAC 送信元 MAC

送信元

- DHCPv4 送信元

送信元

1. 送信元 OmniTWAG 送信元 NSWOW 送信元 UE IP
2. 送信元 PGW 送信元 APN 送信元 PAA
3. 送信元 NSWOW 送信元 AP 送信元 ip_pool_id 送信元 IP 送信元 RADIUS 送信元
4. 送信元 DHCP 送信元 DHCPv4 送信元

UE 送信元 DNS

送信元 UE 送信元 IPv4 送信元

送信元

- PGW PCO 送信元 IPv4 DNS 送信元 OmniTWAG 送信元 8.8.8.8
- UE 送信元 PGW 送信元 DNS 送信元

送信元

1. 送信元 DHCP 送信元 REST API 送信元 DNS 送信元
2. 送信元 PGW PCO 送信元 DNS 送信元
3. 送信元 DNS 送信元 UE 送信元

UE 送信元 NAK

送信元 UE 送信元 REQUEST 送信元 NAK 送信元 [:dhcp, :nak] 送信元

送信元

- UE 送信元 IP 送信元 50 送信元
- 送信元

□□□□

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 を NSWO TWAG として OmniTWAG として IP を IPv4 CIDR DNS REST API Web

OmniTWAG DETS OmniTWAG

機能

-
-
-
-
- REST API
- Web
-

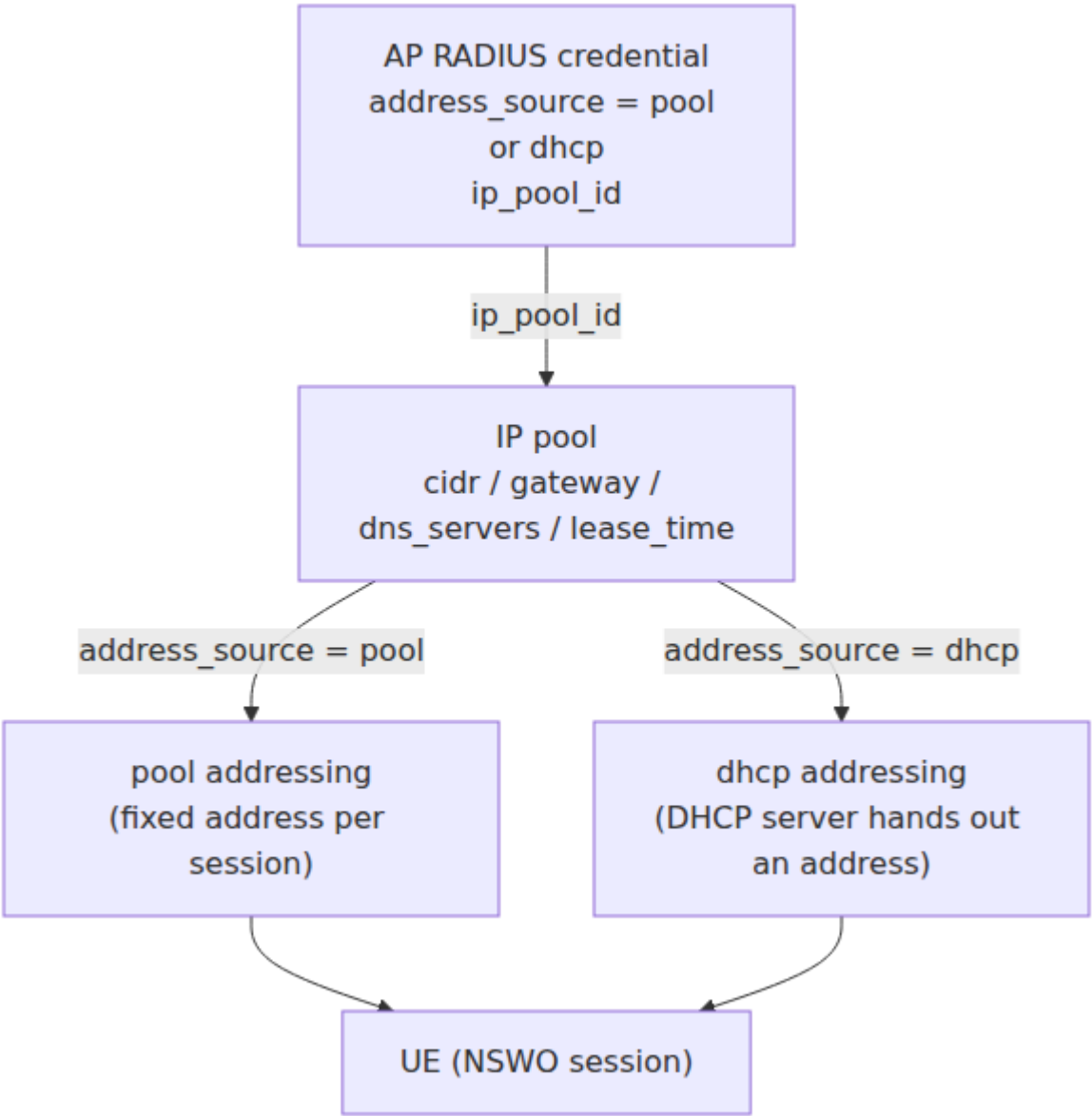
インストール

IPv4 CIDR DNS

IP AP RADIUS `ip_pool_id` `id` RADIUS NSWO

`address_source` OmniTWAG UE

項目	説明
pool	OmniTWAG によって管理される IP アドレスプール
dhcp	OmniTWAG DHCP サーバから UE へ DHCP DISCOVER を送信する

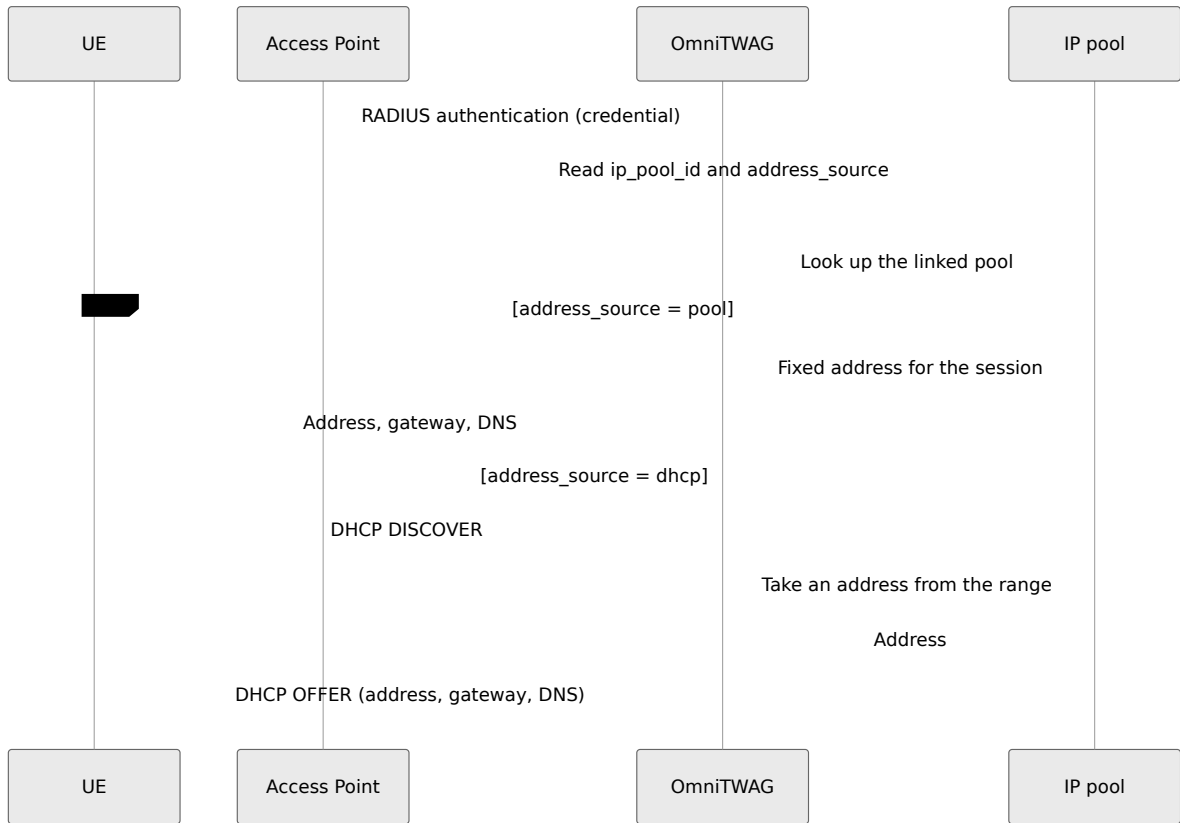


動作概要

OmniTWAG は AP から RADIUS から NSWO を取得する

1. RADIUS から RADIUS を取得する

- OmniTWAG 取得 `ip_pool_id`
- OmniTWAG 取得 `address_source`
- 取得 `address_source` 為 `pool` 時 OmniTWAG 取得 IP 池 ID
- 取得 `address_source` 為 `dhcp` 時 OmniTWAG 向 UE 發 DHCP DISCOVER 取得 DHCP 回應
- OmniTWAG 取得 DNS 資訊 向 UE 發



IP 池

OmniTWAG 的 CIDR 池

每個 CIDR 池是 /30 的，每個池有 2 個 IP 地址
OmniTWAG 使用

當 OmniTWAG 的 `pool_exhausted` 為 true 時

當 `allocated` 為 true 時 OmniTWAG 使用

API

API仕様書

項目	型	説明
id	整数 1	一意なUUIDを返す
name	文字列 1	ネットワーク名 (例: Melbourne guest WiFi)
cidr	文字列 1	IPv4 CIDR 形式 (例: 10.60.0.0/24) または /30 形式のネットワーク
gateway	文字列 1	UE の IPv4 ネットワークアドレス CIDR 形式
dns_servers	文字列 1	UE の IPv4 DNS サーバ
lease_time	整数 1	リース時間 (秒) 3600
default	ブール値 1	true: DHCP サーバとして動作する false: 静的 IP 設定
allocated	ブール値 1	IP アドレスが割り当てられているかどうか
inserted_at	文字列 1	Unix 時刻 (ISO 8601) にフォーマットされた挿入日時
updated_at	文字列 1	Unix 時刻 (ISO 8601) にフォーマットされた更新日時

レスポンス: { name: string, cidr: string, gateway: string, dns_servers: string, lease_time: integer, id: integer, allocated: boolean, inserted_at: string, updated_at: string }

REST API

OmniTWAG REST API 透過 API 端點 8444 透過 HTTPS 與伺服器通訊
/ip_pools

動作	URL	動作名稱	說明
GET	/ip_pools	index	取得所有 IP 池資訊，返回 name 與 cidr 清單
GET	/ip_pools/{id}	show	取得 id 的 IP 池資訊
POST	/ip_pools	create	新增 IP 池
PUT	/ip_pools/{id}	update	更新 IP 池
DELETE	/ip_pools/{id}	delete	刪除 IP 池

參數

POST 與 PUT 透過 JSON 格式傳送參數

參數名稱	類型	是否必填	說明
name	字串	否	IP 池名稱
cidr	字串	否	IPv4 CIDR 格式，例如 /30 或 192.168.1.0/24
gateway	字串	否	IPv4 網關地址，CIDR 格式
dns_servers	陣列	否	IPv4 DNS 伺服器地址
lease_time	數字	否	租約時間，單位為秒，預設 3600
default	字串	否	是否預設 DHCP 租約，true 或 false

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

请求头

请求体

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

5/5

- CIDR 0000000000000000
- CIDR 0000000000000000

□□□□□

1. `allocated` CIDR `10.0.0.0/24`
2. CIDR `10.0.0.0/24`
3. `10.0.0.0/24`

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

□□□ □□□□□□□□□□ IP □□□□□□

□□□□□

- TWAG NSWO OmniTWAG
- AP ip_pool_id

□□□□□

1. 配置 TWAG 和 NSWO 的 IP 地址
2. 配置 RADIUS 服务器的 IP 地址和 ID

UE □□□□□

UE IPv4

□ □ □ □ □

- `address_source` UE `dhcp` UE DHCP DISCOVER `pool` OmniTWAG
-
- `ip_pool_id`

□ □ □ □ □

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
- NSW0 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 通过以下配置可以监控 RADIUS 设备
[:radius, :access, :request, :count] 配置 radius_access_request_count

RADIUS 简介

RFC 2865 是 RADIUS Access-Request/Response 协议

名称	单位	类型	描述
radius_access_request_count	计数	整数	RADIUS Access-Request 请求计数
radius_access_accept_count	计数	整数	RADIUS Access-Accept 响应计数
radius_access_reject_count	计数	整数	RADIUS Access-Reject 响应计数
radius_access_challenge_count	计数	整数	RADIUS Access-Challenge 响应计数 EAP 认证失败

更多

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

# EAP
rate(radius_access_challenge_count[5m])

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

RADIUS

RADIUS RADIUS

radius_auth_unauthorized_source_count	source		
radius_auth_bad_secret_count	source, credential_id		
radius_access_rate_limited_count	nas_ip		

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

# 增加 AP 坏密钥计数
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 識別子送信 1回
eap_aka_challenge_count	回 分	EAP-AKA チャレンジ送信 2回
eap_aka_sync_failure_count	回 分	EAP-AKA 同期失敗(SQLN) 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 受信失敗)
eap_aka_client_error_count	回 分	UE からの EAP-AKA 送信エラー
eap_auth_rejected_count	回 分	失敗した HSS USER_UNKNOWN 認証 回数(UE)
eap_process_crash_count	回 分	EAP 処理エラー(UE からの EAP 送信失敗)
eap_sim_start_count	回 分	EAP-SIM 送信
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 送信

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 の回数が増加したときに通知
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 失敗理由の数を通知
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
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 接続ログ解析

接続ログから Twag.Session.Client を取得する

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	描述
twag_liveness_arp_probe_failure_count	个		失败 ARP 探测
twag_handover_in_count	个	source	Handover-in 次数
twag_handover_out_count	个	cause	Handover-out 次数
twag_handover_ap_change_count	个		切换 AP 次数

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 項目

diameter_message_count

項目 項目

필드	설명
application	Diameter 프로토콜의 application_id, status_code
command	Diameter 프로토콜의 command_code, command_name
direction	inbound (수신) 또는 outbound (발신)

예시

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

NSWO 필드

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

DHCP 필드

DHCPv4 관련 필드 discover.pool_allocated는 NSWO에서 DHCPv4 풀 상태를 나타내며, 이는 NSWO가 DHCPv4 풀을 관리하는 방식에 따라 달라집니다.

項目	単位	説明
dhcp_lease_allocate_count	個 分	リース割り当て
dhcp_lease_deallocate_count	個 分	リース解放
dhcp_discover_count	個 分	クライアントからの DISCOVER 受信
dhcp_discover_no_allocation_count	個 分	クライアントからの DISCOVER 受信 （リースなし）
dhcp_discover_pool_allocated_count	個 分	クライアントからの DISCOVER 受信 （NSWO あり）
dhcp_ack_count	個 分	クライアントからの ACK 受信
dhcp_nak_count	個 分	クライアントからの NAK 受信
dhcp_release_count	個 分	クライアントからの RELEASE 受信
dhcp_decline_count	個 分	クライアントからの DECLINE 受信
dhcp_inform_count	個 分	クライアントからの INFORM 受信
dhcp_packet_malformed_count	個 分	クライアントからの DHCP パケット （不正）

合計

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

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

DHCP 租约回收 dhcp_lease_reap Prometheus 指标
/metrics 指标

DHCPv6 与 IPv6 RA 指标

指标	单位	描述
dhcpv6_info_request_received_count	每秒	DHCPv6 信息请求接收计数
dhcpv6_solicit_received_count	每秒	DHCPv6 Solicit 接收计数
dhcpv6_request_received_count	每秒	DHCPv6 请求接收计数
dhcpv6_info_reply_sent_count	每秒	DHCPv6 信息回复发送计数
dhcpv6_advertise_sent_count	每秒	DHCPv6 广告发送计数
dhcpv6_reply_sent_count	每秒	DHCPv6 回复发送计数
ipv6_ra_prefix_add_count	每秒	IPv6 RA 前缀添加计数
ipv6_ra_prefix_remove_count	每秒	IPv6 RA 前缀移除计数
ipv6_ra_advertisement_sent_count	每秒	IPv6 RA 广告发送计数

WLCP 指标

无线局域网 WLAN 指标

項目	単位	型	説明
<code>wlcp_packet_received_count</code>	個	int	受信した WLCP パケットの総数
<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 パケットの総数
<code>wlcp_packet_unregistered_count</code>	個	int	登録されていない WLCP パケットの総数
<code>wlcp_pdn_request_count</code>	個	int	PDN 接続リクエストの総数
<code>wlcp_pdn_accept_count</code>	個	int	PDN 接続承諾の総数
<code>wlcp_pdn_reject_count</code>	個	int	PDN 接続拒否の総数
<code>wlcp_pdn_disconnect_request_count</code>	個	int	PDN 切断リクエストの総数
<code>wlcp_pdn_disconnect_accept_count</code>	個	int	PDN 切断承諾の総数

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

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	説明
<code>vm_statistics_run_queue</code>	ランキューの長さ
<code>vm_total_run_queue_lengths_total</code>	ランキューの長さの合計
<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

```
# 受け入れ
sum(rate(radius_access_accept_count[5m]))

# 拒絶
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: "高認証失敗率 (> 10%)"

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

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

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

```

- 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 内存使用 1GB"

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

```

配置

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

- **AP**

Radius Gy (Gy)

00

OmniTWAG 0000000000 (OCS) 0000000000OmniTWAG 00 Diameter Ro (Gy) 000
0000000 (CCR) 0000000000 (CCA)0000000000 TWAG 000 WLAN 000000000000

00000000 TWAG 000000000000 **NSWO**0000000 0 TWAG 000000000000TWAG 000000
00000000 TWAG 0000000000 000000**GTP**0000 00PGW 000000000000000000000000
:disabled00000000000000000000 00000 0000

00000000 Gy 00000000 UE 00000

- 0000000 (online_charging.mode)0000000 OmniTWAG 0000000000000000
- 00 **AP** 0000000 0 AP 0000 (charging_mode)0000000 OmniTWAG 0000000 UE 00 Gy 0
0000000 **RADIUS** 0000

0000000000 CCR-Initial 0000000000 00 AP 0000000000000000000000

00

- 0000
- 000000
- 00 **AP** 0000000
- 00000000
- 0000000000000000
- 000000
- 000000000000
- 0000
- 000000000000
- 000000
- 0000

概要

Gy 側は Diameter 側から OCS 側へ 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 OCS 側へ

- **CCR-Initial (CCR-I)**側から OmniTWAG へ OCS 側へ
- **CCR-Update (CCR-U)**側から OmniTWAG 側へ
- **CCR-Termination (CCR-T)**側から OmniTWAG 側へ

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 側へ OmniTWAG 側へ CCR-I 側へ

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

:online_charging

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

AP 在 CCR-U 上

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

AP 在 CCR-U 上 charging_mode	在 CCR-U 上	在 CCR-U 上 Gy 在 CCR-U 上
:charged	AP 在 CCR-U 上 在 CCR-U 上	OmniTWAG 在 CCR-U 上 Gy 在 CCR-U 上 (CCR-Initial)
:uncharged	AP 在 CCR-U 上 在 CCR-U 上	OmniTWAG 在 CCR-U 上 Gy 在 CCR-U 上
:auth_only	AP 在 CCR-U 上 在 CCR-U 上	OmniTWAG 在 CCR-U 上 Gy 在 CCR-U 上

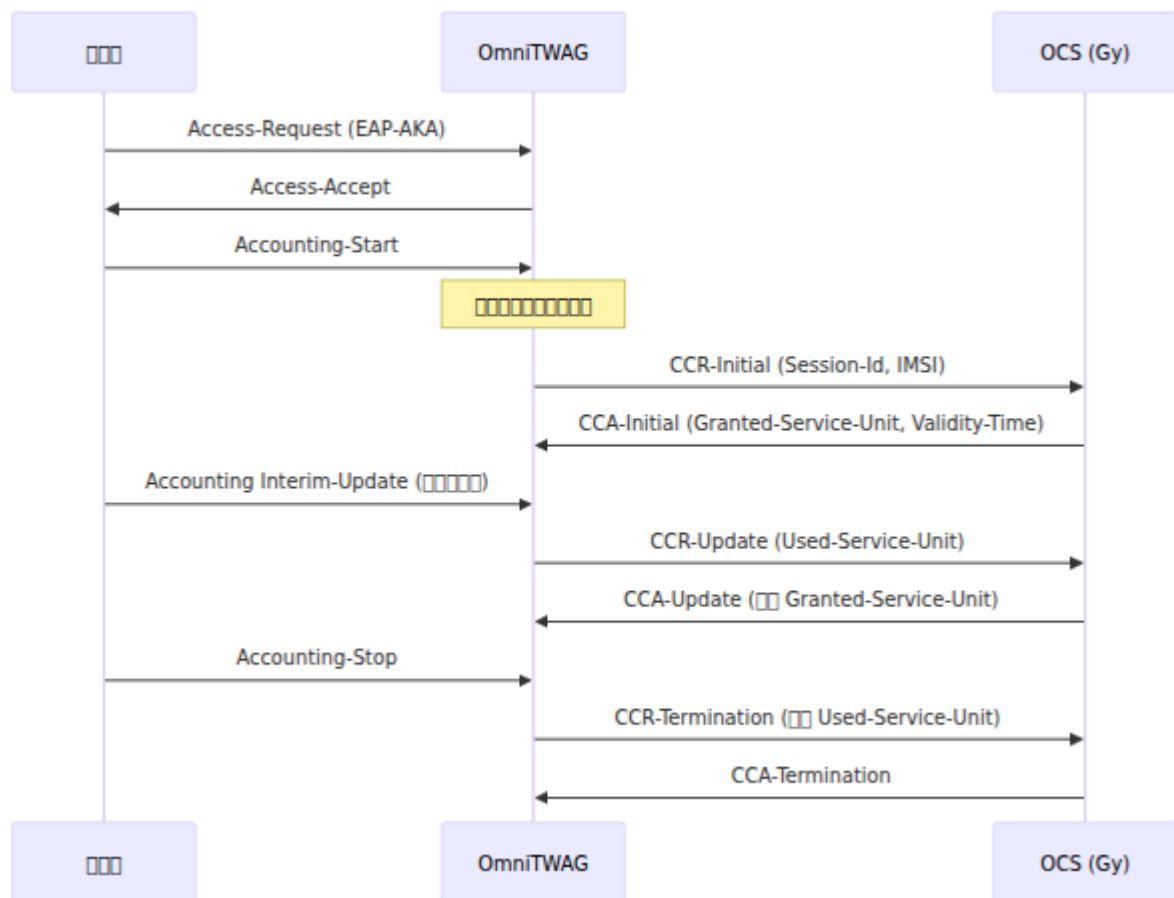
在 CCR-U 上注册 CCR-I 的 OmniTWAG 在 CCR-T 上

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

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

Sequence Diagram

Sequence Diagram illustrating the interaction between a User Equipment (UE), OmniTWAG, and OCS (Gy) for RADIUS accounting and charging.



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 (charging_mode: charged).

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

CCR-Update

The UE sends Accounting Interim-Update to OmniTWAG. OmniTWAG sends CCR-Update to OCS (Gy) after receiving Accounting Interim-Update from the UE. The CCR-Update message includes the Used-Service-Unit.

- 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

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

OmniTWAG

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

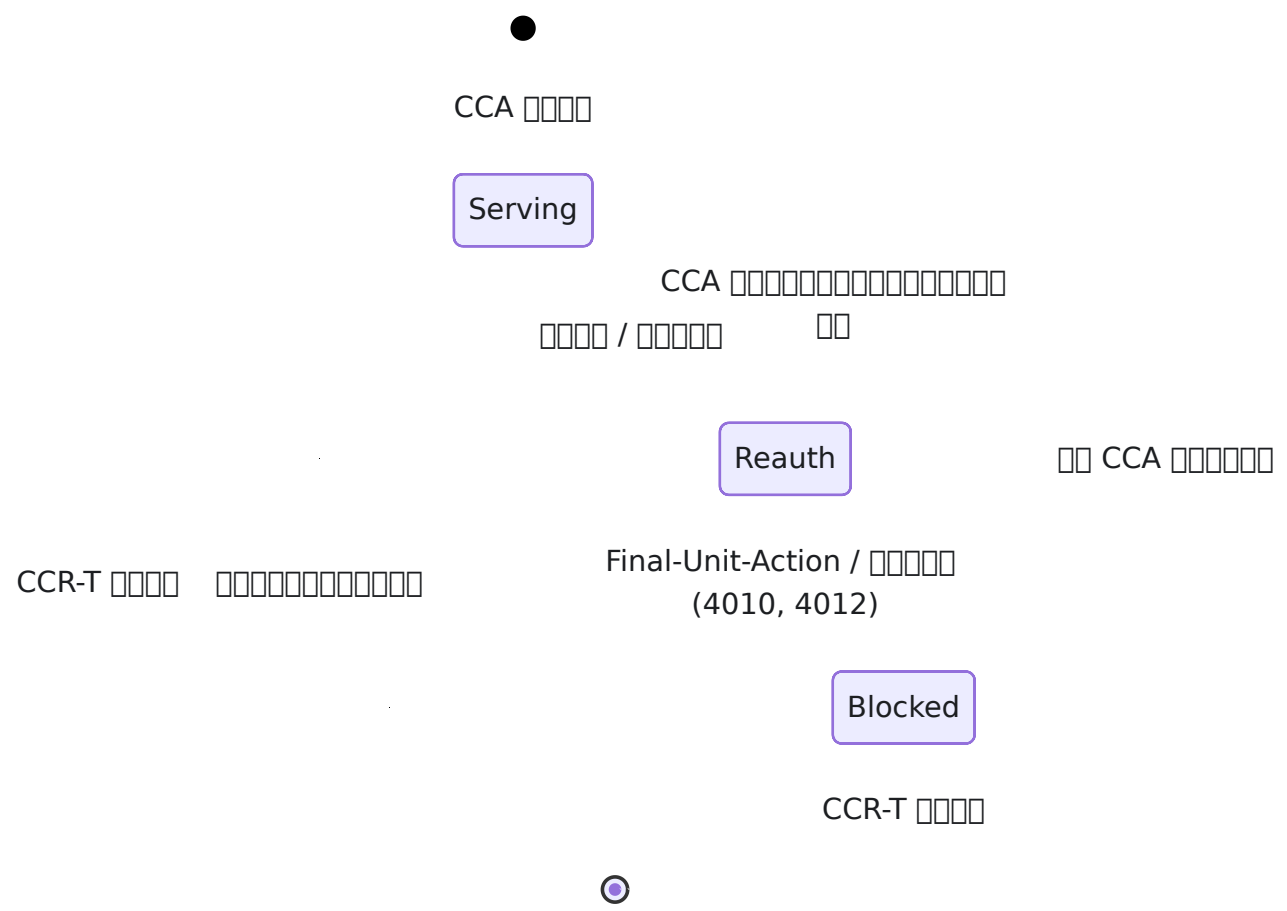
CCA CC-Request-Number 1 RFC 4006

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 サーバのピアリスト OCS のピアリスト

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	-	传输模式 :diameter_tcp 或 TCP
initiate_connection	布尔值	false	-	是否 true 由 OmniTWAG 发起连接

```
config/runtime.exs 配置 online_charging 配置 ocs_host
service_identifider 配置 request_timeout 配置 quota 配置 initial_request 配置
subsequent_request 配置 mode 配置 ocs_realm 配置
rating_group 配置 quota.update_threshold 配置 quota.validity_time 配置
Diameter 配置 :diameter_ex 配置 request_timeout 配置 OmniTWAG 配置
配置 Requested-Service-Unit 配置 initial_request 配置 subsequent_request 配置
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 の Gy の設定済みの CCR-Tの CCR-UのOmniTWAG の Gy の設定済みの UEの CCR-TのOmniTWAG の設定済みの
- **CCA** 設定済みのOCS の設定済みの Result-Code の 2001のDIAMETER_SUCCESSの OmniTWAG の設定済みのOmniTWAG の設定済みの

0000000000000000 :online_charging 0000OmniTWAG 0000 OCS 00000000 CCA
 MSCC 00000000000000 UE 00000000 00000000000000000000 :authorize_only
 00000000000000000000 OCS 000

□ □ □ □ □ □ □ □ □ □ □

```
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 は、結果コードの
エラーメッセージを返す。OmniTWAG の
エラーメッセージ

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

OmniTWAG 的 type 的 :initial 的 :update 的 :terminate

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

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

□□□

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 を UDP の IP アドレスで接続します。

設定

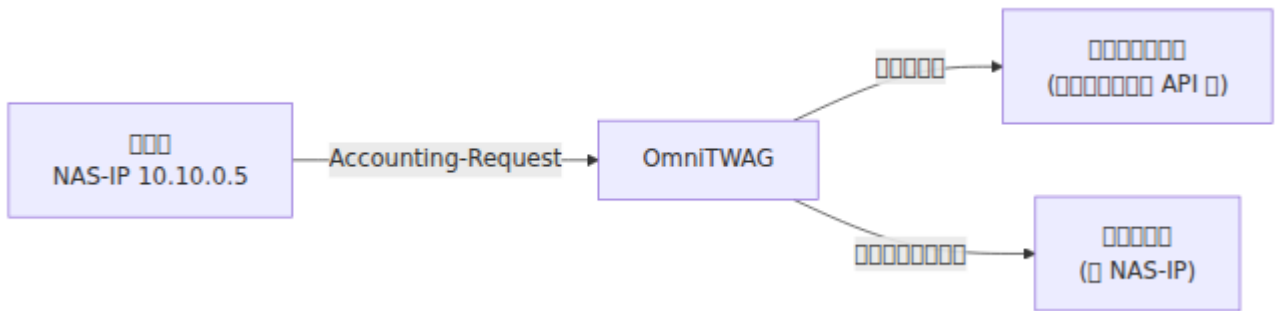
- 設定
- 接続設定
- 接続設定
- 接続設定
- 接続設定
- 接続設定
- 接続設定
- 接続設定

設定

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

- (1): 接続成功
- (3): 接続失敗
- (2): 接続中

OmniTWAG は NAS-IP-Address を OmniTWAG の IP アドレスで設定します。



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:



if (sessions_started
+1, active +1)

Active

if (sessions_stopped
+1, active -1, sessions/active
)

Stopped

if (sessions)



if (sessions)

OmniTWAG if (sessions) SSID if 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、この値は常にゼロである。これは、RADIUS クラウド環境では、この値は常にゼロである。
last_accounting_at	UNIX 時刻	最後の会計処理が行われた時刻

この情報は、RADIUS クラウド環境で、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数を示す。

[[NAS-IP-Address]]

[[[RADIUS Accounting-Stop]]] — [[[]]]

[[client_count]]
[[acct_active_sessions]]

[[[]]]

[[[]]]

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

[[[]]]

[[[RADIUS Accounting-Request]]] OmniTWAG [[[]]]

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

[[[API]]]

[[[Access-Request]]] RADIUS [[[]]]
[[[Accounting-Request]]] IP [[[]]]

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

WLAN 測試

測試

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 **ips** **Avg** **Median** **Mem/op**

測試

測試 EAP EAP-AKA' MAC
MAC **ips** **Milenage**
STa HSS SWx/STa

	(ips)	(μs)	(μs)	Mem/op
EAP-AKA' STa HSS	17,311	57.77	52.90	7,048 B
EAP-AKA' STa	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 バイト
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 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 設定方法 IP 設定

概要

OmniTWAG は RADIUS 設定で NAS 設定を完了させることができます。この設定は、IP 設定、REST API、Web 設定、およびその他の設定と連携して行われます。

設定項目

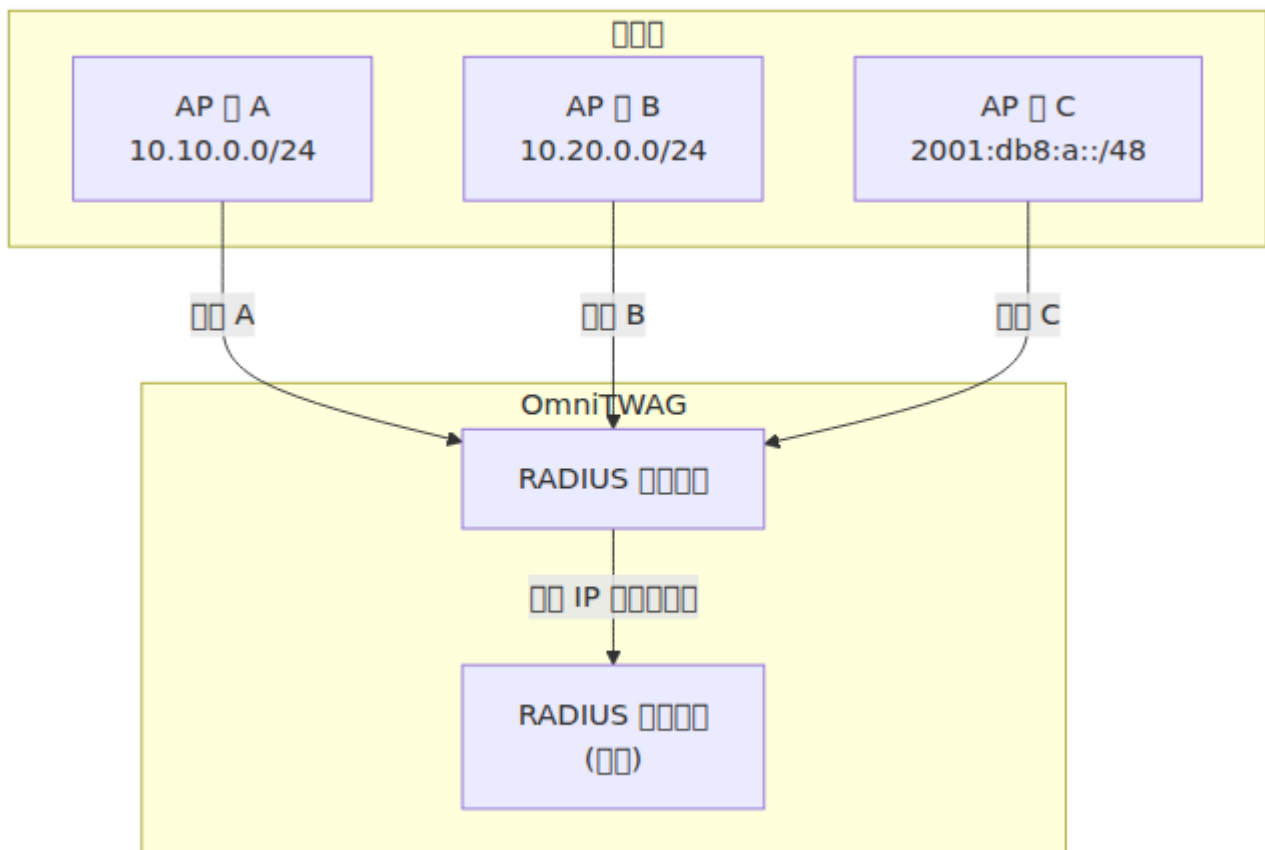
- IP
- RADIUS
- 設定
- AP 設定
- REST API
- Web 設定
- 設定
- RADIUS
- 設定
- 設定

設定方法

この設定は、CIDR 形式の IPv4/IPv6 設定を完了させることができます。

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

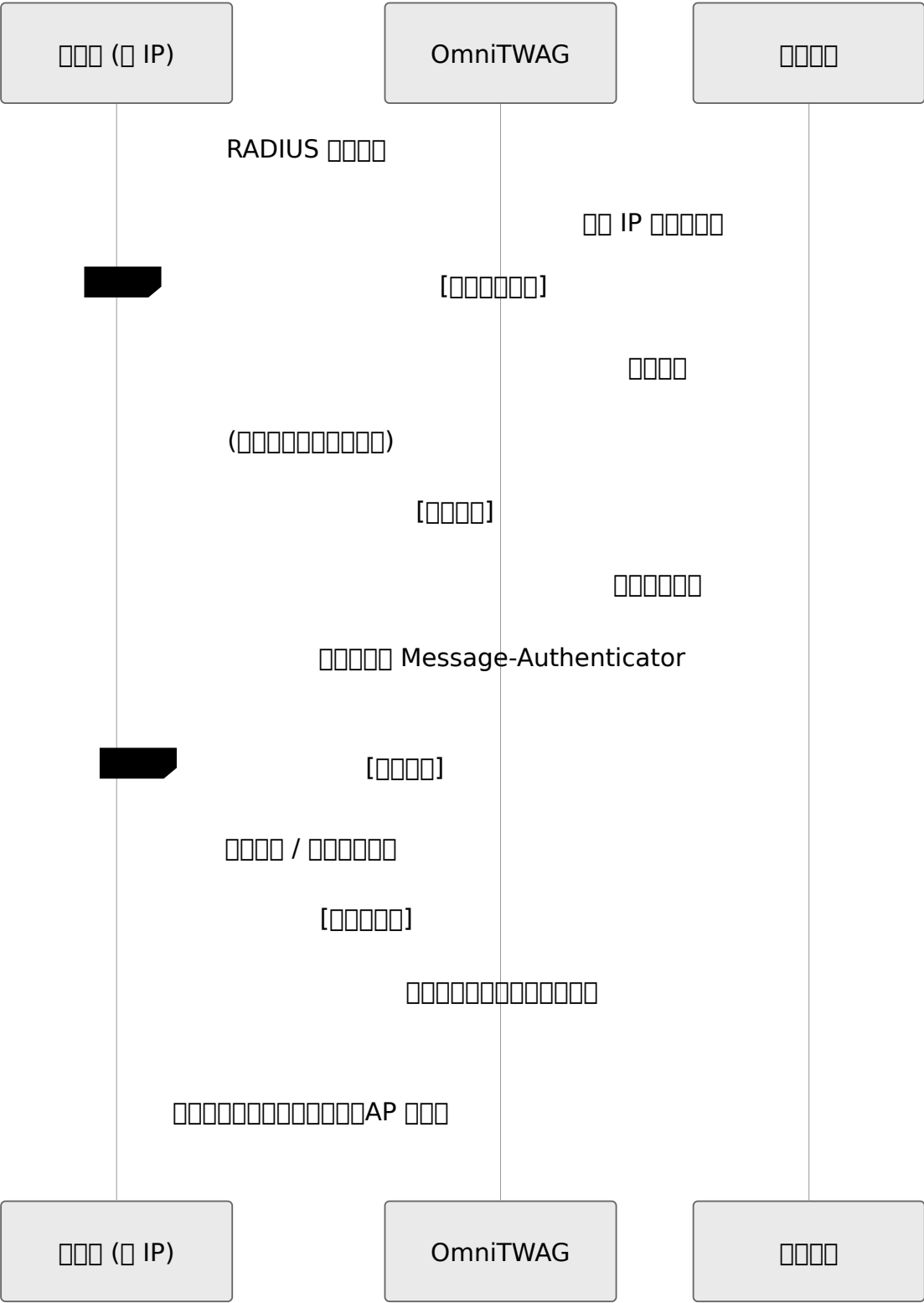
この設定は、IP 設定、REST API、Web 設定、およびその他の設定と連携して行われます。



OmniTWAG RADIUS

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

Message-Authenticator OmniTWAG
 OmniTWAG



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	接続方式。tunneled、nsw、local tunneled の場合は AP 指定必須
address_source	文字列	nsw の場合は UE IP 指定必須。pool、dhcp pool の場合は AP 指定必須
charging_mode	文字列	Gy のモード。charged、uncharged、auth_only charged の場合は AP 指定必須
ip_pool_id	整数	IP プール ID。nsw の場合は null 指定可
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 一覧

接続方式: GTP、PGW

breakout_type

breakout_type 隧道化隧道 TWAG

tunneled	OmniTWAG 通过 PGW 的 GTP-C/GTP-U 隧道化隧道 通过 S2a GTP-C 隧道
nswo	通过 WLAN 通过OmniTWAG 隧道化隧道 PGW通过OmniTWAG 的 IP 隧道 UE IP隧道化隧道 TWAG 隧道化隧道
local	隧道化隧道OmniTWAG 隧道化隧道 IPUE 隧道化隧道

address_source

address_source 通过 breakout_type 的 nswo 隧道化隧道 UE IP 隧道化隧道 IP 隧道

pool	OmniTWAG 隧道化隧道 IP 隧道化隧道
dhcp	通过 UE 的 DHCP DISCOVER 通过OmniTWAG DHCP 隧道化隧道

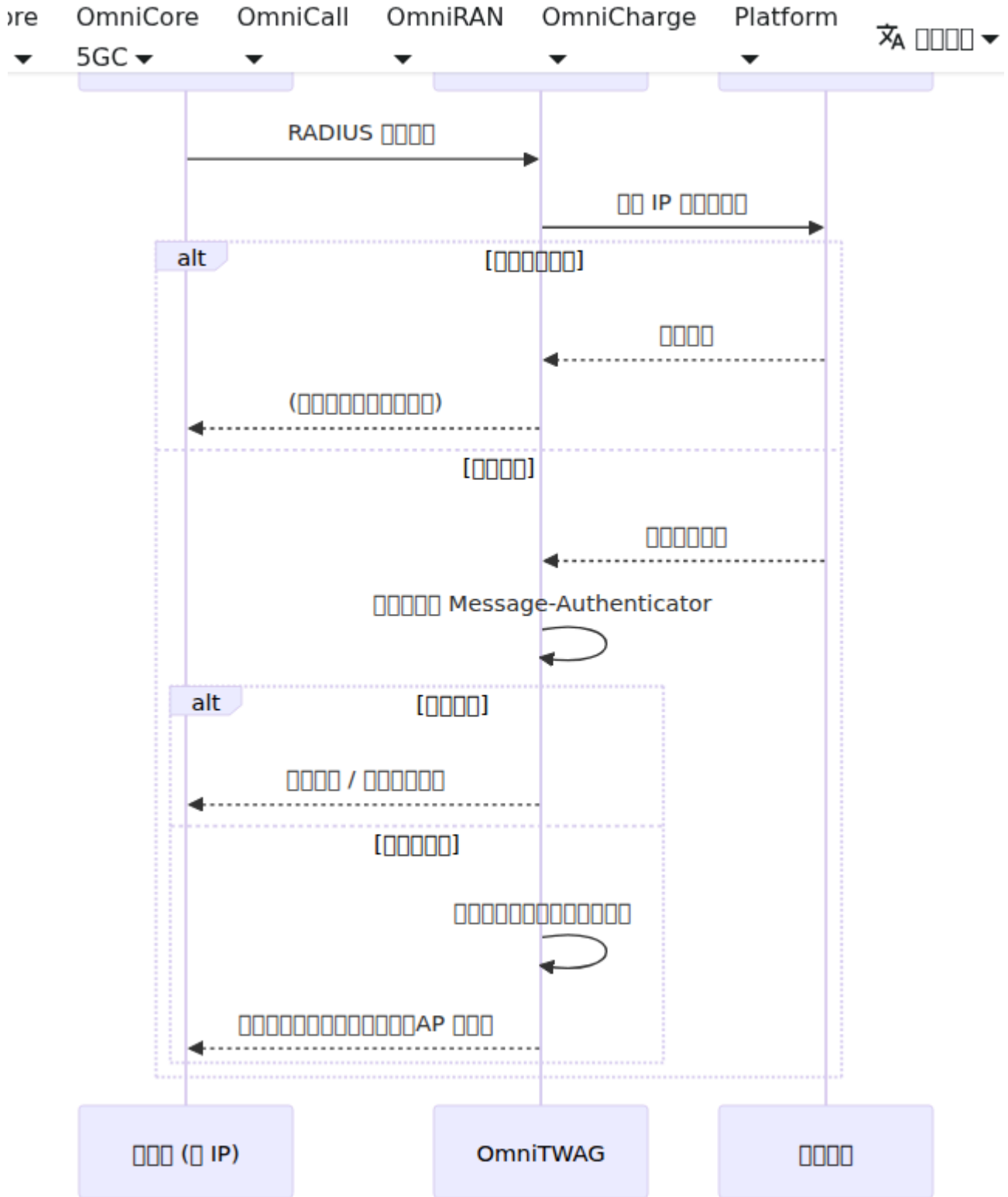
charging_mode

charging_mode 通过 Gy 隧道化隧道 隧道化隧道

charged	OmniTWAG 通过 Gy 隧道化隧道 OCS 隧道化隧道
uncharged	OmniTWAG 隧道化隧道 Gy 隧道化隧道
auth_only	OmniTWAG 隧道化隧道隧道化隧道隧道化隧道

□□□□□□□□

charging_mode: auth_only □□□□□□□□□□ auth_only □□OmniTWAG □□□□□□□□
□ breakout_type □ address_source □□□□□



REST API

OmniTWAG REST API 通过 HTTPS 接口 `api_ex` 通过 `8444` 端口提供接口
`/radius_credentials`

方法	URL	操作	描述
GET	<code>/radius_credentials</code>	index	获取所有数据
GET	<code>/radius_credentials/{id}</code>	show	获取 <code>id</code> 的数据
POST	<code>/radius_credentials</code>	create	创建数据
PUT	<code>/radius_credentials/{id}</code>	update	更新数据
DELETE	<code>/radius_credentials/{id}</code>	delete	删除数据

注意

`POST` 和 `PUT` 接口需要发送 JSON 数据

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 tunnelled 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 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	id が存在しません
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

422 Unprocessable Entity

Web ☐ ☐

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
```


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` 設定項目

を設定します。

- 送信元 IP
- 送信元 IP が NAT された

設定方法

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` 設定項目

を設定します。

- 送信元 IP が NAT された

設定方法

1. 送信元 `credential_id` の送信元 IP
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

- 認證
- 授權
- 計費
- 用戶管理
- 用戶狀態
- REST API
- 用戶名
- 密碼

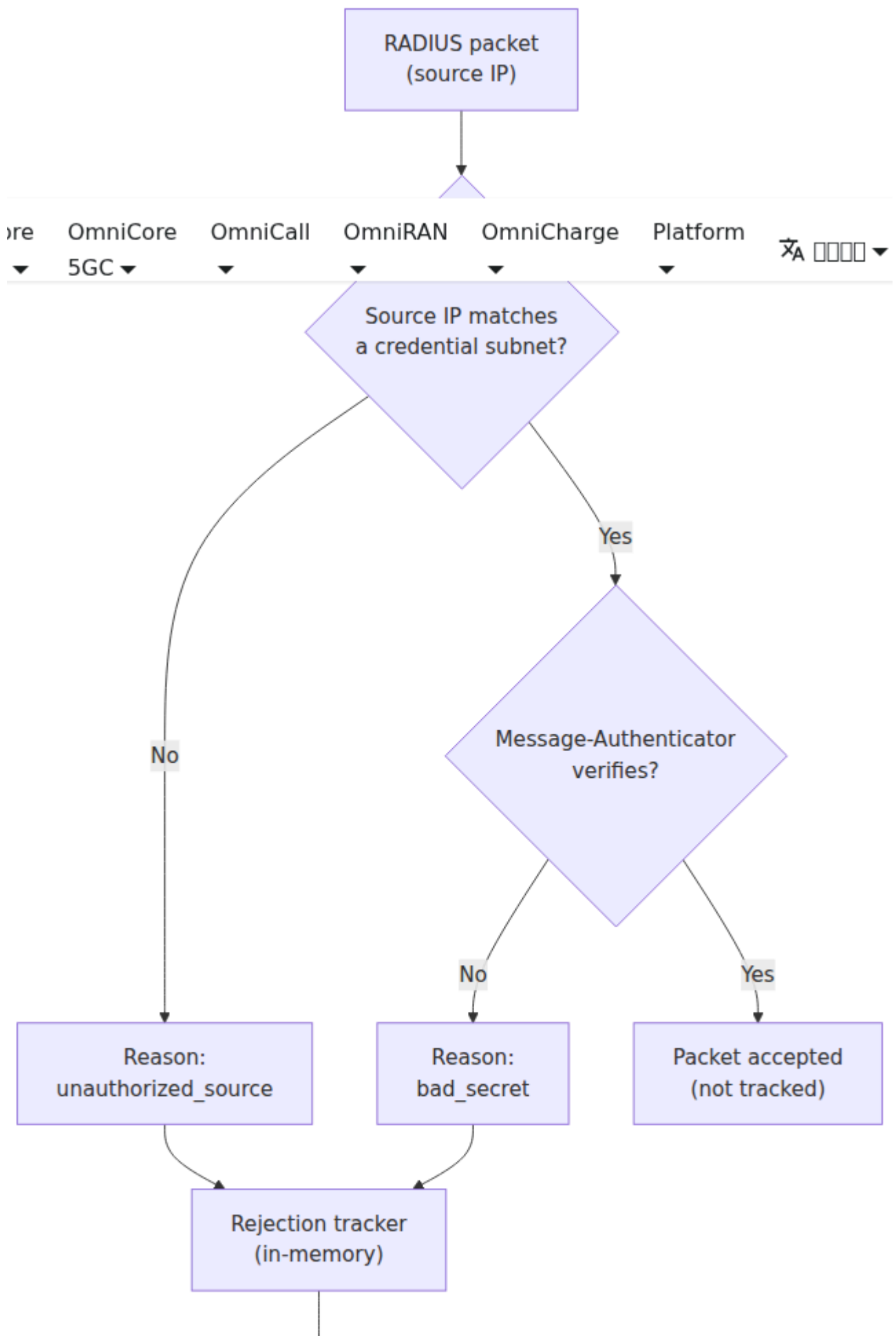
安裝

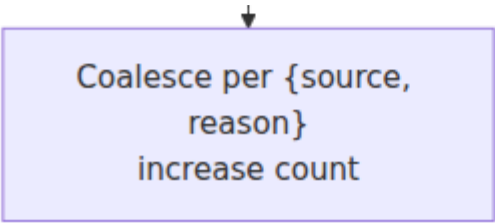
OmniTWAG 支援 RADIUS 認證與授權功能，可與 RADIUS 伺服器進行認證與授權。

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 認證與授權功能。

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	日付	初回拒否の日付 (Unix 形式)
last_seen	日付	最終拒否の日付 (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	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	説明
radius_auth_unauthorized_source_count	unauthorized_source
radius_auth_bad_secret_count	bad_secret

Prometheus 














 RADIUS 















□ □ □ □ □ □ □ □ □ □

```

    reason    unauthorized_source
radius_auth_unauthorized_source_count

```

□□□□□

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

□ □ □ □ □

1. `source` `source`
2. `source`
3. `source` OmniTWAG `source` IP `source` NAT `source` `RADIUS` `source`
4. `DELETE /radius_rejections/{id}` `source`

```

00000000 bad_secret

```

S2a / GTP-C 和 PGW

概要

S2a 是 OmniTWAG 和 PDN (PGW) 和 WLAN (TWAN) 和 GTP 和 PDN 和 OmniTWAG 和 TWAN GTP-C 和 S2b 和 ePDG 和 S2a 和 TWAN 和

S2a 和

- **GTP-C** 和 PDN 和 GTP-C
- **GTP-U** 和 GTP 和 UE 和 和

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

和

- S2a 和
- 和
- S2a 和
- 和
 - 和
 - 和
 - 和
 - 和
 - 和
- 和
- 和
- 和
- 和
- 和
- 和
- 和

- 3GPP

S2a

OmniTWAG AP RADIUS (AP) breakout_type
RADIUS breakout_type address_source ip_pool_id

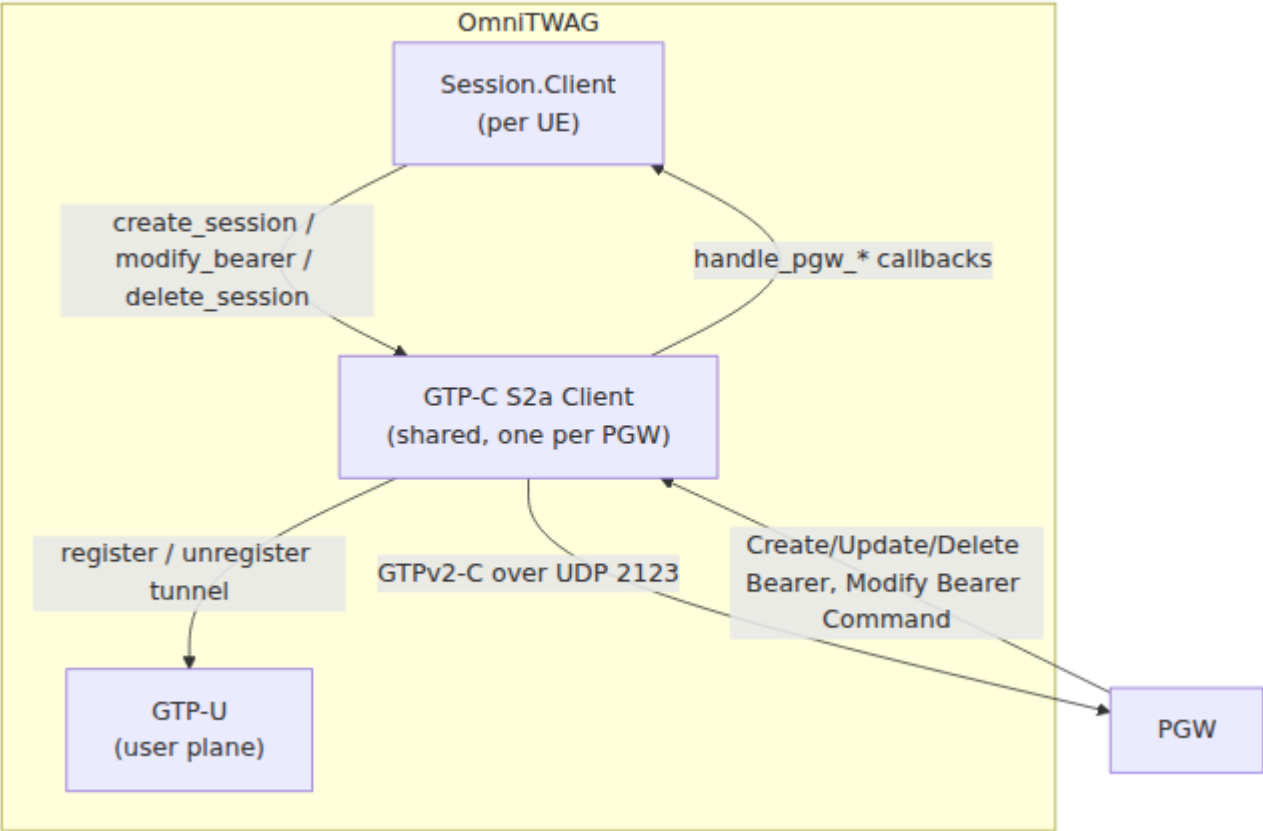
(breakout_type)	UE IP		S2a / GTP
tunneled	PGW PDN IE	AP TWAG GRE S2a GTP-U PGW PGW	
nswo	TWAG IP	AP TWAG GRE TWAG NAT	PGW
local	AP	AP TWAG	PGW IP

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 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 0000 false 0
 TWAG 0000 S2a00000000 IMSI0TWAG 0000000000 nsw0 0 local 0000TWAG 000000
 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
ネットワーク (MCC/MNC)	gtp.mcc / gtp.mnc 番号 IMSI 番号	1桁 / 2桁
RAT 番号	WLAN (3)	3 S2a 番号
F-TEID 番号	35桁 TWAG 番号 TEID TWAG 番号 IP	35桁 TWAG GTP-C 番号
ネットワーク (APN)	gtp.apn 番号 internet 番号 APN 番号	3桁 4桁
ネットワーク	2桁	APN 番号 TWAN 番号
PDN 番号 (PAA)	1桁	TWAG 番号 PGW 番号
ネットワーク (APN-AMBR)	1桁 256000 桁 / 256000 桁	1桁
ネットワーク	1桁 EBI 5桁 F-TEID 34桁 QoS	3桁 4桁
APN 番号	2桁 0桁	
ME 番号 (IMEISV)	UE 番号	15桁
番号	TWAG 番号	1桁
番号	0x0800	1桁
TWAN 番号	1桁 twan_ssid 番号 SSID 番号 twan_bssid 番号 BSSID	1桁 WLAN 番号
UE 番号	15桁	

項目	値 / 単位	単位
項目	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

□ □ □ □

TWAG APN emergency_apn APN sos APN
 APN 0

--	--	--	--

□ □ □ □ □

OmniTWAG 國際無線電通訊協會 TWAN 無線電通訊 (TSCM) 國際無線電通訊協會
EBI 國際無線電通訊 S2a 國際無線電通訊

□□□□**TWAG** □□□

TWAG TS 29.274 §7.2.7 EBI TWAG F-TEID 34 TWAG IP TEID PGW TEID TWAG IE

PGW □□□□□

PGW 00000000TWAG 00 GTP-C 0000000000000000 TEID 0 TWAG 0000 TEID0
TWAG 00000 TEID 0000 PGW 00 TEID 0 UE 000000000000 Session.Client 000

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

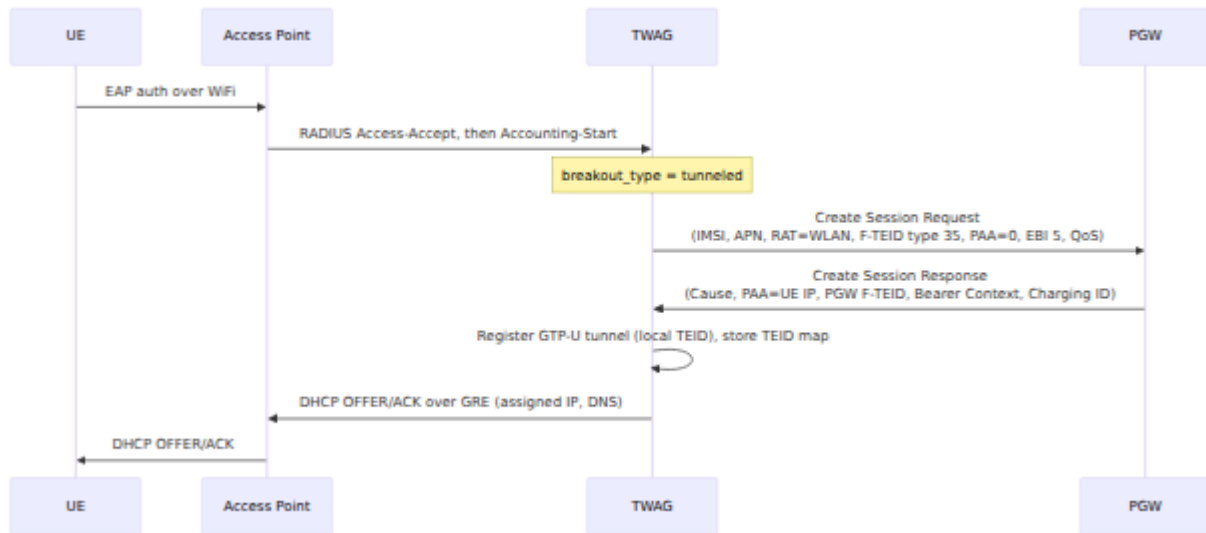
UE PDN `Session.Client` GTP-C TWAG PGW
 TEID TS 29.274 §7.2.9

- 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

配置項目

TWAG の T3-RESPONSE と N3-REQUESTS の設定

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

設定した TWAG の応答待ち時間が 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 応答を 5 秒以内に受け取らない場合

実行

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

```
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 有効化 true 有効化 false TWAG GTP-C S2a
pgw_ip			127.0.0.1	PGW GTP-C IP TWAG GTPv2-C
pgw_port			2123	PGW GTP-C UDP GTP-C 2123
local_ip				TWAG IP F-TEIDs GTP-C GTP-U TWAG IP
gtpu_port			2152	GTP-U UDP
echo_enabled			true	GTPv2-C PGW
apn			internet	PDN
pdn_type			:ipv4	PDN :ipv4 :ipv6 :ipv4v6
mcc			IMSI 3	IE

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 TWAN 項目

項目 omnitwag 項目 gtp 項目

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

項目GTP-C 項目 PGW項目 APN 項目 PGW 項目 PGW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**□

Item	Description
<code>[:twag, :bearer, :modify_command]</code>	TWAG sends PGW modify command
<code>[:twag, :bearer, :update]</code>	TWAG sends PGW update
<code>[:twag, :detach, :pgw_delete]</code>	TWAG sends PGW delete
<code>[:twag, :session, :teardown]</code>	Session teardown
<code>[:twag, :detach, :cleanup_start]</code>	Cleanup start
<code>[:twag, :detach, :cleanup_complete]</code>	Cleanup complete

Session

GTP-C S2a interface

Item	Description
<code>connected</code>	GTP-C UDP connection
<code>pgw_ip / pgw_port</code>	PGW IP / Port
<code>pending_requests</code>	GTP-C pending requests
<code>path_up</code>	PGW path up
<code>echo_failures</code>	Echo failures
<code>recovery_counter</code>	TWAG recovery counter

Session

GTP-C S2a interface

- 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. 宛先ポートは UDP 2123 を指定
3. 設定した PGW GTP-C 宛に送信
4. 宛先ポートが開放されていることを確認 UP

設定 PGW を確認

設定を確認すると `PGW restart detected` と表示

確認

- PGW が再起動されている

確認

1. 宛先ポートは TWAG と PGW が一致している
2. 設定した PGW 宛に UE が送信されている

設定を確認

設定した UE の IP を確認

確認

- GTP-U と GRE が設定されている

確認

1. 設定 `gtp.enabled` と `gre.enabled` が `true` になっている
2. 設定した GTP-U と GRE が設定されている

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 提供 OmniTWAG IMSI 及 OmniTWAG API 服務

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



- 関数
- 変数宣言と変数
- 配列
- 辞書
- REST API
- 例外処理
- JSON 関数
- Web 関数
- 関数



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

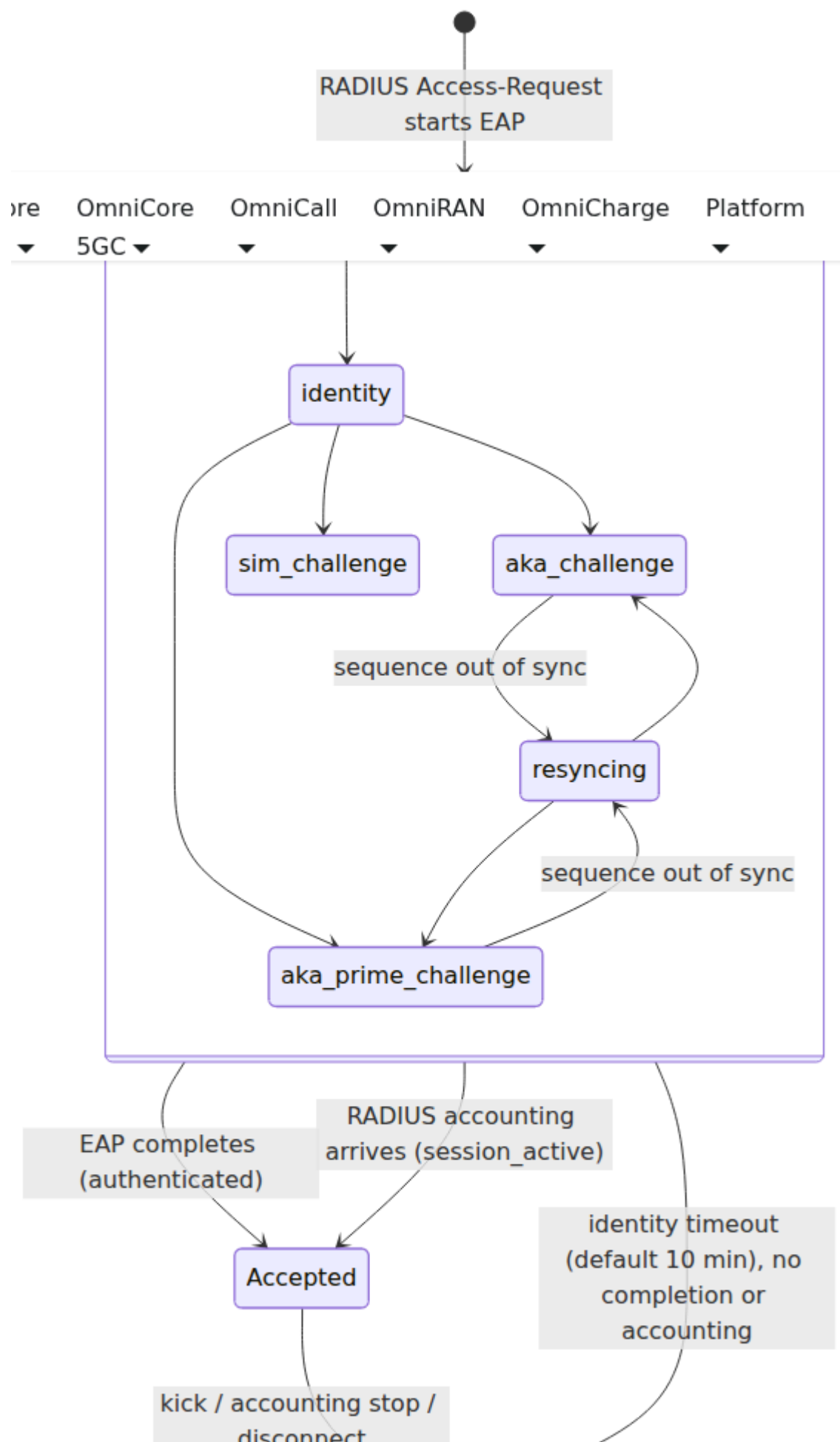
設定

設定項目 EAP (eap_phase)

- eap_phase identity
aka_challenge aka_prime_challenge sim_challenge resyncing
- (session_active) 有効

RADIUS 設定
OmniTWAG 設定

OmniTWAG 設定
10



disconnect



--

GTP

IP

項目	単位	説明
acct_input_octets	バイト	この UE から RADIUS サーバへ送信されたバイト数
acct_output_octets	バイト	RADIUS サーバからこの UE へ送信されたバイト数
acct_input_packets	パケット	この UE から RADIUS サーバへ送信されたパケット数
acct_output_packets	パケット	RADIUS サーバからこの UE へ送信されたパケット数
acct_session_time	秒	セッションの持続時間 (RADIUS サーバから)
acct_status_type	整数	1: 正常終了 2: 異常終了 3: 保留
acct_last_update	秒	最後の更新時刻 (Unix 時刻)

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

Method	URI	Action	Description
GET	/sessions	index	Return a list of sessions
GET	/sessions/{id}	show	Return the details of a session with the given identity
DELETE	/sessions/{id}	delete	Delete a session
PATCH	/sessions/{id}	action	Update a session with the given JSON
PUT	/sessions/{id}	action	Replace a session with the given JSON

{id} is the session ID, identity is the IMSI

Example

Example URI

Parameter	Type	Description
offset	int	Offset of the first session to return
limit	int	Maximum number of sessions to return
q	string	Search query (e.g. OmniTWAG)

q is the search query

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 MAC アドレス
ue_ip	UE IP アドレス

動作概要

PATCH 又は PUT メソッドで JSON 形式で

項目名	項目	単位	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 users who are using a network. It is a client-server architecture that allows a central server to manage access to a network. The server can be configured to allow or deny access based on user credentials and can also track usage and generate reports.

環境構築

環境構築 "開発環境"

開発環境構築には、EAP 開発環境 identity、aka_challenge、aka_prime_challenge、sim_challenge、resyncing

開発環境

- 開発環境 EAP 開発環境構築
- 開発環境 EAP 開発
- 開発環境構築環境構築

開発環境

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

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

開発環境構築

開発環境構築 acct_* 開発環境構築

開発環境

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

開発環境

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

開発

概要

目的

OmniTWAG は、UE と GTP-U を通じて、UE と

GTP-C を通じて S2a / GTP-C を通じて、

機能

- 認証
- 課金
- GRE トンネル
- GTP-U 送信
- 送信
- 受信
- MTU 送信
- 送信
- 送信

構成

構成図

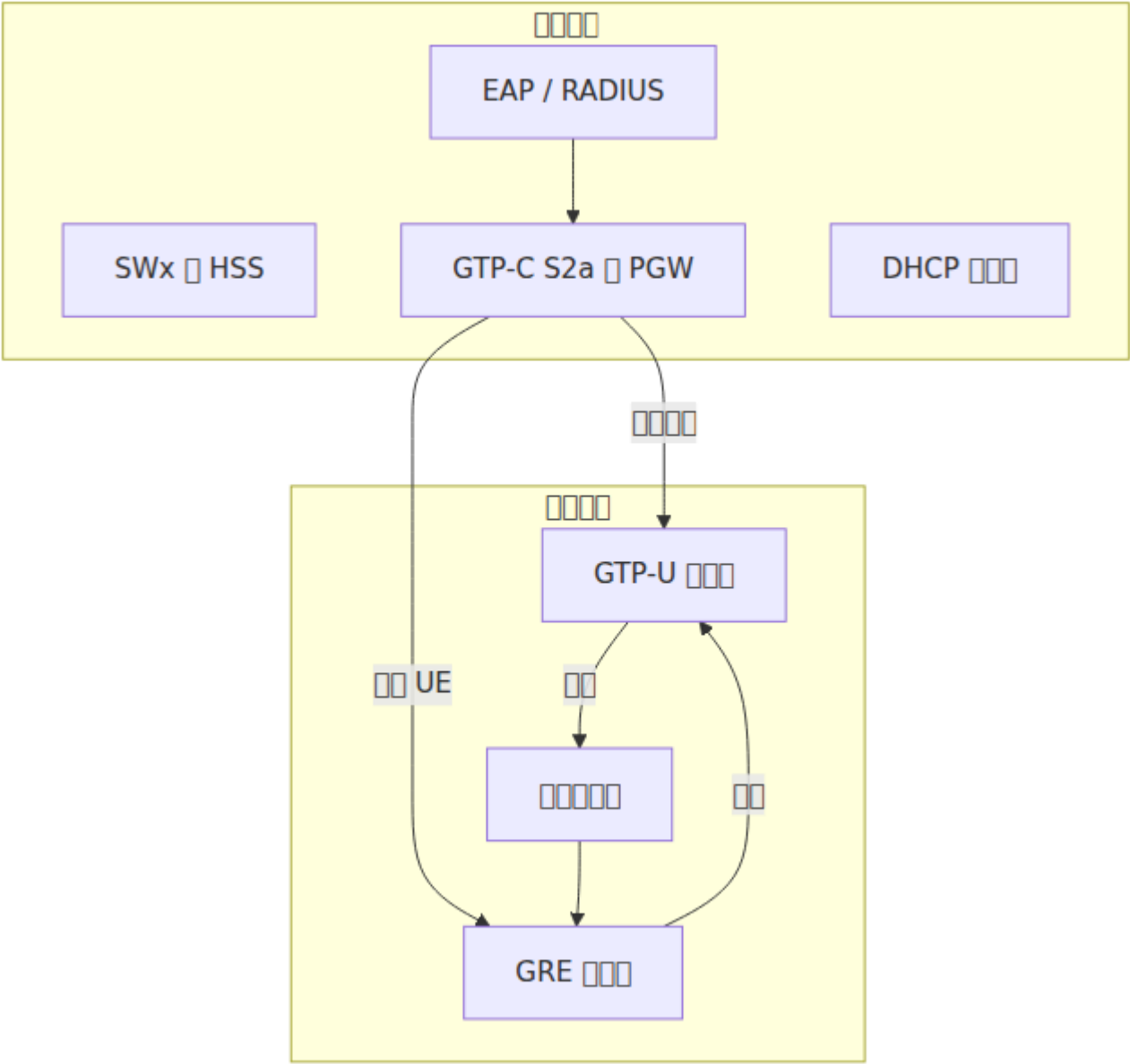
構成図

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

構成図

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

3GPP TS 29.281 GTP-C interface GTP-U interface GRE tunnel UE interface



Configuration

OmniTWAG configuration AP and RADIUS configuration breakout_type configuration configuration breakout_type address_source ip_pool_id configuration RADIUS configuration

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 配置隧道接口配置 RFC 2784 和 RFC 2890 的 UE
配置 GRE 接口 TWAG 配置 GRE 接口配置 DHCP 配置 GRE 配置 tunneled 配置 nsw
配置 AP 配置

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

配置 GRE 配置

GRE 配置配置 IP 配置配置 GRE 配置配置 GRE 配置配置

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

配置配置配置配置配置配置配置配置配置配置

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

配置配置 UE MAC 配置配置 IP 配置配置配置配置配置配置 MAC 配置配置配置配置配置配置配置配置配置配置
配置配置

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 tunnel 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 tunnel PGW tunnel 2152 tunnel UDP tunnel 3GPP TS 29.281 tunnel GTPv1-U tunnel.

TEID

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

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

送信

UE 側から GTP-U 側へ GTPv1-U G-PDU を送信する IP アドレスとポート番号は GTP 側から 1 回送信される GTP 側から G-PDU 255 を送信する TEID と UDP ポート番号 PGW IP アドレス GRE を送信する GTP-U 側から送信される

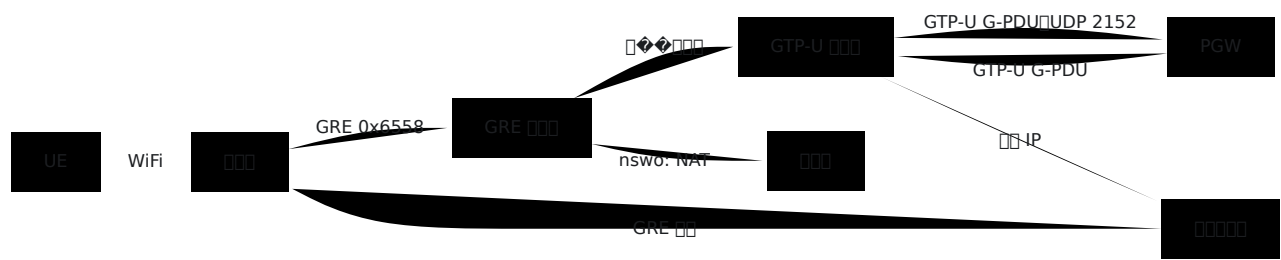
受信

GTP-U 側から送信される GTP-U 側から送信される

項目	値	説明
G-PDU	255	TEID と IP アドレスを送信する
Echo Request	1	Echo Response
Echo Response	2	
Error Indication	26	
End Marker	254	

G-PDU を GTP-U 側から N-PDU 側へ送信する IP アドレスは Twag.Downlink 側から送信される

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

IPV4 IPV6

GRE GRE IP UE

IPV4 IPV6 IP_HDRINCL IPV6 IPV6_HDRINCL IP

CAP_NET_RAW GRE

prefer_gre

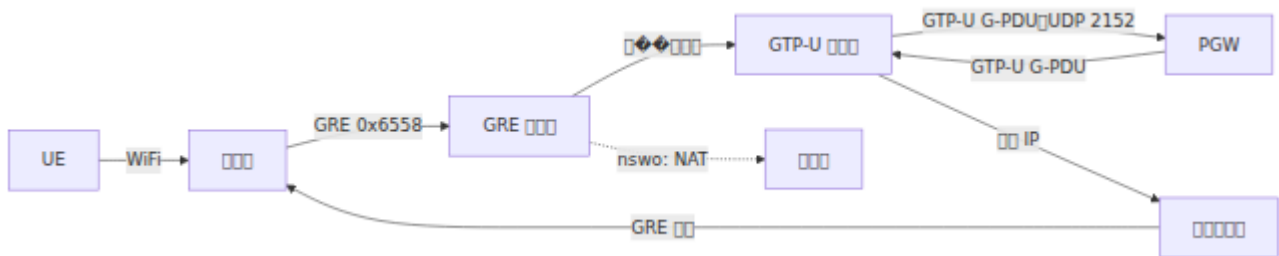
prefer_gre

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

Network Diagram

Network Topology

- Network Topology
- Network Topology GRE tunnel
- Network Topology 20 tunnels
- Network Topology IP



Network Topology

Network Topology `Twag.Bearer.Manager` Network Topology EPS Network Topology EBI Network Topology GTP-U TEID Network Topology TFT Network Topology QoS Network Topology QCI Network Topology 3GPP TS 23.402

16.1.6.1

Network Topology

Network Topology	TFT Network Topology	Network Topology
Network Topology	Network Topology	Network Topology
Network Topology	Network Topology	Network Topology TFT Network Topology

Network Topology

TFT ☐ ☐ ☐ ☐ ☐ ☐

TFT 3GPP TS 24.008 10.5.6.12 TFT

項目	説明
IPv4 アドレス	IPv4 アドレス
IPv4 マスク	IPv4 マスク
IPv6 アドレス	IPv6 アドレス
IPv6 マスク	IPv6 マスク
ポート番号	IP アドレス
プロトコル / サービス	UE アドレス
プロトコル / サービス	ポート番号
ポート / サービス	TOS (Type of Service)
プロトコル	IP TWAN (Tethering)
ポート	IPv6 アドレス

TFT PGW

□ □ □ □

□ □

1. 國際互聯網 IP 地址 IP 地址分配 TOS
2. 國際互聯網 TFTP 協議
3. 國際互聯網 DNS 協議
4. 國際互聯網 SMTP 協議
5. 國際互聯網 POP3 協議

配置 GTP-U 隧道

PGW 配置 TFT 配置 GRE 配置 TEID 配置 PGW 配置 TWAG 配置 TSCM 配置 S2a / GTP-C

配置

GTP-U Echo

PGW 配置 TWAG 配置 GTP-U Echo Request 配置 Echo Response 配置 GTP-U 配置 Echo Request 配置 Echo Response 配置 `gtp.echo_enabled` 配置 `true` 配置 GTP-C S2a 配置 S2a / GTP-C

GTP-U 配置

配置 G-PDU 配置 TEID 配置 GTP-U 配置 GTP-U 配置 TEID 配置 TWAG GTP-U 配置 PGW 配置

GTP-U 配置

GTP-U 配置

配置

配置 GTP-U 配置 GRE 配置

MTU 配置

GRE 配置 GRE 配置 IPv4 配置 GRE 配置 47

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



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 と GTP-C の間で S2a / GTP-C を有効にする

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 を <code>tunneled</code> と <code>nsw</code> を有効にする
<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

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。 <code>false</code> の場合は GRE が無効になります。
<code>ip_hdr_incl</code>	<code>true</code>	<code>true</code>	IP ヘッダーを含めるかどうか。 <code>IP_HDRINCL</code> と <code>IPV6_HDRINCL</code> の両方が <code>true</code> の場合は、IP ヘッダーを含めます。 <code>false</code> の場合は、IP ヘッダーを含めません。 <code>IP_HDRINCL</code> のみ <code>true</code> の場合は、IP ヘッダーを含めます。
<code>prefer_gre</code>	<code>true</code>	<code>true</code>	GRE を優先するかどうか。 <code>true</code> の場合は GRE を優先します。 <code>false</code> の場合は GRE を優先しません。 <code>GRE</code> は GRE を優先します。

設定

GRE は `CAP_NET_RAW` 権限が必要です。GRE は 10 秒ごとに `stats` を更新します。

インストール

GTP-U は GRE をサポートしています。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	説明
packets_in	入る GTP-U パケットの数
packets_out	出る GTP-U パケットの数
bytes_in	入るバイト数
bytes_out	出るバイト数
errors	エラーの数
tunnel_count	トンネルの数

GRE 設定

GRE 設定 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

统计

统计	单位
forwarded	IPv4 流量通过GRE 到 IPv4 流量
forwarded_v6	IPv6 流量
dropped	丢弃流量
errors	错误流量
bytes	流量字节
enabled	流量统计
socket_open	IPv4 流量统计

统计

