OmniTWAG □□□□□□□

□ **Omnitouch** □□

□□□□□□□□□□□□□□□□□□□□□□□ OmniTWAG □□□□

□□

□□

OmniTWAG□□□□□ WiFi □□□□□□□□□□□ 3GPP TWAG □□□□□□□□□□□□□□□□□□□□

- 零配置：与兼容的 SIM 一起开箱即用
- 无缝体验：像蜂窝网络一样自动连接
- 安全：始终使用加密的 WiFi（WPA2）
- 无密码：基于 SIM 的认证

对于移动运营商：

- 网络容量缓解：减少蜂窝基站的负载
- 受控卸载：只有授权的用户才能连接
- 改善用户体验：WiFi 通常提供更高的带宽
- 成本效率：WiFi 基础设施比蜂窝网络便宜
- 一致的身份：WiFi 和蜂窝使用相同的 IMSI
- 计费集成：如有需要，可以对 WiFi 使用进行收费

对于场所/企业：

- 运营商级安全：没有密码共享的风险
- 可扩展性：支持数千个用户，无需手动配置
- 简化管理：无需分发 WiFi 密码

什么是 **WiFi** 卸载？

WiFi 卸载允许移动网络运营商将用户数据流量从拥挤的蜂窝网络重定向到 WiFi 网络。

TWAG 如何实现卸载

TWAG 作为认证网关，连接：

- **WiFi** 接入点（通过 RADIUS 协议）
- 移动核心网络 HSS/HLR（通过 Diameter SWx 接口）

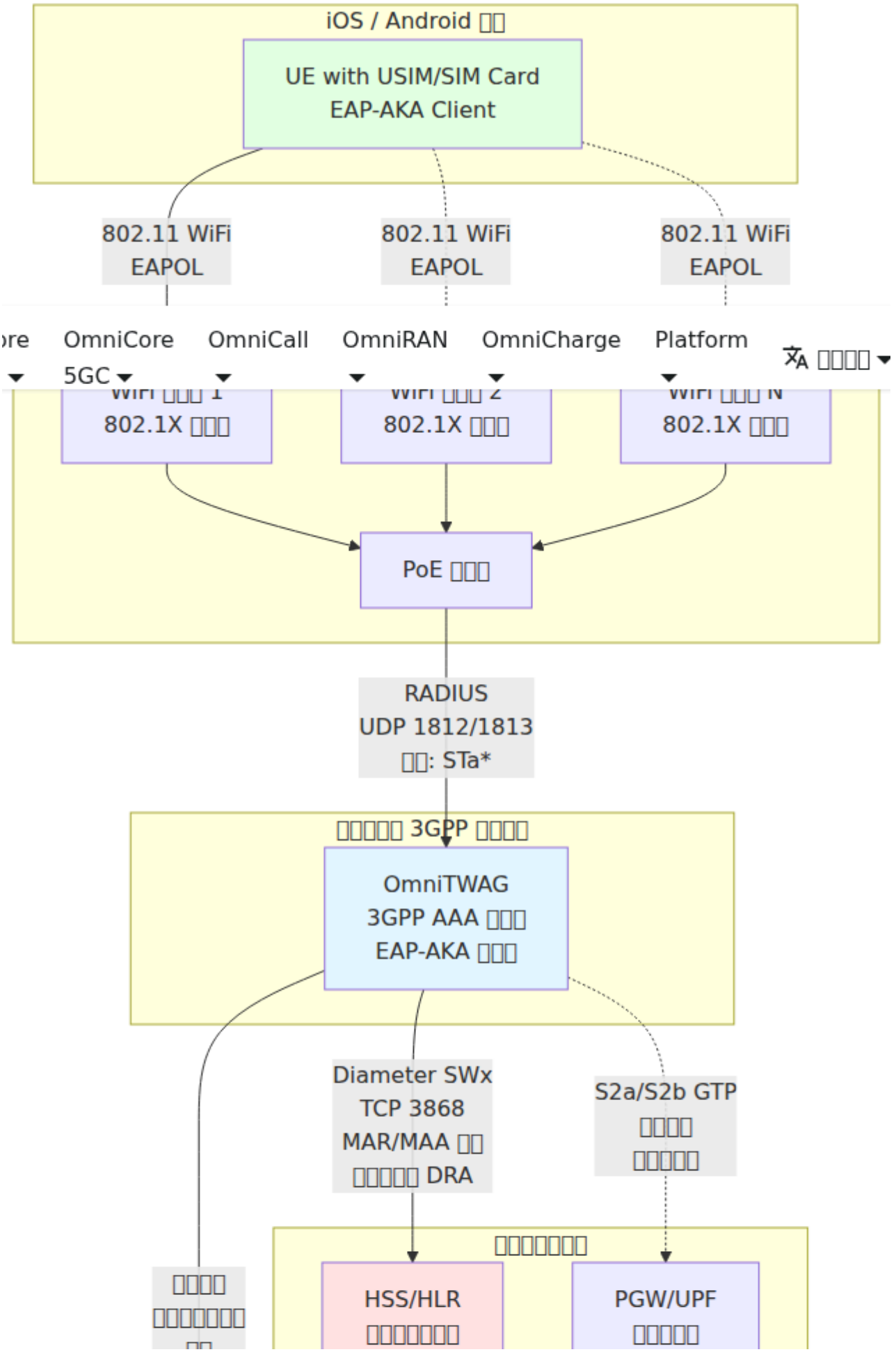
当用户的设备连接到配置为卸载的 WiFi AP 时：

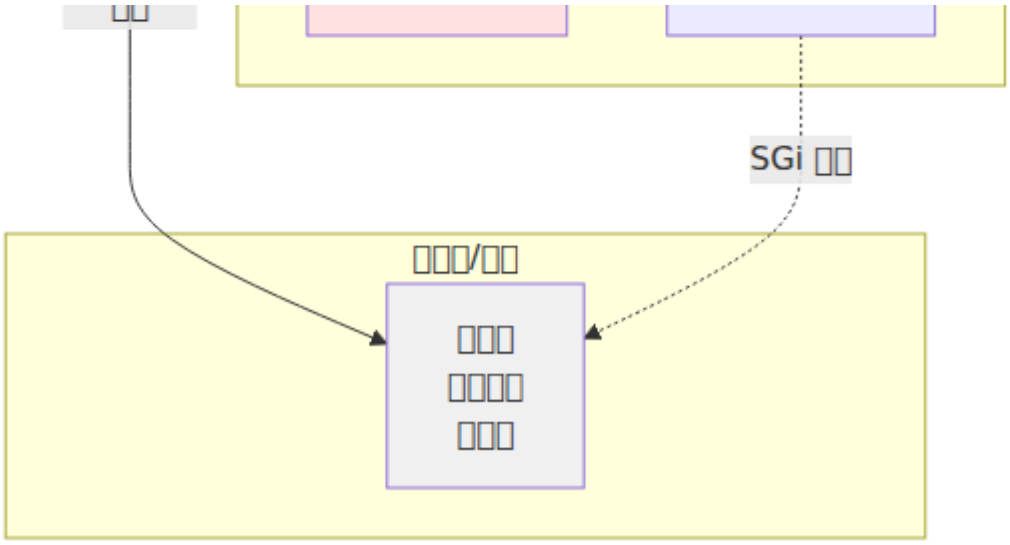
1. 设备使用其 IMSI（来自 SIM 卡）进行身份识别
2. WiFi AP 通过 RADIUS 将认证请求转发给 TWAG

3. TWAG 与运营商的 HSS 通信以检索认证向量
4. 设备与 TWAG 之间进行 EAP-AKA 挑战-响应认证
5. 认证成功后，设备获得 WiFi 访问权限
6. 可选择性地，流量可以隧道回移动核心或□地断开

□□□□

□□□□


iOS / Android □□
UE with USIM/SIM Card
EAP-AKA Client
802.11 WiFi
EAPOL
802.11 WiFi
EAPOL
802.11 WiFi
EAPOL
ore
OmniCore
5GC
OmniCall
OmniRAN
OmniCharge
Platform
802.1X □□□
802.1X □□□
802.1X □□□
PoE □□□
RADIUS
UDP 1812/1813
□□: STa*
□□□□□ 3GPP □□□□
OmniTWAG
3GPP AAA □□□
EAP-AKA □□□
Diameter SWx
TCP 3868
MAR/MAA □□
□□□□□ DRA
S2a/S2b GTP
□□□□
□□□□□
□□□□□□□□
HSS/HLR
□□□□□□□
PGW/UPF
□□□□□

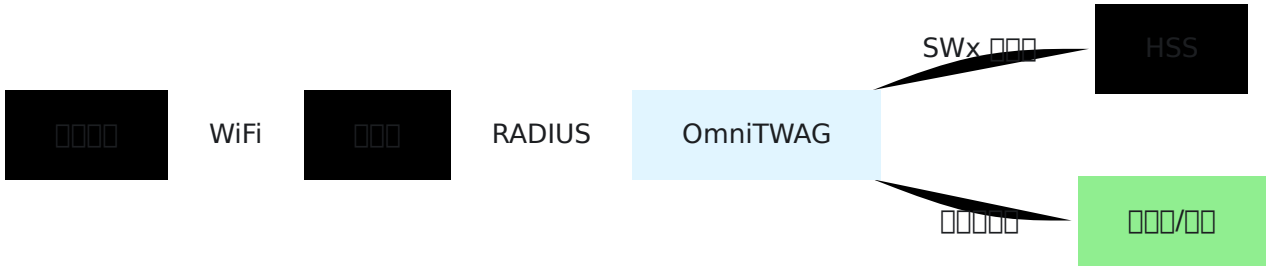


接口图例：

- **STa***: WiFi AP 和 TWAG 之间的 RADIUS/Diameter 接口（非 3GPP 到 AAA）
- **SWx**: TWAG（3GPP AAA 服务器）与 HSS 之间的 Diameter 接口
- **S2a/S2b**: 用于回程到家庭网络的 GTP 隧道接口（可选）
- **SGi**: 到外部数据包数据网络（互联网）的接口
- **802.11**: WiFi 无线接口
- **EAPOL**: EAP over LAN（802.1X 认证）

部署场景

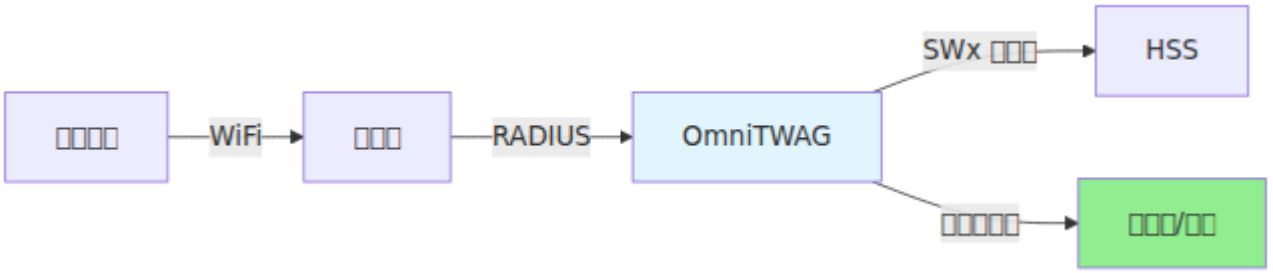
场景 **1**：本地断开（推荐用于性能）



□□□

- □□□□□□□□□□□□□□□□
- □□□□□□□□□
- □□□□□□□□□□□□□□□□
- □□□□□□□□□

□□ 2□□□□□□□□□GTP □□□□□



□□□

- □□□□□□□□
- □□□□/□□
- □□ VPN/□□□□□□
- □ WiFi □□□□□□□□□

SWx □□□□

□□ 1□□□□□□ **HSS**

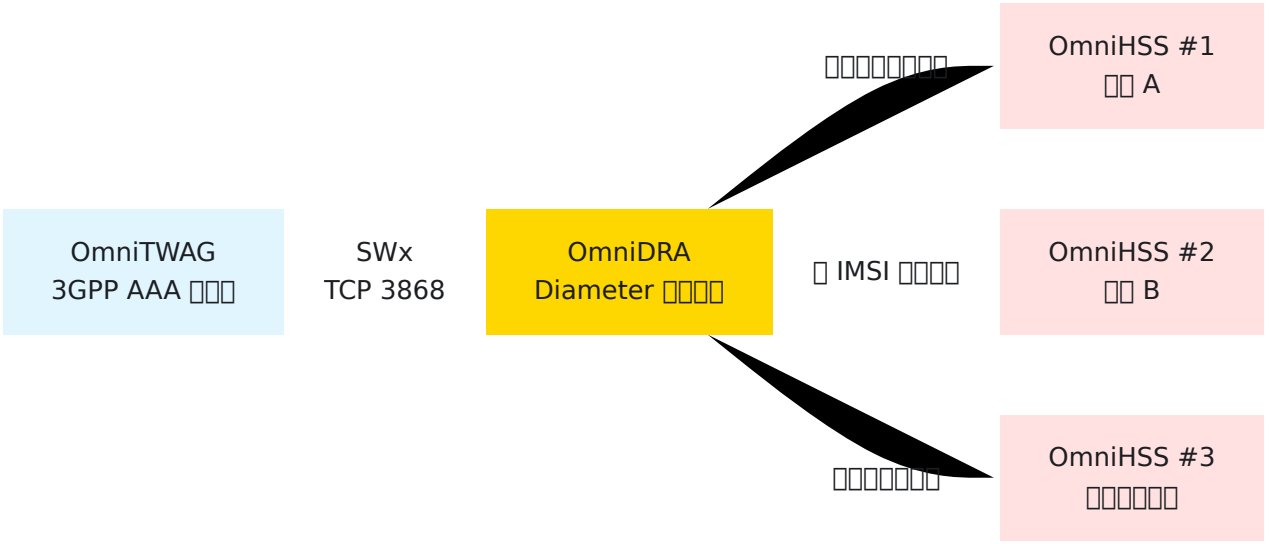
OmniTWAG 3GPP AAA □□□	□□ SWx TCP 3868 MAR/MAA	OmniHSS □□□□□

□□□□□□□□□□□□□□□□□□ HSS

好处：

- 更低的延迟（无需通过 DRA）
- 简化配置
- 更容易故障排除

选项 **2**：通过 **DRA**（**Diameter** 路由代理）



用例：多 HSS 部署、漫游场景、大规模网络

好处：

- 集中路由逻辑
- 在多个 HSS 之间负载均衡
- 支持漫游（路由到家庭 HSS）
- 冗余和故障转移
- 会话粘性

计费流程

TWAG 可以完全集成以向在线计费系统（OCS）发送基于 Diameter `Gy` 的在线计费请求。

这允许对 WiFi 上消耗的所有数据进行计费，并与客户的余额进行核对，通过 AP 以 RADIUS 发送，并由 TWAG 转换为 `Gy` 并转发给 DRA/OCS。

在所有模式下，使用情况由 TWAG 指标跟踪。

计费模式

TWAG 支持三种在线计费模式：

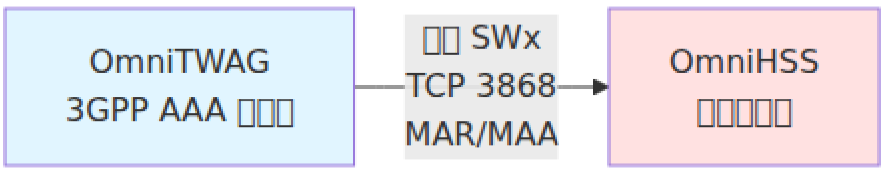
1. 禁用计费

不发送信用控制请求。未执行余额授权。

用例：

- 开放/免费 WiFi 网络
- 实验室/测试环境
- 仅支持离线计费的网络（RADIUS 计费到账单）

流程：



2. 仅授权

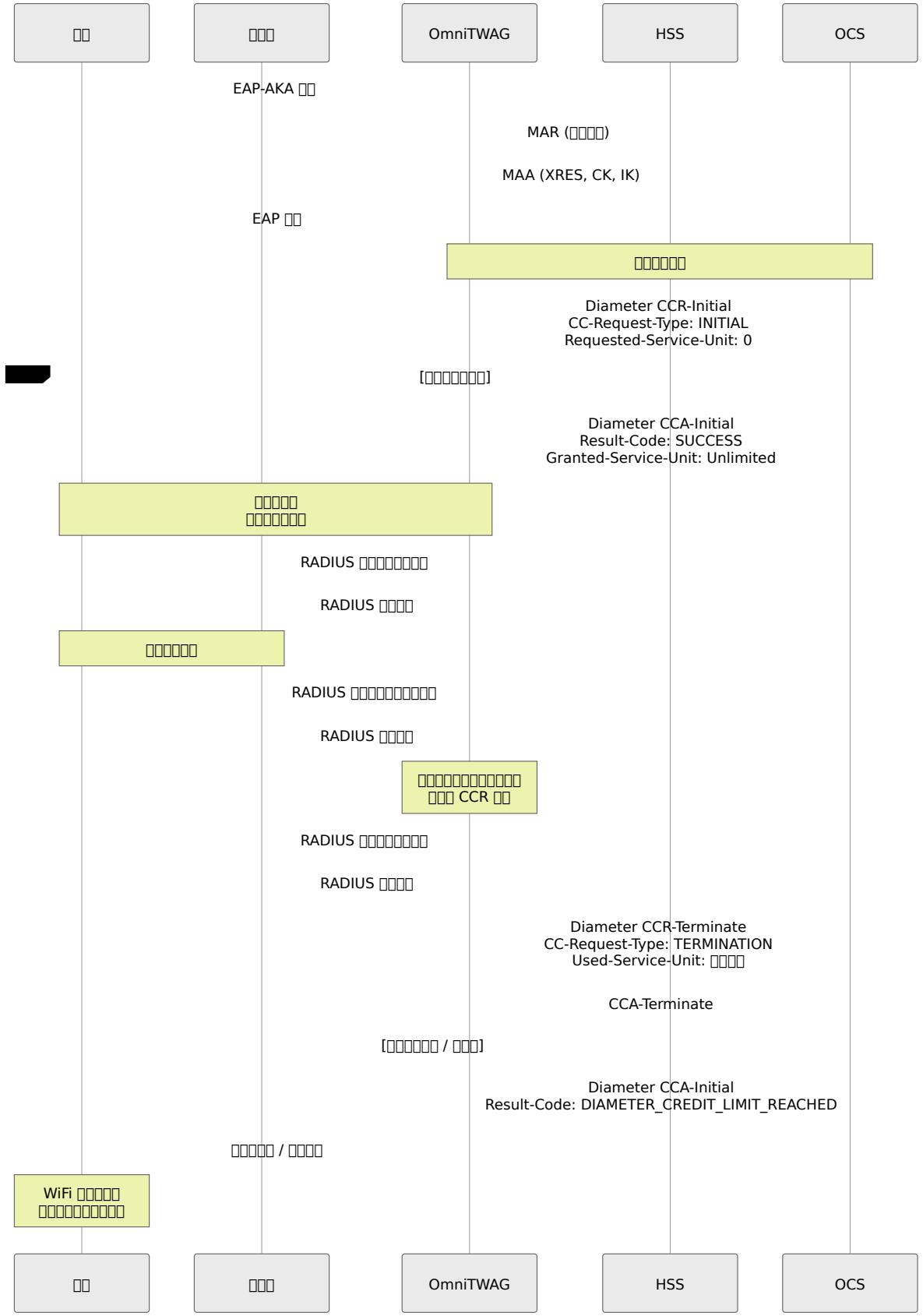
在 WiFi 会话开始时向 OCS 发送 CCR-Initial（信用控制请求），以验证用户是否有余额，但在会话期间不会减少余额。

用例：

- 验证用户是否有活动账户/余额
- 防止暂停账户的 WiFi 访问
- 检查服务资格而不跟踪配额
- 允许 WiFi 作为付费客户的奖励/无限制服务

流程:

配置:



配置:

- 会话开始时（CCR-I）和结束时（CCR-T）查询 OCS
- 会话期间不发送 CCR 更新消息

- □□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□

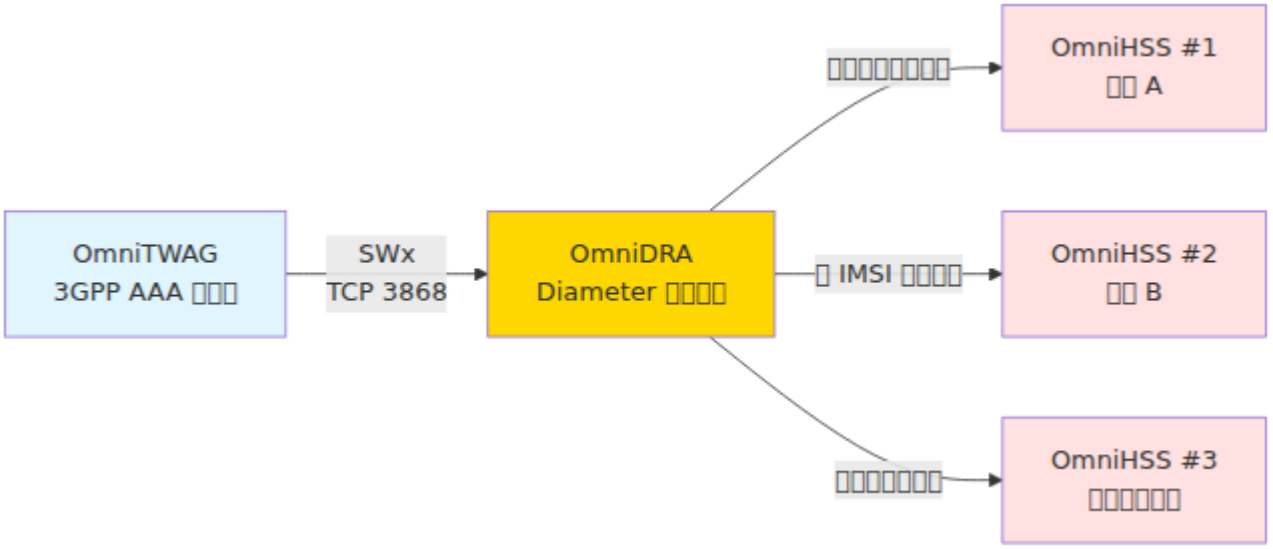
3. □□ Gy □□□□□□□□□□□

□□□□□□ 3GPP □□□□□□□□WiFi □□□□□□□□□□□□□□□ OCS □□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□

- □□□□□□□□
- □□□□□□□ WiFi
- □□□□□□□□□□□□□□□ 10GB □□□
- □□□□□□□□

□□□

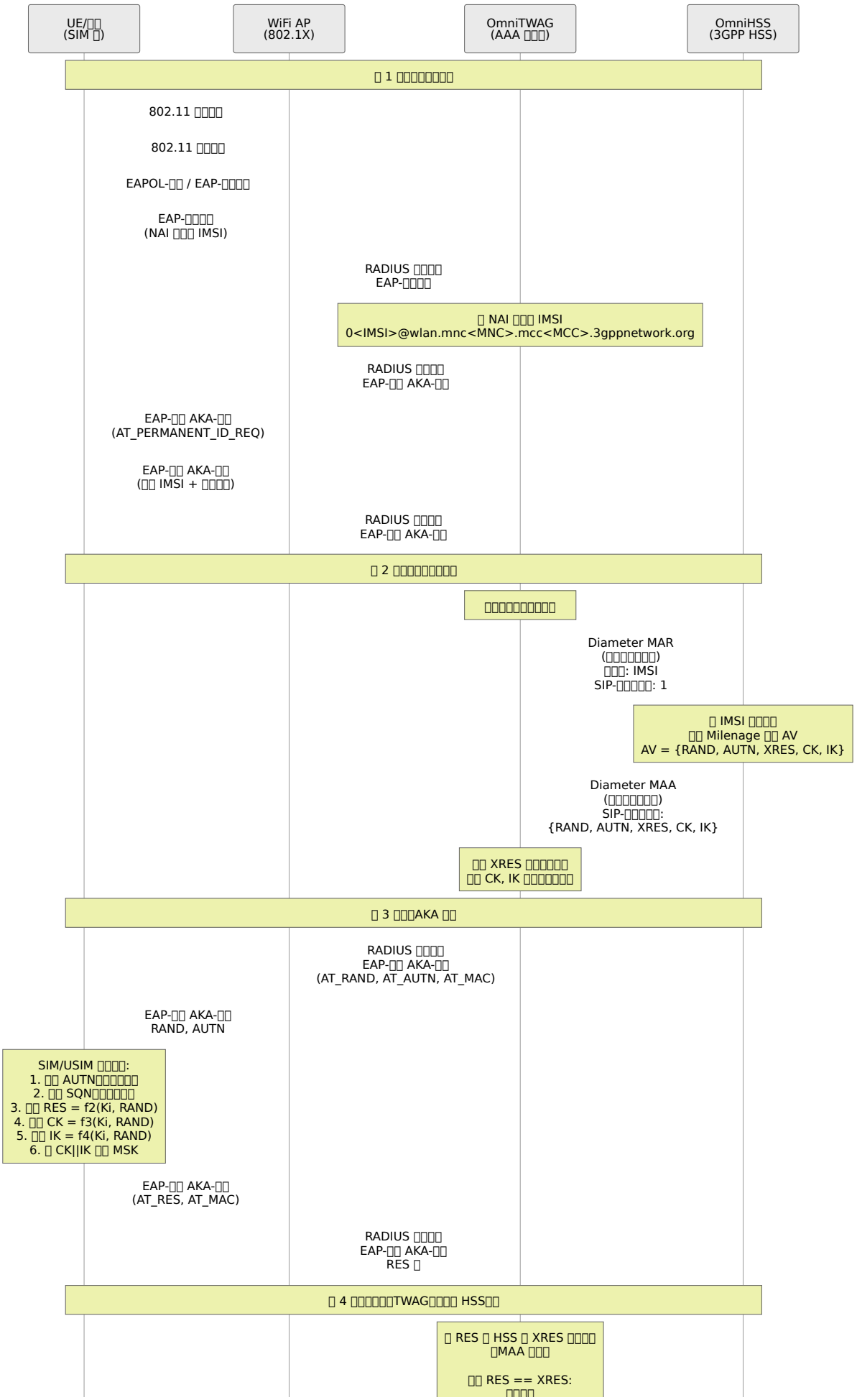


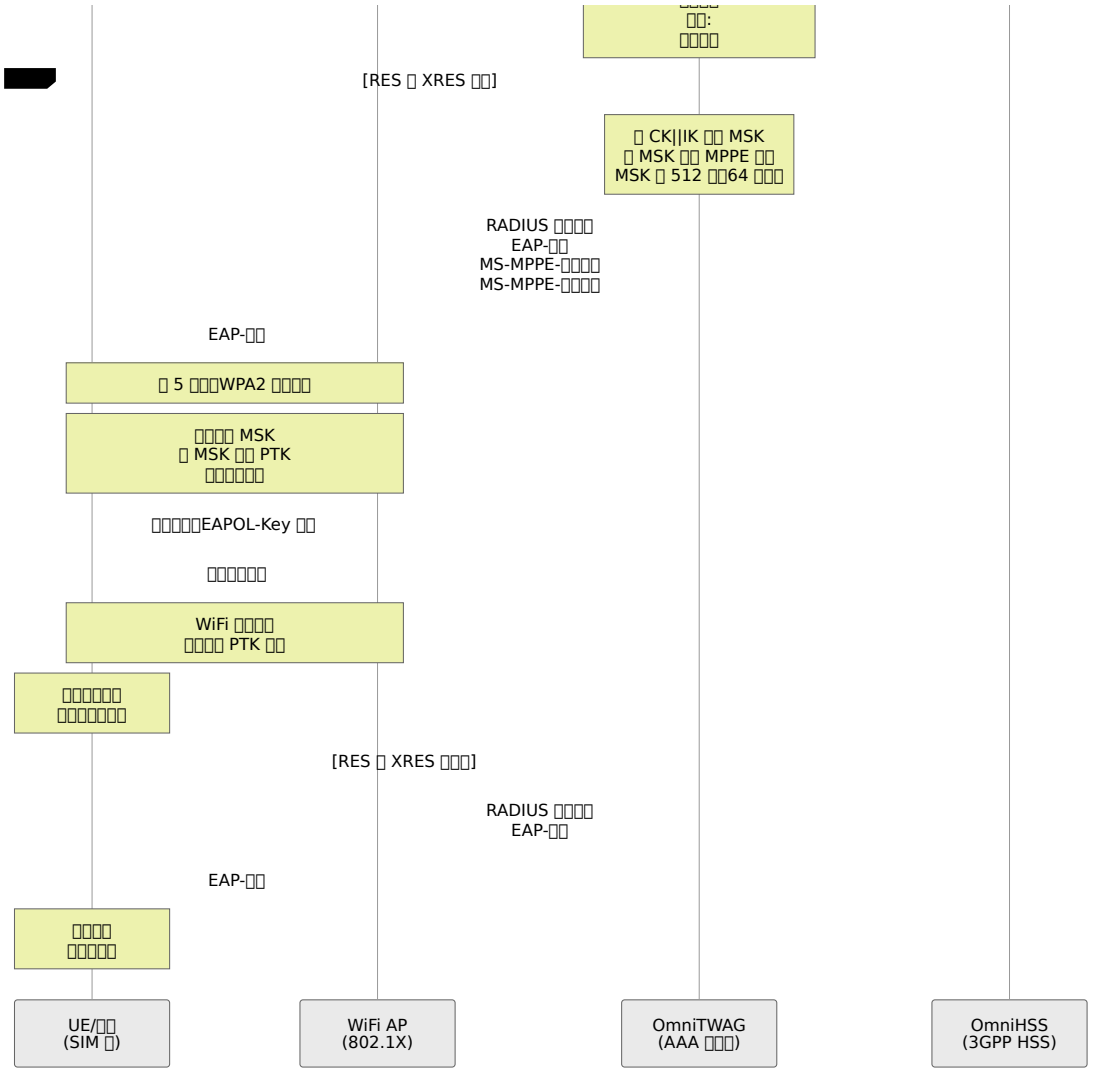
□□□

- □□□□□□□□CCR-I□□□□□□□□CCR-U□□□□□□□CCR-T□□□ OCS
- □□□□□□□□□□□□□□□□10MB□50MB□100MB□
- □□□□□□□□□□□□□□□□ 80%□□□□ CCR □□
- □□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□
- □□□□□□□

认证流程

完整的 **EAP-AKA** 认证序列


UE/设备
(SIM 卡)
WiFi AP
(802.1X)
OmniTWAG
(AAA 服务器)
OmniHSS
(3GPP HSS)
第 1 阶段: 关联和身份
802.11 关联请求
802.11 关联响应
EAPOL-开始 / EAP-请求身份
EAP-响应身份
(NAI 格式的 IMSI)
RADIUS 访问请求
EAP-响应身份
从 NAI 中提取 IMSI
0<IMSI>@wlan.mnc<MNC>.mcc<MCC>.3gppnetwork.org
RADIUS 访问挑战
EAP-请求 AKA-身份
EAP-请求 AKA-身份
(AT_PERMANENT_ID_REQ)
EAP-响应 AKA-身份
(完整 IMSI + 网络信息)
RADIUS 访问请求
EAP-响应 AKA-身份
第 2 阶段: 认证向量检索
需要此用户的认证向量
Diameter MAR
(多媒体认证请求)
用户名: IMSI
SIP-认证数据项: 1
按 IMSI 查找用户
使用 Milenage 生成 AV
AV = {RAND, AUTN, XRES, CK, IK}
Diameter MAA
(多媒体认证响应)
SIP-认证数据项:
{RAND, AUTN, XRES, CK, IK}
存储 XRES 以便后续验证
存储 CK, IK 以进行密钥派生
第 3 阶段: AKA 挑战
RADIUS 访问挑战
EAP-请求 AKA-挑战
(AT_RAND, AT_AUTN, AT_MAC)
EAP-请求 AKA-挑战
RAND, AUTN
SIM/USIM 执行加密:
1. 验证 AUTN (网络认证)
2. 检查 SQN (重放保护)
3. 计算 RES = f2(Ki, RAND)
4. 派生 CK = f3(Ki, RAND)
5. 派生 IK = f4(Ki, RAND)
6. 从 CK||IK 派生 MSK
EAP-响应 AKA-挑战
(AT_RES, AT_MAC)
RADIUS 访问请求
EAP-响应 AKA-挑战
RES 值
第 4 阶段: 验证（TWAG, 而不是 HSS! ）
将 RES 与 HSS 的 XRES 进行比较
(MAA 响应)
如果 RES == XRES:
□□□□



关键步骤说明

1. **MAR/MAA 与 HSS 交互**：向归属网络请求认证向量 XRES 与 MAA，并将认证结果返回给TWAG 以决定是否放行
2. **TWAG 比对 RES**：将HSS 返回的期望值XRES与从 TWAG 转发的 UE 响应的 RES 进行比对，HSS 不参与这一步
3. **密钥下发**：**TWAG**向接入点下发会话 HSS 生成的密钥——这是由 3GPP 定义的AAA 协议将TWAG与接入

身份格式

采用 NAI 格式，由前缀和领域组成，IMSI为

```
505570000000000001@wlan.mnc057.mcc505.3gppnetwork.org
```

□□□`0&lt;IMSI>@wlan.mnc&lt;MNC>.mcc&lt;MCC>.3gppnetwork.org`

□□ - IMSI □□□□□□□□□□□□□□□ 0□□□□□ IMSI SIM □/□□□□□□□□□□□□□□

□□□□□□MSK□

□□□□□□MSK□□□□ EAP-AKA □□□□□ 512 □□64 □□□□□□□□□□□□□□ WiFi □□□□□□□□□

MSK □□□

1. UE □ TWAG □□□□□□□ MSK
2. UE □ SIM □□□ CK/IK □□
3. TWAG □ HSS □□□ CK/IK □□
4. MSK = PRF'(CK || IK, "Full Authentication", IMSI, ...)

MSK □□□

1. **PMK** □□□PMK = MSK □□ 256 □□32 □□□
2. **WPA2** □□□□□UE □ AP □□ PMK □□ PTK
3. □□□□□□□ WiFi □□□□□□□□ PTK □□□□□□TK□□□

□□□ **MSK □□□**

- □□□□□□ MSK□WiFi □□□□□□
- □□□□□□□□ WiFi □
- □□□□□□ EAP □□□ WiFi □□□□
- □□□□□□□□ MSK □□□□□□
- □□□□□□□□□ MSK □□□□□□□□□ MSK

□□□□□

□□□□□□□□□□□□□□□SQN □□□□□□□□□□□□□□□

1. 设备计算 AUTS（认证令牌 - 同步）
2. 发送 EAP-AKA 同步失败，带 AT-AUTS
3. TWAG 将 AUTS 转发给 HSS
4. HSS 重新同步序列号并生成新的向量
5. 使用新向量重试认证

这对最终用户是透明的，无需运营商干预。

配置指南

TWAG 通过 `config/` 目录中的 Elixir 配置文件进行配置。主要运行时配置在 `config/runtime.exs` 中。

对于生产部署，配置集中管理。以下仅为参考，生产节点上更改的任何值将在下次运行自动编排时丢失。

Diameter 配置

位于 `config :diameter_ex`:

```
config :diameter_ex,
  diameter: %{
    # Diameter 栈的服务名称
    service_name: :omnitouch_twag,

    # 绑定 Diameter 服务的本地 IP 地址
    listen_ip: "10.5.198.200",

    # Diameter 连接的本地端口（标准为 3868）
    listen_port: 3868,

    # Diameter Origin-Host
    host: "omnitwag",

    # Diameter Origin-Realm（与您的网络领域匹配）
    realm: "epc.mnc057.mcc505.3gppnetwork.org",

    # Diameter 对等体（HSS、DRA、AAA 服务器）
    peers: [
      %{
        # 对等体 Diameter Origin-Host
        host: "omni-hss01.epc.mnc057.mcc505.3gppnetwork.org",

        # 对等体 Diameter Origin-Realm
        realm: "epc.mnc057.mcc505.3gppnetwork.org",

        # 对等体 IP 地址（可以是 HSS 直接或 DRA）
        ip: "10.179.2.140",

        # 对等体端口（标准为 3868）
        port: 3868,

        # 使用 TLS 进行传输安全
        tls: false,

        # 传输协议（:diameter_tcp 或 :diameter_sctp）
        transport: :diameter_tcp,

        # 向对等体发起连接（true）或等待对等体连接（false）
        initiate_connection: true
      }
```

```
    ]
  }
```

领域格式遵循 3GPP TS 23.003:

```
epc.mnc<MNC>.mcc<MCC>.3gppnetwork.org
```

其中:

- MNC = 移动网络代码（例如，057）
- MCC = 移动国家代码（例如，505 代表澳大利亚）

关于 **DRA** 使用的注意事项：要使用 OmniDRA，将对等体 IP 配置为指向 DRA，而不是直接指向 HSS。然后 DRA 将根据路由规则（目的领域、IMSI 范围等）将消息路由到适当的 HSS。

RADIUS 配置

位于 `config :omnitwag`:

```
config :omnitwag,
  radius_config: %{
    # 允许 RADIUS 客户端的源 IP 子网列表
    # 空列表 = 允许所有（不推荐用于生产）
    allowed_source_subnets: ["10.7.15.0/24", "192.168.1.0/24"],

    # RADIUS 客户端的共享密钥
    # 所有 AP 必须使用此密钥
    secret: "YOUR_STRONG_SECRET_HERE"
  }
```

安全最佳实践：

- 使用强大的RADIUS共享密钥（20个字符以上）
- 配置`allowed_source_subnets`以限制AP访问
- 使用防火墙规则进一步限制对端口1812/1813的访问

示例子网配置：

```
allowed_source_subnets: ["10.7.15.0/24", "192.168.1.0/24"]
```

如果为空，则允许所有源（仅适用于实验室/测试）。

Prometheus 监控配置

位于 `config :omnitwag`:

```
config :omnitwag,
  prometheus: %{
    # Prometheus 指标端口
    port: 9568
  }
```

访问指标: `http://<twag-ip>:9568/metrics`

端口摘要

端口	协议	目的
1812	UDP	RADIUS 认证
1813	UDP	RADIUS 计费
3868	TCP	Diameter (SWx 到 HSS/DRA)
443	TCP	HTTPS Web 仪表板
8444	TCP	HTTPS REST API
9568	TCP	Prometheus 指标

接入点设置

支持的接入点

OmniTWAG 与任何支持以下功能的 WiFi AP 一起工作:

- **WPA2-企业**（802.1X 认证）
- **RADIUS** 客户端功能
- **EAP-AKA** 认证方法

测试平台：Cisco Aironet、Aruba、Ubiquiti UniFi、Ruckus、基于 hostapd 的 AP

一般 AP 配置要求

1. **WPA2-企业**（**802.1X**）安全模式
2. **RADIUS** 服务器指向 TWAG IP 地址
3. **RADIUS** 认证端口：1812
4. **RADIUS** 计费端口：1813（可选但推荐）
5. **RADIUS** 共享密钥：必须与 TWAG 配置匹配
6. **EAP** 方法：EAP-AKA（或“全部”）

Cisco AP 配置示例

CLI 配置：

```
! 配置 RADIUS 服务器
radius-server host 10.5.198.200 auth-port 1812 acct-port 1813 key
YOUR_SHARED_SECRET

! 配置 SSID 使用 802.1X
dot11 ssid OPERATOR-WIFI
   vlan 10
   authentication open eap eap_methods
   authentication network-eap eap_methods
   authentication key-management wpa version 2

! 将 SSID 关联到无线接口
interface Dot11Radio0
   encryption mode ciphers aes-ccm
   ssid OPERATOR-WIFI
```

Web 界面：

1. 导航到 安全性 → **AAA** → **RADIUS** 服务器

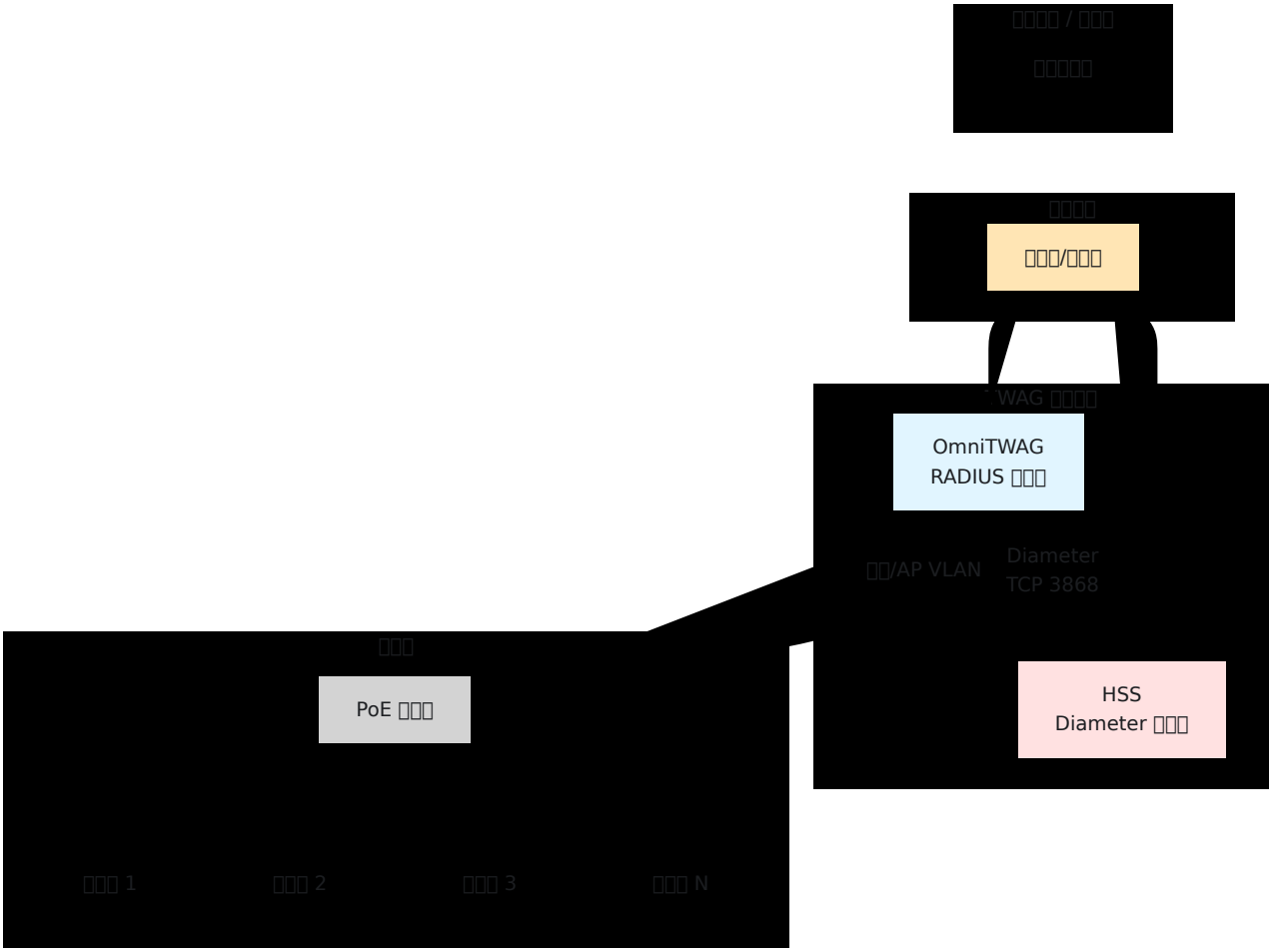
2. 添加 RADIUS 服务器：`10.5.198.200:1812`，共享密钥
3. 导航到 **WLAN** 配置
4. 将安全性设置为 **WPA2-**企业
5. 将 EAP 方法设置为 **EAP-AKA** 或全部
6. 分配 RADIUS 服务器组

hostapd 配置示例

对于基于 Linux 的 AP（OpenWrt、嵌入式系统）：

```
# /etc/hostapd/hostapd.conf

interface=wlan0
driver=nl80211
ssid=OPERATOR-WIFI

# WPA2-企业
wpa=2
wpa_key_mgmt=WPA-EAP
wpa_pairwise=CCMP
ieee8021x=1

# RADIUS 配置
auth_server_addr=10.5.198.200
auth_server_port=1812
auth_server_shared_secret=YOUR_SHARED_SECRET

acct_server_addr=10.5.198.200
acct_server_port=1813
acct_server_shared_secret=YOUR_SHARED_SECRET

# EAP 配置
eap_server=0

# 热点 2.0（可选 - 用于自动卸载）
interworking=1
internet=1
anqp_3gpp_cell_net=505,057
domain_name=wlan.mnc057.mcc505.3gppnetwork.org
nai_realm=0,wlan.mnc057.mcc505.3gppnetwork.org,0,21[2:1][5:7]
roaming_consortium=505057
hs20=1
```

网络架构最佳实践



重要：将 AP 和 TWAG 放置在受信任的网络段上。使用防火墙规则：

- 仅允许 AP 访问 TWAG 的 1812/1813 端口
- 允许 TWAG 访问 HSS 的 3868 端口
- 限制对 TWAG 仪表板（443 端口）的管理访问

热点 2.0 集成

热点 2.0（Passpoint）概述

热点 2.0（也称为 Passpoint 或 802.11u）是 WiFi 联盟标准，允许自动、安全的 WiFi 网络发现和连接，无需用户交互。这是无缝 WiFi 卸载的关键技术。

主要特点：

- □□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□ SIM □□□EAP-AKA□□□□□□□□
- □□□□□□□□OSEN□□ OSU □□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□□□□□□□

□□ 2.0 AP □□

AP □□□□

1. **802.11u** □□□ANQP □□/□□□□
2. **WPA2-**□□□802.1X □□
3. **EAP-AKA** □□□□□□□□ EAP-AKA □□
4. **ANQP** □□□□□□□□□□□□□□□□

□□□□□□□□ **hostapd** □ **AP**□□

```
# 热点 2.0 / Passpoint 配置
interworking=1
internet=1
asra=0
esr=0
uesa=0

# ANQP 配置
anqp_3gpp_cell_net=505,057
domain_name=omnitouchns.com,wlan.mnc057.mcc505.3gppnetwork.org

# NAI 领域配置
nai_realm=0,wlan.mnc057.mcc505.3gppnetwork.org,0,21[2:1][5:7]
# 格式: <编码>,<领域>,<eap-方法>[auth-id:auth-val]
# 21 = EAP-AKA
# 2:1 = 凭据类型: SIM
# 5:7 = 隧道 EAP 方法: 无 (直接 EAP-AKA)

# 漫游联盟
roaming_consortium=505057
# MCC=505 (美国), MNC=057 (运营商特定)

# 场所信息 (可选)
venue_group=1
venue_type=8
venue_name=eng:运营商公共 WiFi

# WPA2-企业配置
wpa=2
wpa_key_mgmt=WPA-EAP
rsn_pairwise=CCMP
ieee8021x=1

# RADIUS 配置 (指向 OmniTWAG)
auth_server_addr=10.5.198.200
auth_server_port=1812
auth_server_shared_secret=YOUR_SHARED_SECRET

acct_server_addr=10.5.198.200
acct_server_port=1813
acct_server_shared_secret=YOUR_SHARED_SECRET

# SSID 配置
```

```
ssid=OperatorWiFi
utf8_ssid=1

# 热点 2.0 指示
hs20=1
hs20_oper_friendly_name=eng:运营商 WiFi 网络
```

自动卸载行为

自动卸载的工作原理：

1. 具有 Passpoint 配置文件的设备定期执行 WiFi 扫描
2. 向检测到的 AP 发送 ANQP 查询
3. 如果 ANQP 响应与配置文件匹配（MCC/MNC、漫游联盟）：
    - 优先级为高（家庭网络）或中（漫游合作伙伴）
4. 如果优先级 ≥ 阈值且信号 > 最小值：
    - 自动 EAP-AKA 认证
5. 如果认证成功且优先级 > 当前连接：
    - 切换到 WiFi，断开蜂窝数据
6. 监控信号质量并保持连接

优先级因素：

1. 家庭与漫游：家庭网络（MCC/MNC 匹配）优先于漫游
2. 信号强度：信号更强者优先
3. 安全性：WPA2-企业优先于开放/WPA2-PSK
4. 策略：运营商可以配置首选网络
5. 用户覆盖：用户可以手动禁用 WiFi 或优先选择蜂窝

监控和管理

Web 仪表板

访问实时监控仪表板：`https://<twag-ip>/`

功能：

- **RADIUS** 客户端视图：活动用户、认证状态、会话详细信息
- 接入点视图：已连接的 AP、客户端数量、SSID 信息
- 客户端使用视图：计费数据、会话时间、数据使用
- **Diameter** 对等体视图：HSS/DRA 连接状态

Prometheus 集成

配置 Prometheus 抓取 TWAG 指标：

```yaml
# prometheus.yml
scrape_configs:
  - job_name: 'omnitwag'
    static_configs:
      - targets: ['10.5.198.200:9568']
    metrics_path: '/metrics'
    scrape_interval: 15s
```

可用指标：

RADIUS 服务器指标：

- `radius_access_request_count` - 接收到的 RADIUS 访问请求包总数
- `radius_access_accept_count` - 发送的访问接受包总数
- `radius_access_reject_count` - 发送的访问拒绝包总数
- `radius_access_challenge_count` - 发送的访问挑战包总数
- `radius_accounting_request_count{status_type}` - 总计费请求包（按状态标记：开始、停止、临时更新、计费开启、计费关闭）
- `radius_active_clients_count` - 当前认证的客户端（每 5 秒轮询一次）
- `radius_access_points_count` - 注册的接入点（每 5 秒轮询一次）

EAP-AKA 认证指标：

- `eap_aka_identity_count` - EAP-AKA 身份交换
- `eap_aka_challenge_count` - EAP-AKA 挑战交换
- `eap_aka_sync_failure_count` - 同步失败（SQN 重新同步事件）

- `eap_aka_auth_success_count` - □□□□
- `eap_aka_auth_reject_count` - □□□□□□

Diameter □□□□□

- `diameter_message_count{application, command, direction}` - □ Diameter □□□□□□□□□□□□□□□□□□

Erlang VM □□□□□

- `vm_memory_total` - □□□□□□□□□□□
- `vm_memory_processes` - Erlang □□□□□□□□□□□
- `vm_memory_processes_used` - Erlang □□□□□□□□□□□□□□□□□□□□□□□□□
- `vm_memory_system` - Erlang □□□□□□□□□□□□□
- `vm_memory_atom` - □□□□□□□□□□□
- `vm_memory_atom_used` - □□□□□□□□□□□□□□□□□□□□□□□□□
- `vm_memory_binary` - □□□□□□□□□□□□
- `vm_memory_code` - □□□□□□□□□□□□□
- `vm_memory_ets` - ETS □□□□□□□□□□

Erlang VM □□□□□

- `vm_system_info_process_count` - □□ Erlang □□□□
- `vm_system_info_port_count` - □□□□□□
- `vm_system_info_atom_count` - □□□□□□
- `vm_system_info_schedulers` - □□□□□□
- `vm_system_info_schedulers_online` - □□□□□□□□□□

Erlang VM □□□□□□

- `vm_statistics_run_queue` - □□□□□□□□□□
- `vm_total_run_queue_lengths_total` - □□□□□□□□□□□□□□□□
- `vm_total_run_queue_lengths_cpu` - CPU □□□□□□□□□□□
- `vm_total_run_queue_lengths_io` - IO □□□□□□□□□□□□

□□□□□

- RADIUS 和 EAP-AKA 指标在事件发生时实时发出
- 活动客户端和接入点数量每 5 秒轮询一次
- VM 指标每 5 秒从 Erlang 运行时轮询一次
- 所有指标以 Prometheus 格式在 `http://<twag-ip>:9568/metrics` 处公开

日志记录

TWAG 使用 Elixir 的 Logger 进行结构化日志记录。

查看日志（systemd）：

```
# 实时日志尾部
journalctl -u twag -f

# 最近 100 行
journalctl -u twag -n 100

# 自上次启动以来的日志
journalctl -u twag -b

# 特定时间范围的日志
journalctl -u twag --since "2025-10-12 10:00:00" --until "2025-10-12 11:00:00"
```

关键日志消息：

- `RADIUS 服务器监听端口 1812` - 服务器已启动
- `来自 {IP}: 访问请求已接收` - 来自 AP 的 RADIUS 请求
- `第 1 阶段: 身份响应` - 初始 EAP 身份
- `第 2 阶段: AKA 挑战` - 发送给设备的挑战
- `认证接受` - 成功认证
- `认证拒绝` - 认证失败
- `注册的 AP: {IP}` - 检测到新 AP

故障排除

认证失败

症状：客户端无法连接到 WiFi

诊断步骤：

1. 检查 TWAG 日志：`journalctl -u twag -f`
2. 验证 AP 和 TWAG 之间的 RADIUS 共享密钥是否匹配
3. 确认 RADIUS 数据包是否到达 TWAG：`tcpdump -i eth0 port 1812`
4. 检查 HSS/配置中的用户预配置

常见原因：

- RADIUS 共享密钥不正确
- 防火墙阻止 UDP 1812/1813
- RES/XRES 不匹配（错误的 SIM Ki 或 HSS 配置）
- 序列号（SQN）不同步（应通过重新同步自动恢复）
- AP 和 TWAG 之间的网络连接问题

Diameter 连接问题

症状：Diameter 对等体无法连接到 HSS/DRA

诊断步骤：

1. 验证网络连接：`telnet <hss-ip> 3868`
2. 检查 Diameter 配置（Origin-Host、Origin-Realm、对等体 IP）
3. 查看 HSS/DRA 日志以获取连接尝试
4. 验证防火墙允许 TCP 3868

常见原因：

- 配置中的对等体 IP/端口不正确
- 防火墙阻止 TCP 3868
- Origin-Host/Realm 不匹配

- HSS/DRA 不接受来自 TWAG 的连接

性能问题

症状：认证缓慢（>5 秒）

诊断步骤：

1. 检查 HSS 响应时间
2. 测量网络延迟：`ping <hss-ip>`，`mtr <hss-ip>`
3. 监控 TWAG 资源使用情况：`top`，`htop`
4. 查看 Diameter 请求超时设置

常见原因：

- HSS 查询超时或响应缓慢
- 高网络延迟
- TWAG 资源耗尽（CPU/内存）
- 同时认证的用户过多

调试工具

数据包捕获

```
# 捕获 RADIUS 流量
tcpdump -i eth0 -n port 1812 or port 1813 -w radius.pcap

# 捕获 Diameter 流量
tcpdump -i eth0 -n port 3868 -w diameter.pcap

# 从特定 AP 捕获
tcpdump -i eth0 -n host 10.7.15.72 and port 1812 -w radius-ap1.pcap
```

使用 Wireshark 分析（支持 RADIUS 和 Diameter 解码器）。

交互式控制台

附加到正在运行的 TWAG 进行实时调试:

```
# 远程 shell 到正在运行的 TWAG
iex --sname debug --remsh twag@hostname --cookie <cookie>
```

从 IEx 控制台:

```
# 列出所有认证的客户端
CryptoState.keys()

# 获取特定客户端状态
CryptoState.get("0505338057900001867@wlan.mnc057.mcc505.3gppnetwork.c

# 列出所有 AP
APState.list()

# 列出计费会话
ClientUsage.list()
```

□□□□□□

□□□□	□□	□□□□
`□□□□□□□□□□`	□□□□□□□	□□ AP □ TWAG □□ RADIUS □□□□□□
`RES □□□□□□□ <XRES>□□□ <RES>`	□□□□□□□	□□ SIM Ki□□□ HSS □□□
`Diameter □□□□□□□`	□□□□ HSS	□□□□□□□□□HSS □□
`□□□□ EAP □□`	□□□□□ EAP □□□	□□ AP □□□□□□□□□ AP
`□□ EAP-AKA □□□`	□□□□ EAP-AKA □□	□□□□□□□ EAP-AKA □□
`□□□□□□□`	SQN □□□	□□□□□□□□□□□□□

□□□□□

OmniTWAG □□□□ 3GPP □ IETF □□□

- **3GPP TS 23.402**□□ 3GPP □□□□□□□
- **3GPP TS 24.302**□□□□ 3GPP □□□□□□ EPC
- **3GPP TS 29.273**□□□ Diameter □ SWx/SWm □□
- **3GPP TS 33.402**□□ 3GPP □□□□□□□
- **3GPP TS 35.206**□Milenage □□□□
- **RFC 2865**□RADIUS □□
- **RFC 2866**□RADIUS □□
- **RFC 3579**□RADIUS □ EAP □□□
- **RFC 4187**□EAP-AKA □□□□
- **RFC 5448**□EAP-AKA'□□□□□□

□□

OmniTWAG□□ Omnitouch □□□□□□□□□□□□□□□ 3GPP WiFi □□□□□□□□

1. □□□□□□□□□□□□□□□□□□□□
2. □□□□□□□□ 3GPP SWx□EAP-AKA□RADIUS □□
3. □□□□□□□□ SIM □□□□□□□□□□□□□
4. □□□□MSK □□□□□□□□ WPA2 □□
5. □□□□ **2.0**□□□□□□□□□□□□□□
6. □□□□□□□□□□□□□□□□□□□□□
7. □□□□□□□□ HSS □□□□□ OmniDRA □□□□/□□□□
