

OmniPGW

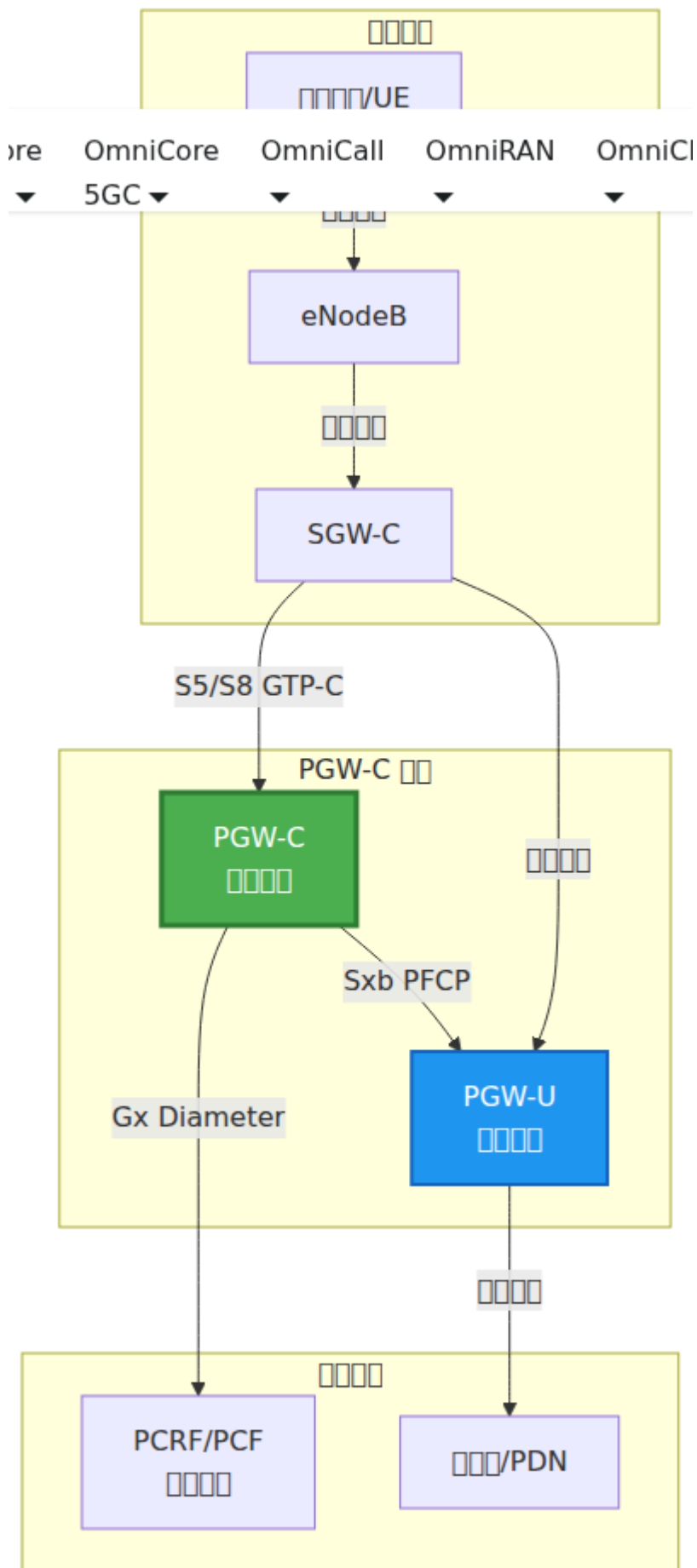
OmniPGW - (PGW-C)

Omnitouch

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.
7. (OAM API)
- 8.
- 9.
- 10.

OmniPGW (PGW-C) 3GPP LTE (EPC) Omnitouch

- - UE
- IP - IP
- - PCRF
- - PGW-U



PGW-C

- **SGW-C** S5/S8 (GTP-C)
 - **UE IP**
 - **Gx (Diameter) PCRF**
 - **Sxb (PFCP) PGW-U**
 - **QoS QoS**
 -
-

- [illegible]

PGW-C 3GPP

S5/S8 □□ (GTP-C v2)

□□□ SGW-C □ PGW-C □□□□□□□□

UDP → GTP-C → 2

□ □ □ □ □

- 000000/00
- 000000/00
- 000000/00
- 000000/00

□□□ □□□ S5/S8 □□

Sxb (PFCP)

PGW-C PGW-U

UDP PFCP

□ □ □ □ □

- □□□□□□/□□
- □□□□□□/□□

- 3GPP TS 29.261
- 3GPP TS 29.262
- 3GPP TS 29.263

3GPP TS 29.261 PFCP/Sxb

Gx (Diameter)

3GPP TS 29.262 (PCRF)

3GPP TS 29.263 Diameter (IETF RFC 6733)

3GPP TS 29.264

- 3GPP TS 29.264 (CCR-I/CCA-I)
- 3GPP TS 29.264 (CCR-T/CCA-T)

3GPP TS 29.265 Diameter Gx

- **APN** - 網路位址
- - QoS
- **ID** -
- **TEID** ID - S5/S8
- **SEID** ID - Sxb

網路位址

網路位址 QoS

- - PDN
- - QoS
- **EBI** EPS ID -
- **QoS** - QCI ARP MBR GBR

PFCP

PGW-C PGW-U

- **PDR** -
- **FAR** -
- **QER** QoS -
- **BAR** -

網路位址 PFCP

IP

UE IP

- **APN** - APN
- - IP
- - UE IP
- - IP

網路位址 UE IP IPv6 / IPv4 IPv6 IPv4v6 PDN

□□

□□□□

- Elixir ~1.16
- Erlang/OTP 26+
- □ SGW-C□PGW-U □ PCRF □□□□
- □□ LTE EPC □□

□□ OmniPGW

1. □ `config/runtime.exs` □□□□□□□□
2. □□□□□□□□

```
mix deps.get
mix compile
```

3. □□□□□□□□

```
mix run --no-halt
```

□□□□

□□□□□□□□□□□□

```
[info] □□ OmniPGW...
[info] □ 127.0.0.42:42069 □□□□□□□□
[info] □ 127.0.0.10 □□ S5/S8 □□
[info] □ 127.0.0.20 □□ Sxb □□
[info] □□ Gx □□
[info] □□ PFCP □□□□□□
[info] OmniPGW □□□□
```

□□□□□ □ `http://127.0.0.42:42069/metrics` □□□□□□□□



config/runtime.exs

??

```

# config/runtime.exs
import Config

# 配置
config :logger, level: :info

config :pgw_c,
  # 配置 (Prometheus)
  metrics: %{
    enabled: true,
    ip_address: "10.0.0.20", # 配置
    port: 9090,
    registry_poll_period_ms: 5_000
  },

  # Diameter/Gx (PCRF 配置)
  diameter: %{
    listen_ip: "0.0.0.0",
    host: "omnipgw.epc.mnc001.mcc001.3gppnetwork.org",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    peer_list: [
      %{
        host: "pcrf-primary.epc.mnc001.mcc001.3gppnetwork.org",
        realm: "epc.mnc001.mcc001.3gppnetwork.org",
        ip: "10.0.1.30",
        initiate_connection: true
      },
      %{
        host: "pcrf-backup.epc.mnc001.mcc001.3gppnetwork.org",
        realm: "epc.mnc001.mcc001.3gppnetwork.org",
        ip: "10.0.2.30",
        initiate_connection: true
      }
    ]
  },

  # S5/S8 (SGW-C 配置)
  s5s8: %{
    local_ipv4_address: "10.0.0.20"
  },

  # Sxb/PFCP (PGW-U 配置)
  sxb: %{

```

```

    local_ip_address: "10.0.0.20"
  },

  # UPF
  upf_selection: %{
    rules: [
      %{
        name: "internet",
        priority: 1,
        match_field: :apn,
        match_regex: ~r/^internet/,
        upf_pool: [
          %{remote_ip_address: "10.0.0.21", remote_port: 8805,
weight: 100},
          %{remote_ip_address: "10.0.0.22", remote_port: 8805,
weight: 0} #
        ]
      }
    ],
    fallback_pool: [
      %{remote_ip_address: "10.0.0.21", remote_port: 8805, weight:
100}
    ]
  },

  # UE IP
  ue: %{
    subnet_map: %{
      "internet" => [
        "100.64.0.0/18" # 16,382 IP
      ],
      "ims" => [
        "100.64.64.0/22" # 1,022 IP
      ],
      "enterprise.corp" => [
        "10.100.0.0/16" # 65,534 IP
      ],
      default: [
        "100.64.127.0/24" # 254 IP
      ]
    }
  },

  #

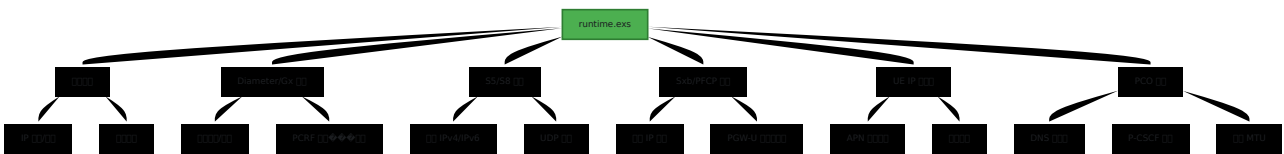
```



```
pco: %{\n  primary_dns_server_address: "8.8.8.8",\n  secondary_dns_server_address: "8.8.4.4",\n  p_cscf_ipv4_address_list: ["10.0.0.50", "10.0.0.51"],\n  ipv4_link_mtu_size: 1400\n}
```

网络配置

网络



网络

网络	网络	网络
metrics	Prometheus 网络	网络
diameter	Gx 网络 PCRF	Diameter Gx 网络
s5s8	GTP-C 网络 SGW-C	S5/S8 网络
sxb	PFCEP 网络 PGW-U	PFCEP 网络
ue	UE IP 网络	IP 网络
pco	网络	PCO 网络
CDR	网络	CDR 网络

网络 网络

OmniPGW (OAM API)

OmniPGW 提供 **OAM REST API** 用于管理 OmniPGW 配置。❗❗
API 文档/接口 - 管理 OmniPGW 配置

访问 OAM API

API 通过 HTTPS/TLS 提供，地址为 `/api`。

```
https://<omnipgw-ip>:8443/
```

API 文档 Swagger UI 地址

```
https://<omnipgw-ip>:8443/api/docs
```

验证证书 -k 选项

```
curl -k https://localhost:8443/api/status  
{ "result": "ok" }
```

返回

項目	API	説明
ユーザ検索 (IMSI/MSISDN/IP)	POST /api/ue_search	UE 検索
セッション PDN 検索	GET /api/sessions[?search=<imsi msisdn ip>]	PGW 検索
IMSI によるセッション検索	GET /api/sessions/<imsi>	PGW 検索 セッション
セッション履歴	GET /api/session_history[?type=&search=&page=&page_size=]	セッション
ネットワークトポロジ	GET /api/topology	セッション
UE IP 検索	GET /api/ip_pools, GET /api/ip_pools/<name>	IP 検索
UPF / PCF 検索	GET /api/upf, GET /api/upf/<ip>	PCF 検索 / UPF 検索
UPF 選択 PCO 検索	GET /api/upf_selection	UPF 検索
Diameter (Gx/Gy) 検索	GET /api/diameter[?status=connected disconnected], GET /api/diameter/<origin-host>	Diameter 検索
チャージング (Gy) 検索	GET /api/charging[?search=<imsi>], GET /api/charging/<imsi>	Gy 検索
P-CSCF DNS 検索	GET /api/pcscf_monitor	P-CSCF 検索
システムステータス	GET /api/status	-

API の詳細は、API 仕様書をご覧ください。

```
watch -n 2 'curl -sk https://localhost:8443/api/sessions'
watch -n 1 'curl -sk https://localhost:8443/api/diameter?
status=disconnected'
```

GET /api/charging 取得 Gy/充電電圧

□ □ □ □

□ □ □ □ □ □ □ □

- `omniPGW` `omniPGW`
- `watch/cron/omniPGW`
- `connected/associated` `JSON`

□ □ □ □ □ □

- GET /api/sessions?search=... IMSI IP MSISDN APN
- ?status=connected|disconnected Diameter
- ?search=<imsi>

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

- `POST (/api/sessions/<imsi> /api/upf/<ip> /api/diameter/<origin-host> /api/charging/<imsi>) 返回JSON`
- `POST /api/charging/<imsi>`

□□□□□

- 使用 HTTPS/TLS 8443 端口
- 使用固定 IP 地址
- 使用 NOC/网络管理系统

□□□□□

□ □ □ □ □ □ □ □ □ □ □ □

1. `POST /api/ue_search` JSON `IMSI` `MSISDN` `IP`
3. `POST /api/ue_search`
 - a) `QCI` - `QCI` `MBR/GBR`
 - b) `APN` - `APN` `TAC` `ID`
 - c) `QCI` - `QCI` `MBR/GBR`
 - `QCI` `MBR/GBR`
 - `APN` `AMBR`
 - d) `QCI` - `Gy` `ID` `GET /api/charging/<imsi>`
 - e) `QCI` - `Gx` `PCC`
 - f) `QCI` - `GET /api/session_history?search=<imsi>`
4. `POST /api/ue_search` → `GET /api/diameter?status=disconnected` `PCRF`

PCRF

1. `POST /api/ue_search`
2. `GET /api/sessions?search=<imsi|msisdn>` `GET /api/sessions/<imsi>`
3. `POST /api/ue_search`
 - `UE IP`
 - `QoS`
 - `QCI`
4. `POST /api/ue_search` → `POST /api/ue_search`

PCRF

1. `GET /api/upf` → `PGW-U` `"OK"`
2. `GET /api/diameter?status=connected` → `PCRF` `"OK"`
3. `GET /api/sessions` → `QCI`
4. `GET /api/status` → `result` `ok`

PCRF

- `GET /api/sessions` `QCI`
- `QCI` `QCI`
- `QCI`
- `APN` `GET /api/ip_pools` `QCI`

API 목록

OAM API 목록

- 사용자 검색 (POST /api/ue_search)
- 세션 검색 (GET /api/sessions/<imsi>)
- 사용자 정보 (GET /api/upf GET /api/diameter)
- 시스템 상태 (GET /api/status)
- IMSI/MSISDN/IP 정보 조회
- QoS 정보 MBR GBR QCI
- 충전 정보 (GET /api/charging)
- 세션 기록 (GET /api/session_history)
- IP 풀 정보 (GET /api/ip_pools)
- UPF 선택 (GET /api/upf_selection)

Prometheus 목록

- 사용자
- 세션
- 사용자 정보
- 충전
- 세션 기록

목록 - OAM API 목록 Prometheus 목록

목록

OAM API OmniPGW Prometheus 목록

목록

- 사용자
 - teid_registry_count - S5/S8

- `seid_registry_count` - PFCP 計数
- `session_id_registry_count` - Gx 計数
- `address_registry_count` - UE IP 計数
- `charging_id_registry_count` - Charging ID 計数

- 入信メッセージ

- `s5s8_inbound_messages_total` - S5/S8 GTP-C 入信メッセージ
- `sxb_inbound_messages_total` - S5/S8 PFCP 入信メッセージ
- `gx_inbound_messages_total` - Gx Diameter 入信メッセージ
- 入信メッセージの合計

- 入信エラー

- `s5s8_inbound_errors_total` - S5/S8 入信エラー
- `sxb_inbound_errors_total` - S5/S8 PFCP 入信エラー
- `gx_inbound_errors_total` - Gx Diameter 入信エラー

- PDN セッション

- `pgw_session_create_total{result,cause,pgn_type,pgn_type}` - PDN セッション作成の総数
 {result,cause} は success/failure のいずれか、pgn_type は GTP による PDN のタイプ
 {ipv4,ipv6,ipv4v6} のいずれか、pgn_type は PDN のタイプ
- `pgw_session_delete_total{pgn_type}` - PDN セッションの削除の総数
- `pgw_session_modify_total` - PDN セッションの修正の総数

- UE IP 管理

- `ue_pool_addresses_allocated{pool,ip_version}` - APN ごとに IP アドレスが割り当てられた UE の数
- `ue_pool_addresses_total{pool,ip_version}` - APN ごとに IP アドレスの総数
- `ue_pool_addresses_utilization_ratio{pool,ip_version}` - APN ごとに IP アドレスの利用率
 (0.0-1.0) の範囲で、小数点第 2 位まで表示される
- `ue_ip_allocation_failures_total{ip_version,reason}` - IP アドレスの割り当て失敗の総数
 {reason} は pool_exhausted, already_registered のいずれか

- 入信 UPF / PFCP

- `upf_peer_associated{peer_ip}` - 000000 PFCP 0000 1/0
- `upf_peer_healthy{peer_ip}` - 0000000000 1/0
- `upf_peer_missed_heartbeats{peer_ip}` - 000000000000
- `upf_heartbeat_rtt` - 0 `peer_ip` 0 PFCP 00000000

- **QoS** 00

- `pgw_bearer_create_total{type,qci}` - 0000 `default/dedicated` 00 QCI 000000
- `pgw_bearer_delete_total{type}` - 0000000000

- **Diameter** 0000

- `diameter_peers_connected{application}` - 000000 `gx/gy/all` 0000 Diameter 00

00000

0000 HTTP 000000000000

```
curl http://127.0.0.42:42069/metrics
```

000 00000000 0000000000000000

00000

00000000 OmniPGW 0000000000000000000000

0000

OmniPGW 00

└─ OPERATIONS.md (000)

└─ docs/

└─ 00000

└─ configuration.md

runtime.exs 00

└─ ue-ip-allocation.md

IP 000

└─ pco-configuration.md

DNSP-CSCFP-MTU 00

└─ 0000

└─ pfcf-interface.md

Sxb/PFCF (PGW-U 00)

└─ diameter-gx.md

Gx (PCRF 00)

└─ diameter-gy.md

Gy/Ro (OCS 00)

└─ s5s8-interface.md

S5/S8 (SGW-C 00)

└─ 00

└─ session-management.md

PDN 000000

└─ monitoring.md

Prometheus 00000

000000

0 0000

00	00	00
OPERATIONS.md	00000000000000	00000000

⚙ 00

파일	내용	줄 수
configuration.md	runtime.exs 관련	1,600+
ue-ip-allocation.md	UE IP 할당	943
ipv6-dual-stack.md	PGW-C 및 UPF 관련 IPv6 / IPv4v6 PDN	-
pco-configuration.md	DNS, P-CSCF, MTU	344

네트워크

파일	내용	줄 수
pfcp-interface.md	PFCP/Sxb 관련 PGW-U	1,355
diameter-gx.md	Diameter Gx 관련 PCRF	941
diameter-gy.md	Diameter Gy/Ro 관련 OCS	1,100+
s5s8-interface.md	GTP-C S5/S8 관련 SGW-C	456

기타

파일	내용	줄 수
session-management.md	PDN 관련	435
monitoring.md	Prometheus 및 Grafana	807
data-cdr-format.md	CDR 관련 URR	847
qos-bearers.md	QoS 관련	448
troubleshooting.md	문제 해결	687

目 次

目次	目次	目次
pcscf-monitoring.md	P-CSCF 監視	894

目 次

目 次 Mermaid

Mermaid 目 次

- 目 次
- 目 次
- 目 次
- 目 次

目 次

目 次

- 目 次
- 目 次
- 目 次

目 次

目 次

- 目 次
- 目 次
- 目 次

目 次

目 次

目次

概要

1. [OPERATIONS.md](#) - 運用
2. [configuration.md](#) - 設定
3. [monitoring.md](#) - 監視
4. [session-management.md](#) - セッション管理

ネットワーク

1. [OPERATIONS.md](#) - ネットワーク運用
2. [pfcg-interface.md](#) - PFCGインターフェース
3. [diameter-gx.md](#) - Diameter-GX
4. [diameter-gy.md](#) - Diameter-GY
5. [s5s8-interface.md](#) - S5/S8インターフェース
6. [ue-ip-allocation.md](#) - UE IP 割り当て

セキュリティ

1. [configuration.md](#) - セキュリティ設定
2. [ue-ip-allocation.md](#) - UE IP 割り当て
3. [pco-configuration.md](#) - PCO設定
4. [monitoring.md](#) - セキュリティ監視

仕様

- 仕様書 14
- 仕様数 ~10,900+
- 仕様数 ~265 KB
- **Mermaid** 仕様 75+
- 仕様数 150+

開発環境

OS

- 消息/消息头
- 消息体/消息内容
- 消息长度
- 消息类型 GenServer 消息

消息

- 消息头/消息体
- 消息体/消息内容
- 消息长度
- 消息类型 Diameter RFC 6733

3GPP 消息

- Sx/PGW-C ↔ PGW-U
- Gx/PGW-C ↔ PCRF
- Gy/Ro/PGW-C ↔ OCS
- S5/S8/SGW-C ↔ PGW-C

消息

- 消息头
 - 消息体
 - 消息长度
 - 消息类型
 - 消息内容
-

□□□□

3GPP □□

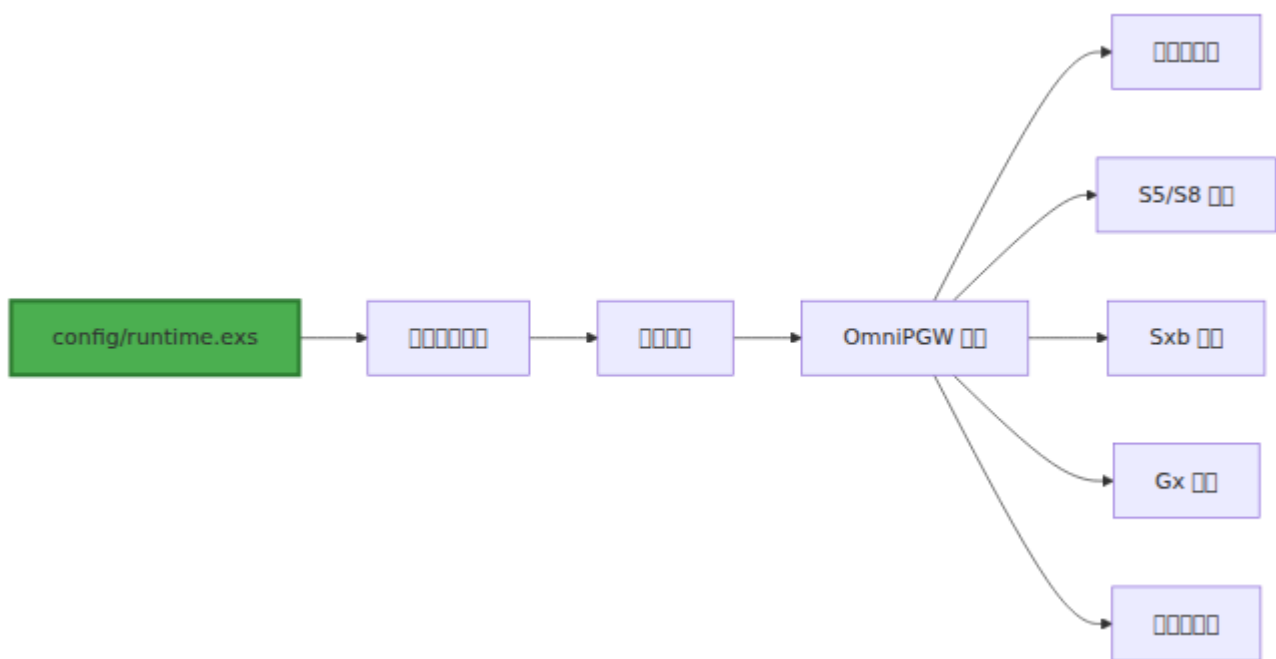
□□	□□
TS 29.274	GTP-C v2□S5/S8 □□□
TS 29.244	PFCP□Sxb □□□
TS 29.212	Diameter Gx □□□□□□□□
TS 32.299	Diameter □□□□□Gy/Ro□
TS 32.251	□□□□□□□□
TS 23.401	EPC □□

□□□□

- `configuration.md` - □□□ `config/runtime.exs` □□
-

OmniPGW 的 配置文件 是 `config/runtime.exs` 文件。该文件定义了 OmniPGW 的 配置。该文件位于 `lib/omni_pgw/config` 目录下。

□□□□



□□□□□

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

□□□□□□□□

□□□□□

```
pgw_c/
├─ config/
│   ├── config.exs      # □□□□□□□□ runtime.exs□
│   ├── dev.exs         # □□□□□□□□
│   ├── prod.exs        # □□□□□□□□
│   └─ runtime.exs      # ← □□□□□□□□
```

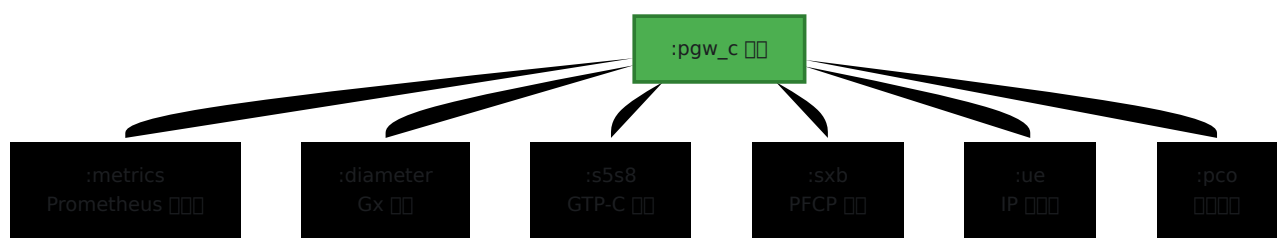

配置

```
# config/runtime.exs
import Config

config :logger, level: :info

config :pgw_c,
  metrics: %{...},
  diameter: %{...},
  s5s8: %{...},
  sxb: %{...},
  ue: %{...},
  pco: %{...}
```

架构图



部署

部署

部署 Prometheus 和 OmniPGW

配置

```
config :pgw_c,
  metrics: %{
    # 是否开启
    enabled: true,

    # 监听 HTTP 请求 IP 地址
    ip_address: "0.0.0.0",

    # 监听端口
    port: 9090,

    # 注册中心轮询周期
    registry_poll_period_ms: 10_000
  }
```

参数

参数	类型	默认值	说明
<code>enabled</code>	布尔	<code>true</code>	是否开启
<code>ip_address</code>	字符串 IP	<code>"0.0.0.0"</code>	监听地址 0.0.0.0 = 所有地址
<code>port</code>	整数	<code>9090</code>	<code>/metrics</code> 监听 HTTP 端口
<code>registry_poll_period_ms</code>	整数	<code>10_000</code>	注册中心轮询周期

部署

部署 - 指定 IP

```
metrics: %{
  enabled: true,
  ip_address: "10.0.0.20", # 10.0.0.20
  port: 9090,
  registry_poll_period_ms: 5_000 # 5 000ms
}
```

□□ - □□□□□□◆◆

```
metrics: %{
  enabled: true,
  ip_address: "127.0.0.1",
  port: 42069, # 0000
  registry_poll_period_ms: 10_000
}
```

□□□□□

```
metrics: %{
  enabled: false
}
```

□□□□

```
# curl http://<ip_address>:<port>/metrics
```

```
# curl http://10.0.0.20:9090/metrics
```

Diameter/Gx 配置

配置

Gx 配置 Diameter 配置 PCRF 配置

配置

```
config :pgw_c,  
  diameter: %{  
    # Diameter 配置 IP 配置  
    listen_ip: "0.0.0.0",  
  
    # OmniPGW 配置 Diameter 配置 Origin-Host  
    host: "omnipgw.epc.mnc001.mcc001.3gppnetwork.org",  
  
    # OmniPGW 配置 Diameter 配置 Origin-Realm  
    realm: "epc.mnc001.mcc001.3gppnetwork.org",  
  
    # PCRF 配置  
    peer_list: [  
      %{  
        # PCRF Diameter 配置  
        host: "pcrf.epc.mnc001.mcc001.3gppnetwork.org",  
  
        # PCRF 配置  
        realm: "epc.mnc001.mcc001.3gppnetwork.org",  
  
        # PCRF IP 配置  
        ip: "10.0.0.30",  
  
        # 配置 PCRF 配置  
        initiate_connection: true  
      }  
    ]  
  }  
}
```

PCRF

PCRF	PCRF	PCRF	PCRF
listen_ip	PCRF IP	PCRF	Diameter PCRF
host	PCRF FQDN	PCRF	OmniPGW PCRF Origin-Host FQDN
realm	PCRF Realm	PCRF	OmniPGW PCRF Origin-Realm
peer_list	PCRF	PCRF	PCRF PCRF

PCRF

PCRF	PCRF	PCRF	PCRF
host	PCRF FQDN	PCRF	PCRF Diameter PCRF
realm	PCRF Realm	PCRF	PCRF PCRF
ip	PCRF IP	PCRF	PCRF IP PCRF
initiate_connection	PCRF	PCRF	OmniPGW PCRF PCRF

FQDN PCRF

Diameter PCRF FQDN

```
# PCRF
host: "omnipgw.epc.mnc001.mcc001.3gppnetwork.org"

# PCRF
host: "omnipgw" # PCRF FQDN
host: "10.0.0.20" # PCRF IP
```

3GPP PCRF

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

□□□

- omnipgw.epc.mnc001.mcc001.3gppnetwork.org (MCC=001, MNC=001)
- pgw-c.epc.mnc260.mcc310.3gppnetwork.org (MCC=310, MNC=260 - □□ T-Mobile)

□□

□□ **PCRF**□

```
diameter: %{  
  listen_ip: "0.0.0.0",  
  host: "omnipgw.epc.mnc001.mcc001.3gppnetwork.org",  
  realm: "epc.mnc001.mcc001.3gppnetwork.org",  
  peer_list: [  
    %{  
      host: "pcrf.epc.mnc001.mcc001.3gppnetwork.org",  
      realm: "epc.mnc001.mcc001.3gppnetwork.org",  
      ip: "10.0.0.30",  
      initiate_connection: true  
    }  
  ]  
}
```

□□ **PCRF**□□□□□

```
diameter: %{
  listen_ip: "0.0.0.0",
  host: "omnipgw.epc.mnc001.mcc001.3gppnetwork.org",
  realm: "epc.mnc001.mcc001.3gppnetwork.org",
  peer_list: [
    %{
      host: "pcrf-primary.epc.mnc001.mcc001.3gppnetwork.org",
      realm: "epc.mnc001.mcc001.3gppnetwork.org",
      ip: "10.0.1.30",
      initiate_connection: true
    },
    %{
      host: "pcrf-backup.epc.mnc001.mcc001.3gppnetwork.org",
      realm: "epc.mnc001.mcc001.3gppnetwork.org",
      ip: "10.0.2.30",
      initiate_connection: true
    }
  ]
}
```

PCRF 配置

```
diameter: %{
  listen_ip: "0.0.0.0",
  host: "omnipgw.epc.mnc001.mcc001.3gppnetwork.org",
  realm: "epc.mnc001.mcc001.3gppnetwork.org",
  peer_list: [
    %{
      host: "pcrf.epc.mnc001.mcc001.3gppnetwork.org",
      realm: "epc.mnc001.mcc001.3gppnetwork.org",
      ip: "10.0.0.30",
      initiate_connection: false # PCRF 不主动连接
    }
  ]
}
```

配置 Diameter Gx 接口

S5/S8

SGW-C GTP-C

```
config :pgw_c,  
  s5s8: %{  
    # S5/S8 IPv4  
    local_ipv4_address: "10.0.0.20",  
  
    # IPv6  
    local_ipv6_address: nil,  
  
    # GTP-C 2123  
    local_port: 2123,  
  
    # GTP-C 500  
    # GTP-C  
    request_timeout_ms: 500,  
  
    # GTP-C 3  
    # = request_timeout_ms * request_attempts  
    request_attempts: 3  
  }
```


이

필드	타입	값	설명
local_ipv4_address	문자열 IPv4	10.0.0.20	S5/S8 노드의 IPv4 주소
local_ipv6_address	문자열 IPv6	nil	S5/S8 노드의 IPv6 주소 (없음)
local_port	정수	2123	GTP-C를 위한 UDP 포트 번호 2123
request_timeout_ms	정수	500	요청 시간 제한 (ms)
request_attempts	정수	3	요청 시도 횟수

이

- GTP-C를 위한 포트 2123
- UDP
- 2123
- SGW-C와 연결

이

이 IPv4

```
s5s8: %{\n  local_ipv4_address: "10.0.0.20"\n}
```

IPv4 + IPv6

```
s5s8: %{\n  local_ipv4_address: "10.0.0.20",\n  local_ipv6_address: "2001:db8::20"\n}
```

配置示例

```
s5s8: %{\n  local_ipv4_address: "10.0.0.20",\n  local_port: 2124 # 端口\n}
```

配置示例

```
s5s8: %{\n  local_ipv4_address: "10.0.0.20",\n  request_timeout_ms: 1500, # 请求超时 1.5 秒\n  request_attempts: 3      # 请求次数 4.5 秒\n}
```

配置示例

S5/S8 配置示例 GTP-C 配置/配置示例

配置示例

配置示例 = request_timeout_ms × request_attempts
配置500ms × 3 = 1.5 秒

配置示例

配置示例	配置示例	配置示例
配置示例<50ms	200-300	600-900
配置示例50-150ms	500	1.5
配置示例>150ms	1000-2000	3-6
配置/配置	2000-3000	6-9

Wireshark

- Wireshark 安裝及設定
- Wireshark 抓包及分析
- Wireshark 抓包及分析

Wireshark

- Wireshark 抓包及分析
- Wireshark 抓包及分析
- Wireshark 抓包及分析

Wireshark

IP

- IP 地址
- IP 地址
- IP 地址

Wireshark

```
# SGW-C 到 GTP-C
iptables -A INPUT -p udp --dport 2123 -s <sgw_c_network> -j ACCEPT
```

Gn/Gp 網路 (GGSN)

Wireshark

Wireshark 抓包及分析

配置

```
config :pgw_c,
  gn: %{
    # Gn 网元 IPv4 地址
    local_ipv4_address: "10.0.0.20",

    # 网元 IPv6 地址 2G/3G 网元 IPv4 地址
    local_ipv6_address: nil,

    # GTP-C 网元 S5/S8 网元
    local_port: 2123
  },

  # PC0 网元 DNS 网元 S5/S8 网元
  dns: %{
    primary_ipv4: {8, 8, 8, 8},
    secondary_ipv4: {8, 8, 4, 4}
  }
}
```

数据

属性	类型	值	描述
local_ipv4_address	网元 IPv4 地址	10.0.0.20	Gn 网元 IPv4 地址 GGSN 网元
local_ipv6_address	网元 IPv6 地址	nil	Gn 网元 IPv6 地址
local_port	端口	2123	GTP-C v1 网元 UDP 网元 S5/S8 网元

DNS

IP	Port	IP	Port
primary_ipv4	53	{8, 8, 8, 8}	PCO 53 DNS 53
secondary_ipv4	53	{8, 8, 4, 4}	PCO 53 DNS 53

3GPP

- GTP-C 1 (3GPP TS 29.060)
- UDP
- 2123 GTP-C v2
- SGSN

S5/S8

OmniPGW PGW 4G GGSN 2G/3G

- UDP 2123
- GTP
- IP 4G 2G/3G
- IP UPF/PFCP

Gn/Gp

Sxb/PFCP

PGW-U PFCP

配置

```
config :pgw_c,
  sxb: %{
    # PFCP 地址 IP
    local_ip_address: "10.0.0.20",

    # PFCP 端口 8805
    local_port: 8805
  }
```

表

名称	类型	值	说明
local_ip_address	字符串	IP	PFCP 地址
local_port	整数	8805	PFCP UDP 端口

配置

- 在 **UPF** 配置文件中添加 **upf_selection** 配置项 + 配置值
- 配置 UPF 配置项
 - 配置值 "UPF-<ip>:<port>"
 - 在 PFCP 配置项中配置 UPF 地址
 - 5 配置项
- UPF 配置项中配置 **upf_selection** 配置项 + **UPF 地址**
- 配置 UPF 配置项中配置 DNS 配置项 UPF

表

配置

```
sxb: %{  
  local_ip_address: "10.0.0.20"  
}  
  
# [] upf_selection [] UPF []  
# - []"UPF-10.0.0.21:8805"  
# - [] PFCP [] UPF []  
# - 5 []
```

[] **PFCP** []

```
sxb: %{  
  local_ip_address: "10.0.0.20",  
  local_port: 8806 # [] PFCP []  
}
```

[] **UPF** []

```
sxb: %{
  local_ip_address: "10.0.0.20"
},
upf_selection: %{
  rules: [
    %{
      name: "IMS ",
      priority: 10,
      match_field: :apn,
      match_regex: ~r/^ims$/,
      upf_pool: [
        %{remote_ip_address: "10.0.1.21", remote_port: 8805,
weight: 100},
        %{remote_ip_address: "10.0.1.22", remote_port: 8805,
weight: 100}
      ]
    }
  ],
  fallback_pool: [
    %{remote_ip_address: "10.0.2.21", remote_port: 8805, weight:
100}
  ]
}
# 3 UPF 10.0.1.21 10.0.1.22 10.0.2.21
```

DNS

```
sxb: %{
  local_ip_address: "10.0.0.20"
},
upf_selection: %{
  dns_enabled: true,
  dns_query_priority: [:ecgi, :tai],
  dns_suffix: "epc.3gppnetwork.org",
  fallback_pool: [
    %{remote_ip_address: "10.0.2.21", remote_port: 8805, weight:
100}
  ]
}
# DNS UPF
```


UPF 部署

UPF 部署架构图 UPF 部署 `upf_selection` 部署图

部署

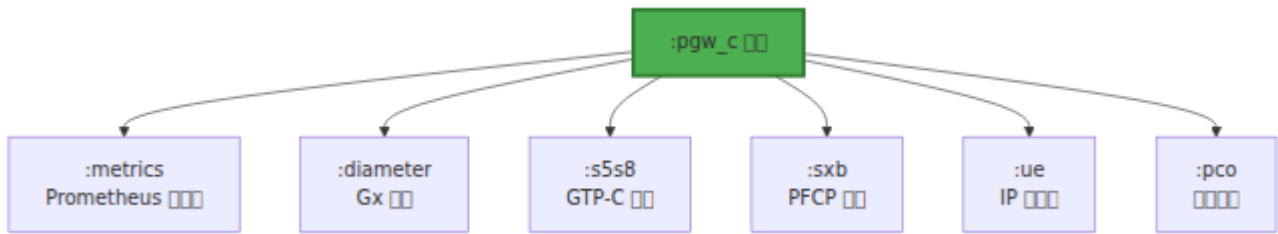
UPF 部署 `upf_selection` 部署图

- 部署 - 部署图
- DNS** 部署 - 部署图 UPF 部署
- 部署 - 部署图 DNS 部署图

部署图

- 部署图 - 部署图
- 部署 **DNS** 部署图 - 部署图 UPF 部署
- 部署图   - 部署图 DNS 部署图

UPF 部署图



部署图

部署图

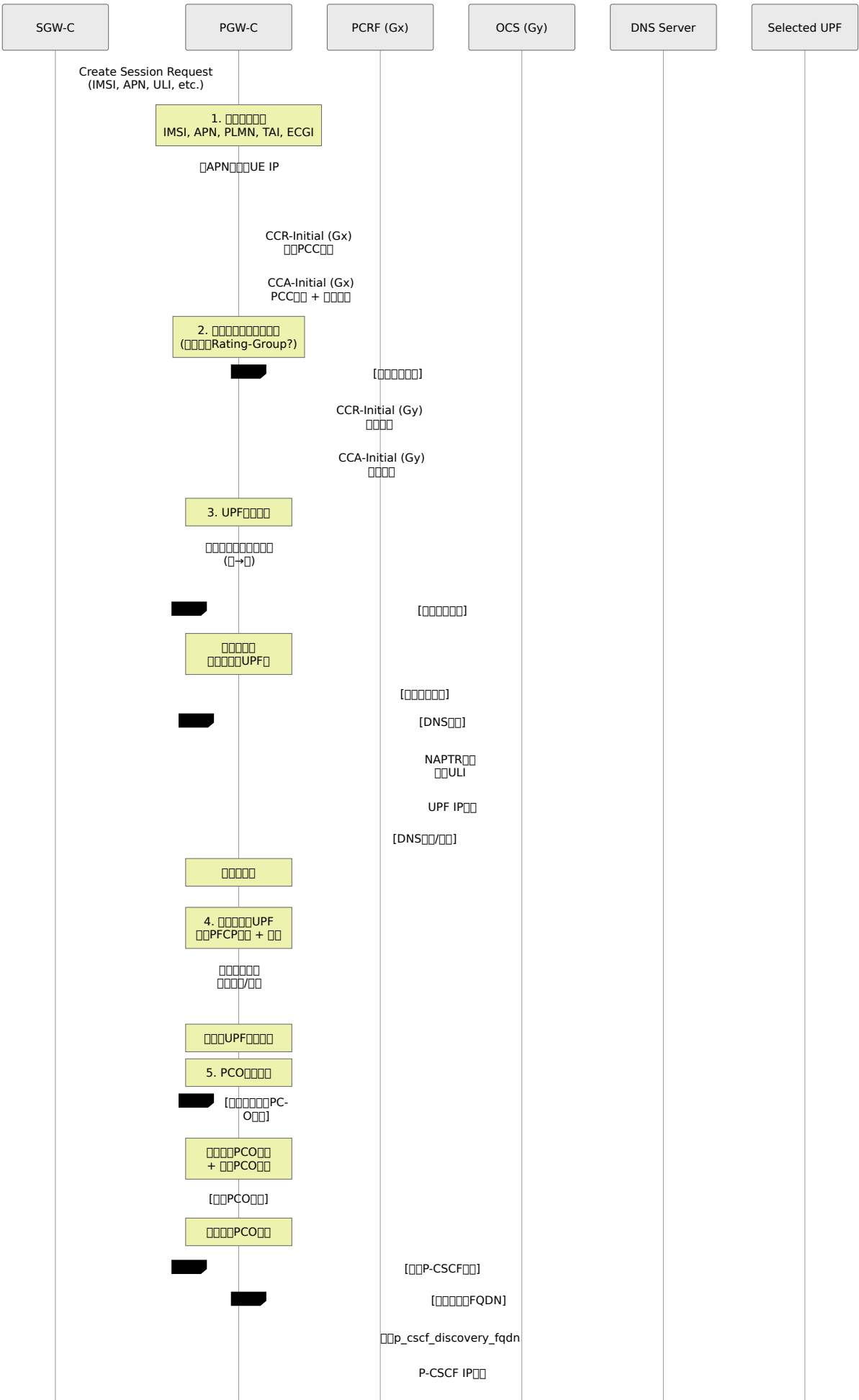
項目名	説明	形式
:imsi	IMSI	^001001.*
:apn	APN / DNN	^internet\. ^ims\.
:serving_network_plmn_id	PLMN ID	^001001\$
:sgw_ip_address	SGW IP	^10\.100\..*
:uli_tai_plmn_id	PLMN ID	^313.*
:uli_ecgi_plmn_id	E-UTRAN PLMN ID	^313.*

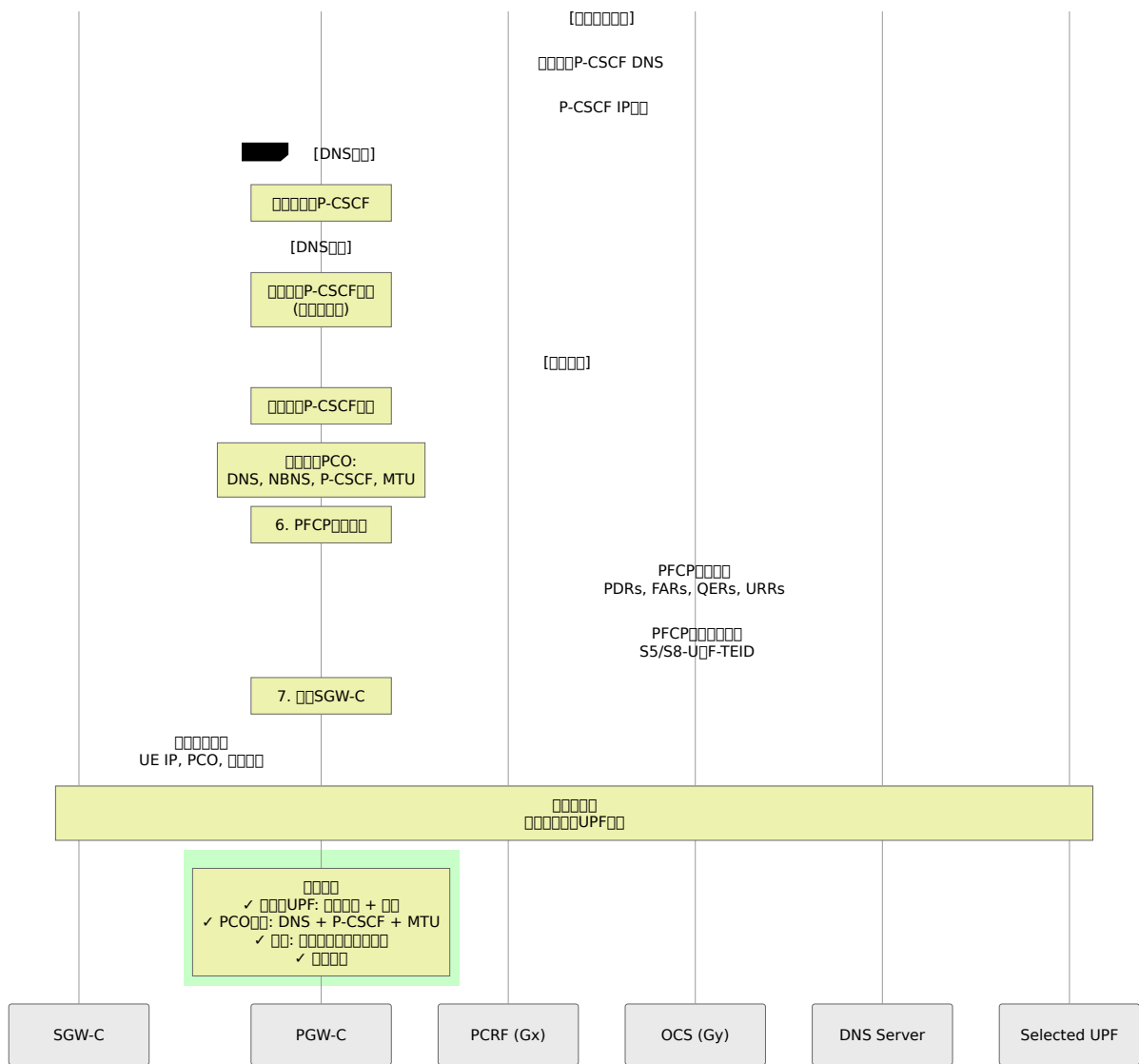
ネットワーク

項目名	説明	形式	項目名
UPF	UPF	UPF	UPF
APN	APN	IMS/	
IMSI	IMSI		
DNS	MEC/		DNS
		UPF	

□□□□□□□□

□□□□□□□UPF□□□PCO□□□□□□□□□□□□□□□□□





Sequence

1. UPF

- DNS → P-CSCF
- P-CSCF
- P-CSCF / P-CSCF
- P-CSCF PFCP UPF

2. PCO

- PCO → P-CSCF DNS → PCO
- P-CSCF
- PCO P-CSCF

3. P-CSCF

- IP FQDN → DNS → PCO → PCO
- P-CSCF 注册

4. 注册

- PCRF 注册 Rating-Group + Online=1
- OCS 注册
- PGW-C 注册 CCR-Update
- Diameter Gx 与 Diameter Gy 注册

注册

UPF 注册

```

config :pgw_c,
  # PCF - UPF upf_selection
  sxb: %{
    local_ip_address: "127.0.0.20"
  },

  # UPF - UPF
  upf_selection: %{
    # DNS
    dns_enabled: false,
    dns_query_priority: [:ecgi, :tai, :rai, :sai, :cgi],
    dns_suffix: "epc.3gppnetwork.org",
    dns_timeout_ms: 5000,

    # Rules
    rules: [
      # 1 IMS - 
      %{
        name: "IMS Traffic",
        priority: 20,
        match_field: :apn,
        match_regex: "^ims",
        upf_pool: [
          weight: 80,
          %{remote_ip_address: "10.100.2.21", remote_port: 8805,
          weight: 20}
          %{remote_ip_address: "10.100.2.22", remote_port: 8805,
          weight: 20}
        ]
      },

      # 2 APN
      %{
        name: "Enterprise Traffic",
        priority: 15,
        match_field: :apn,
        match_regex: "^(enterprise|corporate)\\.apn",
        upf_pool: [
          weight: 100,
          %{remote_ip_address: "10.100.3.21", remote_port: 8805,
          weight: 100}
        ]
      },

      # 3 - 

```

```

    %{
      name: "Internet Traffic",
      priority: 5,
      match_field: :apn,
      match_regex: "^internet",
      upf_pool: [
        %{remote_ip_address: "10.100.1.21", remote_port: 8805,
weight: 33},
        %{remote_ip_address: "10.100.1.22", remote_port: 8805,
weight: 33},
        %{remote_ip_address: "10.100.1.23", remote_port: 8805,
weight: 34}
      ]
    },

    # DNS - 本地DNS服务器
    fallback_pool: [
      %{remote_ip_address: "127.0.0.21", remote_port: 8805,
weight: 100}
    ]
  }

```

配置

配置

- 配置 `upf_selection` 为 UPF 列表
- 配置 `UPF` 列表
- 配置 `upf_pool` 为 UPF 列表
- 配置 `fallback_pool` 为本地 DNS
- 配置 **DNS** 为本地 DNS
- 配置 `DNS` 为 UPF
- 配置 `UPF` 为 5 个 UPF

配置

- 配置 `sxb.peer_list` 为 UPF 列表
- 配置 `selection_list` 为 UPF 列表

- 每个UPF都维护一个 `upf_selection` 列表

UPF 列表

1. 初始化的UPF列表

- $list = PFCP_list + 本地UPF \times 3$
- 每个UPF都维护一个
- 每个UPF都维护一个UPF列表

2. 每个/每个 `weight: 0` 的UPF

- 每个UPF `weight > 0` 的UPF
- 每个UPF `weight == 0` 的UPF
- 每个UPF `weight: 1`

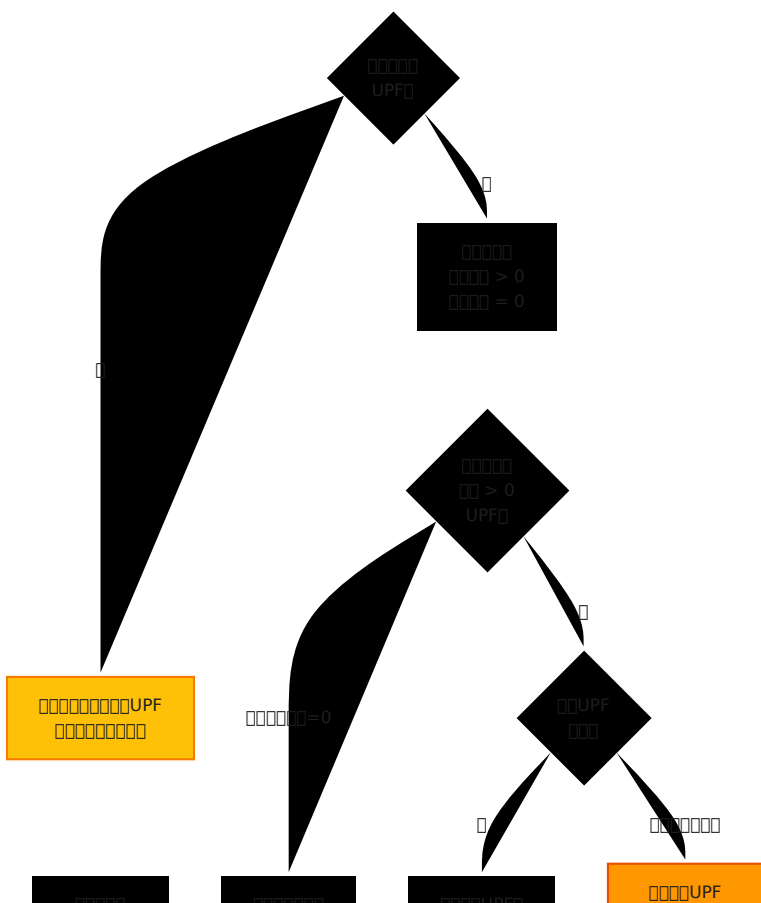
3. 每个/每个UPF

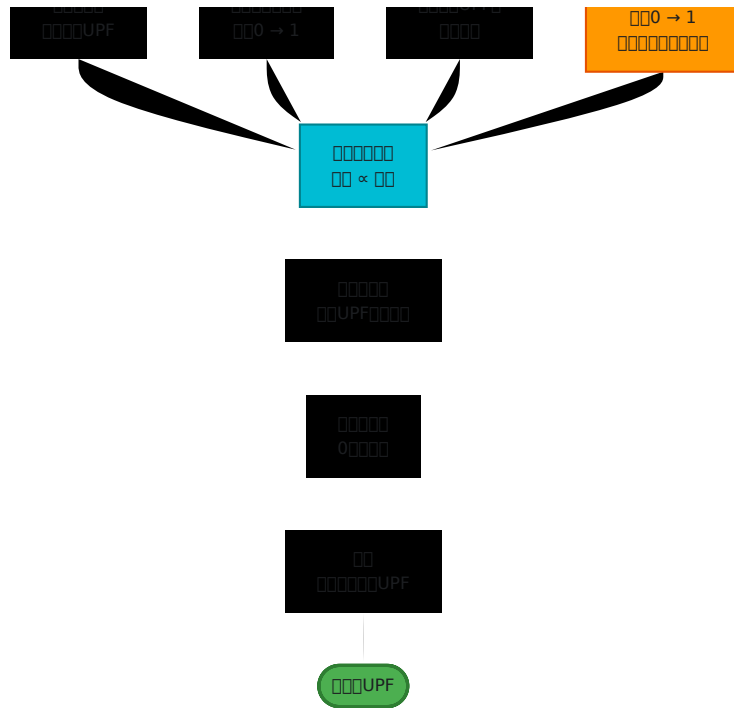
- 每个UPF 70% 的.21 20% 的.22 10% 的.23
- 每个UPF = 每个UPF
- 每个UPF = 每个UPF

4. 每个/每个UPF

- 每个UPF `"UPF-<ip>:<port>"`
- 每个UPF PFCP 5 个
- 每个UPF 每个UPF

□□□□□□□□/□□





UPF

[
 UPF-A: 50, ✓
 UPF-B: 30, ✓
 UPF-C: 20, ✓
]

UPF: 50 + 30 + 20 = 100

UPF:
 UPF-A: 0-49 (50%)
 UPF-B: 50-79 (30%)
 UPF-C: 80-99 (20%)

UPF: 63 → UPF-B
 UPF: 15 → UPF-A
 UPF: 91 → UPF-C

UPF/UPF

```
{}: [  
  UPF-A: {} 100, {} ✓    ({})  
  UPF-B: {} 0, {} ✓      ({})  
]
```

```
{}1: UPF-A{}  
→ {}: [UPF-A: 100]  
→ {}UPF-A
```

```
{}2: UPF-A{}  
→ {}UPF{}  
→ {}: [UPF-B: 1]  
→ {}UPF-B  
→ {}: "{}UPF{}UPF"
```

```
{}3: {}  
→ {}UPF  
→ {}: [UPF-A: 100, UPF-B: 0]  
→ {}  
→ {}: "{}UPF{}"
```

{} {}

```

# 25%
upf_pool: [
    %{remote_ip_address: "10.0.1.1", remote_port: 8805, weight: 1},
    %{remote_ip_address: "10.0.1.2", remote_port: 8805, weight: 1},
    %{remote_ip_address: "10.0.1.3", remote_port: 8805, weight: 1},
    %{remote_ip_address: "10.0.1.4", remote_port: 8805, weight: 1}
]

# 90% / 10%
upf_pool: [
    %{remote_ip_address: "10.0.1.21", remote_port: 8805, weight: 90},
    %{remote_ip_address: "10.0.1.22", remote_port: 8805, weight: 10}
]

# 100% 0%
upf_pool: [
    %{remote_ip_address: "10.0.1.21", remote_port: 8805, weight: 100}, #
    %{remote_ip_address: "10.0.1.22", remote_port: 8805, weight: 0}
]

#
upf_pool: [
    %{remote_ip_address: "10.0.1.1", remote_port: 8805, weight: 100}, #
    %{remote_ip_address: "10.0.1.2", remote_port: 8805, weight: 0},
    %{remote_ip_address: "10.0.1.3", remote_port: 8805, weight: 0}
]

# 100% 00 50/50%
# A/B 50% / 50%
upf_pool: [
    %{remote_ip_address: "10.0.1.100", remote_port: 8805, weight: 50}, #
    %{remote_ip_address: "10.0.1.200", remote_port: 8805, weight: 50} #
]

```

111

- α/α weight: 0 UPF
- **HA** UPF PFCP
- UPF
- UPF
- 95% 5%
- UPF
- UPF

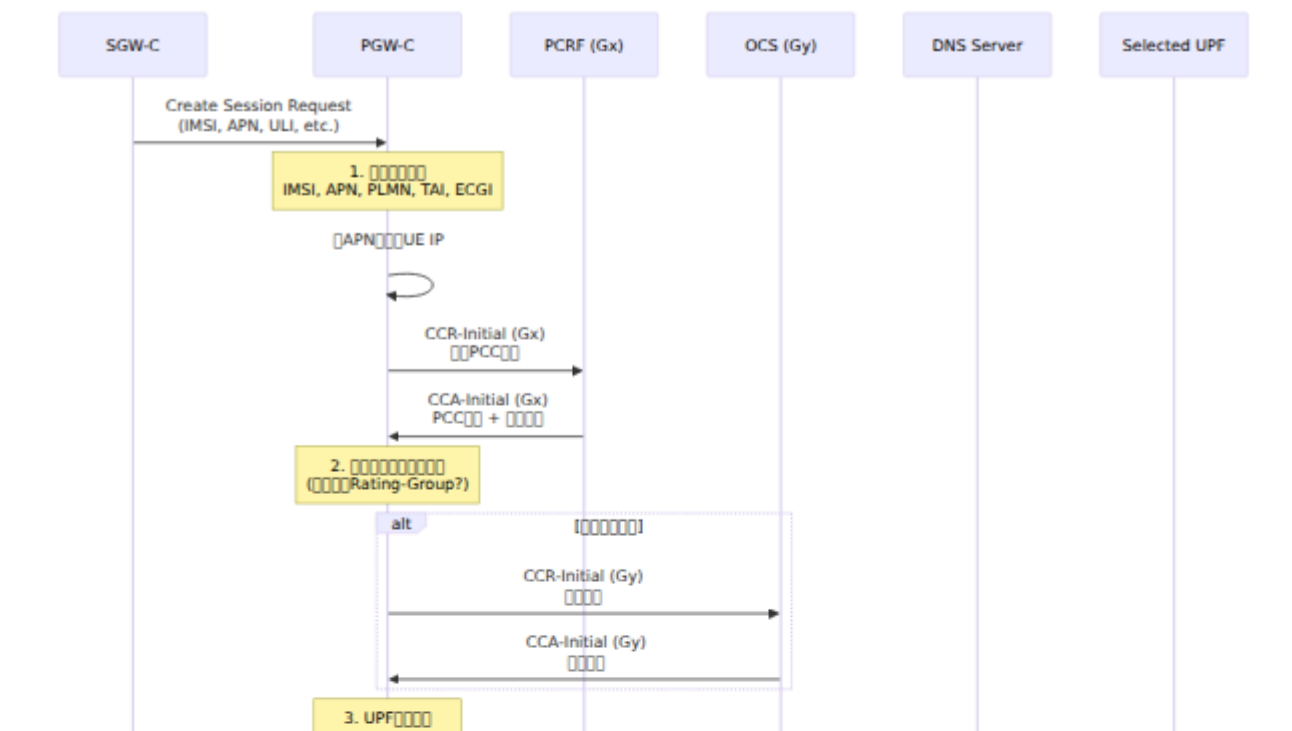
PCO□□□□□□□□□□

UPF PCO PCO APN

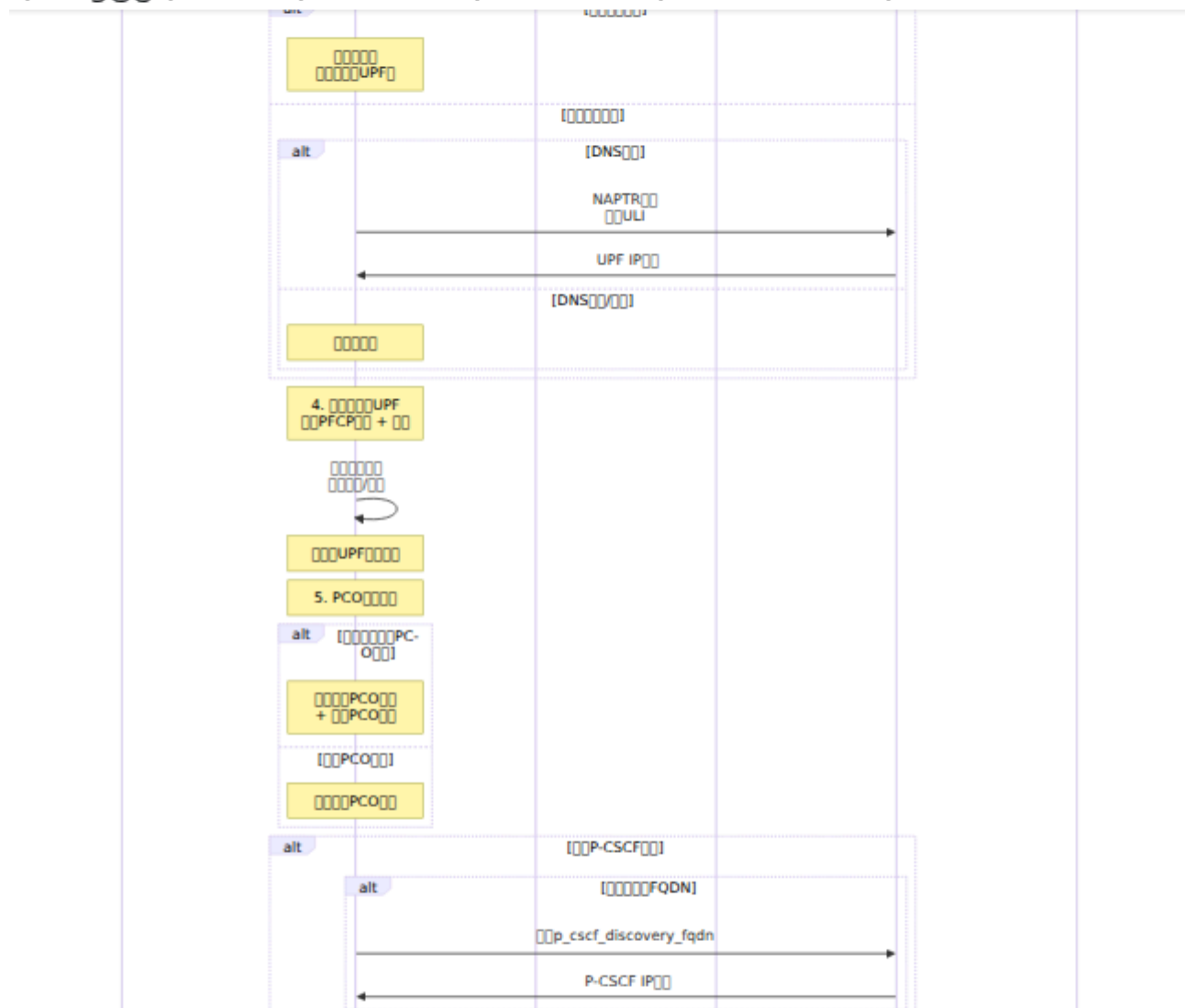
PCO□□□□□□□□

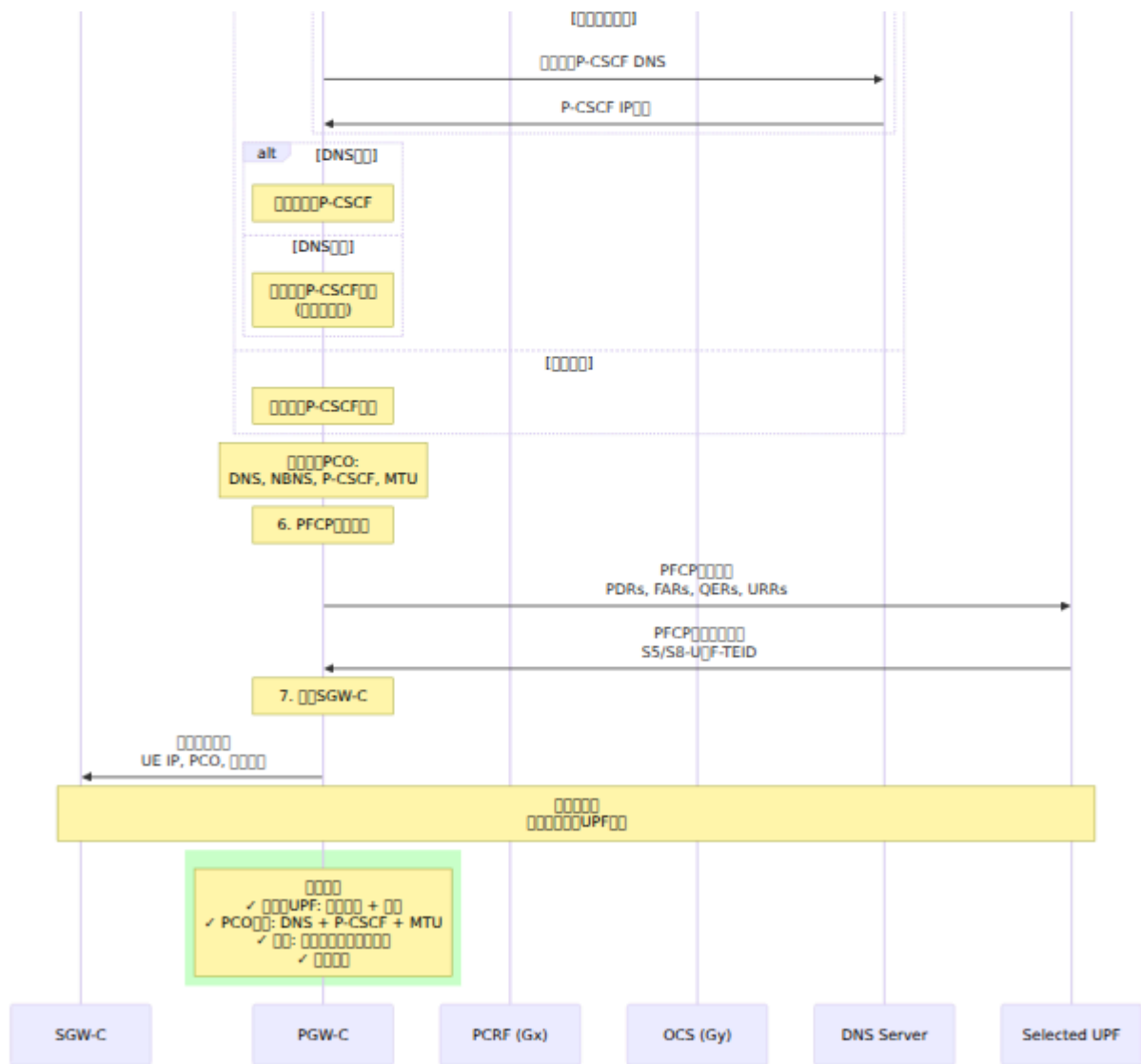
1. □□□□□□□□□□PCO□□
2. □□□□□□□□□□□□pc□□□□□
3. □□□□□□□□□□□□□□PCO□□
4. □□□□□□□□PCO□□□□□□PCO

PCO□□□□



ore OmniCore OmniCall OmniRAN OmniCharge Platform 5G A





PCO

1. PCO
2. P-CSCF DNS P-CSCF
3. PCO /

IMS DNS

└─ DNS□□□□□□□□□□□□□□

```
|— P-CSCF[]DNS[]p_cscf_discovery_fqdn[]
```

L MTU

```
|─ DNS[ ][ ][ ][ ][ ][ ][ ][ ]192.168.1.10, 192.168.1.11[ ]
```

└ MTU□□□□□□1500□

```
└─ DNS□□□□□□□□
```

MTU□□□□□

- `primary_dns_server_address` - DNS IP IPv4

- secondary_dns_server_address - DNS IP IPv4

- primary_dns_server_ipv6_address - DNS IP IPv6 0x0003

- secondary_dns_server_ipv6_address - DNS IP IPv6 0x0003

- `primary_nbns_server_address` - □WINS□□□IP

- secondary_nbns_server_address - □WINS□□□IP

- `p_cscf_ipv4_address_list` - P-CSCF IP IPv4 IMS 0x000C - PCO P-CSCF P-CSCF

- `p_cscf_ipv6_address_list` - P-CSCF IP IPv6 address list (IMS) 0x0001

- `ipv4_link_mtu_size` - MTU

PCOUPFPP-P-CSCF

- `p_cscf_discovery_fqdn` - DNS P-CSCF FQDN
`"pcscf.mnc001.mcc001.3gppnetwork.org"`

ขั้นตอนที่ 1

1. PGW-C FQDN DNS
2. DNS P-CSCF IP
3. P-CSCF PCO UE
4. DNS PCO `p_cscf_ipv4_address_list` PCO
5. / P-CSCF

ขั้นตอนที่ 2

- **IMS APNs** - IMS P-CSCF
- - P-CSCF
- - DNS UE P-CSCF
- - DNS P-CSCF

ขั้นตอนที่ 3 **P-CSCF** IMS

```

rules: [
  %{
    name: "IMS Traffic",
    priority: 20,
    match_field: :apn,
    match_regex: "^ims",
    upf_pool: [
      %{remote_ip_address: "10.100.2.21", remote_port: 8805,
weight: 80},
      %{remote_ip_address: "10.100.2.22", remote_port: 8805,
weight: 20}
    ],
    # P-CSCF 00000000 DNS 0000 P-CSCF 00
    # DNS 000000 FQDN 0000 P-CSCF IP
    p_cscf_discovery_fqdn: "pcscf.mnc001.mcc001.3gppnetwork.org",
    # IMS 00000000 P-CSCF 000000 DNS 00000000
    pco: %{
      p_cscf_ipv4_address_list: ["203.0.113.100", "203.0.113.101"]
      # DNS0NBNS0MTU 0000 PCO 00000000
    }
  }
]

```

000000 **UE IP** 0000

00 UPF 00000000 **UE IP** 000000000000000000000000

00	00000	0000	00000
1. 00	(0000)	PGW-C0000000 0 UPF0	APN 0000 ue.subnet_map00 00 :default
2. 000	ue_ip_pool: "name"	PGW-C0000000 0 UPF0	0000 ue.subnet_map 0 "name"
3. UPF 00	upf_assigns_ip: true	UPF0PFCP CHV40	UPF 000000

- `upf_assigns_ip` - 00000000 `false`0000 `true` 00PGW-C 00 UE-IP-Address IE00
00 **CHV4**0“UP 00”0000000000000000 UPF 0 Created-PDR 0000000000000000
00**UPF** 0000000000 **UPF** 00000000 UPF 00000000000000 APN / PFCP 00000000
00— PGW-C 0000000000000000
- `ue_ip_pool` - 0000000000000000**PGW-C** 000000 `ue.subnet_map` 00000000
PGW-C 00 UPF 0000000000000000/00 `100.64.20.0/24` 0 UPF 000000000000
000000000000000000000000 `nil` → 0000 APN 00 `ue.subnet_map` 000000 100

`upf_assigns_ip` 0 `ue_ip_pool` 0 0000 — 0000000000 UPF0000000000 PGW-C 00000000
00000000000000000000000000000000 `upf_assigns_ip` 0000 `ue_ip_pool` 000000

0000 PGW-C 00 UPF 000000000000 20 // UPF 000000 3000000000 UPF 0000000000000000
0 OSPF 000000 `100.64.20.0/24`0000000000 UE 000000 `/32` — 0000 UPF 00000000000000
00 UE 00000000000000000000000000000000 UE 0000000000000000

00 — 0/**UPF** 0000 0000 UE 0000000000000000 UPF 0000000000000000 200000 UPF 000000

`upf_pool` 000000 UPF 000000000000 30000 `upf_pool` UPF 000000000000??

`ueip_pool`0omniupf `feature_ueip: true`00000000

0 `route_advertise_pools`000000 UE 000000 UPF 00000000000000000000 UPF 000000000000
00

00 — 00 20**PGW-C** 0000000000 **UPF** 00000000

```
# ue.subnet_map
ue: %{
  subnet_map: %{
    "ims_upf03" => ["100.64.20.0/24"],
    "ims_upf04" => ["100.64.21.0/24"],
    default:      ["100.64.16.0/24"]
  }
},
# UPF
upf_selection: %{
  rules: [
    %{name: "IMS→upf03", priority: 21, match_field: :apn,
match_regex: "^ims",
  upf_pool: [{remote_ip_address: "10.179.1.42", remote_port:
8805, weight: 1}],
  ue_ip_pool: "ims_upf03"},
    %{name: "IMS→upf04", priority: 20, match_field: :apn,
match_regex: "^ims",
  upf_pool: [{remote_ip_address: "10.179.1.45", remote_port:
8805, weight: 1}],
  ue_ip_pool: "ims_upf04"}
  ]
}
```

3UPF CHV4 UE IP

```
rules: [
  %{
    name: "IMS",
    priority: 20,
    match_field: :apn,
    match_regex: "^ims",
    upf_pool: [
      %{remote_ip_address: "10.179.1.42", remote_port: 8805,
weight: 50},
      %{remote_ip_address: "10.179.1.45", remote_port: 8805,
weight: 50}
    ],
    upf_assigns_ip: true # UPF UE IPCHV4 UPF
  }
]
```

配置 DNS 规则

```
rules: [  
  {%  
    name: "Enterprise Traffic",  
    priority: 15,  
    match_field: :apn,  
    match_regex: "^(enterprise|corporate)\.apn",  
    upf_pool: [  
      {%remote_ip_address: "10.100.3.21", remote_port: 8805,  
weight: 100}  
    ],  
    # 配置 DNS 规则 MTU  
    pco: {%  
      primary_dns_server_address: "192.168.1.10",  
      secondary_dns_server_address: "192.168.1.11",  
      ipv4_link_mtu_size: 1500  
      # P-CSCF/NBNS 规则 PCO 配置  
    }  
  }  
]
```

配置 PCO 规则


```

rules: [
  %{
    name: "IoT APN - Fully Custom",
    priority: 10,
    match_field: :apn,
    match_regex: "^iot\\.m2m",
    upf_pool: [
      %{remote_ip_address: "10.100.5.21", remote_port: 8805,
weight: 100}
    ],
    # IoT PCO
    pco: %{
      primary_dns_server_address: "8.8.8.8",
      secondary_dns_server_address: "8.8.4.4",
      primary_nbns_server_address: "10.0.0.100",
      secondary_nbns_server_address: "10.0.0.101",
      p_cscf_ipv4_address_list: [], # IoT P-CSCF
      ipv4_link_mtu_size: 1280 # MTU
    }
  }
]

```

•

- **IMS/VoLTE** P-CSCF
- **APNs** DNS
- **IoT/M2M** DNS MTU
- DNS
-

DNS **UPF**

ULI UPF DNS NAPTR DNS `upf_selection`

UPF UPF PFCP PFCP

```
upf_selection: %{
  # DNS
  dns_enabled: true,

  #
  dns_query_priority: [:ecgi, :tai, :rai, :sai, :cgi],

  # 3GPP NAPTR DNS
  dns_suffix: "epc.3gppnetwork.org",

  # DNS
  dns_timeout_ms: 5000,

  # ... fallback_pool ...
}
```

DNS

1. DNS
2. UE DNS NAPTR
 - ECGI `eci-<hex>.ecgi.epc.mnc<MNC>.mcc<MCC>.epc.3gppnetwork.org`
 - TAI `tac-lb<hex>.tac-hb<hex>.tac.epc.mnc<MNC>.mcc<MCC>.epc.3gppnetwork.org`
 - RAI/SAI/CGI 3GPP TS 23.003
- 3.
4. DNS
- 5.

DNS DNS

```
; NAPTR PLMN 001-001 TAC 100
tac-lb64.tac-hb00.tac.epc.mnc001.mcc001.epc.3gppnetwork.org IN
NAPTR 10 50 "a" "x-3gpp-upf:x-sxb" "" upf-edge-1.example.com.

; UPF A
upf-edge-1.example.com IN A 10.100.1.21
```

□□□

- □□□□□□□**MEC**□□□□□□□□□□□□□□□□ UPF
- □□ **UPF** □□□□□/□□ UPF □□□□□□□ PGW-C
- □□□□□□□□□□ UPF □□□□□□
- □□□□□□□□□□□□□□□□□□□□□□ UPF

UPF □□□□

□□□□□□□□□□PGW-C □□□□□□ UPF □□□□□□□□□□□□□□□□ UPF □□□□□□□□

□□□□□□

□□□□□□□□□□□□UPF □□□ □□□

1. **PFCP** □□□□□□UPF □□□ PFCP □□
2. □□□□□□□□ 3 □□□□□□□□□□
3. □□□□□□UPF □□□ GenServer □□□□□□

□□□□□□□□□□□□ UPF □□□ □□□□

- PFCP □□□□□□ `associated: false`□
- 3 □□□□□□□□□□□□
- UPF □□□□□□□□□□□□

□□□□

□□□□□□ **UPF**□□ `upf_selection` □□□

- □□□□□□□□□□□□□□
- PFCP □□□□□□□
- □ 5 □□□□□□□□
- `missed_heartbeats_consecutive` □□□□□□□□□□
- □□□□□□□□□□□□ UPF □□□□□□

□□ **DNS** □□□ **UPF**□□□□□□□□

- □□□□□□□□□□□□□□

- 000000000000
- 000000000000

0000

00/00000000 weight: 0 000

1. 000000 UPF
2. 00 000000 > 000 000000 == 00
3. 000000000 UPF00000000 UPF
4. 0000000 UPF 000000000000 UPF000000 10
5. 00000000 UPF0000000000

000000000000 > 000

1. 00000000 UPF
2. 0000 UPF 000000000000
3. 00000000 UPF0000000000

000000

```
[debug] 0000 UPF 002/3 00 UPF01 000
[info] 0000 UPF 000000000000 UPF01 00 UPF00000 0 00 10
[warning] 00000000 UPF00 3 00000000000000
```

00 **UPF** 00

00000000

```
# Verify UPF status
iex> PGW_C.PFCP_Node.is_peer_healthy({10, 100, 1, 21})
true

# Verify peer health
iex> PGW_C.PFCP_Node.get_peer_health({10, 100, 1, 21})
%{
  associated: true,
  missed_heartbeats: 0,
  healthy: true,
  registered: true
}
```

Verify OAM API

- Verify `/api/upf_selection` endpoint `curl -k https://localhost:8443/api/upf_selection`
- Verify UPF status
- Verify Active-UP, Standby-Ready, Active-DOWN, Not Associated
- Verify ACTIVE count > 0, STANDBY count == 0, DYNAMIC DNS
- Verify UPF status

Verify UPF

1. Verify `upf_selection` endpoint **UPF** status

```
upf_selection: %{
  rules: [
    %{
      name: "Internet Traffic",
      priority: 10,
      match_field: :apn,
      match_regex: "^internet",
      upf_pool: [
        %{remote_ip_address: "10.100.1.21", remote_port: 8805,
weight: 100}
      ]
    }
  ],
  fallback_pool: [
    %{remote_ip_address: "10.100.2.21", remote_port: 8805,
weight: 100}
  ]
}
# [] UPF []
# - 5 []
# - []
# - []
```

2. [] **UPF** [] 0 []

```
upf_pool: [
  %{remote_ip_address: "10.1.1.1", remote_port: 8805, weight:
100}, # []
  %{remote_ip_address: "10.1.1.2", remote_port: 8805, weight:
0} # []
]
```

3. [] **OAM API** [] `/api/upf_selection` [] UPF []

4. [] 3 []

[] **UPF** []

[] PGW-C [] DNS [] UPF[] `upf_selection` []

#####

DNS##### UPF

1. ##### **PFCP** ##### UPF #####
2. ##### **PFCP** ##### UPF ##### PFCP #####
3. ##### UPF #####
4. #####10 #####
5. ##### UPF #####

UPF

UPF

```
%{
  name: "Dynamic-UPF-<IP>",                                # ##### "Dynamic-UPF-10-100-1-21"
  remote_ip_address: <discovered_ip>,                      # ##### DNS ##### IP
  remote_port: 8805,                                       # ##### PFCP #####
  initiate_pfcip_association_setup: true,                 # PGW-C #####
  heartbeat_period_ms: 10_000                             # 10 #####
}
```

UPF ##### `upf_selection`

DNS ##### **UPF**

```
# DNS #####upf-edge-2.example.com -> 10.200.5.99
# ##### UPF ##### upf_selection #####

# #####
# 1. ##### UPF 10.200.5.99
# 2. #####“UPF {10, 200, 5, 99} #####...”
# 3. ##### 10.200.5.99:8805 ##### PFCP #####
# 4. ##### UPF #####
# 5. ##### UPF #####
```

#####

1. DNS UPF
 2. PGW-C
 3.
 4. UPF
 5. UPF

UPF PFCP

[error] PFCP UPF {10, 200, 5, 99}:timeout
 [error] UPF {10, 200, 5, 99} :timeout
 UPF upf_selection

PGW-C

UPF	upf_selection
DNS UPF	
/ UPF	
UPF	upf_selection
/ UPF	UPF

UPF

UPF

[info] PFCP -UPF-10-200-5-99 ({10, 200, 5, 99}:8805)
 [info] UPF -UPF-10-200-5-99 PID #PID<0.1234.0>

PGW-C PFCP Node 注册 UPF 对端

```
# 注册 PFCP Node
PGW_C.PFCP_Node.registered_peer_count()

# 获取 UPF 对端信息
PGW_C.PFCP_Node.get_peer({10, 200, 5, 99})
# 返回{:ok, #PID<0.1234.0>} 或 :error
```

UPF 注册 PGW-C

UPF 注册 PFCP 对端

```
# 注册 UPF
PGW_C.PFCP_Node.register_dynamic_peer({10, 200, 5, 99}, 9999)
```

PGW-C DNS 注册 PFCP 对端 8805 PFCP 对端

UPF 注册 PGW-C

UPF 注册 PFCP 对端

```
config :pgw_c,
  # 注册 PFCP 对端
  upf_selection_dry_run: true
```

注册 PFCP 对端

1. 注册 PFCP 对端 UPF
2. 注册 PFCP 对端
3. 注册 `assign_sxb_peer/1` 或 `{:error, :dry_run_mode}`
4. 注册 PFCP 对端
5. 注册 PFCP 对端 DNS 对端

注册 PFCP 对端

- `session-ipv4-address` UPF
- `session-ipv4-address` UPF `session-ipv4-address`
- `session-ipv4-address`

`session-ipv4-address`

- `:imsi` - IMSI
- `:apn` - APN/DNN
- `:serving_network_plmn_id` - PLMN ID
- `:sgw_ip_address` - SGW IP
- `:uli_tai_plmn_id` - PLMN ID
- `:uli_ecgi_plmn_id` - E-UTRAN PLMN ID

`session-ipv4-address`

```
# session-ipv4-address
heartbeat_period_ms: 2_000 # 2 s

# session-ipv4-address
heartbeat_period_ms: 5_000 # 5 s

# session-ipv4-address
heartbeat_period_ms: 10_000 # 10 s
```

`session-ipv4-address` PFCP `session-ipv4-address`

UE IP `session-ipv4-address`

`session-ipv4-address`

`session-ipv4-address` UE `session-ipv4-address` IP `session-ipv4-address` APN `session-ipv4-address`

例

```
config :pgw_c,
  ue: %{
    subnet_map: %{
      # APN "internet"
      "internet" => [
        "100.64.0.0/20" # 4094 IP
      ],

      # APN "ims"
      "ims" => [
        "100.64.16.0/22" # 1022 IP
      ],

      # 未指定 APN
      default: [
        "42.42.42.0/24" # 254 IP
      ]
    }
  }
}
```

例

項目	型	値	説明
subnet_map	Map	1	1 APN ごとに指定
default	List (型)	1	未指定 APN 指定

例

CIDR 形式 <network_address>/<prefix_length>

例 **IP** 形式

CIDR	IP	IP	
/24	256	254	100.64.1.1 - 100.64.1.254
/23	512	510	100.64.0.1 - 100.64.1.254
/22	1024	1022	100.64.0.1 - 100.64.3.254
/21	2048	2046	100.64.0.1 - 100.64.7.254
/20	4096	4094	100.64.0.1 - 100.64.15.254
/16	65536	65534	100.64.0.1 - 100.64.255.254

```
ue: %{\n  subnet_map: %{\n    "internet" => ["100.64.1.0/24"],\n    default: ["42.42.42.0/24"]\n  }\n}
```

```

ue: %{
  subnet_map: %{
    # 互联网 - 互联网
    "internet" => [
      "100.64.0.0/18" # 16,382 IP
    ],

    # IMS (VoLTE)
    "ims" => [
      "100.64.64.0/22" # 1,022 IP
    ],

    # 企业 APN
    "enterprise.corp" => [
      "10.100.0.0/16" # 65,534 IP
    ],

    # IoT 网络
    "iot.m2m" => [
      "100.64.72.0/20" # 4,094 IP
    ],

    # 默认
    default: [
      "42.42.42.0/24" # 254 IP
    ]
  }
}

```

互联网 APN 互联网

```
ue: %{
  subnet_map: %{
    "internet" => [
      "100.64.0.0/22",    # 1022 IP
      "100.64.4.0/22",    # 1022 IP
      "100.64.8.0/22",    # 1022 IP
      "100.64.12.0/22"    # 1022 IP
    ],
    # 4096 IP
    default: ["42.42.42.0/24"]
  }
}
```

インターネット

インターネット APN (^) インターネット

```
ue: %{
  subnet_map: %{
    # "ims" APN "ims" "ims.apn" "ims.something"
    "^ims" => [
      "100.64.10.0/24"
    ],

    # "m2m." APN "m2m.test" "m2m.prod"
    "^m2m\." => [
      "100.64.20.0/24"
    ],

    #
    "enterprise.corp" => [
      "10.100.0.0/16"
    ],

    default: ["42.42.42.0/24"]
  }
}
```

企業

- (^) インターネット

- `^` 行頭に一致する
- `$` 行末に一致する
- `.` 任意の文字に一致する
- `\\` 文字列をそのまま扱う

例

- `^ims` `ims` `ims.apn` `ims.foo.bar`
- `^.*\\.corp$` `foo.corp` `bar.corp`
- `^.*test.*` `test` `foo.test.bar`

IPv6

```
ue: %{
  subnet_map: %{
    # IPv4
    "internet" => [
      "100.64.0.0/20"
    ],

    # IPv6
    "internet.ipv6" => [
      "2001:db8:1::/48"
    ],

    default: [
      "42.42.42.0/24"
    ]
  }
}
```

IP

RFC 6598 NAT

- `100.64.0.0/10`
- `~400` IP
- NAT

IP RFC 1918

- 10.0.0.0/8 - 1600 IP
- 172.16.0.0/12 - 100 IP
- 192.168.0.0/16 - 65,534 IP

UE IP

PCO

UE PCO

□□□

```
config :pgw_c,  
  pco: %{  
    # DNS □□□  
    primary_dns_server_address: "8.8.8.8",  
    secondary_dns_server_address: "8.8.4.4",  
  
    # IPv6 DNS □□□□□□□□ 0x0003□  
    primary_dns_server_ipv6_address: nil,  
    secondary_dns_server_ipv6_address: nil,  
  
    # NBNS □□□□□□□□□□ Windows □□□  
    primary_nbns_server_address: nil,  
    secondary_nbns_server_address: nil,  
  
    # IMS □ P-CSCF □□  
    p_cscf_ipv4_address_list: ["10.0.0.50", "10.0.0.51"], # □□  
    0x000C  
    p_cscf_ipv6_address_list: [], # □□  
    0x0001  
  
    # IPv4 MTU □□  
    ipv4_link_mtu_size: 1400  
  }  
}
```

primary_dns_server_address	 IPv4		DNS
secondary_dns_server_address	 IPv4		DNS
primary_dns_server_ipv6_address	 IPv6	nil	IPv6 DNS 0x0003
secondary_dns_server_ipv6_address	 IPv6	nil	IPv6 DNS 0x0003
primary_nbns_server_address	 IPv4	nil	NBNS NetBIOS
secondary_nbns_server_address	 IPv4	nil	NBNS
p_cscf_ipv4_address_list	 IPv4	[]	IMS IPv4 P- CSCF 0x000C
p_cscf_ipv6_address_list	 IPv6	[]	IMS IPv6 P- CSCF 0x0001
ipv4_link_mtu_size		1400	

DNS

```
pco: %{
  primary_dns_server_address: "8.8.8.8",
  secondary_dns_server_address: "8.8.4.4",
  ipv4_link_mtu_size: 1400
}
```

DNS

```
pco: %{
  primary_dns_server_address: "10.0.0.10",
  secondary_dns_server_address: "10.0.0.11",
  ipv4_link_mtu_size: 1400
}
```

IMS

```
pco: %{
  primary_dns_server_address: "10.0.0.10",
  secondary_dns_server_address: "10.0.0.11",

  # IMS/VoLTE P-CSCF
  p_cscf_ipv4_address_list: [
    "10.0.0.50", # P-CSCF
    "10.0.0.51" # P-CSCF
  ],

  ipv4_link_mtu_size: 1400
}
```

NBNS Windows

```
pco: %{
  primary_dns_server_address: "10.0.0.10",
  secondary_dns_server_address: "10.0.0.11",
  primary_nbns_server_address: "10.0.0.20",
  secondary_nbns_server_address: "10.0.0.21",
  ipv4_link_mtu_size: 1400
}
```

MTU

```
# 標準
ipv4_link_mtu_size: 1500

# 標準
ipv4_link_mtu_size: 1400

# Jumbo
ipv4_link_mtu_size: 9000
```

PCO

OAM API

OmniPGW OAM REST API API

```
API config_env() :test:test_mock :test_impl
```

REST API 配置

```
config :api_ex,  
  api: %{  
    # 配置  
    port: 8443,  
    listen_ip: "0.0.0.0",  
  
    # API 配置  
    product_name: "PGW-C",  
    title: "API - PGW-C",  
    hostname: "localhost",  
  
    # TLS 配置  
    enable_tls: true,  
    tls_cert_path: "priv/cert/omnitouch.crt",  
    tls_key_path: "priv/cert/omnitouch.pem",  
  
    # 路由配置  
    routes: [  
      %{  
        path: "/status",  
        module: ApiEx.Api.StatusController,  
        actions: [:index]  
      }  
    ]  
  }  
}
```

API

Property	Type	Value	Description
port	Integer	8443	REST API port HTTPS
listen_ip	String	"0.0.0.0"	API listen IP 0.0.0.0 = all IP
product_name	String	"PGW-C"	Product name API
title	String	"API - PGW-C"	API title
hostname	String	"localhost"	API hostname
enable_tls	Boolean	true	Enable API HTTPS
tls_cert_path	String		SSL/TLS certificate path
tls_key_path	String		SSL/TLS key path
routes	Array	[]	API routes

Routes

routes is an array of route objects

Property	Type	Value	Description
path	String	URL path	"/status"
module	String	Module name	ApiEx.Api.StatusController
actions	Array	Actions	[:index, :show]

□□

□□□□□□□□

```
# API □ localhost:8443
config :api_ex,
  api: %{
    port: 8443,
    listen_ip: "127.0.0.1", # □□□□□□
    hostname: "localhost",
    enable_tls: true,
    tls_cert_path: "priv/cert/dev-cert.crt",
    tls_key_path: "priv/cert/dev-key.pem",
    routes: [
      %{path: "/status", module: ApiEx.Api.StatusController,
actions: [:index]}
    ]
  }
```

□□□□□

```
# API □□□□□
config :api_ex,
  api: %{
    port: 8443,
    listen_ip: "10.0.0.20", # □□□□
    product_name: "OmniPGW-C",
    hostname: "pgw-c-api.example.com",
    enable_tls: true,
    tls_cert_path: "/etc/ssl/certs/pgw-c-api.crt",
    tls_key_path: "/etc/ssl/private/pgw-c-api.key",
    routes: [
      %{path: "/status", module: ApiEx.Api.StatusController,
actions: [:index]},
      %{path: "/sessions", module: ApiEx.Api.SessionController,
actions: [:index, :show]}
    ]
  }
```


API

API

```
# 本地测试
curl -k https://localhost:8443/api/status

# 生产环境
curl https://pgw-c-api.example.com:8443/api/status
```

API 接口

- /api/status
- /api/sessions
- /api/upf
- /api/diameter
- /api/charging
- /api/ip_pools
- /api/session_history
- /api/topology
- /api/ue_search
- POST /api/pcscf_monitor
- /api/upf_selection

API 文档 Swagger UI

<https://localhost:8443/api/docs>

TLS

TLS

```
# 生成私钥和证书
openssl req -x509 -newkey rsa:4096 -keyout priv/cert/omnitouch.pem \
-out priv/cert/omnitouch.crt -days 365 -nodes \
-subj "/CN=localhost"
```

TLS

CA 证书

- Let's Encrypt
- CA DigiCert GlobalSign
- 自建 CA

準備

1. 設定ファイルに **TLS** - 項目 `enable_tls: true`
2. 項目 **listen_ip** - 設定ファイルに IP 指定 0.0.0.0
3. 設定ファイル - 設定ファイルのパス
4. 設定ファイル - 設定ファイルのパス8443
5. 設定ファイル - 項目 4096 の RSA 鍵
6. 設定ファイル - 設定ファイルのパス

準備

設定ファイル API

```
# 設定ファイルのパス
netstat -tulpn | grep 8443

# 設定ファイルのパス
ls -la priv/cert/

# 設定ファイルのパス
tail -f /var/log/pgw_c/application.log
```

SSL/TLS 設定

- 設定ファイルのパス
- 設定ファイルのパス
- 設定ファイルのパス `openssl x509 -in cert.crt -text -noout`
- 設定ファイルのパス PGW-C 設定ファイル

設定ファイル OAM API 設定ファイル

□□□□

□□□□□□

```
# config/runtime.exs
import Config

# □□□□
config :logger, level: :info

config :pgw_c,
  # □□□Prometheus□
  metrics: %{
    enabled: true,
    ip_address: "10.0.0.20", # □□□□
    port: 9090,
    registry_poll_period_ms: 5_000
  },

  # Diameter/Gx□PCRF □□□
  diameter: %{
    listen_ip: "0.0.0.0",
    host: "omnipgw.epc.mnc001.mcc001.3gppnetwork.org",
    realm: "epc.mnc001.mcc001.3gppnetwork.org",
    peer_list: [
      %{
        host: "pcrf-primary.epc.mnc001.mcc001.3gppnetwork.org",
        realm: "epc.mnc001.mcc001.3gppnetwork.org",
        ip: "10.0.1.30",
        initiate_connection: true
      },
      %{
        host: "pcrf-backup.epc.mnc001.mcc001.3gppnetwork.org",
        realm: "epc.mnc001.mcc001.3gppnetwork.org",
        ip: "10.0.2.30",
        initiate_connection: true
      }
    ]
  },

  # S5/S8□SGW-C □□□
  s5s8: %{
```

```

    local_ipv4_address: "10.0.0.20"
  },

  # Sxb/PFCP PGW-U
  sxb: %{
    local_ip_address: "10.0.0.20"
  },

  # UPF
  upf_selection: %{
    rules: [
      %{
        name: "internet",
        priority: 1,
        match_field: :apn,
        match_regex: ~r/^internet/,
        upf_pool: [
          %{remote_ip_address: "10.0.0.21", remote_port: 8805,
weight: 100},
          %{remote_ip_address: "10.0.0.22", remote_port: 8805,
weight: 0} #
        ]
      }
    ],
    fallback_pool: [
      %{remote_ip_address: "10.0.0.21", remote_port: 8805, weight:
100}
    ]
  },

  # UE IP
  ue: %{
    subnet_map: %{
      "internet" => [
        "100.64.0.0/18" # 16,382 IP
      ],
      "ims" => [
        "100.64.64.0/22" # 1,022 IP
      ],
      "enterprise.corp" => [
        "10.100.0.0/16" # 65,534 IP
      ],
      default: [
        "100.64.127.0/24" # 254 IP
      ]
    }
  }
}

```

```
    ]
  }
},

# 配置
pco: %{
  primary_dns_server_address: "8.8.8.8",
  secondary_dns_server_address: "8.8.4.4",
  p_cscf_ipv4_address_list: ["10.0.0.50", "10.0.0.51"],
  ipv4_link_mtu_size: 1400
}
```

配置

配置

OmniPGW 配置

```
[info]  runtime.exs 配置
[info] 配置...
[info] 配置
[info]  OmniPGW...
```

配置

配置 **IP** 配置

```
[error] s5s8.local_ipv4_address 配置 IP 配置"10.0.0"
```

配置

```
[error] 配置sxb.local_ip_address
```

配置 **CIDR**

```
[error] ue.subnet_map 100.64.1.0/33"
```

100 **Diameter** 100

```
[error] Diameter 100000 FQDN10000 IP"10.0.0.20"
```

10000

1000000000

```
# 10000
mix compile

# 10000000
iex -S mix
iex> Application.get_env(:pgw_c, :metrics)
```

□□□□□□

□□

```
# config/dev.exs
import Config

config :logger, level: :debug

config :pgw_c,
  metrics: %{
    enabled: true,
    ip_address: "127.0.0.1",
    port: 42069,
    registry_poll_period_ms: 10_000
  },
  diameter: %{
    listen_ip: "0.0.0.0",
    host: "omnipgw-dev.local",
    realm: "local",
    peer_list: [
      %{
        host: "pcrf-dev.local",
        realm: "local",
        ip: "127.0.0.1",
        initiate_connection: true
      }
    ]
  },
  s5s8: %{
    local_ipv4_address: "127.0.0.10"
  },
  sxb: %{
    local_ip_address: "127.0.0.20"
  },
  upf_selection: %{
    fallback_pool: [
      %{remote_ip_address: "127.0.0.21", remote_port: 8805,
weight: 100}
    ]
  },
  ue: %{
```

```
    subnet_map: %{"internet" => ["100.64.1.0/24"],
    default: ["42.42.42.0/24"]}
  },
  pco: %{"primary_dns_server_address": "8.8.8.8",
  secondary_dns_server_address": "8.8.4.4",
  ipv4_link_mtu_size": 1400}
}
```


□□□□□□

```
# config/runtime.exs
import Config

config :pgw_c,
  metrics: %{
    enabled: System.get_env("METRICS_ENABLED", "true") == "true",
    ip_address: System.get_env("METRICS_IP", "0.0.0.0"),
    port: String.to_integer(System.get_env("METRICS_PORT",
"9090")),
    registry_poll_period_ms: 10_000
  },
  diameter: %{
    listen_ip: System.get_env("DIAMETER_LISTEN_IP", "0.0.0.0"),
    host: System.get_env("DIAMETER_HOST") || raise("DIAMETER_HOST
required"),
    realm: System.get_env("DIAMETER_REALM") ||
raise("DIAMETER_REALM required"),
    peer_list: [
      %{
        host: System.get_env("PCRF_HOST") || raise("PCRF_HOST
required"),
        realm: System.get_env("PCRF_REALM") ||
System.get_env("DIAMETER_REALM"),
        ip: System.get_env("PCRF_IP") || raise("PCRF_IP
required"),
        initiate_connection: true
      }
    ]
  }
# ... □□□□
```

□□□

```
export DIAMETER_HOST="omnipgw.epc.mnc001.mcc001.3gppnetwork.org"
export DIAMETER_REALM="epc.mnc001.mcc001.3gppnetwork.org"
export PCRF_HOST="pcrf.epc.mnc001.mcc001.3gppnetwork.org"
export PCRF_IP="10.0.0.30"

mix run --no-halt
```

□□□□

□□□□

- **PFCP** □□ - Sxb/PFCP □□□UPF □□□□□□
- **Diameter Gx** □□ - PCRF □□□□PCC □□□QoS □□
- **Diameter Gy** □□ - OCS □□□□□□□□□□□□
- **S5/S8** □□ - GTP-C v2 □□□SGW-C □□□4G/LTE□
- **Gn/Gp** □□ - GTP-C v1 □□□SGSN □□□2G/3G GGSN□

□□□□

- **UE IP** □□ - IP □□□□□□ APN □□□□DHCP
- **PCO** □□ - □□□□□□□DNS□P-CSCF□MTU
- **P-CSCF** □□ - P-CSCF □□□□□IMS □□□□

□□□□

- □□□□ - PDN □□□□□□□□□□
- □□□□ - Prometheus □□□□□□□□
- □□ **CDR** □□ - □□□□□□□□□□

□□□□□□

OmniPGW □□□□ - *□ Omnitouch □□□□□□*

PGW-C CDR

PGW-C

OmniPGW / Omnitouch

目次

1. 概要
2. CDR の形式
3. CDR のフィールド
4. CDR の例
5. 注意事項
6. 関連情報
7. CDR の取得方法
8. 関連する他の CDR
9. 問い合わせ先
10. 最終更新

概要

本 CDR は、PGW-C における CDR の形式とフィールドの定義を記述しています。

本 CDR は、SGW-C CDR と EPC CDR と関連しています。

注意事項

- 本 CSV は、PGW-C 向けです。
- 本 CDR は、PGW-C 向けです。
- 本 CDR は、PGW-C 向けです。

- 0000 - 0000000000000000
- 00 **3GPP** 00 - 00 3GPP TS 32.251 PS 00000 TS 32.298 CDR 000

00

00	00
0000	00000000 CDR
00	00000000
0000	0000000000
0000	0000000000
0000	0000000000

CDR 0000

0000000

<epoch_timestamp>

00:

1726598022

0000000000 Unix 00000000000000

0000

0000:

- PGW-C: /var/log/pgw_c/cdrs/

cdm_config cdr_directory config/runtime.exs

cdm

CDR

```
# CDR :
# : HH:MM:SS (unix_timestamp)
# : HH:MM:SS (unix_timestamp)
# : <gateway_name>
#
epoch,imsi,event,charging_id,msisdn,ue_imei,timezone_raw,plmn,tac,eci
```

cdm:

- - CDR Unix
 - - Unix
 - - PGW-C pgw_name
 - - CSV
-

CDR 表

表名

項目	データ型	長さ	説明
0	epoch	整数	UNIX 時刻
1	imsi	文字列	IMSI
2	event	整数	CDR イベント番号“default_bearer_start”
3	charging_id	整数	充電 ID
4	msisdn	文字列	MSISDN
5	ue_imei	文字列	UE IMEI
6	timezone_raw	文字列	UE 時刻領域
7	plmn	文字列	PLMN
8	tac	文字列	TAC
9	eci	文字列	E-UTRAN セル ID
10	sgw_ip	文字列	SGW-C S5/S8 インターフェイス IP
11	ue_ip	文字列	UE IP アドレスIPv4 IPv6
12	pgw_ip	文字列	PGW-C S5/S8 インターフェイス IP
13	apn	文字列	APN
14	qci	整数	QoS 値

項目	フィールド名	型	説明
15	octets_in	uint64_t	受信バイト数
16	octets_out	uint64_t	送信バイト数

CDR 形式

形式

CDR 形式

項目	フィールド名	型	説明
1	<type>_bearer_start	uint64_t	開始時刻
2	<type>_bearer_update	uint64_t	更新時刻
3	<type>_bearer_end	uint64_t	終了時刻/秒

形式:

- default - デフォルト PDN 形式
- dedicated - デフォルト PDN 形式

形式

- default_bearer_start - デフォルト
- default_bearer_update - デフォルト
- default_bearer_end - デフォルト
- dedicated_bearer_start - デフォルト
- dedicated_bearer_update - デフォルト
- dedicated_bearer_end - デフォルト

□□□□

□□ **CDR** □□

```
# □□ CDR □□:  
# □□□□□□: 18:53:42 (1726598022)  
# □□□□□□: 19:53:42 (1726601622)  
# □□□□: sgw-c-prod-01  
# epoch,imsi,event,charging_id,msisdn,ue_imei,timezone_raw,plmn,tac,e  
1726598022,310260123456789,default_bearer_start,12345,15551234567,123  
1726598322,310260123456789,default_bearer_update,12345,15551234567,12  
1726598622,310260123456789,default_bearer_update,12345,15551234567,12  
1726598922,310260123456789,default_bearer_end,12345,15551234567,12345
```

□□□□

CDR □□□□□□□□□□□□□□□□

-

00

0000

PGW-C CDR 000 config/runtime.exs 0000

00	0 0	00	000	000
pgw_name	0 0 0 0	PGW 00000 00000 CDR 00000 00	"omni- pgw01"	000000000 ID
cdr_file_duration	0 0	000000 000000	3600000	360000001 000
cdr_directory	0 0 0	CDR 000000 0	"/tmp/pgw_c"	/var/log/pgw_c/cdr
usage_report_interval	0 0	URR 0 000000 00 - PGW- U 000 000000 0	60000	6000001 000

0000

0000 (config/runtime.exs):

```

config :pgw_c,
  # CDR
  pgw_name: "omni-pgw01",
  cdr_file_duration: 3_600_000,          # 1
  cdr_directory: "/var/log/pgw_c/cdrs",

  # URR PGW-U
  usage_report_interval: 60_000          # 60

```

CDR:

```

config :pgw_c,
  pgw_name: "pgw-c-prod-01",
  cdr_file_duration: 3_600_000,          # 1
  cdr_directory: "/var/log/pgw_c/cdrs",
  usage_report_interval: 60_000          # 1

```

CDR:

```

config :pgw_c,
  pgw_name: "pgw-c-dev",
  cdr_file_duration: 300_000,            # 5
  cdr_directory: "/tmp/pgw_c_cdrs",
  usage_report_interval: 30_000          # 30

```

CDR:

```

config :pgw_c,
  pgw_name: "pgw-c-prod-heavy",
  cdr_file_duration: 1_800_000,          # 30
  cdr_directory: "/mnt/fast-storage/cdrs",
  usage_report_interval: 300_000         # 5

```

URR

PGW-C **PF**CP URR PGW-U URR PGW-U CDR

URR 配置:

1. `usage_report_interval` 配置 PFCP 上报间隔
2. PGW-C 配置 URR
3. PGW-U 配置
4. 配置 `bearer_update` CDR
5. 配置 `bearer_end` CDR

配置: `usage_report_interval: 60_000`

- PGW-U 60 秒上报
- 60 秒 CDR
- 配置

URR 定义:

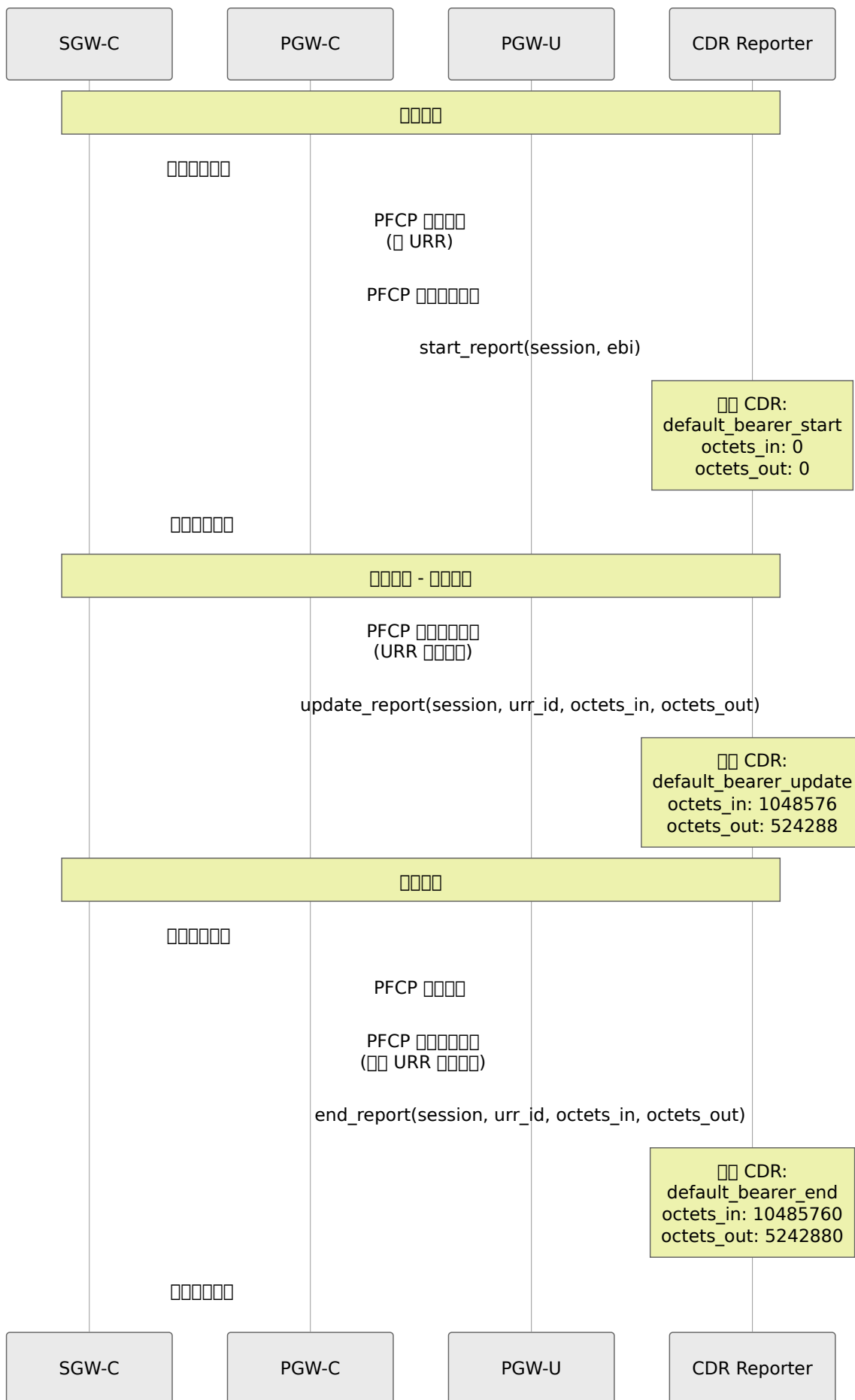
```
defmodule PGW_C.Session.Types.URR do
  typedstruct do
    field :urr_id, non_neg_integer()
    field :measurement_method, :duration | nil
    field :reporting_triggers, :time_threshold | nil
    field :time_threshold, non_neg_integer() | nil #
  end
end
```

配置 PFCP 配置 URR PFCP 配置 URR

CDR 配置

配置 CDR

PGW-C CDR 配置:



CDR 項目

1. 項目:

- 項目: 項目
- 項目: 項目
- **octets_in:** 0
- **octets_out:** 0

2. 項目:

- 項目: PGW-U 項目 PFCP 項目項目URR 項目
- 項目: 項目
- **octets_in:** 項目
- **octets_out:** 項目
- 項目: URR 項目 `usage_report_interval` 項目

3. 項目:

- 項目: PGW-U 項目 PFCP 項目
- 項目: 項目
- **octets_in:** 項目
- **octets_out:** 項目

項目項目

1. epoch (項目)

項目: Unix 項目

項目: CDR 項目

項目:

1726598022 → 2025-09-17 18:53:42 UTC

2. imsi (□□□□)

00: 000000 15 0000

□□: MCCMNC + MSIN

□□: □□□□□□□□□□

Diagram illustrating the mapping of a 9-digit number to three categories:

- MCC** (3 digits): 310
- MNC** (2 digits): 260
- MSIN** (4 digits): 123456789

A horizontal line is divided into three segments by two vertical tick marks. Below the first segment is the label 'MCC', below the second is 'MNC', and below the third is 'MSIN'.

(310) (260) (123456789)

UE 2020

3. event (CDR □□□□)

□□ : □□□

□□: <bearer_type>_bearer_<event>

- default_bearer_start
- default_bearer_update
- default_bearer_end
- dedicated_bearer_start
- dedicated_bearer_update
- dedicated_bearer_end

□□:

- EBI EPS ID LBI ID default
- EBI LBI dedicated

EBI LBI

4. charging_id ()

: 32

:

:

12345

: PGW-C

:

- SGW PGW
- Diameter Gy/Gz
-

5. msisdn ()

: E.164

: ISDN

: +

:

15551234567



CC (1)

(5551234567)

UE HSS MME

6. ue_imei (IMEI)

IMEI: 15551234567

IMEI: TAC (8) + SNR (6) + Spare (1)

IMEI: 15551234567

IMEI:

123456789012345



TAC

SNR

S

UE MME

7. timezone_raw (UE Timezone)

Timezone: UTC+08:00

Timezone: UE Timezone

Timezone: CSV Timezone

Timezone: Timezone

Timezone:

, ()

8. plmn ()

:

:

:

MCC: 505, MNC: 57

↓

"50557"

↓

: "055570"

↓

: $0x055570 = 349552$

:

349552 → MCC: 505, MNC: 57

: MME UE

:

9. tac ()

: 16

: UE

: 0 - 65535

:

1234

UE: UE MME

UE:

-
-
- TAI

10. eci (E-UTRAN)

UE: 28

UE: E-UTRAN UE

UE: eNodeB ID20 + ID8

UE: 0 - 268,435,455

UE:

5678

UE: MME UE

UE:

-
-
-

11. sgw_ip (SGW IP)

UE: IPv4 IPv6

例: SGW-C の S5/S8 間 IP 接続 F-TEID

例: 接続 IPv4 接続 IPv6

例:

10.0.0.15	(IPv4)
2001:db8::15	(IPv6)

例: 接続 S5/S8

12. ue_ip (UE IP 接続)

例: 接続 IPv4|IPv6 接続

例: 接続 UE の PDN 接続 IP 接続

例: <ipv4>|<ipv6>

例:

172.16.1.100	(IPv4)
2001:db8::1	(IPv6)
172.16.1.100 2001:db8::1	(接続)

例: PGW-C の PDN 接続 PAA

例:

- IPv4: 接続 IPv4 接続
- IPv6: 接続 IPv6 接続
- 接続: 接続 PDN 接続

13. pgw_ip (PGW 接続 IP)

例: 接続 IPv4 の IPv6 接続

例: PGW-C の S5/S8 インターフェイス IP アドレス F-TEID

例: インターフェイス IPv4 アドレス / インターフェイス IPv6

例:

```
10.0.0.20          (IPv4)
2001:db8::20       (IPv6)
```

例: PGW-C の IP アドレス

14. apn (アクセスポイント名)

例: デフォルト 100

例: インターフェイス PDN アドレス

例: DNS アドレス

例:

```
internet
ims
mms
enterprise.corporate
```

例: MME の IP アドレス

例:

- インターフェイス
- インターフェイス
- インターフェイス IP アドレス

15. qci (QoS 参数)

范围: 000 8 000

范围: QoS 参数000000000000

范围: 1 - 9000000128-2540000000

范围 QCI 范围:

QCI	名称	类型	延迟	丢包率	说明
1	GBR	2	100 ms	10^-2	语音
2	GBR	4	150 ms	10^-3	语音
3	GBR	3	50 ms	10^-3	语音
4	GBR	5	300 ms	10^-6	视频
5	Non-GBR	1	100 ms	10^-6	IMS 语音
6	Non-GBR	6	300 ms	10^-6	视频
7	Non-GBR	7	100 ms	10^-3	视频
8	Non-GBR	8	300 ms	10^-6	视频
9	Non-GBR	9	300 ms	10^-6	语音

范围:

9 → 0000000000

范围: PGW-C 范围 QoS 范围

16. octets_in (字节)

格式: 字节 64 字节

格式: 网络地址 → UE 网络地址

格式: 网络地址

格式:

1048576 → 1 MB 字节

格式: PGW-U 网络地址 PFCP 网络地址 URR 网络地址

格式:

- 格式 update 网络地址
- 格式 end 网络地址
- 格式 start 网络地址 0
- 格式 URR 网络地址 usage_report_interval 网络地址

17. octets_out (字节)

格式: 字节 64 字节

格式: 网络地址 UE → 网络地址

格式: 网络地址

格式:

524288 → 512 KB 字节

格式: PGW-U 网络地址 PFCP 网络地址 URR 网络地址

格式:

- `update` `usage_report_interval`
 - `end`
 - `start` 0
 - URR `usage_report_interval`
-

11

1: 11

11:

1. 11
2. 5 11: 10 MB 5 MB 11
3. 11

CDR 11:

```
# 11 CDR 11:
# 11: 10:00:00 (1726570800)
# 11: 11:00:00 (1726574400)
# 11: pgw-c-01
# epoch,imsi,event,charging_id,msisdn,ue_imei,timezone_raw,plmn,tac,e
1726570800,310260111111111,default_bearer_start,10001,1555111111,11
1726571100,310260111111111,default_bearer_update,10001,1555111111,11
1726571400,310260111111111,default_bearer_end,10001,1555111111,1111
```

2: 11

11:

1. 11 IPv4 + IPv6
2. 11
3. 11

CDR 00:

```
1726570800,3102602222222222,default_bearer_start,10002,15552222222,222
1726571100,3102602222222222,default_bearer_update,10002,15552222222,22
1726571400,3102602222222222,default_bearer_update,10002,15552222222,22
1726571700,3102602222222222,default_bearer_update,10002,15552222222,22
1726572000,3102602222222222,default_bearer_end,10002,15552222222,22222
```

00 3: 00000000

000:

1. 00000000QCI 9
2. 000000000000QCI 6
3. 0000000000
4. 000000
5. 000000

CDR 00:

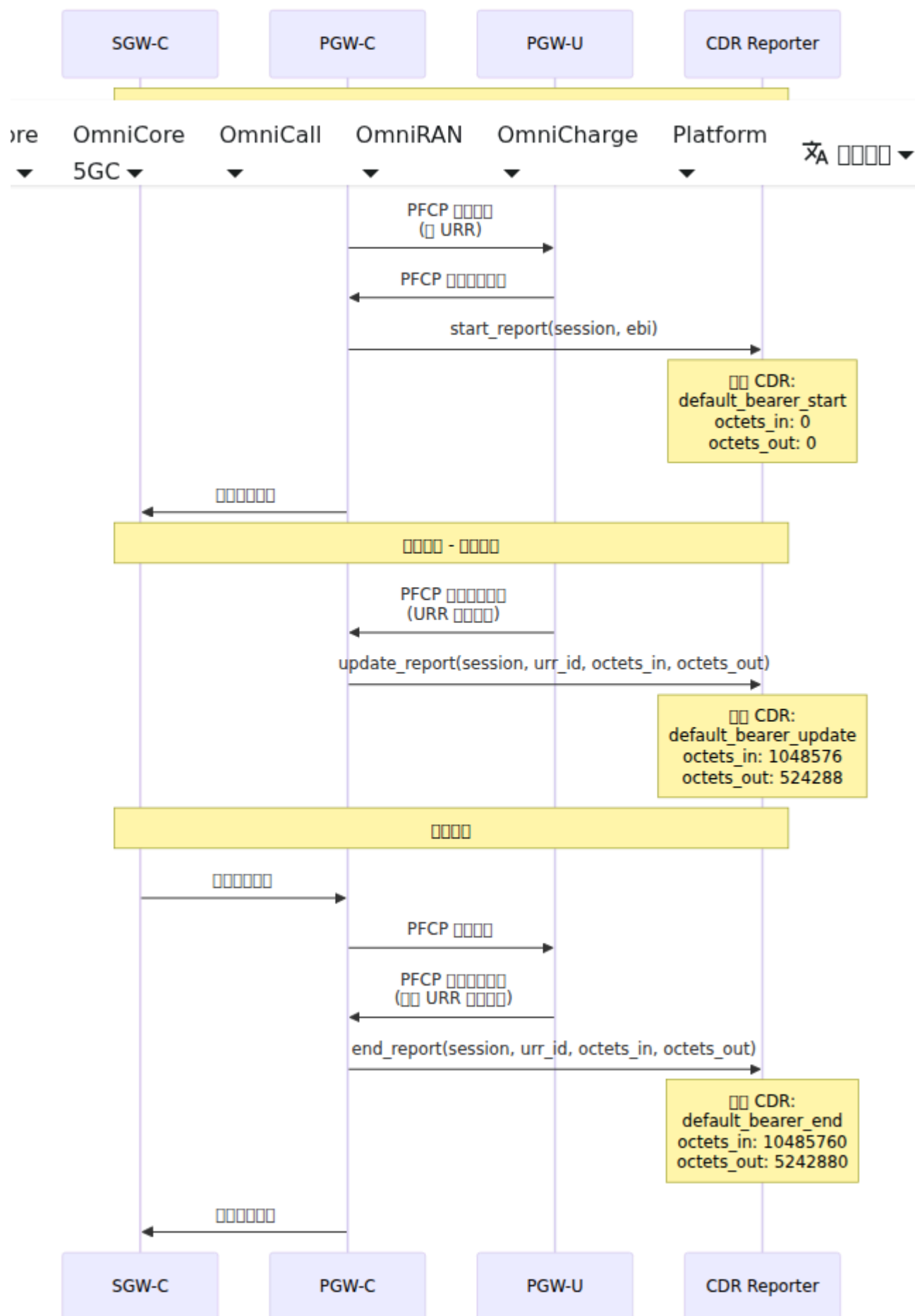
```
1726570800,3102603333333333,default_bearer_start,10003,15553333333,333
1726571100,3102603333333333,dedicated_bearer_start,10004,15553333333,3
1726571400,3102603333333333,default_bearer_update,10003,15553333333,33
1726571400,3102603333333333,dedicated_bearer_update,10004,15553333333,
1726571700,3102603333333333,dedicated_bearer_end,10004,15553333333,333
1726572000,3102603333333333,default_bearer_end,10003,15553333333,33333
```

00:

- 0000001000300000000010 MB 0004 MB 000
- 00000010004000000000200 MB 0002 MB 000
- 000 QCI 009 60000000 QoS 00

□□

CDR □□□□



CDR 配置

1. 配置守护进程:

```
# 配置 CDR 守护进程 (PGW-C)
inotifywait -m /var/log/pgw_c/cdrs/ -e close_write | while read
path action file; do
    # 处理 CDR 文件
    process_cdr "$path$file"
done
```

2. 配置日志:

```
# 配置日志守护进程
tail -F /var/log/pgw_c/cdrs/* | process_cdr_stream
```

配置

- **配置** - 配置 CDR 配置
- **PFCP 配置** - 配置 URR 在 PGW-U 上
- **配置** - CDR 配置
- **配置** - CDR 在 URR 上
- **Diameter Gx 配置** - CDR 在 QCI 配置
- **Diameter Gy 配置** - 配置

3GPP 配置

- TS 32.251 - 配置 PS 配置
- TS 29.274 - 3GPP 配置 EPS 配置 GTP-C 配置
- TS 29.244 - CP 在 UP 配置 PFCP - **URR** 配置
- TS 32.298 - CDR 配置

CDR 00 - PGW-C 00000000

Omnitouch 000000

0000: 1.0 0000: 2025-12-28

Diameter Gx

Policy and Charging Rules Function (PCRF)

Overview

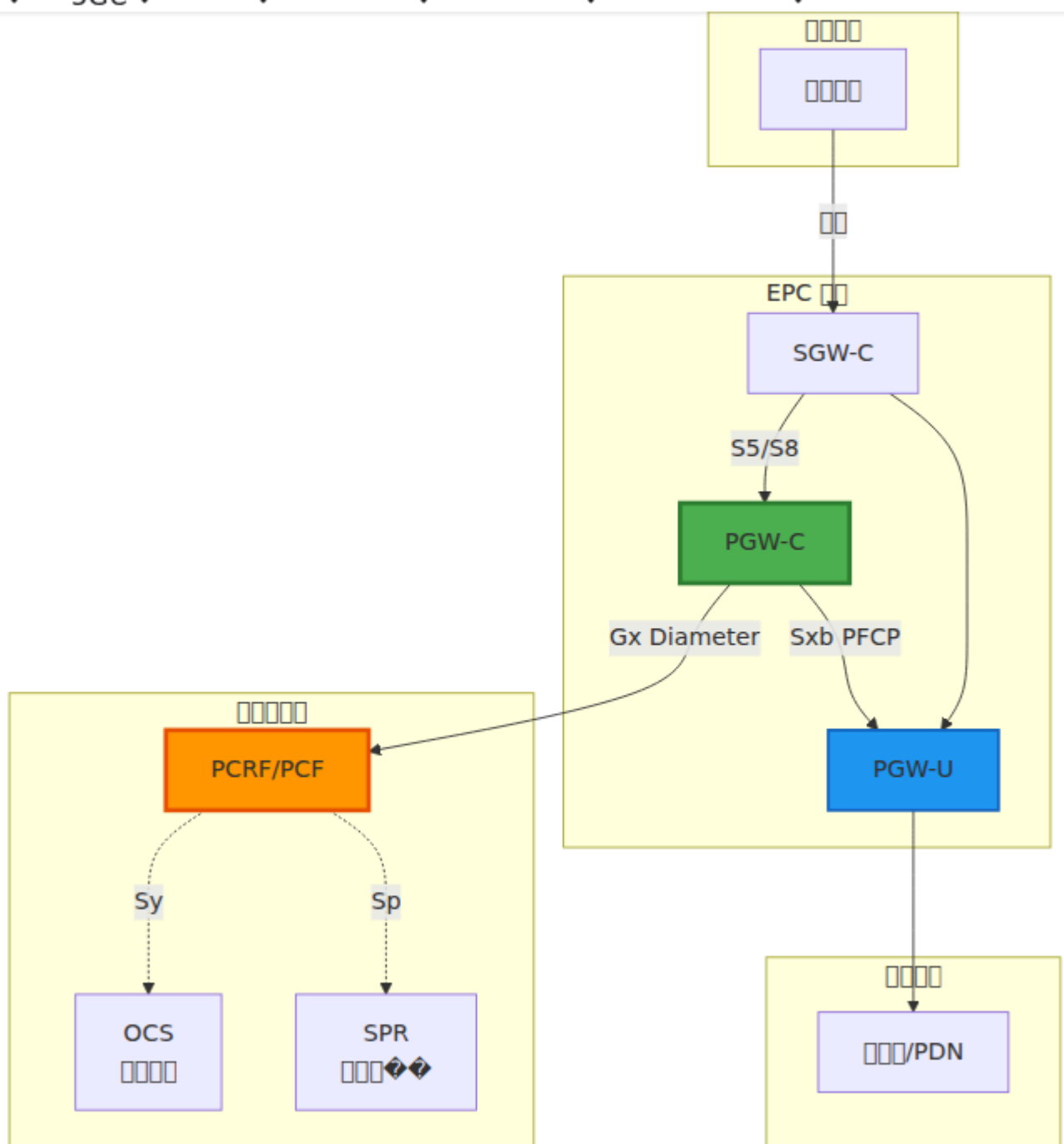
- Introduction
- Gx Interface
- Diameter Protocol
- Network Elements
- PCRF Functions
- PCRF Capabilities
- PCRF Configuration
- PCRF Deployment
- PCRF Security
- PCRF Troubleshooting

Details

Gx interface connects PGW-C to PCRF. PCRF manages QoS and charging for 5G networks.

- QoS Management - PCRF manages QoS parameters.
- Charging Management - PCRF manages charging parameters.
- Policy Management - PCRF manages policy parameters.
- Security Management - PCRF manages security parameters.

□□□□□□ Gx



1.

1.	2.
1.	PCRF 1. PCC 1.
QoS 1.	1. QoS 1.
1.	1./1.
1.	1./1.
1.	1.

Gx 1.

3GPP 1.

- 1. 3GPP TS 29.212
- 1. **Diameter** 1. **ID** 16777238 (Gx)
- 1. Diameter 1. (RFC 6733)

1.

1. UE PDN 1. **Gx** 1. **Session-ID** 1.

- 1. UE 1. (CCR-Initial)
- 1. 1. (CCR-Update) - 1. 1.
- 1. UE 1. (CCR-Termination)

Session ID

Session-ID: <Origin-Host>;<high32>;<low32>[;<optional>]

Example: omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org;1234567890;98765

Fields

- **Origin-Host**: PGW-C or Diameter peer
- **high32**: high 32 bits
- **low32**: low 32 bits

Diameter

Fields

Diameter peer or PGW-C

```

Diameter Header (20 bytes)
├─ Version (1 byte) = 1
├─ Message Length (3 bytes)
├─ Flags (1 byte)
│   ├── R: Request (1) / Answer (0)
│   ├── P: Proxiable
│   ├── E: Error
│   └─ T: Potentially retransmitted
├─ Command Code (3 bytes)
├─ Application ID (4 bytes) = 16777238 (Gx)
├─ Hop-by-Hop ID (4 bytes)
└─ End-to-End ID (4 bytes)

```

```

AVPs (00-00)
├─ AVP Header
│   ├── AVP Code
│   ├── Flags (V, M, P)
│   ├── AVP Length
│   └─ Vendor ID (optional)
└─ AVP Data

```

00 Diameter 00

AVP0000-0000

- Diameter 00000000
- 00000000
- 00000000 AVP

000

- 00/000
- CCR00000000/ CCA00000000

00000

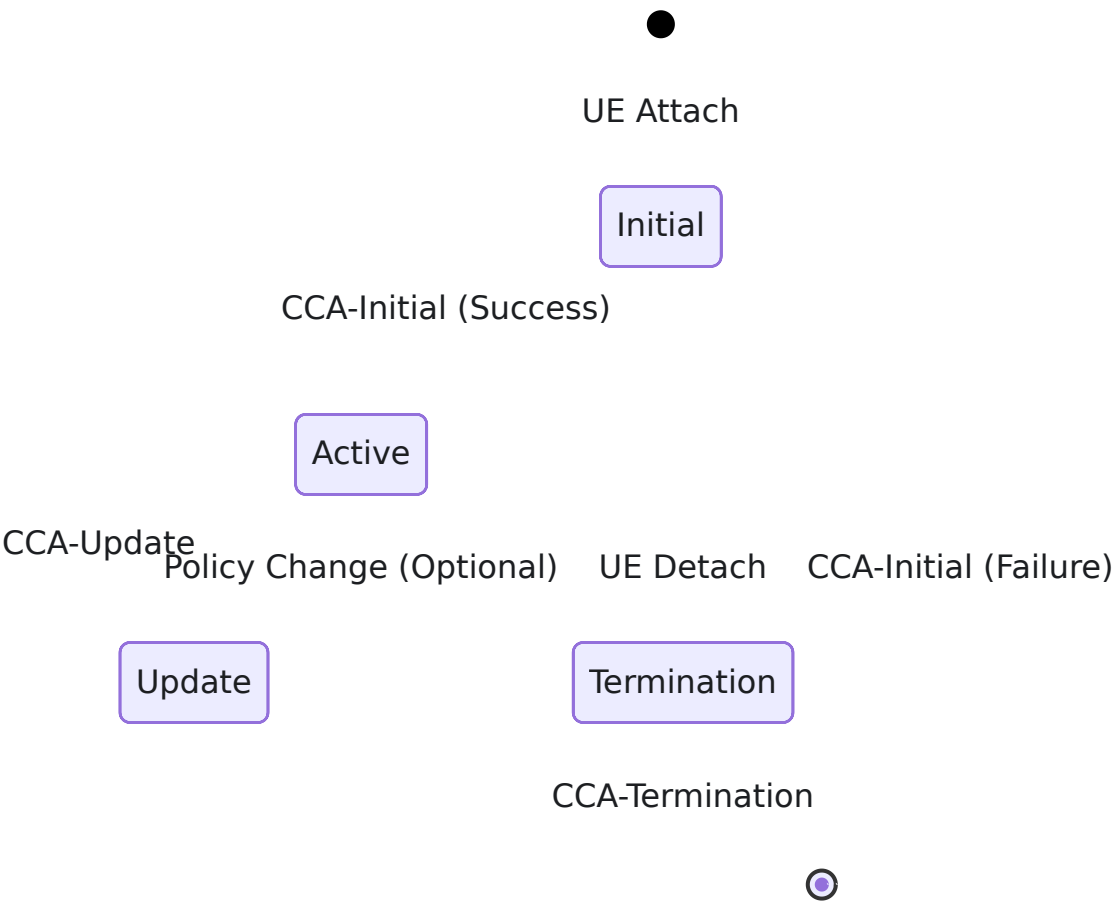
- 2001 - DIAMETER_SUCCESS
- 3xxx - 0000
- 4xxx - 0000

- 5xxx - 5G Core

5G Core

PGW-C uses **Diameter** protocol (RFC 4006) for Gx

5G Core



CCR-Initial - 5G Core

5G Core UE PDN

5G Core

- 5G Core
- PCRF 5G Core

- QoS

PGW-C **AVP**

AVP 이름	AVP 번호	타입	비고
Session-Id	263	UTF8String	Gx 인터페이스
Auth-Application-Id	258	Unsigned32	16777238 (Gx)
Origin-Host	264	DiamIdent	PGW-C Diameter 호스트
Origin-Realm	296	DiamIdent	PGW-C Diameter 도메인
Destination-Realm	283	DiamIdent	PCRF 도메인
CC-Request-Type	416	Enumerated	1 = INITIAL_REQUEST
CC-Request-Number	415	Unsigned32	순서번호 0 포함
Subscription-Id	443	Grouped	UE 식별자 (IMSI/MSISDN)
Called-Station-Id	30	UTF8String	APN
Framed-IP-Address	8	OctetString	UE IPv4 주소 (IPv4 및 IPv4v6 PDNs)
Framed-IPv6-Prefix	97	OctetString	UE IPv6 주소 (IPv6 PDNs)
3GPP-User-Location-Info	22	OctetString	UE 위치 정보 (TS 29.061)
User-Equipment-Info	458	Grouped	UE 정보 (IMEISV) 포함
IP-CAN-Type	1027	Enumerated	5 = 3GPP-EPS
RAT-Type	1032	Enumerated	GTP RAT 식별자 (1004 = EUTRAN)

AVP 名称	AVP 值	格式	描述
QoS-Information	1016	Grouped	QoS (APN-AMBR UL/DL)
Default-EPS-Bearer-QoS	1049	Grouped	QCI/ARP (QCI 5)
Network-Request-Support	1024	Enumerated	网络请求支持
Supported-Features	628	Grouped	Gx 特性 (Rel-8/9/10 Gx)

AVP 值 UE 请求 Framer-IP-Address IPv4 IPv4v6 PDN IPv4 Framer-IPv6-Prefix IPv6 PDN IPv4v6 IPv6 Gx 特性 IPv6 / 特性

CCR-I 请求

```
CCR (Command Code: 272, Request)
├─ Session-Id: "pgw_c.example.com;123;456"
├─ Auth-Application-Id: 16777238
├─ Origin-Host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org"
├─ Origin-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ Destination-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ CC-Request-Type: INITIAL_REQUEST (1)
├─ CC-Request-Number: 0
├─ Subscription-Id (Grouped)
│   └─ Subscription-Id-Type: END_USER_IMSI (1)
│       └─ Subscription-Id-Data: "310260123456789"
├─ Called-Station-Id: "internet"
├─ Framed-IP-Address: 100.64.1.42
├─ IP-CAN-Type: 3GPP-EPS (5)
├─ RAT-Type: EUTRAN (1004)
├─ QoS-Information (Grouped)
│   └─ APN-Aggregate-Max-Bitrates-UL: 100000000 (100 Mbps)
│       └─ APN-Aggregate-Max-Bitrates-DL: 50000000 (50 Mbps)
├─ Network-Request-Support: 1
└─ Supported-Features: [...]
```

CCA-Initial

PCRF CCR-I

-
- PCC
- QoS

PGW-C AVP

AVP	AVP	
Result-Code	268	(2001)
Experimental-Result	297	
QoS-Information	1016	QoS ()
Charging-Rule-Install	1001	PCC
Charging-Rule-Definition	1003	
Default-EPS-Bearer-QoS	1049	QoS

```

CCA (Command Code: 272, Answer)
├─ Session-Id: "pgw_c.example.com;123;456"
├─ Result-Code: DIAMETER_SUCCESS (2001)
├─ Origin-Host: "pcrf.example.com"
├─ Origin-Realm: "example.com"
├─ Auth-Application-Id: 16777238
├─ CC-Request-Type: INITIAL_REQUEST (1)
├─ CC-Request-Number: 0
├─ QoS-Information (Grouped)
│   ├─ APN-Aggregate-Max-Bitrate-UL: 50000000 (50 Mbps - reduced)
│   └─ APN-Aggregate-Max-Bitrate-DL: 100000000 (100 Mbps -
increased)
├─ Charging-Rule-Install (Grouped)
│   ├─ Charging-Rule-Name: "default_internet_rule"
│   └─ Charging-Rule-Name: "video_streaming_rule"
└─ Charging-Rule-Definition (Grouped)
    ├─ Charging-Rule-Name: "default_internet_rule"
    ├─ QoS-Information: {...}
    └─ Precedence: 1000

```

CCR-Termination -

UE PDN

- PCRF
- /

CCR-I

- CC-Request-Type: TERMINATION_REQUEST (3)
-
- AVP

CCR-T


```
CCR (Command Code: 272, Request)
├─ Session-Id: "pgw_c.example.com;123;456"
├─ Auth-Application-Id: 16777238
├─ Origin-Host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org"
├─ Origin-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ Destination-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ CC-Request-Type: TERMINATION_REQUEST (3)
├─ CC-Request-Number: 1
└─ Termination-Cause: DIAMETER_LOGOUT (1)
```

CCA-Termination

PCRF CCR-T

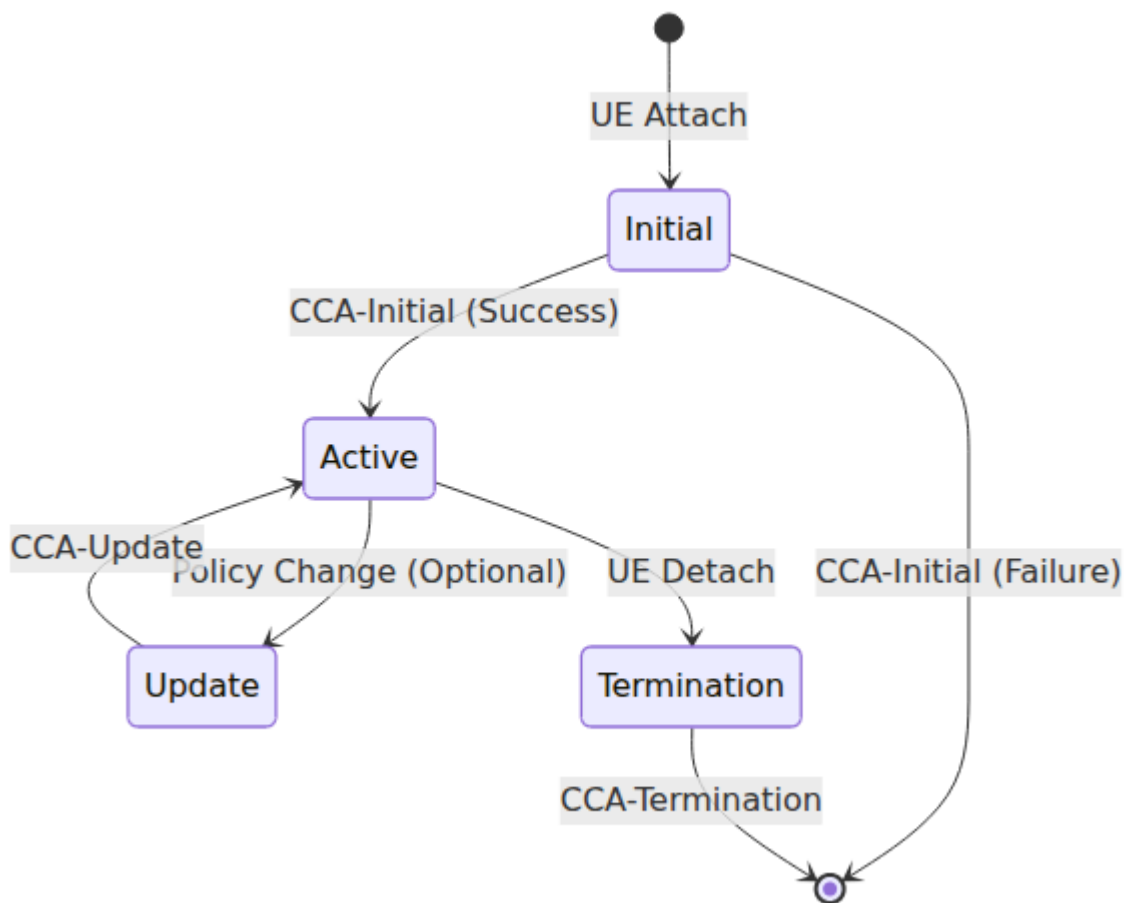
-
-

CCA-T

```
CCA (Command Code: 272, Answer)
├─ Session-Id: "pgw_c.example.com;123;456"
├─ Result-Code: DIAMETER_SUCCESS (2001)
├─ Origin-Host: "pcrf.example.com"
├─ Origin-Realm: "example.com"
├─ Auth-Application-Id: 16777238
├─ CC-Request-Type: TERMINATION_REQUEST (3)
└─ CC-Request-Number: 1
```

PCC

PCC



□□□□

1. □□□□

- □□□□□□□□
- □□□ "video_streaming_rule"

2. □□□□

- □□□□ = □□□□□
- □□□0-65535
- □□□□□□□□□□

3. □□□□□TFT - □□□□□□□

- □□□□□□□□□□□□
- □□□
 - IP 5 □□□□□□□□/□□ IP□□/□□□□

- "permit out ip from any to 8.8.8.8 80"

4. QoS

- **QCI** **QoS** 1-9 128-254
 - QCI 1
 - QCI 5 IMS
 - QCI 9
- **ARP**
- **MBR/GBR**

5.

- OCS - Diameter Gy
-
- OCS Diameter Gy CDR - CDR

6.

-
-

PCRF

1.

```
Charging-Rule-Install (Grouped)
├─ Charging-Rule-Name: "gold_subscriber_internet"
└─ Charging-Rule-Name: "video_qos_boost"
```

2.

```
Charging-Rule-Definition (Grouped)
├─ Charging-Rule-Name: "dynamic_rule_123"
├─ Precedence: 100
├─ Flow-Information (Grouped)
│   ├─ Flow-Description: "permit out ip from any to 192.0.2.0/24"
│   └─ Flow-Direction: DOWNLINK
├─ QoS-Information (Grouped)
│   ├─ QoS-Class-Identifier: 5
│   ├─ Max-Requested-Bandwidth-UL: 100000000
│   └─ Max-Requested-Bandwidth-DL: 50000000
└─ Rating-Group: 1000
```

QoS 和 AVP

APN-AMBR 和 GBR

APN 和 GBR 的 QoS

```
QoS-Information (Grouped)
├─ APN-Aggregate-Max-Bitrate-UL: 100000000 # 100 Mbps
└─ APN-Aggregate-Max-Bitrate-DL: 200000000 # 200 Mbps
```

PGW-C 和 UE

- APN 和 AMBR
- PGW-U 和 QER

PGW-C 和 UE

PGW-C 和 UE 之间的交互 (RAT) 和 PLMN 的 UE 位置信息
CCR-Update 和 3GPP-User-Location-Info AVP

PGW-C 和 PCRF

PGW-C 和 PCRF 之间的交互 — 3GPP TS 29.212 §4.5.7 / §5.3.7 中的 PCRF 和 CCA 和 RAR 事件触发 AVP 和 PGW-C 之间的 Gx 接口

CCRF-Update

- 1. Event-Trigger
- 2. Gx

PCRF Event-Trigger PGW-C — GTP-C PCRF PCRF

PGW-C PCRF Event-Trigger AVP — CCA-Initial CCA-Update PCRF RAR Event-Trigger Event-Trigger AVP PCRF RAR

Event-Trigger		...	AVP
RAT_CHANGE	2	RAT-Type EUTRAN → UTRAN	RAT-Type
PLMN_CHANGE	4	MCC/MNC	3GPP-SGSN-MCC-MNC
USER_LOCATION_CHANGE	13		3GPP-User-Location-Info
UE_TIME_ZONE_CHANGE	25	UE	3GPP-MS-TimeZone
TAI_CHANGE	26		3GPP-User-Location-Info
ECGI_CHANGE	27	E-UTRAN	3GPP-User-Location-Info

USER_LOCATION_CHANGE TAI_CHANGE ECGI_CHANGE CCR-Update Event-Trigger AVP

□ □ □ □

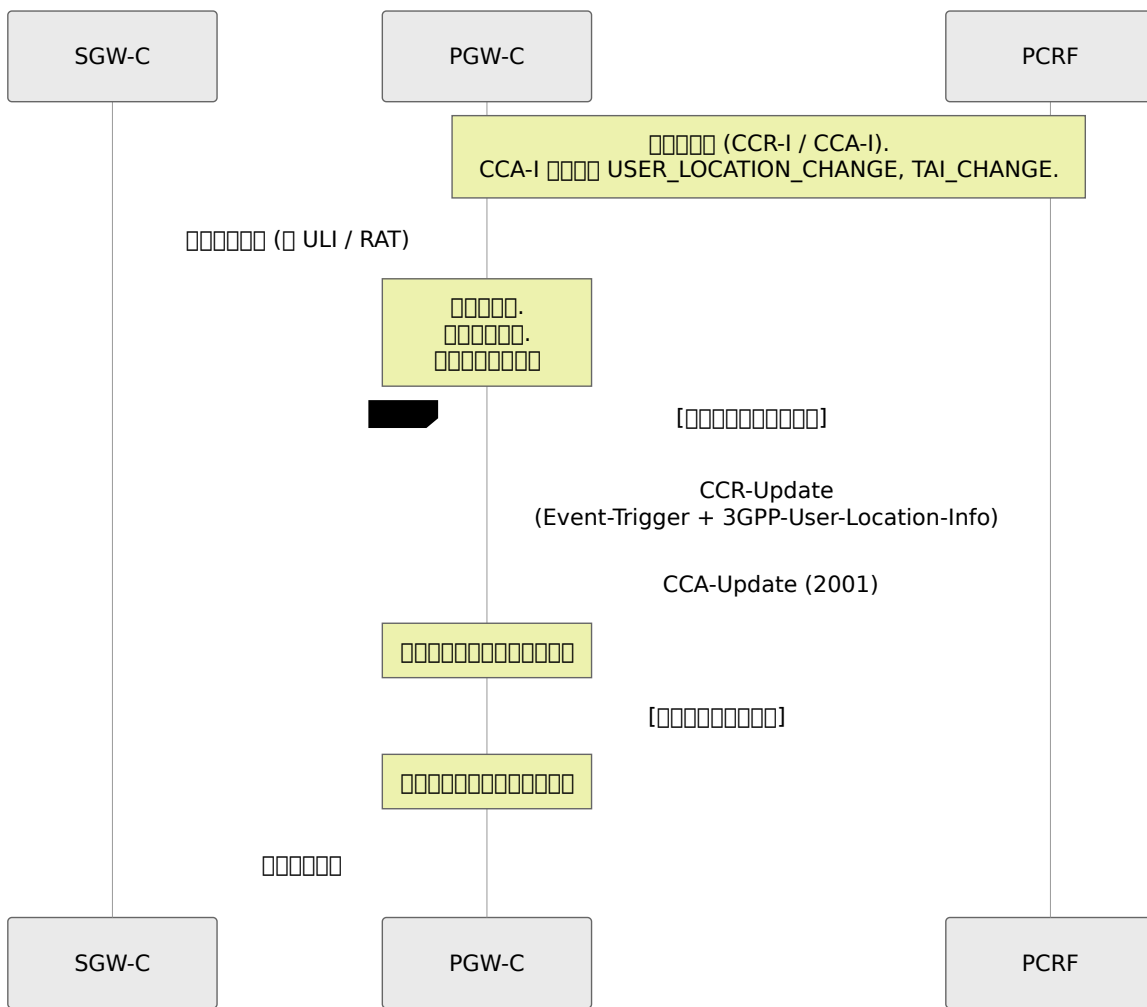
- **CCR-Initial** 3GPP-User-Location-Info PCRF
- **GTP-C** S5/S8 RAT-Type UE PGW-C
- IE

3GPP-User-Location-Info ☐

AVP 3GPP TS 29.061 §16.4.7.2 PLMN 3GPP TS 24.008 §10.5.1.3

項目	値	説明
CGI	0	PLMN + LAC + CI (2G)
SAI	1	PLMN + LAC + SAC (3G)
RAI	2	PLMN + LAC + RAC
TAI	128	PLMN + TAC (LTE)
ECGI	129	PLMN + ECI (LTE)
TAI + ECGI	130	PLMN + TAC + PLMN + ECI (LTE+3G)

0000



PCRF → GTP-C → CCR-Update → PCRF → PGW-C → SGW-C —

API

```

API (GET /api/sessions | GET /api/sessions/:imsi) | JSON |
| state | PCRF |
state.gx.requested_event_triggers | Gx | session_id |
cc_request_number | state.uli |

```

API

PATCH /api/sessions/:imsi PCRF
GET — gx

```
PATCH /api/sessions/3102600000000001
```

```
Content-Type: application/json
```

```
{ "gx": { "requested_event_triggers": [13, 26, 27] } }
```

-  Gx  gx 
-  ([]) 
-  state 

UPF  APN-AMBR  QoS  QER  QCI / MBR / GBR  QER  PFPCP 

```
PATCH /api/sessions/3102600000000001
```

```
Content-Type: application/json
```

```
{ "ambr": { "uplink": 200000, "downlink": 200000 } }
```

```
PATCH /api/sessions/3102600000000001
```

```
Content-Type: application/json
```

```
{ "default_bearer_qos": { "qci": 5, "mbr_ul": 300000, "mbr_dl": 250000 } }
```

 default_bearer_qos  MBR/GBR  GBR  QER  APN-AMBR  MBR

 PGW-C  PCRF  PCRF  CCA  RAR  PCRF

□□

□□ **Gx** □□

□□ config/runtime.exs□

```
config :pgw_c,  
  diameter: %{  
    # □□ Diameter □□□ IP □□  
    listen_ip: "0.0.0.0",  
  
    # PGW-C □ Diameter □□ (Origin-Host)  
    host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org",  
  
    # PGW-C □ Diameter □□ (Origin-Realm)  
    realm: "epc.mnc999.mcc999.3gppnetwork.org",  
  
    # PCRF □□□□□  
    peer_list: [  
      %{  
        # PCRF Diameter □□  
        host: "pcrf.epc.mnc999.mcc999.3gppnetwork.org",  
  
        # PCRF □□□□□□ PGW-C □□□□□  
        realm: "epc.mnc999.mcc999.3gppnetwork.org",  
  
        # PCRF IP □□  
        ip: "10.0.0.30",  
  
        # PGW-C □□□□□ PCRF □□□  
        # true = PGW-C □□□ PCRF  
        # false = □□ PCRF □□  
        initiate_connection: true  
      }  
    ]  
  }  
}
```

□□ **PCRF** □□□

□□□□□□□□□□

```

config :pgw_c,
  diameter: %{
    listen_ip: "0.0.0.0",
    host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org",
    realm: "epc.mnc999.mcc999.3gppnetwork.org",
    peer_list: [
      %{
        host: "pcrf-primary.example.com",
        realm: "epc.mnc999.mcc999.3gppnetwork.org",
        ip: "10.0.1.30",
        initiate_connection: true
      },
      %{
        host: "pcrf-backup.example.com",
        realm: "epc.mnc999.mcc999.3gppnetwork.org",
        ip: "10.0.2.30",
        initiate_connection: true
      }
    ]
  }
}

```

□□□□□

- Diameter □□□□□□□□□□
- □□□□□□□□□□
- □□□□□□□□□□□□□□

□□□□□□

Diameter □□□□□ **FQDN**□□□□□□□□□□

```

# □□ - FQDN □□
host: "pgw_c.epc.mnc999.mcc999.3gppnetwork.org"

# □□□ - □□□□□ Diameter □□
host: "pgw_c"
host: "10.0.0.20" # □□□ IP □□

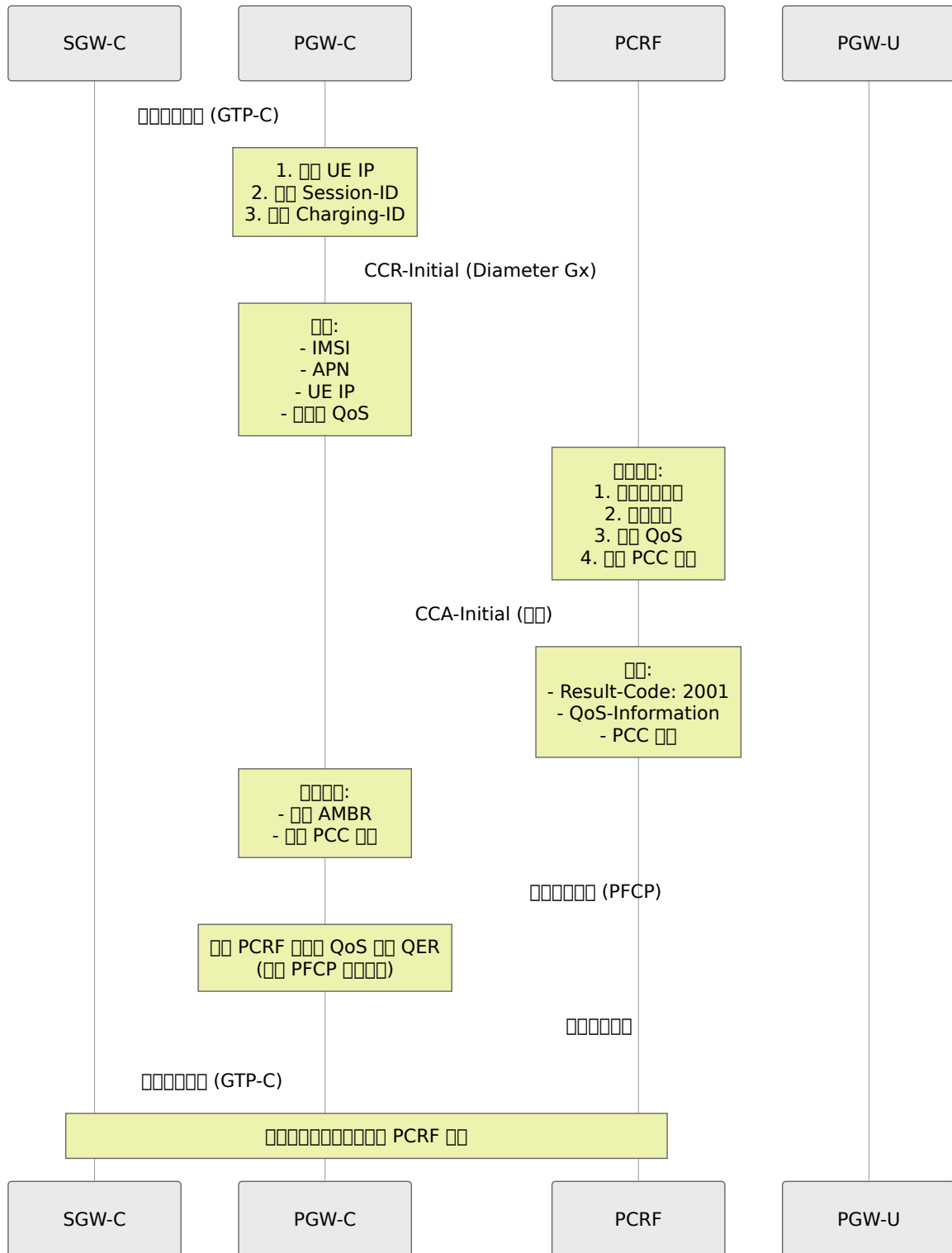
```

□□□□□

- `0000000000`
 - `3GPP PLMN 000000 epc.mncXXX.mccYYY.3gppnetwork.org`
-

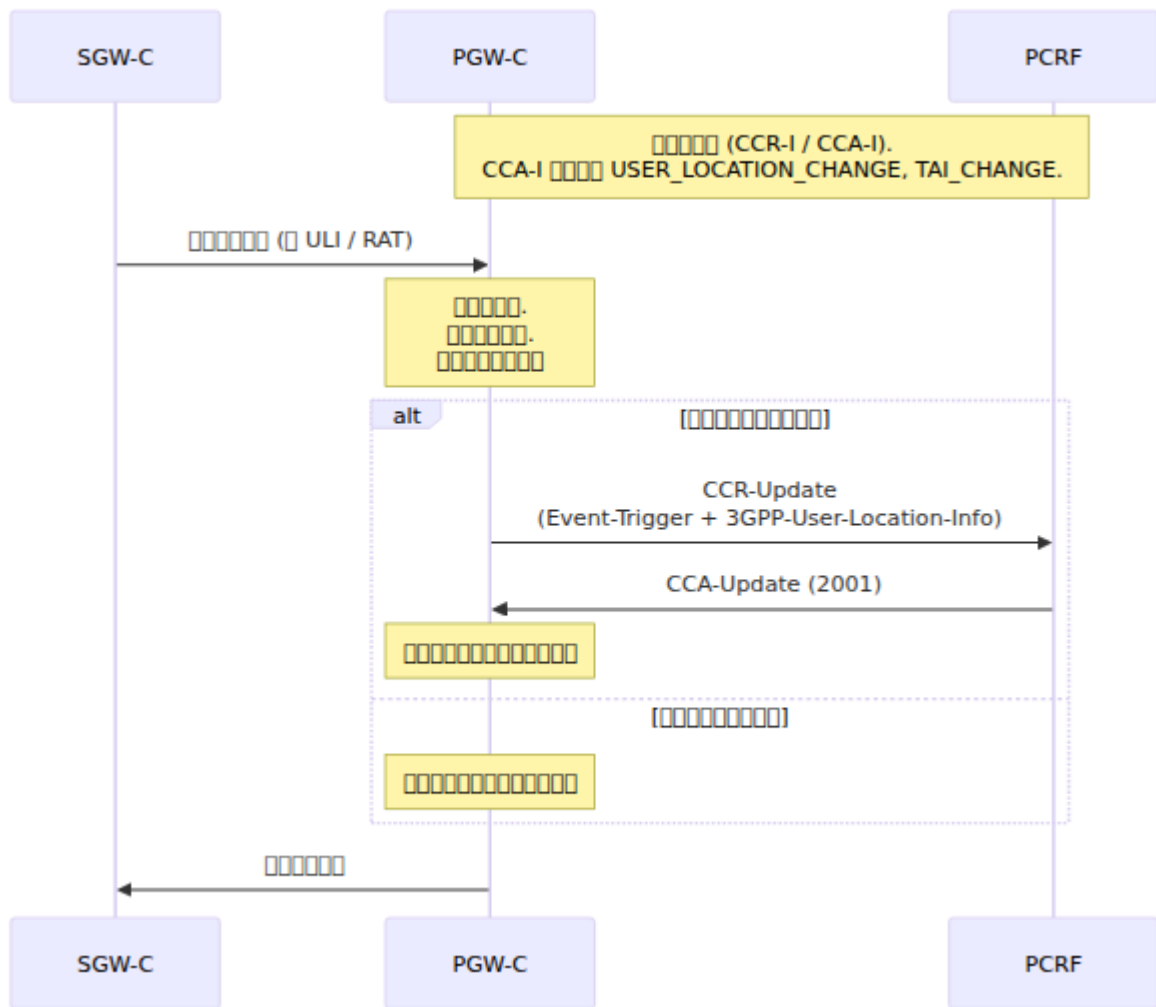
UE

Network

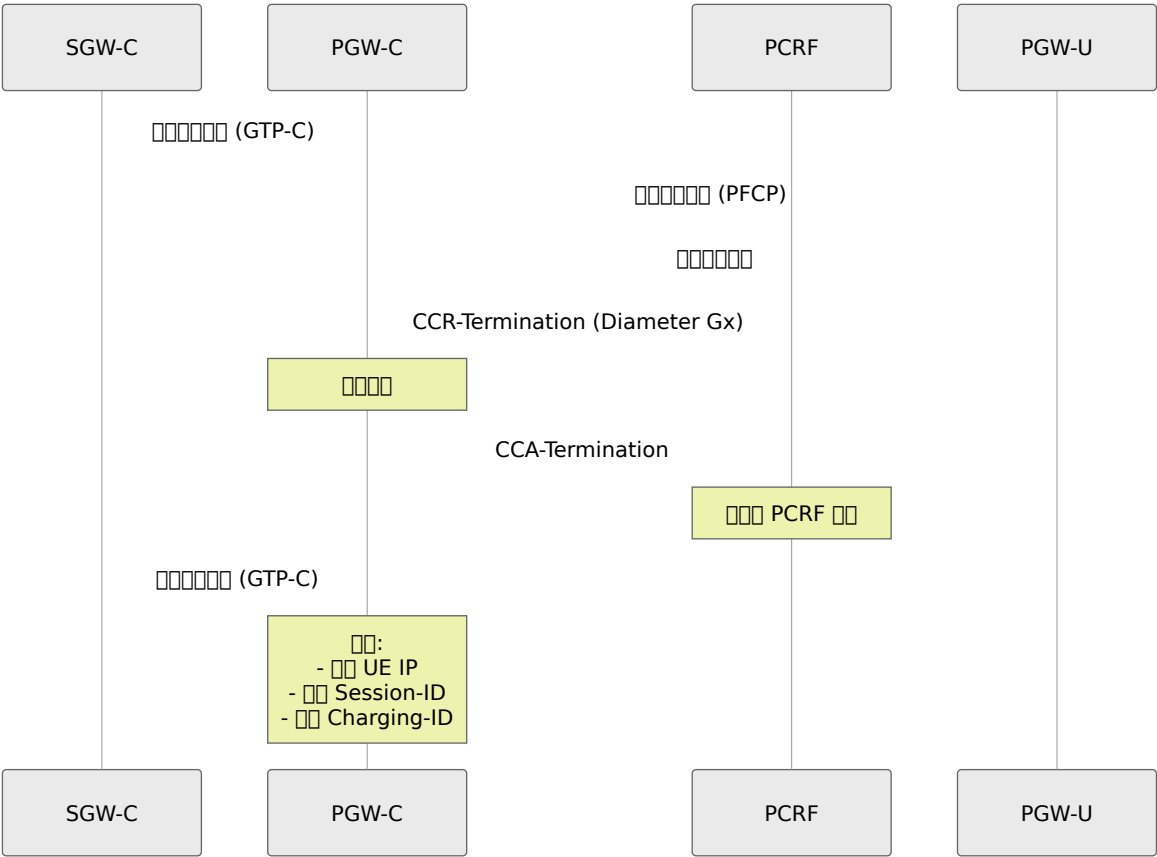


PCRF QoS QER QoS PFCP PGW-U PFCP QER

□□□□□□□□□□



???



PGW-C

CCA

PGW-C CCA Diameter

2001

Code	Message	Result
2001	DIAMETER_SUCCESS	Success

5xxx

PGW-C	PGW-C	PGW-C
5002	DIAMETER_UNKNOWN_SESSION_ID	
5030	DIAMETER_USER_UNKNOWN	
5140	DIAMETER_ERROR_INITIAL_PARAMETERS	
5003	DIAMETER_AUTHORIZATION_REJECTED	

PGW-C (4xxx)

PGW-C	PGW-C	PGW-C
4001	DIAMETER_AUTHENTICATION_REJECTED	
4010	DIAMETER_TOO_BUSY	
4012	DIAMETER_UNABLE_TO_COMPLY	

PGW-C

PGW-C

Experimental-Result (Grouped)
└─ Vendor-Id: 10415 (3GPP)
└─ Experimental-Result-Code: <vendor-specific code>

PGW-C 3GPP

PGW-C	PGW-C	PGW-C
5065	IP_CAN_SESSION_NOT_AVAILABLE	PCRF
5143	INVALID_SERVICE_INFORMATION	

□ □ □ □

CCR-I

PCRF CCR-Initial

1. PGW-C 5
2. CCA
 - "CCR-Initial Session-ID: ..."
 - SGW-C
 -
3. SGW-C

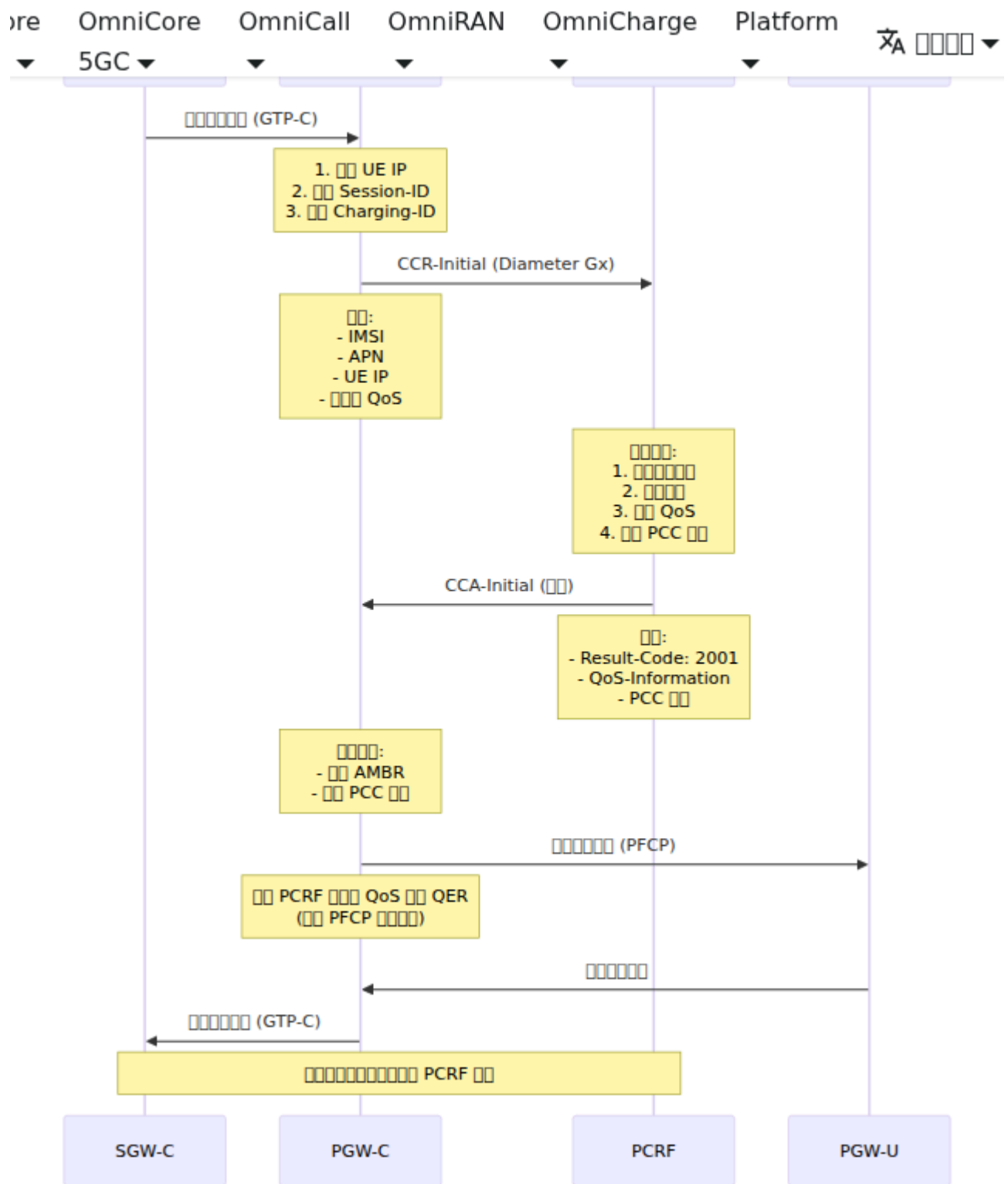
SGW-C

CCR-Initial PGW-C SGW-C

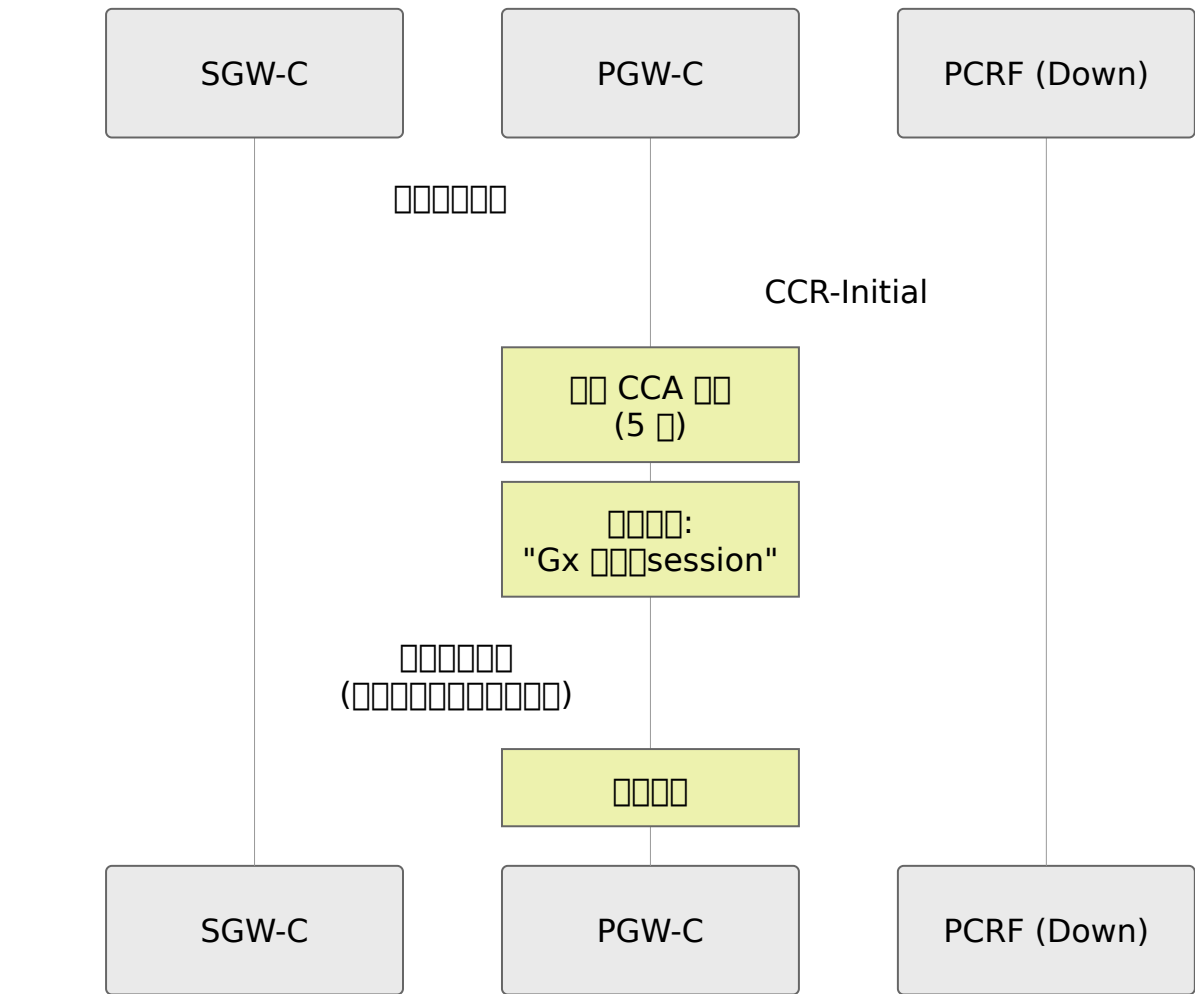
```
:remote_peer_not_responding[]
```

□ □ □ □

1 PCR



2. PCRF



SGW-C

PGW-C

1. Diameter Session

SGW-C

- "Diameter Session"
- CCR-Initial

PGW-C

- PCRF IP
- PCRF IP

- ping Diameter 3868
- Diameter 3868/3868

ping

```
# ping
ping <pcrf_ip>

# ping Diameter 3868 (TCP 3868)
telnet <pcrf_ip> 3868

# ping Diameter 3868
# ping 3868 FQDN 3868 IP
```

ping

```
config :pgw_c,
  diameter: %{
    # ping FQDN 3868 IP
    host: "pgw_c.epc.mnc999.mcc999.3gppnetwork.org",
    realm: "epc.mnc999.mcc999.3gppnetwork.org",
    peer_list: [
      %{
        host: "pcrf.epc.mnc999.mcc999.3gppnetwork.org",
        ip: "10.0.0.30"
      }
    ]
  }
}
```

2. CCR-Initial

ping

- ping 3868
- ping "CCR-Initial 3868"

ping

- PCRF 3868

- 0000
- PCRF 0000 Session-ID

00000

1. 00 PCRF 000000
2. 00 PCRF 00000000
3. 0000000 ping <pcrf_ip>
4. 0000000000000000

3. 000 PCRF 00

000

- CCA-Initial 0 Result-Code != 2001
- 00000000

0000000

0000	0000	0000
5030	IMSI 000000000	0 HSS/SPR 00000
5003	00000	000000
4010	PCRF 00	00000 PCRF 00

00000

```
# PGW-C 00000
[error] Diameter Gx 000Result-Code 5030 (DIAMETER_USER_UNKNOWN)
[error] IMSI 310260999999999 0 PCRF 00
```

4. QoS 000

000

- 000000 QoS 00

- ــ

ــــــــــــــــ

1. ــــــــ **CCA-Initial** ــــــــ

- ــــــــ QoS-Information AVP ــــــــ
- ــــــــ APN-Aggregate-Max-Bitrate-UL/DL ــــــــ

2. ــــــــ **PFCP** ــــــــــــــــ

- ــــــــ QER ــــــــــــــــ MBR ــــــــ
- ــــــــ PGW-U ــــــــــــــــ QER ــــــــ

3. ــــــــ **PCRF** ــــــــ

- ــــــــ PCRF ــــــــ
- ــ QoS

5. Diameter ــــــــ

ــــــــ

- Diameter ــــــــــــــــ PCRF
- ــــــــ“ــ”

ــــــــ

- ــ

ــــــــــــــــ

ــــــــــــــــ

```
# 配置
config :pgw_c,
  diameter: %{
    realm: "epc.mnc999.mcc999.3gppnetwork.org", # PGW-C 配置
    peer_list: [
      %{
        realm: "epc.mnc999.mcc999.3gppnetwork.org" # PCRF 配置
      }
    ]
  }
}
```

CCR-Initial 消息

```
Origin-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
Destination-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
```

接口 Gx 消息

配置

```

# Gx
rate(gx_inbound_messages_total{message_type="gx_CCA"}[5m])
rate(gx_outbound_messages_total{message_type="gx_CCR"}[5m])

# Gx
rate(gx_inbound_errors_total[5m])

# Gx
sum(rate(gx_outbound_responses_total{result_code_class="2xxx"}[5m])) /
sum(rate(gx_outbound_responses_total[5m])) * 100

# PCRF Gx
rate(gx_outbound_responses_total{result_code_class!="2xxx"}[5m])
by (diameter_host)

# Gx
session_id_registry_count

# Gx
histogram_quantile(0.95,
rate(gx_inbound_handling_duration_bucket[5m]))

```

Annotations

`gx_outbound_responses_total` PCRF Diameter

- `message_type` gx_RAA gx_CCA
- `result_code_class` 2xxx 3xxx 4xxx 5xxx
- `diameter_host` PCRF

Legend

- 2001** DIAMETER_SUCCESS -
- 3001** DIAMETER_COMMAND_UNSUPPORTED -
- 5012** DIAMETER_UNABLE_TO_COMPLY -
- 5030** DIAMETER_USER_UNKNOWN -

Annotations

```

# Gx 错误率
- alert: GxErrorRateHigh
  expr: rate(gx_inbound_errors_total[5m]) > 0.1
  for: 5m
  annotations:
    summary: "Gx 错误率"

# Gx 响应失败率
- alert: GxResponseFailureRate
  expr: |

sum(rate(gx_outbound_responses_total{result_code_class!="2xx"}
[5m])) /
  sum(rate(gx_outbound_responses_total[5m])) > 0.1
  for: 5m
  annotations:
    summary: "Gx 响应失败率"
    description: "10% Gx 响应失败率"

# PCRF 失败率
- alert: GxPCRFFailures
  expr:
rate(gx_outbound_responses_total{result_code_class=~"4xx|5xx"}
[5m]) by (diameter_host) > 0.05
  for: 3m
  annotations:
    summary: "PCRF {{ $labels.diameter_host }} 失败率"
    description: "PCRF 失败率"

# 会话拒绝率
- alert: GxSessionRejection
  expr: rate(gx_inbound_errors_total{result_code="5030"}[5m]) >
0.01
  for: 5m
  annotations:
    summary: "PCRF USER_UNKNOWN"

```

错误率

PCRF Diameter 失败率


```
# config/runtime.exs
config :logger, level: :debug

# 配置日志
iex> Logger.configure(level: :debug)
```

输出

- [debug] 应用 CCR-InitialSession-ID: ...
- [debug] 应用 CCA-Initial: Result-Code 2001
- [error] Diameter 应用: ...

Diameter 应用

Diameter Gx 应用通过 PCRF 应用 **OAM REST API** 应用 API 应用 HTTPS/TLS 应用
`https://<host>:8443` 应用 `/api` 应用 API 应用 Swagger UI 应用
`https://<host>:8443/api/docs` 应用

Diameter 应用

```
GET /api/diameter # 应用 + 应用
GET /api/diameter?status=disconnected # 应用
GET /api/diameter/<origin-host> # 应用 Origin-Host 应用
```

应用

```
curl -k https://localhost:8443/api/diameter
```

`GET /api/diameter` 应用 Diameter 应用 `?status=connected` 应用
`?status=disconnected` 应用 Origin-Host 应用 `GET`
`/api/diameter/<origin-host>` 应用

应用

項目	説明
host	送信元 Diameter ホスト (Origin-Host)
status	接続状態 / 原因 / 詳細
ip	送信元 IP アドレス
port	送信元ポート
transport	TCP または SCTP
connection_initiation	OmniPGW からの接続開始
realm	送信元 CEA の Diameter ホスト
product_name	送信元 CEA の製品名
application_ids	送信元 CEA の Diameter ホスト ID (Gx = 16777238)

送信元 PCRF ホストと Gx 接続を確立するための接続開始メッセージ

送信元

送信元 **PCRF** ホスト

```
curl -k https://localhost:8443/api/diameter
```

送信元 PCRF ホストは connection_initiation メッセージを送信し、application_ids を Gx16777238 と設定します。

送信元 **Gx** ホストは "PCRF ホスト" を送信元として送信します。

```
curl -k https://localhost:8443/api/diameter/<origin-host>
```

- status が 0000 → 送信元 PCRF ホストは TCP 3868

- `status` 0 (idle) → 1 (connected) PCRF 状態変更

00 **Diameter** 0000 PCRF 000000000000 0000

```
curl -k https://localhost:8443/api/diameter/<origin-host>
```

00 `realm` 00000000 `application_ids` 0000 Gx0000 `product_name` 00000000 PCRF
0000

00000000 00 PCRF 0000 `GET /api/diameter?status=disconnected` 0000 0000 0000
0 PCRF 00 00000000000000000000000000000000 `status` 0000 0000

00 **Diameter** 000000

- 00000000 0000 0 `realm` 00
- `application_ids` 0000 Gx0167772380
- `product_name` 00000 PCRF 0000

0000000000 00 `connection_initiation` 00000000000000000000000000000000
`initiate_connection: true` 00000000000000000000000000000000

00000000

OAM API 0000000000000000 Prometheus 000000

- 00 Gx 0000
- CCR/CCA 00000
- 00000

`GET /api/diameter` 00“000000000000”000000“00000000000000000000000000000000”

PCRF

PCRF 功能

- **PCRF** - Diameter 协议 PCRF 功能
- **PFCP** 协议 - 用于 PCC 策略 QER 和 QoS 控制
- **PCRF** - 策略控制功能
- **QoS** 策略 - 用于 QoS 控制

PCRF 接口

- **Diameter Gy** 接口 - 用于 PCC 策略控制
- **CDR** 接口 - 用于计费数据记录
- **PCO** 接口 - IMS 网络 P-CSCF 接口

PCRF 部署

- **PCRF** - Gx 接口 PCRF 部署
- **S5/S8** 接口 - 用于策略控制

PCRF 部署

3GPP (Gy/Ro OCS)

3GPP (OCS) 3GPP

3GPP

- 3GPP
- 3GPP 3GPP
- Gy/Ro 3GPP
- 3GPP
- 3GPP
- 3GPP
- 3GPP
- 3GPP
- 3GPP
- 3GPP
- 3GPP
- 3GPP

3GPP

Gy 3GPP IMS 3GPP Ro 3GPP PGW-C 3GPP 3GPP (OCS) 3GPP

- 3GPP - 3GPP
- 3GPP - 3GPP
- 3GPP - 3GPP
- 3GPP - 3GPP
- 3GPP - 3GPP

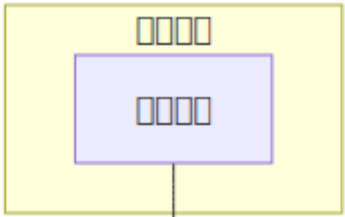
Table 1

Item	Parameter (Gy/Ro)	Parameter (Gz/Rf)
1	Parameter 1	Parameter 2
2	Parameter 3	Parameter 4
3	Parameter 5	Parameter 6
4	Parameter 7	Parameter 8
5	OCS (Parameter 9)	CGF/CDF (Parameter 10)
6	Parameter 11	Parameter 12
7	Parameter 13	Parameter 14
8	Parameter 15	Parameter 16

Table 2 CDR Table 1 Parameter 1

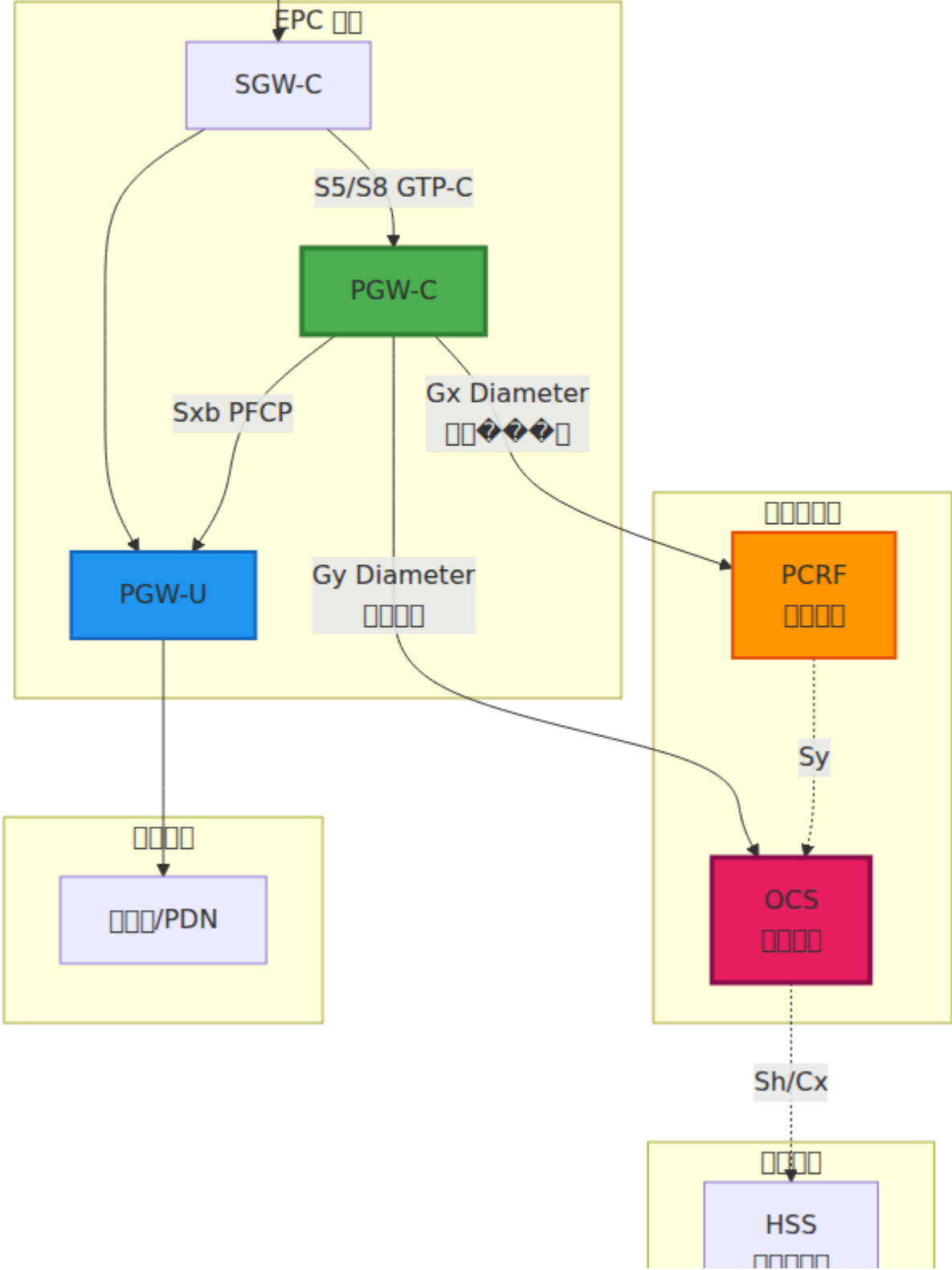
Table 3 Parameter 1 PDN Parameter 2

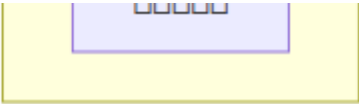
□□□□□ Gy



ore OmniCore OmniCall OmniRAN OmniCharge Platform 5A

▼ 5GC ▼ ▼ ▼ ▼ ▼

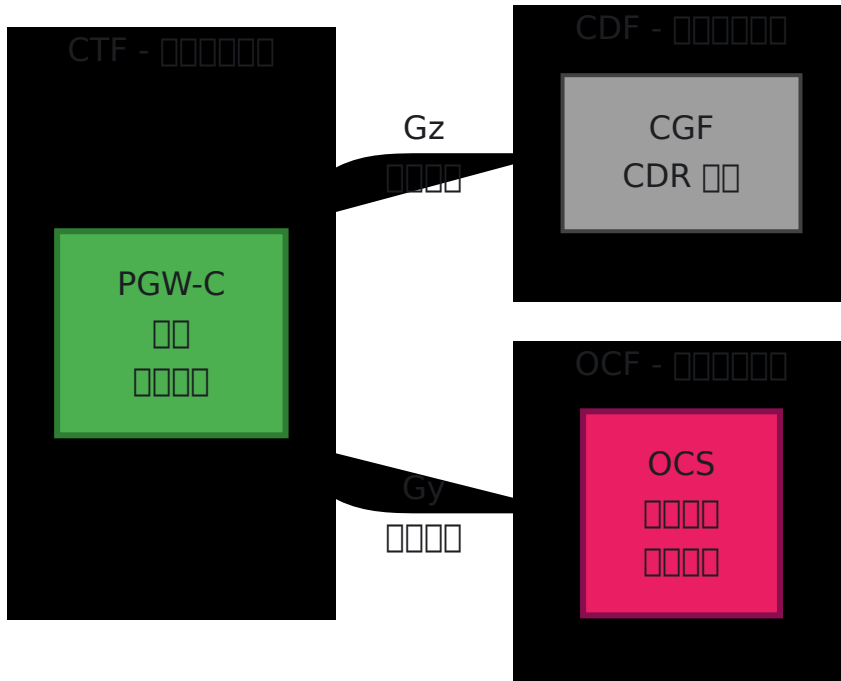




00000

00	00
0000	00000000 OCS 0000
0000	00000000000000000000
000000	000000
0000	00000000000000
0000	000000000000
0000	0000000000000000

3GPP



(CTF)

PGW-C 는 **CTF** ()

1. -
2. -
3. -
4. -
5. -

(OCF)

OCS 는 **OCF** ()

1. -
2. - MB
3. -

4. Gy/Ro - Diameter
5. Gy/Ro - Diameter

Gy/Ro

3GPP

- 3GPP TS 32.299 (Diameter)
- 3GPP TS 32.251 (PS)
- Diameter ID 4** (Gy/Ro - Diameter)
- RFC 4006 (Diameter)

UE PDN Session-ID Gy/Ro

- CCR-Initial
- CCR-Update
- CCR-Termination

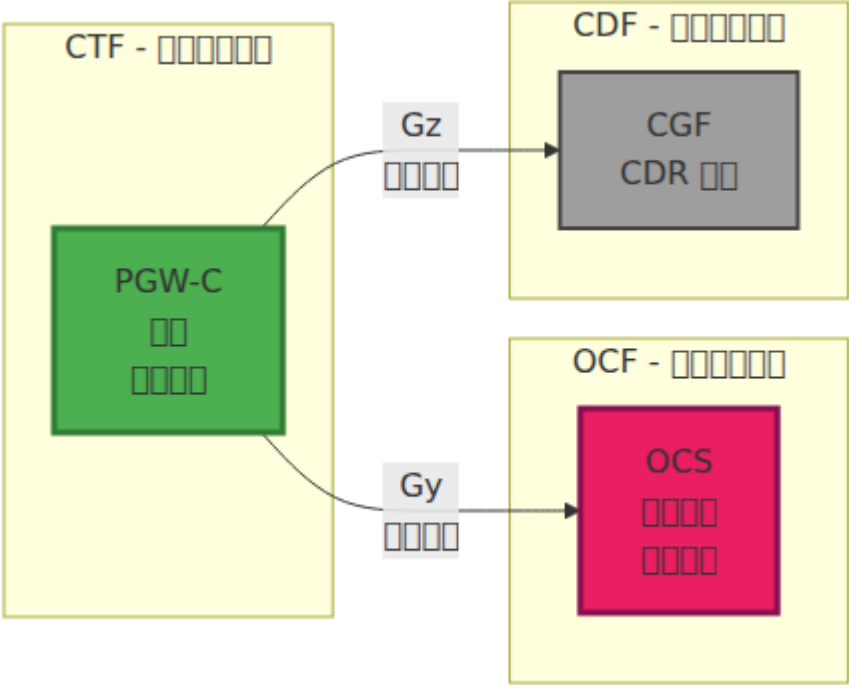
ID

Session-ID: <Origin-Host>;<high32>;<low32>[;<optional>]
omni-
pgw_c.epc.mnc999.mcc999.3gppnetwork.org;9876543210;12345;gy

- Origin-Host:** PGW-C Diameter
- high32:** 32
- low32:** 32
- optional:** "gy" Gx

□□□□□□

□□□□



CCR-Initial (□□□□□□ - □□)

□□□ UE □□ PDN □□□□□□□□□□□□

□□□

- □ OCS □□□□□□□□
- □□□□□□□□□□
- □□ Gy/Ro □□

PGW-C □□□□□ **AVP**□

AVP 名称	AVP 编号	数据类型	说明
Session-Id	263	UTF8String	会话 ID Gy 系统内部
Auth-Application-Id	258	Unsigned32	4 (系统内部)
Origin-Host	264	DiamIdent	PGW-C 的 Diameter 名称
Origin-Realm	296	DiamIdent	PGW-C 的 Diameter 名称
Destination-Realm	283	DiamIdent	OCS 名称
CC-Request-Type	416	Enumerated	1 = INITIAL_REQUEST
CC-Request-Number	415	Unsigned32	系统内部 0 开始
Subscription-Id	443	Grouped	UE 标识 (IMSI/MSISDN)
Service-Context-Id	461	UTF8String	系统内部名称
Multiple-Services-Credit-Control	456	Grouped	系统内部名称
Requested-Service-Unit	437	Grouped	系统内部名称
Used-Service-Unit	446	Grouped	系统内部名称 0 开始
Service-Identifier	439	Unsigned32	系统内部名称
Rating-Group	432	Unsigned32	系统内部名称

系统 CCR-I 系统

```
CCR (Sequence: 272, Seq)
├─ Session-Id: "pgw_c.example.com;123;456;gy"
├─ Auth-Application-Id: 4
├─ Origin-Host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org"
├─ Origin-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ Destination-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ CC-Request-Type: INITIAL_REQUEST (1)
├─ CC-Request-Number: 0
├─ Subscription-Id (Grouped)
│   ├─ Subscription-Id-Type: END_USER_IMSI (1)
│   └─ Subscription-Id-Data: "310260123456789"
├─ Subscription-Id (Grouped)
│   ├─ Subscription-Id-Type: END_USER_E164 (0)
│   └─ Subscription-Id-Data: "15551234567"
├─ Service-Context-Id: "32251@3gpp.org"
├─ Multiple-Services-Credit-Control (Grouped)
│   ├─ Service-Identifier: 1
│   ├─ Rating-Group: 100
│   └─ Requested-Service-Unit (Grouped)
│       └─ CC-Total-Octets: 10000000 (10 MB)
└─ Used-Service-Unit (Grouped)
    └─ CC-Total-Octets: 0 (0)
```

CCA-Initial (Sequence - Seq)

Sequence OCS Seq CCR-I Seq

Seq

- Sequence
- OCS
- Seq

PGW-C Sequence **AVP**

AVP 名称	AVP 代码	说明
Result-Code	268	结果 (2001) 成功
Multiple-Services-Credit-Control	456	多服务信用控制
Granted-Service-Unit	431	授予的服务单元
Validity-Time	448	有效期
Result-Code	268	结果 (2001) 成功
Final-Unit-Indication	430	最终单元指示
Volume-Quota-Threshold	-	流量配额阈值

消息内容

```
CCA (消息 ID: 272, 类型)
├─ Session-Id: "pgw_c.example.com;123;456;gy"
├─ Result-Code: DIAMETER_SUCCESS (2001)
├─ Origin-Host: "ocs.example.com"
├─ Origin-Realm: "example.com"
├─ Auth-Application-Id: 4
├─ CC-Request-Type: INITIAL_REQUEST (1)
├─ CC-Request-Number: 0
├─ Multiple-Services-Credit-Control (Grouped)
│   ├─ Result-Code: DIAMETER_SUCCESS (2001)
│   ├─ Service-Identifier: 1
│   ├─ Rating-Group: 100
│   ├─ Granted-Service-Unit (Grouped)
│   │   └─ CC-Total-Octets: 10000000 (约 10 MB)
│   ├─ Validity-Time: 3600 (有效期 1 小时)
│   └─ Volume-Quota-Threshold: 8000000 (约 8 MB 流量配额 80%)
```

CCR-Update (消息 ID - 类型)

消息

- 80% 80%
-
-
-

-
-
-

CCR-I

- CC-Request-Type: UPDATE_REQUEST (2)
- CC-Request-Number
- Used-Service-Unit
- Requested-Service-Unit

CCR-U

```
CCR (272, )
├─ Session-Id: "pgw_c.example.com;123;456;gy"
├─ Auth-Application-Id: 4
├─ Origin-Host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org"
├─ Origin-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ Destination-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ CC-Request-Type: UPDATE_REQUEST (2)
├─ CC-Request-Number: 1
└─ Multiple-Services-Credit-Control (Grouped)
    ├─ Service-Identifier: 1
    ├─ Rating-Group: 100
    ├─ Used-Service-Unit (Grouped)
    │   └─ CC-Total-Octets: 8000000 (8 MB)
    └─ Requested-Service-Unit (Grouped)
        └─ CC-Total-Octets: 10000000 (10 MB)
```


CCA-Update (信用更新 - 更新)

信用更新 OCS と CCR-U 関係

更新

- 信用更新の仕組み
- 更新の種類
- 更新の処理

更新の種類

1. 信用更新

```
CCA (信用)
├─ Multiple-Services-Credit-Control
│   ├── Result-Code: DIAMETER_SUCCESS (2001)
│   ├── Granted-Service-Unit
│   │   └─ CC-Total-Octets: 10000000 (信用 10 MB)
│   └─ Validity-Time: 3600
```

2. 信用更新の仕組み

```
CCA (信用)
├─ Multiple-Services-Credit-Control
│   ├── Result-Code: DIAMETER_SUCCESS (2001)
│   ├── Granted-Service-Unit
│   │   └─ CC-Total-Octets: 1000000 (信用 1 MB)
│   └─ Final-Unit-Indication
│       └─ Final-Unit-Action: TERMINATE (0)
```

3. 信用更新

CCA ()

```
└─ Result-Code: DIAMETER_CREDIT_LIMIT_REACHED (4012)
└─ Multiple-Services-Credit-Control
    └─ Result-Code: DIAMETER_CREDIT_LIMIT_REACHED (4012)
        └─ Final-Unit-Indication
            └─ Final-Unit-Action: TERMINATE (0)
```

CCR-Termination (-)

- UE
- PDN
-

-
- Gy/Ro
-

- CC-Request-Type: TERMINATION_REQUEST (3)
- Used-Service-Unit
- Requested-Service-Unit
- Termination-Cause

CCR-T

```
CCR (seq: 272, id)
├─ Session-Id: "pgw_c.example.com;123;456;gy"
├─ Auth-Application-Id: 4
├─ Origin-Host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org"
├─ Origin-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ Destination-Realm: "epc.mnc999.mcc999.3gppnetwork.org"
├─ CC-Request-Type: TERMINATION_REQUEST (3)
├─ CC-Request-Number: 5
├─ Termination-Cause: DIAMETER_LOGOUT (1)
└─ Multiple-Services-Credit-Control (Grouped)
    ├─ Service-Identifier: 1
    ├─ Rating-Group: 100
    └─ Used-Service-Unit (Grouped)
        └─ CC-Total-Octets: 18500000 (seq 18.5 MB)
```

CCA-Termination (seq - id)

seq OCS id CCR-T id

seq

- seq
- seq
- seq

seq CCA-T

```
CCA (seq: 272, id)
├─ Session-Id: "pgw_c.example.com;123;456;gy"
├─ Result-Code: DIAMETER_SUCCESS (2001)
├─ Origin-Host: "ocs.example.com"
├─ Origin-Realm: "example.com"
├─ Auth-Application-Id: 4
├─ CC-Request-Type: TERMINATION_REQUEST (3)
└─ CC-Request-Number: 5
```

1. 背景

2. 目的

OCS による課金

項目	AVP	単位	説明
時間	CC-Time	秒	課金対象の時間
総量	CC-Total-Octets	4桁の16進数 + 単位	課金対象の総量
入力/出力量	CC-Input-Octets, CC-Output-Octets	バイト	課金対象の量
サービス単位	CC-Service-Specific-Units	サービス単位	課金対象のAPI 単位
その他	-	その他	課金対象の項目

3. 前提条件

PGW-C による課金

OCS による **Volume-Quota-Threshold** と **Time-Quota-Threshold** PGW-C による PGW-U への PFCP による **PFCP** による

課金

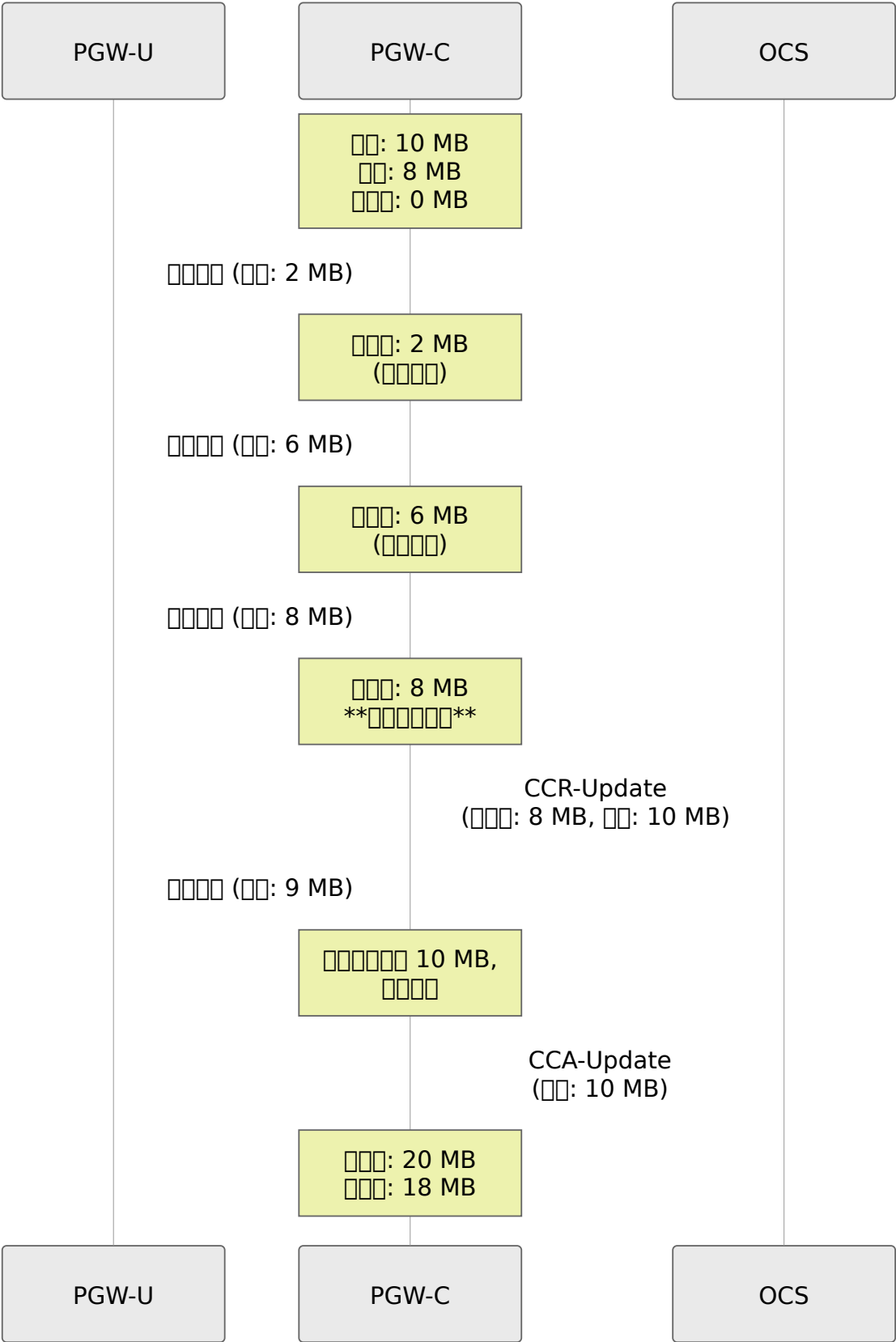
1. OCS 10 MB 80% 8 MB
2. PGW-C PGW-U PFCP
3. 8 MB
 - PGW-C CCR-Update
 -
4. OCS
5. CCR-Update
 - PGW-C

Granted-Service-Unit: 10000000 (10 MB)
Volume-Quota-Threshold: 8000000 (8 MB)

8 MB → CCR-Update
2 MB OCS

PGW-C

PGW-C PGW-U PFCP



PGW-U

PGW-C

OCS 与 CCA 消息 **Final-Unit-Indication** AVP 消息

Final-Unit-Action	值	PGW-C 消息
TERMINATE	0	终止会话
REDIRECT	1	重定向到另一个服务器
RESTRICT_ACCESS	2	限制访问

消息格式

```
CCA (消息)
├─ Multiple-Services-Credit-Control
│   ├── Result-Code: DIAMETER_SUCCESS (2001)
│   ├── Granted-Service-Unit
│   │   └─ CC-Total-Octets: 1000000 (约 1 MB)
│   └─ Final-Unit-Indication
│       ├── Final-Unit-Action: REDIRECT (1)
│       └─ Redirect-Server (Grouped)
│           ├── Redirect-Address-Type: URL (2)
│           └─ Redirect-Server-Address:
│               "http://topup.example.com"
```

PGW-C 消息

- TERMINATE:** 向 CCR-T 发送消息
- REDIRECT:** 向 PCRF 发送 HTTP 重定向 URL
- RESTRICT_ACCESS:** 向 PCRF 发送限制 IP 地址

消息格式

消息格式

3GPP 规范 TS 23.203, TS 29.212, TS 32.251

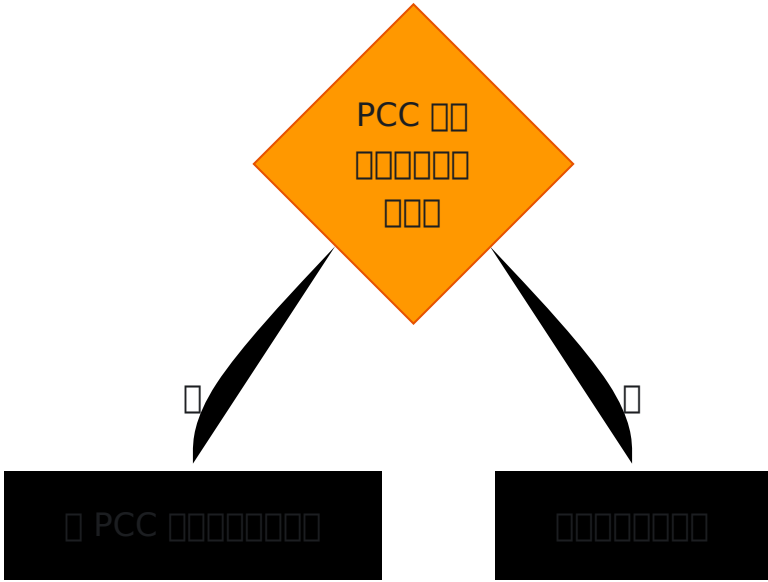
3GPP TS 23.002: PCRF, Gx, PCC, Diameter Gx, PCC

3GPP TS 23.002

PGW-C

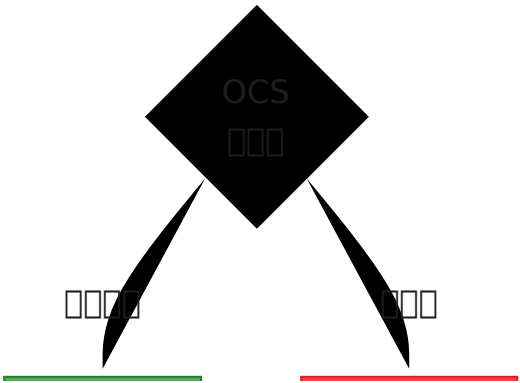
PGW-C ↔ PCRF ↔ CCR-I

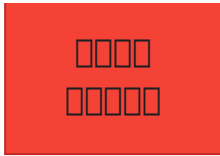
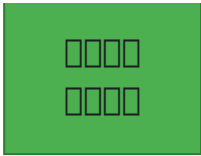
PCRF ↔ PCC ↔



PGW-C ↔ OCS ↔ CCR-I

OCS





PGW-C PCC

PCRF (CCA-I Gx)

```
CCA (Gx )
├─ Charging-Rule-Definition (Grouped)
│   ├── Charging-Rule-Name: "prepaid_data_rule"
│   ├── Rating-Group: 100
│   ├── Online: 1 ( )
│   ├── Offline: 0 ( )
│   ├── Metering-Method: VOLUME (1)
│   ├── Precedence: 100
│   ├── Flow-Information: [...]
│   └─ QoS-Information: [...]
```

PCC AVP

AVP 名称	AVP 代码	数据类型	说明
Rating-Group	432	Unsigned32	计费策略组 OCS 计费策略
Online	1009	0=否, 1=是	在线计费 (Gy)
Offline	1008	0=否, 1=是	离线计费 (Gz)
Metering-Method	1007	0=实时计费, 1=预付费, 2=后付费	计费方式
Reporting-Level	1011	0=否, 1=是	计费报告

计费策略组

策略组	策略	策略名称	策略说明
1	0	策略	计费策略组
0	1	策略	计费策略组
1	1	策略	计费策略组
0	0	-	计费策略组

计费策略

计费 PDN 策略组 计费策略组

计费策略

PGW-C Gy

└─ Rating-Group: 100 ()
└─ Online: 1

PGW-C Gy

└─ Rating-Group: 200 ()
└─ Online: 1

PGW-C Gy

└─ Rating-Group: 300 ()
└─ Online: 1

PGW-C Gy

- CCR-I MSCC

CCR-Initial

└─ Session-Id: "..."
└─ Multiple-Services-Credit-Control
└─ [Rating-Group: 100] →
└─ [Rating-Group: 200] →
└─ [Rating-Group: 300] →

OCS

CCA-Initial

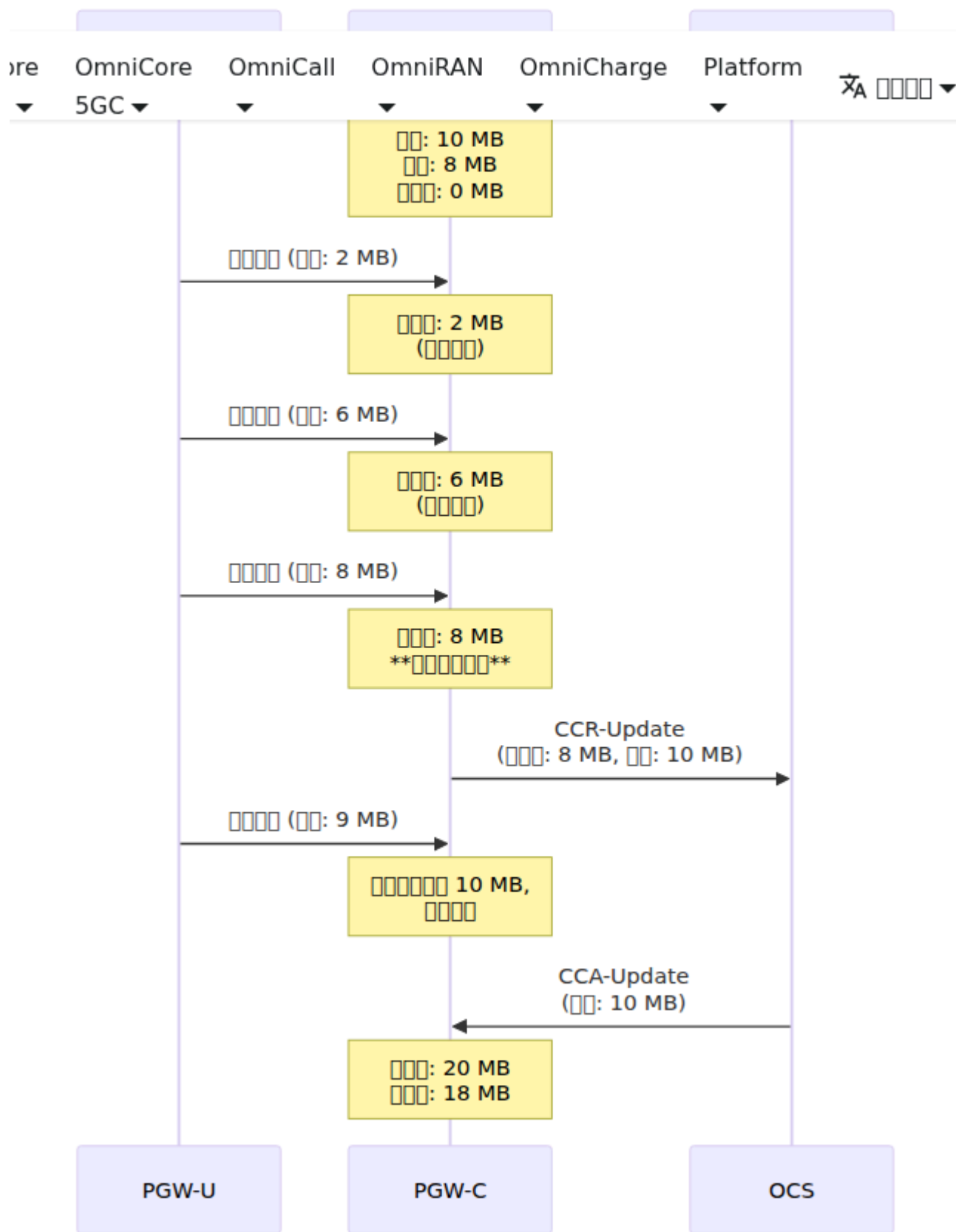
└─ Multiple-Services-Credit-Control
└─ [Rating-Group: 100] → : 10 MB
└─ [Rating-Group: 200] → : 5 MB ()
└─ [Rating-Group: 300] → : 60

PGW-C

PGW-C

```
# []  
state.charging_quotas = %  
  100 => %{granted: 10_000_000, used: 0, threshold: 8_000_000},  
  200 => %{granted: 5_000_000, used: 0, threshold: 4_000_000},  
  300 => %{granted: 60_000, used: 0, threshold: 48_000} # []  
}
```

□□□□□□□□□□



□□□□□□□□

MSCC (□□□□□□□) AVP

□□□ □□□□□/□□□□□□□□□□

□□□

```
Multiple-Services-Credit-Control (Grouped, AVP 456)
├─ Service-Identifier (Unsigned32, AVP 439)
├─ Rating-Group (Unsigned32, AVP 432)
├─ Requested-Service-Unit (Grouped, AVP 437)
│   └─ CC-Time (Unsigned32, AVP 420)
│   └─ CC-Total-Octets (Unsigned64, AVP 421)
│   └─ CC-Input-Octets (Unsigned64, AVP 412)
│   └─ CC-Output-Octets (Unsigned64, AVP 414)
├─ Used-Service-Unit (Grouped, AVP 446)
│   └─ [□ Requested-Service-Unit □□□□□]
├─ Granted-Service-Unit (Grouped, AVP 431)
│   └─ [□ Requested-Service-Unit □□□□□]
├─ Validity-Time (Unsigned32, AVP 448)
├─ Result-Code (Unsigned32, AVP 268)
└─ Final-Unit-Indication (Grouped, AVP 430)
    └─ Final-Unit-Action (Enumerated, AVP 449)
```

Table 1: Service and Rating Group

Service	Rating Group	Price
Service 1	Rating Group 1	100=100, 200=200
Service 1	Rating Group 2	100=100, 200=200
Service 1	Rating Group 3	100=100, 200=200
Service 2	Rating Group 1	100=100, 200=200
Service 2	Rating Group 2	100=100, 200=200
Service 2	Rating Group 3	100=100, 200=200
Service 3	Rating Group 1	100=100, 200=200

Table 2: Service and Rating Group

Service-Identifier: 1 (Service 1)
└─ Rating-Group: 100 (Rating Group 1 - \$0.01/MB)
└─ Rating-Group: 200 (Rating Group 2 - \$0.05/MB)
Service-Identifier: 2 (Service 2)
└─ Rating-Group: 300 (Rating Group 3 - \$0.10/MB)

Table 3: Service and Rating Group

Table 4: Service and Rating Group

Table 5: Service and Rating Group


```

config :pgw_c,
  gy: %{
    # 计费功能
    enabled: true,

    # OCS 鉴权
    timeout_ms: 5000,

    # 计费功能 鉴权 PCRF 鉴权
    default_requested_quota: 10_000_000, # 10 MB

    # 鉴权功能
    # (0.8 = 鉴权 80% 鉴权 CCR-Update)
    quota_threshold_percentage: 0.8,

    # OCS 鉴权
    # 鉴权:block, :allow
    timeout_action: :block,

    # OCS 鉴权
    # 鉴权:terminate, :redirect
    no_credit_action: :terminate,

    # 鉴权 URL 鉴权 no_credit_action: :redirect 鉴权
    topup_redirect_url: "http://topup.example.com"
  },
  diameter: %{
    listen_ip: "0.0.0.0",
    host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org",
    realm: "epc.mnc999.mcc999.3gppnetwork.org",

    # OCS 鉴权
    peer_list: [
      # PCRF 鉴权 (Gx)
      %{
        host: "pcrf.epc.mnc999.mcc999.3gppnetwork.org",
        realm: "epc.mnc999.mcc999.3gppnetwork.org",
        ip: "10.0.0.30",
        initiate_connection: true
      },
      # OCS 鉴权 (Gy)
      %{
        host: "ocs.epc.mnc999.mcc999.3gppnetwork.org",

```

```

        realm: "epc.mnc999.mcc999.3gppnetwork.org",
        ip: "10.0.0.40",
        initiate_connection: true
    }
]
}

```


enabled

- **true**: PCRF가 CCR에 OCS
- **false**: PCRF가 Gy

timeout_ms

- OCS가 CCA
- 3000-5000

default_requested_quota

- PCRF가
- 1-100 MB

quota_threshold_percentage

- % CCR-Update
- 0.75-0.85 [75%-85%]
- =
- =

timeout_action

- **:block** - OCS
- **:allow** - OCS

no_credit_action

- **:terminate** -

- **:redirect** - 重定向

配置

配置

```
config :pgw_c,  
  gy: %{  
    enabled: true,  
    timeout_action: :block,  
    no_credit_action: :terminate,  
    quota_threshold_percentage: 0.8  
  }
```

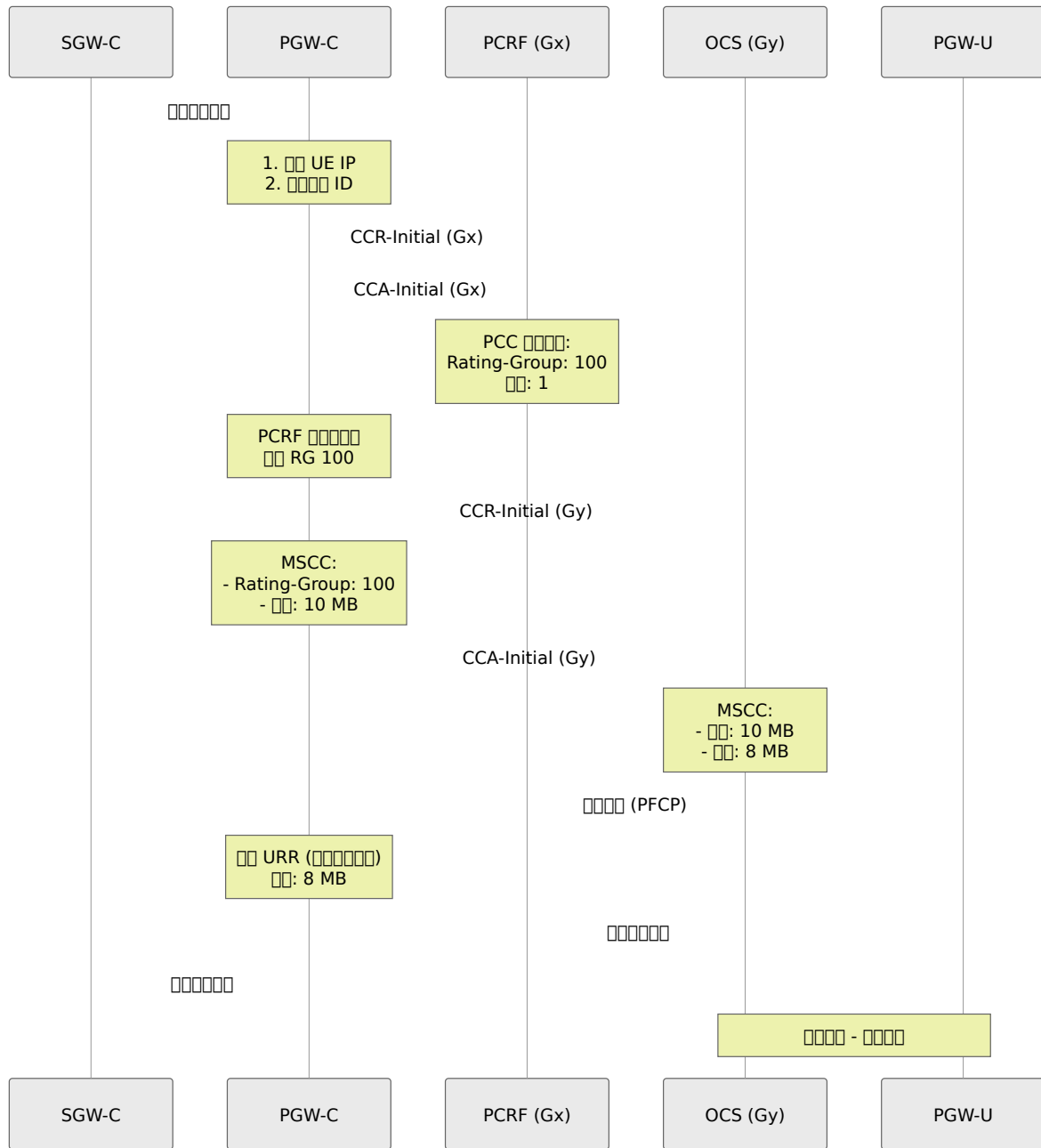
配置

```
config :pgw_c,  
  gy: %{  
    enabled: false # 禁用  
  }
```

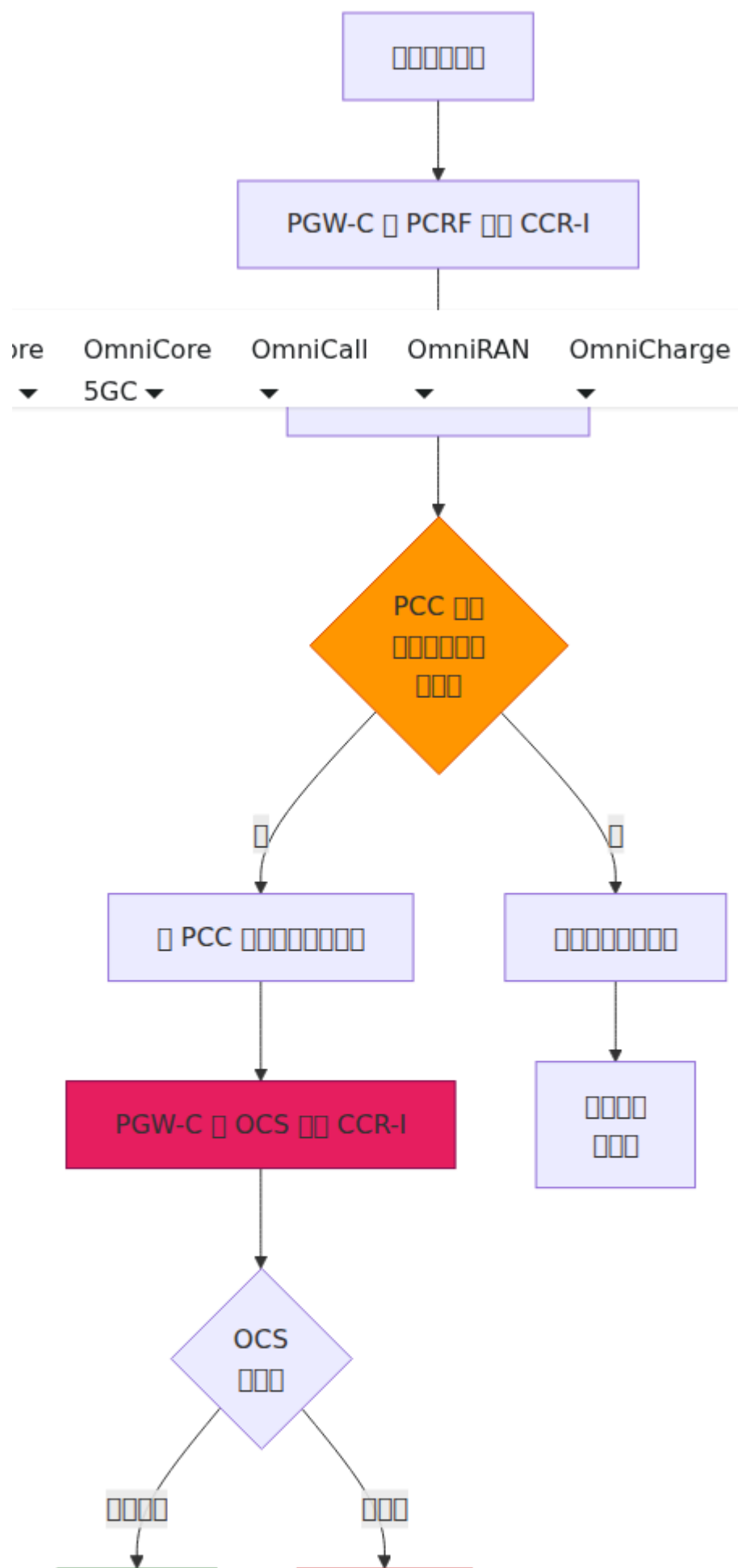
配置

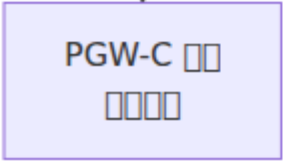
```
config :pgw_c,  
  gy: %{  
    enabled: true, # 启用 PCRF 功能  
    timeout_action: :allow, # OCS 鉴权失败  
    no_credit_action: :terminate  
  }
```

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

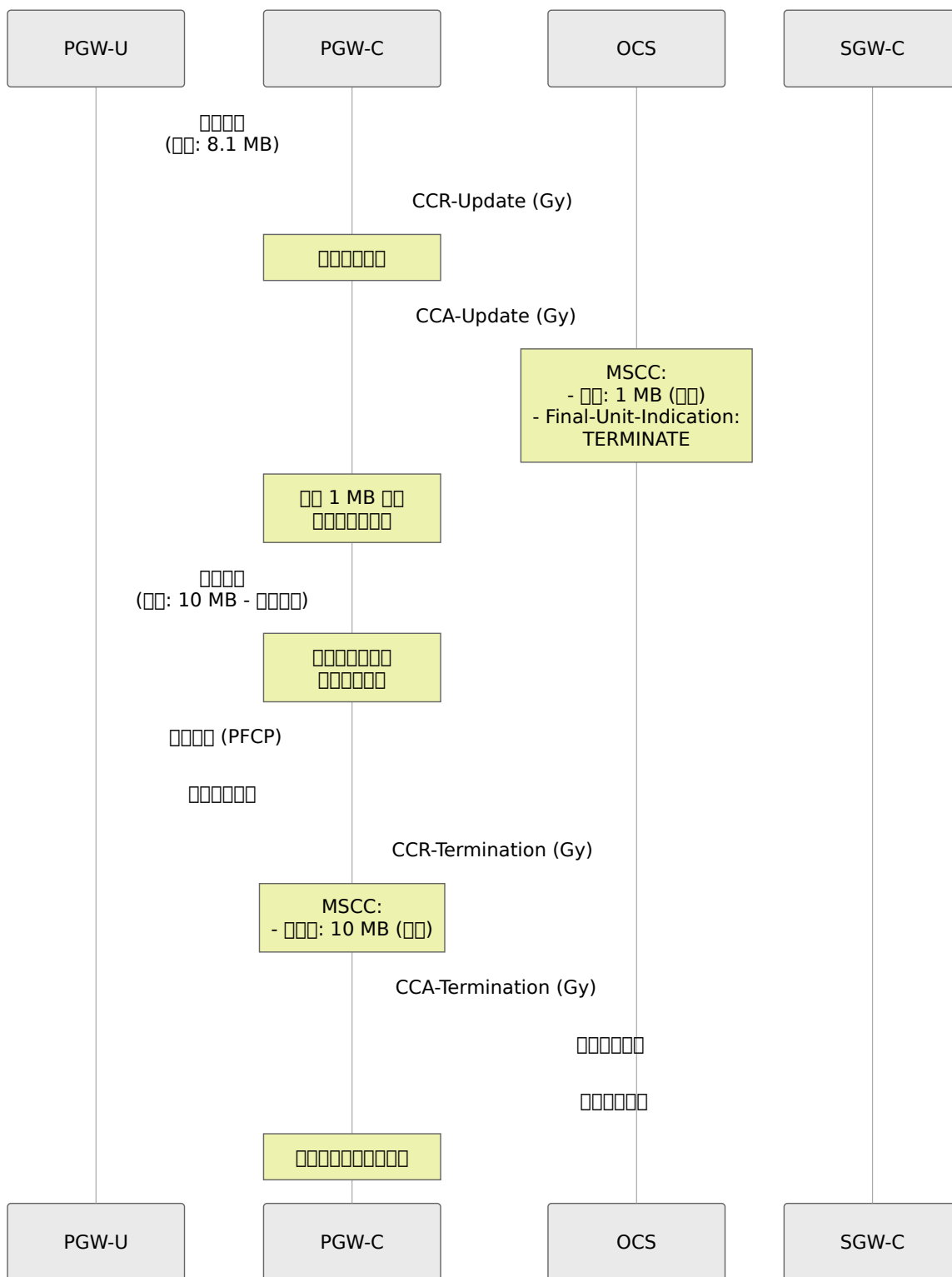


□□□□□□ (CCR-Update)

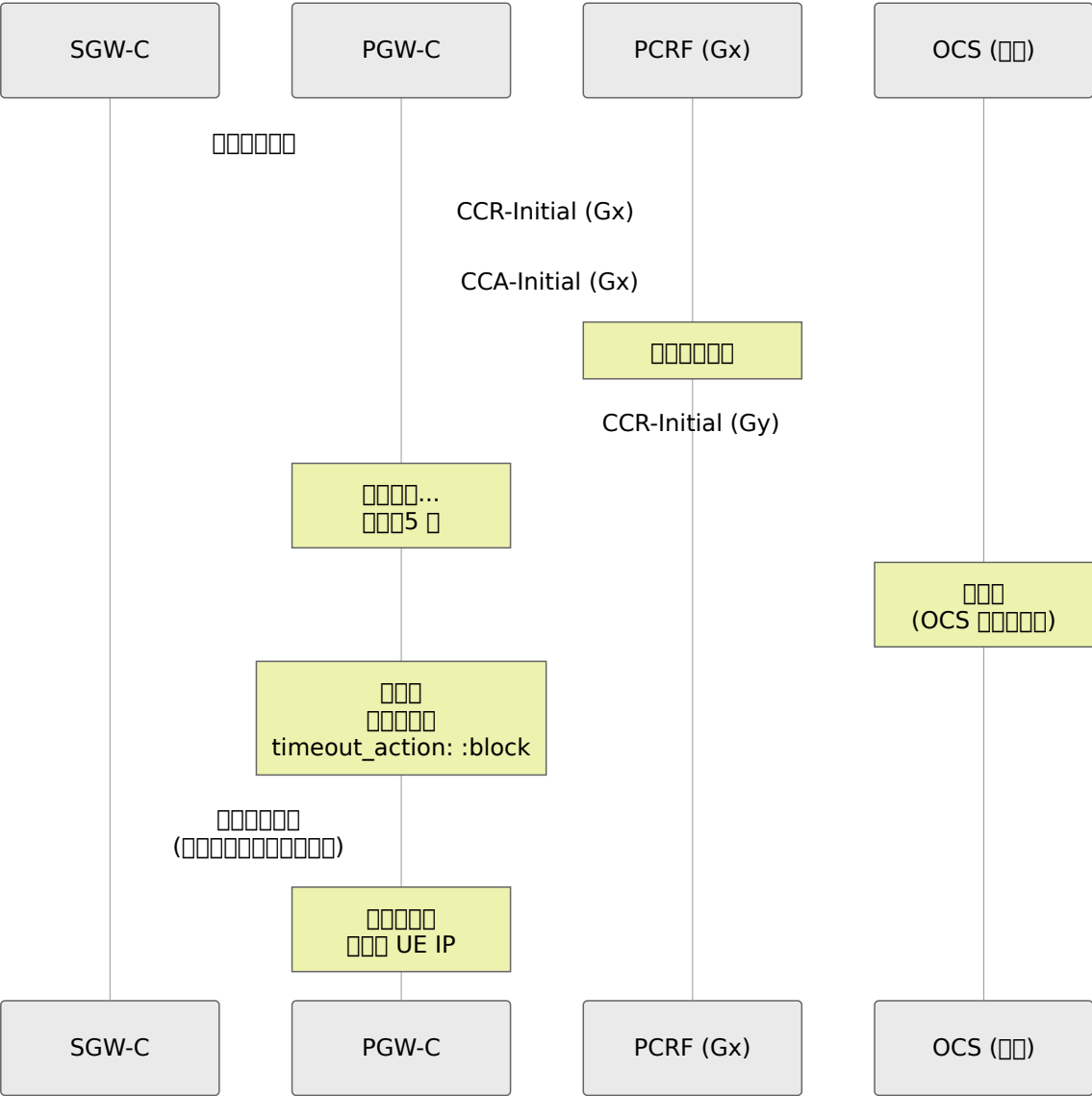




□□□□ (□□□□)



OCS 消息



消息

消息

消息

错误码	错误名称	错误描述
2001	DIAMETER_SUCCESS	成功

4xxx (4xxx)

错误码	错误名称	PGW-C 错误
4010	DIAMETER_TOO_BUSY	太忙
4011	DIAMETER_UNABLE_TO_COMPLY	无法遵从
4012	DIAMETER_CREDIT_LIMIT_REACHED	信用额度已用尽

5xxx (5xxx)

错误码	错误名称	PGW-C 错误
5003	DIAMETER_AUTHORIZATION_REJECTED	授权被拒绝
5031	DIAMETER_USER_UNKNOWN	用户未知

6xxx (6xxx)

Result-Code 错误码 错误描述

1. 错误码 - 错误描述
2. **MSCC** 错误 - 错误描述

错误

CCA-Initial

└─ Result-Code: DIAMETER_SUCCESS (2001) ← OK

└─ Multiple-Services-Credit-Control

└─ [Rating-Group: 100]

└─ Result-Code: DIAMETER_SUCCESS (2001) ← RG 100: OK

└─ [Rating-Group: 200]

└─ Result-Code: DIAMETER_CREDIT_LIMIT_REACHED (4012) ←

RG 200: OK

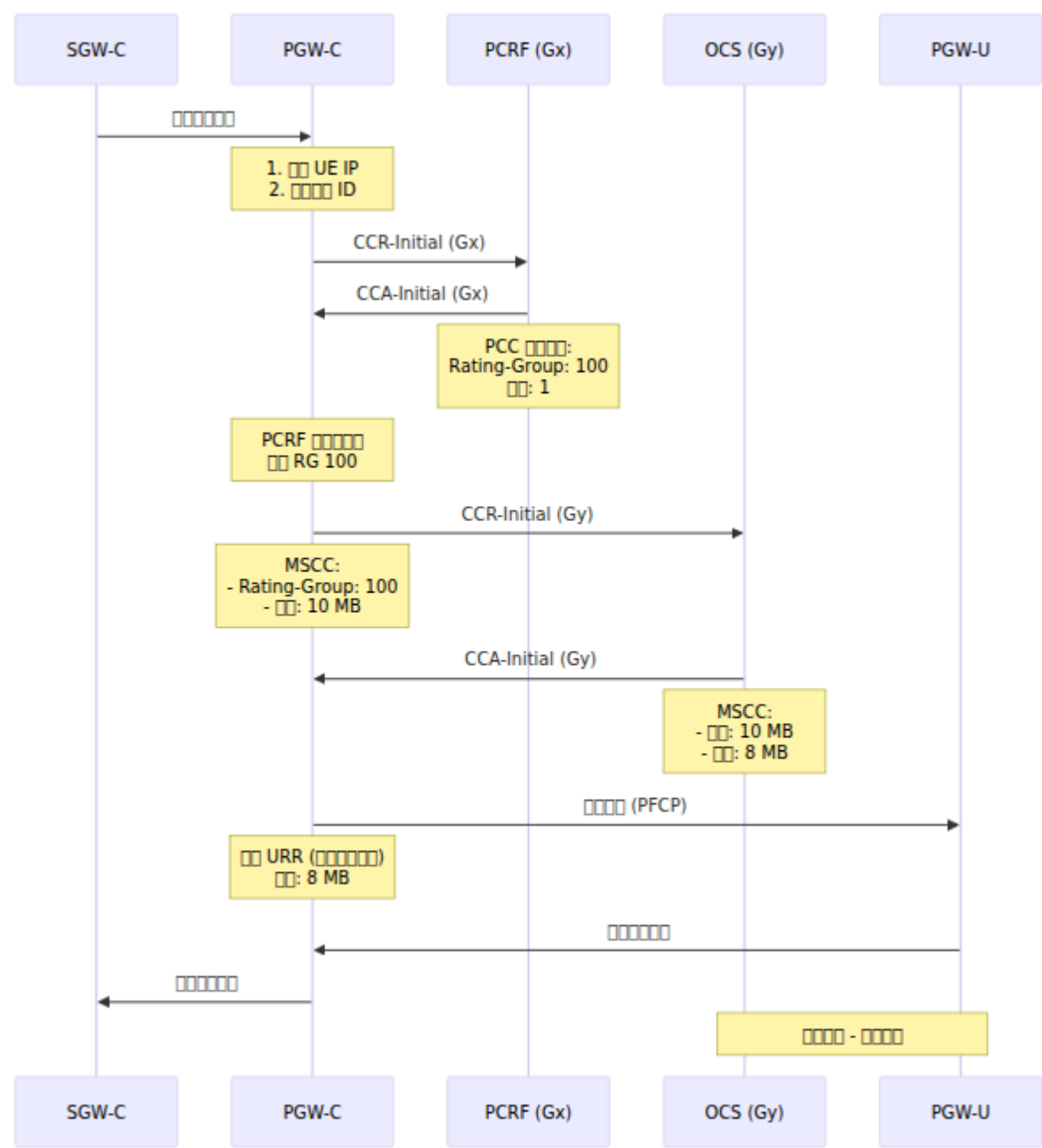
PGW-C OK

- OK RG 100 OK
- OK RG 200 OK

PGW-C Gx OK

Gx OK PCRF OK Gy OK Rating-Group OK Diameter Gx OK
OK OK

Gx & Gy



1.

PGW-C

↓

CCR-I PCRf (Gx)

CCA-I PCC

↓

□□ PCC □□□

- □□□□□□□□

$$- \square\square = 1 \square\square$$

↓

□□□□

CCR-I OCS (Gy)

CCA-I

↓

□□□□□□□□

1000000000

□□□□

□ □ □ □ □ □ □ □ □ □

2. 問題 (PCRF と RAR)

PCRF Gx RAR ()

↓

□□ PCC □□□□□□=1□□□□=200

↓

PGW-C □ OCS (Gy) □□ CCR-U

- □□□□ 200 □□ MSCC

↓

OCS

↓

□ □ □ □ □ □ □ □ □ □

□□□□

□□□□

1. CCR-Initial □ OCS □□

□□□

- □□□□□□“OCS □□”
- □□□“CCR-Initial (Gy) □□”

□□□□□

- OCS □□□□
- □□□□ OCS IP □□□
- □□□□□ Diameter □□□3868□
- OCS □□

□□□□□

```
# □□□□□□
ping <ocs_ip>

# □□ Diameter □□ (TCP 3868)
telnet <ocs_ip> 3868

# □□□□
# □□□ peer_list □□□□ OCS □□□
```

2. □□□ OCS □□

□□□

- CCA-I □ Result-Code != 2001
- □□□□□□□□

□□□□□□□□

IMEI	MSISDN	MSISDN
4012	15000000000000000000	15000000000000000000
5003	15000000000000000000	15000000000000000000
5031	15000000000000000000	15000000000000000000 OCS 15000000000000000000

1. 1. 1. 1.

1. 1. 1. 1.
2. 2. 2. 2.
3. 3. 3. 3.

3. 3. 3. 3.

1. 1. 1. 1.

- 1. 1. 1. 1.
- 2. 2. 2. 2.

1. 1. 1. 1.

- 1. 1. 1. 1.
- 2. 2. 2. 2.
- 3. 3. 3. 3.

1. 1. 1. 1.

1. 1. 1. 1.

```
1. 1. 1. 1.
|  URR-ID: 1
|  Measurement-Method: VOLUME
|  Volume-Threshold: 80000000 (8 MB)
|  Reporting-Triggers: VOLUME_THRESHOLD
```

2. 2. 2. 2.

3. 配置参数 `quota_threshold_percentage`

4. 配置参数

配置

- OCS 配置参数“配置参数”
- 配置

配置

- CCR-I 配置参数 OCS 配置参数
- PCRF 配置参数配置

配置

1. 配置 PCRF 和 PCC 配置参数
 2. 配置 OCS 配置参数配置
 3. 配置 PCC 配置 OCS 配置参数
-

□□

□□□□

```
# Gy □□□□
rate(gy_inbound_messages_total{message_type="cca"}[5m])
rate(gy_outbound_messages_total{message_type="ccr"}[5m])

# Gy □□□□
rate(gy_inbound_errors_total[5m])

# □□□□□□
rate(gy_quota_exhausted_total[5m])

# OCS □□□
rate(gy_timeout_total[5m])

# Gy □□□□□□□□
histogram_quantile(0.95,
rate(gy_inbound_handling_duration_bucket[5m]))
```

□□

```
# Gy □□□□□
- alert: GyErrorRateHigh
  expr: rate(gy_inbound_errors_total[5m]) > 0.1
  for: 5m
  annotations:
    summary: "□□□□ Gy □□□"

# OCS □□□□
- alert: OcsTimeout
  expr: rate(gy_timeout_total[5m]) > 0.05
  for: 2m
  annotations:
    summary: "□□ OCS □□"

# □□□□□□□□
- alert: CreditExhaustionSpike
  expr: rate(gy_quota_exhausted_total[5m]) > 10
  for: 5m
  annotations:
    summary: "□□□□□□"
```

Gy □□□□ (OAM API)

□□□□□ **OAM REST API** □□□□□□□□□□ HTTPS/TLS □□□□□ <https://<host>:8443> □
□□□ API □□□Swagger UI□□□ <https://<host>:8443/api/docs> □□□

□□□□□□□

```
GET /api/charging # □□□□□□□□ + □□□□□□□
GET /api/charging?search=<imsi> # □ IMSI □□□□
GET /api/charging/<imsi> # □ IMSI □□□□□□□□□□□□□
```

□□□

```
curl -k https://localhost:8443/api/charging/310170123456789
```

GET /api/charging OCS OCS /
?search=<imsi> IMSI GET /api/charging/<imsi>

Gy

```
POST /api/gy_simulator
```

OCS—
 OAM API Gy Gy
gy.enabled: true PGW-C Gy OCS

(application/json):

imsi	string	IMS
msisdn	string	MSISDN (E.164) User-Name <msisdn>@<apn>
apn	string	APN
requested_units	integer	(Requested-Service-Unit → CC-Total-Octets)
rating_group	integer	/ 1
service_id	integer	
cc_request_type	string	initial update terminate 1/2/3

— 1 MB

```
curl -k -X POST https://localhost:8443/api/gy_simulator \
-H 'content-type: application/json' \
-d
'{"imsi":"310170123456789","msisdn":"14155551234","apn":"internet","r
```

CCNA-Initial 1 MB 1

```
{
  "session_id": "omni-pgw_c.epc...;734825199;app_gy_sim",
  "cc_request_type": 1,
  "cc_request_type_name": "initial",
  "result_code": "diameter_success",
  "accepted": true,
  "ocs_host": "ocs.epc.example.org",
  "ocs_realm": "epc.example.org",
  "granted": { "total_octets": 1000000, "input_octets": null,
"output_octets": null, "time": null },
  "validity_time": 3600,
  "final_unit_action": null,
  "final_unit_redirect": null,
  "mscc_result_code": "diameter_success"
}
```

422 imsi 503 Gy OCS reason :timeout

- GET /api/charging/<imsi>
- POST /api/gy_simulator OCS IMSI
- GET /api/charging?search=<imsi>

OCS Prometheus

네트워크

네트워크

- **Diameter Gx** - PCRF - PCC
- **CDR** -
- -

네트워크

- - PDN
- **PFCP** - URR - PGW-U
- **S5/S8** - GTP-C

네트워크

- - Gy - OCS
- **UE IP** - IP

네트워크

Gn/Gp 介面

SGSN 與 GTP-C 透過 2G/3G 網路

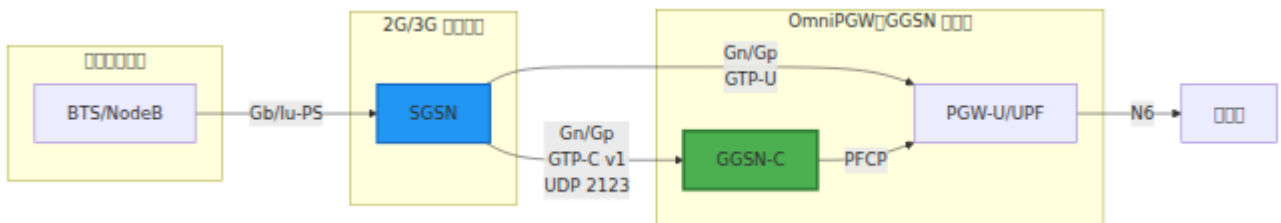
OmniPGW 與 Omnitouch 網路架構

網路

Gn/Gp 網路 OmniPGW 透過 2G/3G 網路 GGSN 透過 GPRS 網路透過 GTP-C v1 網路
透過 GGSN 與 SGSN 網路 PDP 網路

- Gn 網路透過 PLMN 網路 SGSN
- Gp 網路透過 PLMN 網路 SGSN

網路網路網路 - 網路網路網路



PGW (S5/S8) 对比

项目	GGSN (Gn/Gp)	PGW (S5/S8)
传输层	GTP-C v1	GTP-C v2
3GPP 规范	TS 29.060	TS 29.274
网络层	PDP 上下文	PDN 连接 + 会话
接口	NSAPI (0-15)	EBI (5-15)
接口名称	Gi	Gx (PCRF)
QoS 支持	R99 QoS 参数	EPS 网络 QoS (QCI)
端口	2123/TCP	2123

对比

GTP-C 对比 1

- 支持 GTP-C v1 (3GPP TS 29.060)
- 支持 UDP
- 支持 2123/TCP S5/S8 GTP-C v2 接口
- 支持 QoS

TEID 对比

在 PDP 上下文中 TEID 的分配

- GGSN 的 TEID 与 OmniPGW 的 TEID
- GGSN 的 TEID 与 UPF 的 TEID

- **SGSN** 的 **TEID** 的 SGSN 的
- **SGSN** 的 **TEID** 的 SGSN 的

的

SGSN → GGSN: 的 TEID = GGSN 的 TEID

GGSN → SGSN: 的 TEID = SGSN 的 TEID

CUPS 的

OmniPGW 的 2G/3G 的 CUPS 的

- 的 **GSN** 的 OmniPGW 的 IP 的 `gn.local_ipv4_address` 的
- 的 **GSN** 的 UPF 的 IP 的 PFCP 的

的 GGSN-C 的 PGW-U/UPF 的 IP 的 CUPS 的

NSAPI 的

NSAPI 的 PDP 的

- 的 5-15 的 0-4 的
- 的 PDP 的
- 的 LTE/EPC 的 EBI

□□

□□□□

```
# config/runtime.exs
config :pgw_c,
  # □□ gn □□□□Gn/Gp □□□□□□
  gn: %{
    # Gn □□□□ IPv4 □□
    local_ipv4_address: "10.0.0.20",

    # □□□□ IPv6 □□
    local_ipv6_address: nil,

    # GTP-C v1 □□□□ S5/S8 □□□
    local_port: 2123
  },

  # □ PCO □□□□ DNS □□□□□ PGW □□□
  dns: %{
    primary_ipv4: {8, 8, 8, 8},
    secondary_ipv4: {8, 8, 4, 4}
  }
```

Gn □□□□

□□	□□	□ □	□□	□□
local_ipv4_address	□□ □	□	-	□□ Gn □□□□□ IPv4 □□□□□□ SGSN □ □□ GGSN □□□
local_ipv6_address	□□ □	□	nil	□□□ IPv6 □□□□□□ 2G/3G □□□□□ IPv4□
local_port	□□	□	2123	GTP-C □ UDP □□□□□ 3GPP TS 29.060 □□□□□□□ S5/S8 □□□

DNS

IP	IP	IP	IP	IP
primary_ipv4	IP	IP	{8, 8, 8, 8}	IP PCO IP DNS IP IPv4 IP
secondary_ipv4	IP	IP	{8, 8, 4, 4}	IP PCO IP DNS IP IPv4 IP
primary_ipv6	IP	IP	nil	IP DNS IP IPv6 IP
secondary_ipv6	IP	IP	nil	IP DNS IP IPv6 IP

IP

IP

```
# IP SGSN IP GTP-C
iptables -A INPUT -p udp --dport 2123 -s <sgsn_network>/24 -j ACCEPT

# IP SGSN IP GTP-C
iptables -A OUTPUT -p udp --dport 2123 -d <sgsn_network>/24 -j ACCEPT
```

IP

PDP

IP PDP IP

UE SGSN → GGSN

UE PDP Context

UE IEs

IE	Length	Value
IMSI	13	IMSI
NSAPI	5	NSAPI (5-15)
TEID I	4	SGSN TEID
TEID S	4	SGSN TEID
QoS	4	QoS PDN IP
QoS	4	QoS APN
GSN I	IP	SGSN GSN
GSN S	IP	SGSN GSN
QoS	4	QoS QoS
MSISDN	13	MSISDN
QoS	4	APN QoS
QoS	4	QoS

UE

UE PDP Context

- └ IMSI: 310260123456789
- └ NSAPI: 5
- └ TEID MME: 0x12345678
- └ TEID S-GW: 0x87654321
- └ Address: IPv4 address
- └ APN: internet
- └ GSN MME: 10.1.1.100
- └ GSN S-GW: 10.1.1.100
- └ QoS Class: [R99 QoS Class]
- └ PCO: [DNS Class]

UE **PDP** Context

UE GGSN → SGSN

UE PDP Context

UE Context

IE 名称	类型	说明
名称	名称	名称名称
名称名称	名称名称名称	名称名称名称
名称	名称	GSN 名称
TEID 名称 I	名称	GSN 名称名称名称
TEID 名称	名称	GSN 名称名称名称
NSAPI	名称	名称 NSAPI
名称 ID	名称	名称 ID
名称名称	名称	名称 UE IP 名称
GSN 名称名称	IP	GSN 名称名称
GSN 名称名称	IP	GSN 名称名称
QoS 名称	名称	名称 QoS 名称
名称名称	名称	DNS 名称名称

名称名称

PDP Context

- Length: 128
- TEID I: 0xAABBCCDD
- TEID S: 0xDDCCBBAA
- NSAPI: 5
- UE ID: 0x11223344
- IPv4: 100.64.1.42
- GSN MNC: 10.0.0.20
- GSN MCC: 10.0.0.20
- QoS Class: [Default QoS]
- PCO
 - DNS: 8.8.8.8
 - DNS: 8.8.4.4

PDP Context

SGSN → GGSN

PDP Context SGSN QoS Class

Context

- SGSN Context
- QoS Class
- SGSN Context

Context

IE	
NSAPI	PDP Context
TEID I	SGSN TEID
GSN MNC	SGSN
QoS Class	QoS Class

IE PDP

SGSN → GGSN

PDP

IE	
TEID	GGSN TEID
GSN	GGSN
QoS	QoS

IE PDP

SGSN → GGSN

PDP

IE	
NSAPI	PDP

IE PDP

SGSN → GGSN

PDP

IE 00	00
00	00000000

0000

0000 / 00

00000

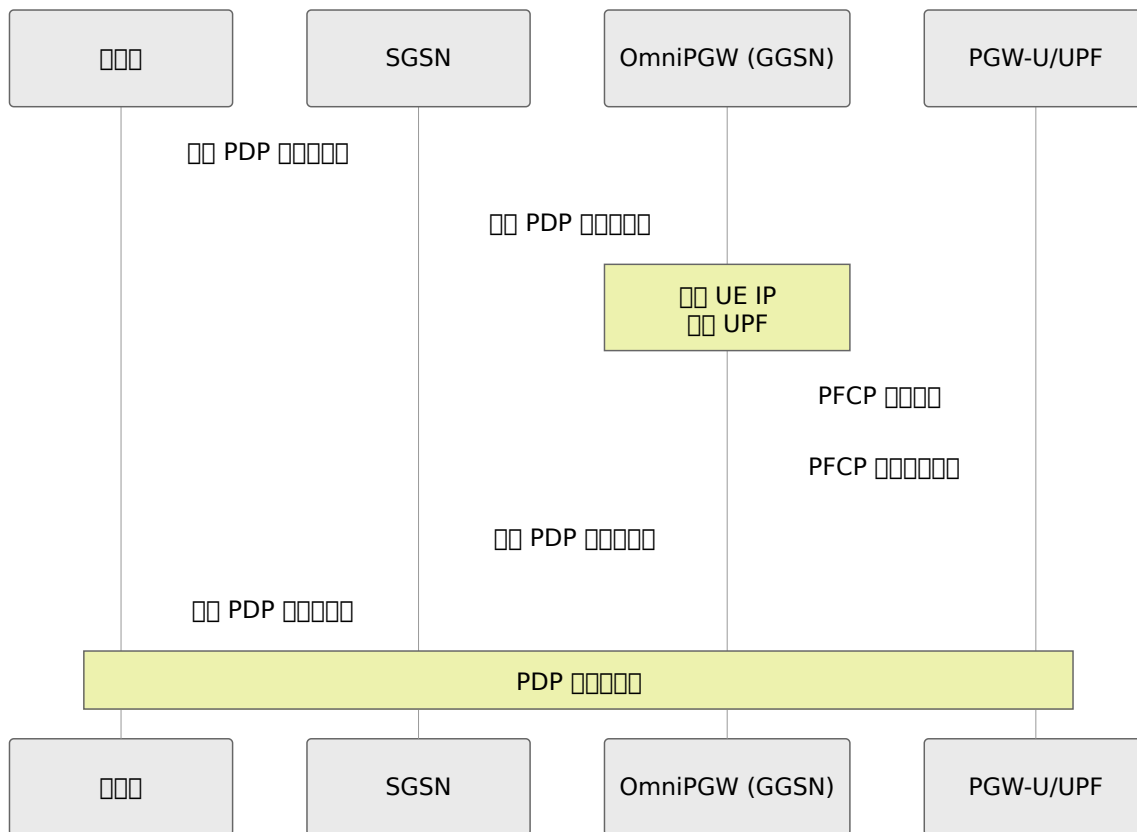
0000000000000000

0000000

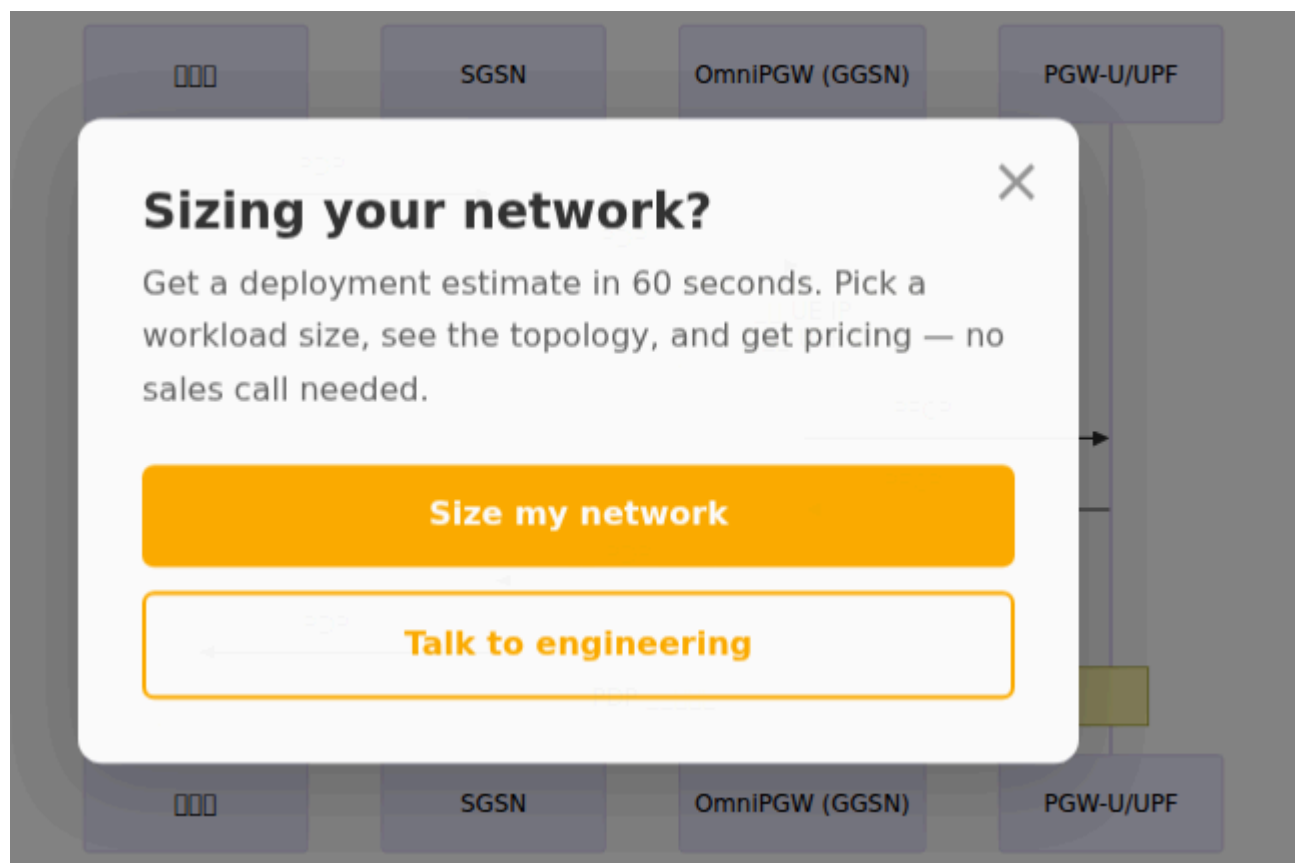
IE 00	00
00	00000000000000



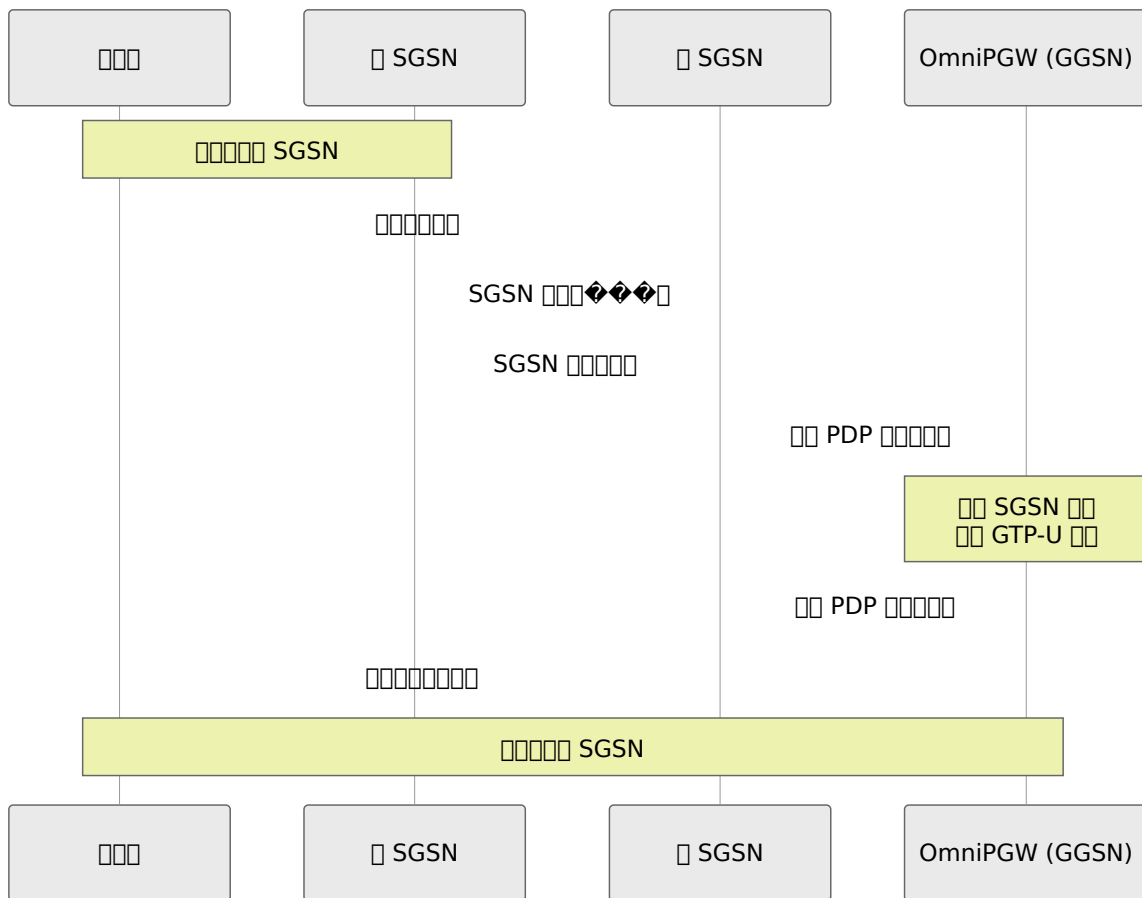
PDP



PDP



SGSN



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

OmniPGW □ GGSN □□□ PGW □□□□□□□□

UE IP

PGW AddressRegistry UE IP

UPF/PFCP ☐☐☐☐

PGW PFCP

- PDP → PDR/FAR/QER →
- upf_selection → UPF

- 3GPP TS 23.060 PFCP 4.10

3GPP TS 23.060 Gy

3GPP TS 23.060 Diameter Gy 4.10

- 3GPP TS 23.060
- PGW 3GPP TS 23.060 Gy 4.10
- 3GPP TS 23.060 Diameter Gy 4.10

CDR 4.10

3GPP TS 23.060 PGW 3GPP TS 23.060 CDR 4.10 3GPP TS 23.060 CDR 4.10

3GPP TS 23.060

3GPP

3GPP	3GPP	3GPP
128	3GPP TS 23.060	3GPP TS 23.060

0000

00	00	0000
192	000	000 PDP 000
193	00000000	000000 GTP 00
194	IMSI 00	0000000
195	MS 0 GPRS 00	MS 000
196	MS 0 GPRS 00	MS 0000
197	MS 00	MS 0000
198	000000	GTP 000000
199	000000	00000
200	000000	0000000
201	00 IE 000	IE 00000
202	00 IE 00	000000 IE
203	00 IE 000	00 IE 00
204	00000	00000
205	00000	000000
206	P-TMSI 000000	00000
207	GPRS 00000	000000
208	0000000	0000000

項目	単位	説明
209	メッセージ数	メッセージ数

##

Gn メトリクス

```
# メトリクス
gn_inbound_messages_total{message_type="create_pdp_context_request"}
gn_inbound_messages_total{message_type="update_pdp_context_request"}
gn_inbound_messages_total{message_type="delete_pdp_context_request"}

gn_outbound_messages_total{message_type="create_pdp_context_response"}
gn_outbound_messages_total{message_type="update_pdp_context_response"}
gn_outbound_messages_total{message_type="delete_pdp_context_response"}

# メトリクス
gn_outbound_transaction_duration_bucket
```

##

PDP メトリクス

```
rate(gn_inbound_messages_total{message_type="create_pdp_context_request"}
[5m])
```

##

```
sum(rate(gn_outbound_messages_total{message_type="create_pdp_context_response"}
[5m]))
/
sum(rate(gn_inbound_messages_total{message_type="create_pdp_context_request"}
[5m]))
```

📡 **PDP** 📡📡

```
gn_inbound_messages_total{message_type="create_pdp_context_request"}  
-  
gn_inbound_messages_total{message_type="delete_pdp_context_request"}
```

📡📡📡📡

📡📡📡📡 **PDP** 📡📡📡📡📡

📡📡

- SGSN 📡📡 PDP 📡📡📡
- 📡📡📡📡
- SGSN 📡📡

📡📡📡📡

1. 📡📡📡📡
2. OmniPGW 📡📡📡 IP 📡📡
3. 📡📡📡📡 UDP 2123
4. UPF/PFCP 📡📡📡

📡📡

```
# 检查 OmniPGW 配置
netstat -ulnp | grep 2123

# 抓包配置
tcpdump -i any -n port 2123

# 检查配置
grep "local_ipv4_address" config/runtime.exs

# 检查 PFCP 配置
curl http://pgw:9090/metrics | grep pfc
```

检查 PDP 配置

检查

- 检查 PDP 配置
- PDP 配置

检查

```
检查 199 配置
→ IP 地址
→ 检查 curl http://pgw:9090/metrics | grep address_registry_count
→ 检查 ue.subnet_map 配置 IP 地址

检查 202 配置 IE 配置
→ SGSN 配置 IEs
→ 检查 SGSN 配置
→ 检查 tcpdump 配置

检查 204 配置
→ UPF/PFCP 配置
→ 检查 PFCP 配置
→ 检查 OmniPGW 配置
```

检查 PDP 配置

检查

- PDP Context
- Context

Context

- PDP Context TEID
- NSAPI
- Context

Context

```
# Context
curl http://pgw:9090/metrics | grep session_count

# Context
journalctl -u omnipgw | grep "PDP Context"
```

Context

Context

1. S5/S8 Context

- Gn S5/S8 UDP 2123
- OmniPGW GTP
- IP 2G/3G 4G

2. SGSN Context

- SGSN GGSN IP
- Gn VLAN
- Gp

3. IP Context

- IP PGW GGSN

- 網路 2G/3G 網路 APN 網路

網路

1. 網路

- 網路 session_count 網路
- 網路 2G/3G 網路
- 網路 2G/3G 網路

2. UDP 網路

- 網路 S5/S8 網路
- 網路 4-8 MB

網路

網路 GGSN 網路 OmniPGW 網路

1. APN 網路

- 網路 APN 網路 GGSN 網路
- 網路 PCO 網路 DNS 網路

2. IP 網路

- 網路 IP 網路
- 網路 APN 網路

3. QoS 網路

- R99 QoS 網路
 - 2G/3G 網路 QCI 網路
-

□□□□

□□□□

- □□□□ - □□□□□□
- □□□□ - □□□□□□□□
- **UE IP** □□ - IP □□□□
- **PCO** □□ - □□□□□□

□□□□

- **S5/S8** □□ - 4G/LTE □□□□
- **PFCP** □□ - □□□□□□
- **Diameter Gy** □□ - □□□□

□□

- □□□□ - □□□□□□
- □□□□ - □□□□
- □□ **CDR** □□ - CDR □□

□□□□

- **3GPP TS 29.060** - GPRS □□□□GTP□□□□ Gn □ Gp □□
- **3GPP TS 23.060** - □□□□□□□□GPRS□□□□□□
- **3GPP TS 24.008** - □□□□□□□ 3 □□□□QoS □□□□□□

□□□□□□

OmniPGW Gn/Gp □□ - □ *Omni*touch □□□□□□

UE APN UE IP Pool Allocation PFCP Gx
PFCP/Sxb Diameter Gx

□ □

- 4G
- PDN 4G
- 4G/5G
- PGW-C 4G
- 4G/5G (OmniUPF)
- 4G
- 4G/5G

11

```

graph LR
    UE[UE] -- PDN Type --> SGW-C[SGW-C]
    SGW-C -- "S5/S8-C SR (GTP-C v2)" --> OmniPGW[OmniPGW]
    OmniPGW -- "Gx (Diameter)" --> PCRF_HSS[PCRF / HSS]
    OmniPGW -- "Sxb (PFCP)" --> OmniUPF[OmniUPF]
    OmniUPF -- "OSPFv3 (FRR)" --> Edge_Router[Edge Router]
    Edge_Router -- IPv6 --> Internet((Internet))
  
```

面 / 面	IPv6 面
OmniPGW — S5/S8	面 PDN 面 UE 面 PDN 面 (PAA)
OmniPGW — Gx	面 UE 面 PCRF面 Framed-IP-Address 面 IPv4面 IPv4/IPv4v6 PDN面 Framed-IPv6-Prefix 面 IPv6 PDN
OmniPGW — Sxb	面 PFCP 面 UE IPv6 面 PDN 面 UPF
OmniUPF — 面	面 IPv6 面/面 IPv6 面
OmniUPF — FRR	面 OSPFv3 面 UE IPv6 面 /128 面

面 3GPP TS 29.274 (GTPv2-C)面 3GPP TS 29.212 (Gx)面 3GPP TS 29.244 (PFCP)面

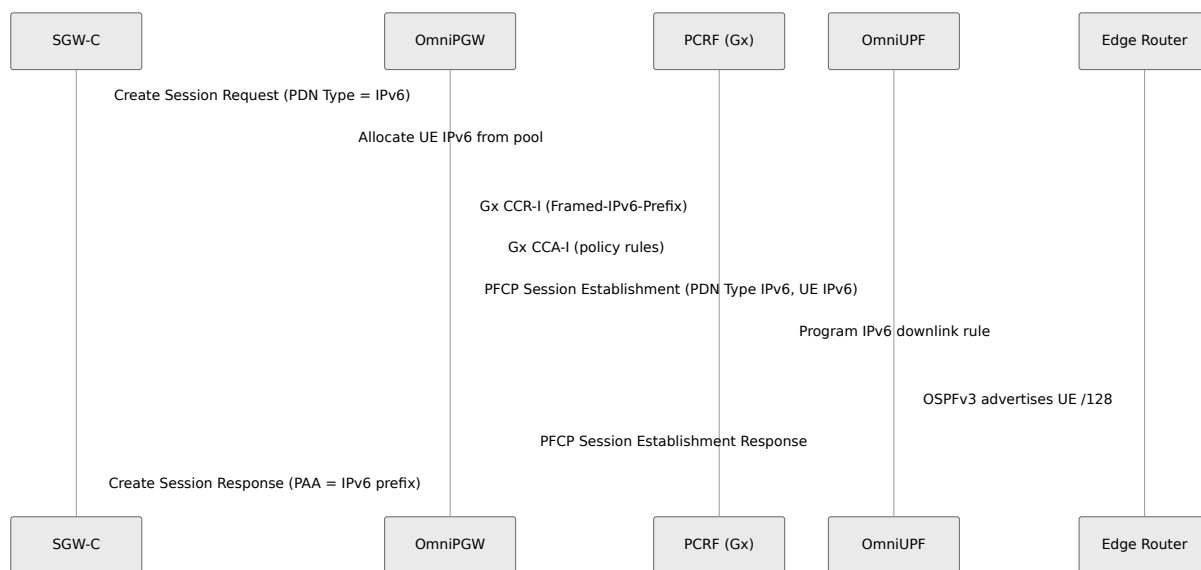
PDN 面

面	PDN 面	UE 面	PAA 面 (TS 29.274 §8.14)
1	IPv4	IPv4 面	4 面 IPv4 面
2	IPv6	IPv6 面 (/64)	面 + 16 面
3	IPv4v6	IPv4 面 + IPv6 面	面 + 16 面 IPv6 + 4 面 IPv4

PGW-C 面 PDN 面 IE 面 PDN 面 PDN 面 IE 面 PAA IE 面 SGW 面 PAA面 UPF 面 PFCP PDN 面 IE面

面 IPv6 面 IPv4v6面 IPv4v6 PDN面 APN 面 IPv6 面 PGW-C 面 PDN 面 IPv4 面 APN 面 (SOS) APN面 APN 面 IPv6 面 IPv4v6 面

PGW-C



PGW-C

PGW-C

UE IPv6

UE `runtime.exs` `ue.subnet_map` APN `runtime.exs` IPv6 `runtime.exs`
APN `runtime.exs` IPv6 CIDR `runtime.exs`

```
config :pgw_c,
  ue: %{
    subnet_map: %{
      # APN ==> CIDR IPv4 / IPv6
      default: [
        "100.64.16.0/24",
        "2001:db8:6f85:70::/64"
      ]
    }
  }
```

UE	Network	IPv4	IPv6	Configuration
ue.subnet_map	Map	1	-	UE APN is default. CIDR is default. APN is default.
IPv4	CIDR	1	-	IPv4 is 100.64.16.0/24. IPv6 is 2001:db8:6f85:70::/64. UE is IPv6. UE is /64.

UE APN is default. APN is default. UE IP Pool Allocation

IPv4, IPv6, CIDR, APN, IPv4, IPv6, IPv4v6, PDN, PGW-C, PDN, IPv4v6

Gx (Framed-IPv6-Prefix)

Gx is PGW-C, UE, Gx, AVP

- IPv6 PDN — Framed-IPv6-Prefix AVP, UE IPv6, RFC 3162, Framed-IP-Address, IPv4
- IPv4, IPv4v6 PDN — UE IPv4, Framed-IP-Address, IPv4v6, IPv6, Gx, CCR, AVP, IPv4, IPv6, PFCP, UPF, PAA, UE

IPv6, PCRF, Framed-IPv6-Prefix, IPv6, IPv6

OmniUPF

IPv6 PDN, OmniUPF, IPv6, PGW-C, PFCP, IPv6, OSPFv3, UE IPv6, /128, UPF

OmniUPF, PGW-C, UPF, UE, OSPFv3, OmniUPF IPv6, PGW-C, UPF, IPv6, PGW-C, PFCP, UE IPv6, PDN, PFCP/Sxb

□□□

PDN □□□□ (PAA) — TS 29.274 §8.14

PDN □□	□□□□□□
IPv4	4 □□□IPv4 □□
IPv6	1 □□□□□□ + 16 □□ IPv6 □□/□□
IPv4v6	1 □□□□□□ + 16 □□ IPv6 + 4 □□ IPv4

Gx □□□ AVP

AVP	□□	□□	□□	□□
Framed-IP-Address	8	IPv4	4 □□□□	RFC 4005
Framed-IPv6-Prefix	97	IPv6	□□□□ + □□□□ + □□	RFC 3162 §2.3

PFCP PDN □□ IE — TS 29.244

□	□□
1	IPv4
2	IPv6
3	IPv4v6

IPv6

Gx CCR 配置 IPv6

配置 IPv4 配置 IPv6/IPv4v6 配置 `remote_peer_not_responding` Gx 配置

配置

- PCRF 配置 `Framed-IPv6-Prefix` AVP 配置
- 配置 PCRF/HSS 配置 IPv6

配置

- 配置 PCRF/HSS 配置
- 配置 PCRF 配置 Gx CCR-I 配置 `Framed-IPv6-Prefix`
- 配置 IPv4 配置 IPv6 配置

IPv6 配置 UE

配置 IPv6 PAA 配置 UE 配置

配置 OmniUPF 配置 — PGW-C 配置 PCF 配置 UE `/128` 配置
配置 UPF 配置 OSPFv3/配置 OmniUPF IPv6 /
配置

配置 IPv6

配置 IPv6 PDN 配置 IPv6 PAA

配置

- 配置 APN `:default` `ue.subnet_map` 配置 IPv6 CIDR
- 配置 IPv6 配置

配置

- 配置 `ue.subnet_map` 配置 APN 配置 IPv6 CIDR 配置 PGW-C 配置

2. 設定確認

APN から IPv6 へ IPv4v6 へ

UE から ue.subnet_map を設定して IPv4 CIDR から IPv6 へ APN から **IPv4v6** PDN から APN から IPv4 へ IPv6 へ

APN から IPv6 へ IPv6 へ IPv6 へ APN から TS 29.274 / TS 23.401 から PDN から

APN から		UE / PCRF / UPF から
APN (SOS) APN	IPv4 — IPv4 から IPv6 から PDN から IPv4	PAAからPFCP PDN から PFCP UE IP から IPv4からGx から Framed-IP-AddressからP-CSCF から PCO から IMS から warning
APN	— {error, :no_ipv6_pool} から error	

APN から APN から "sos" から sos から sos.mnc410.mcc310.gprs から IPv6 から APN から IPv6 から

MME から PGW-C から IPv4v6 から IPv4 から IPv6 から IPv4v6 から SOS から 911/112 から IPv4 から

- APNから APN から IPv6 CIDRから PGW-C から
- IPv4 から (SOS) APNから — IPv4 から SOS PDN から IPv6 CIDRから

OmniPGW 部署架构图

Prometheus 部署图

OmniTouch 部署图

部署

1. 部署
 2. 部署
 3. 部署
 4. Prometheus 部署
 5. Grafana 部署
 6. 部署
 7. 部署
 8. 部署
-

部署

OmniPGW 部署图

1. OAM REST API 部署图

- 部署 (GET /api/sessions)
- PFCP 部署 (GET /api/upf)
- Diameter 部署 (GET /api/diameter)
- 部署 (GET /api/sessions/<imsi>)

2. Prometheus 部署图

- 部署
- 部署

- 
- 

Prometheus OAM REST API HTTPS 8443 /api

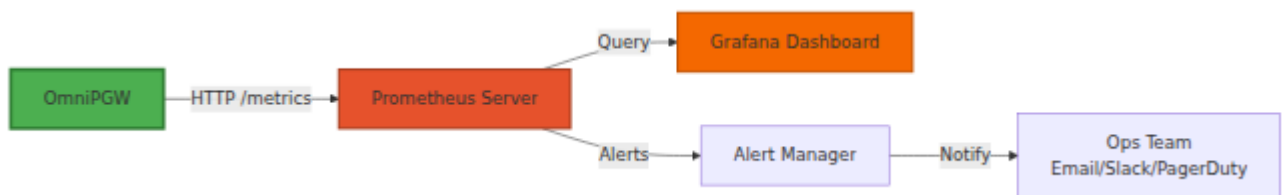
- 3GPP - 3G API
- PCFDP 3G - 3G API
- Diameter Gx - 3GPP API

Prometheus

OmniPGW  Prometheus 

- 1000000 - 1000000000000000
- 10000 - 1000000000
- 10000 - 1000000000
- 100 - 10000000
- 100 - 1000000000000000

□ □ □ □



11

```
❏ config/runtime.exs
```

```

config :pgw_c,
  metrics: %{
    enabled: true,
    ip_address: "127.0.0.42",      # 00000000 "0.0.0.0"
    port: 9090,                  # HTTP
    registry_poll_period_ms: 10_000 # 10s
  }

```

127.0.0.42:9090 Prometheus
 ip_address "0.0.0.0"

HTTP

```

http://<omnipgw_ip>:<port>/metrics

```

```

curl http://10.0.0.20:9090/metrics

```

Prometheus

```
# HELP teid_registry_count The number of TEID registered to
sessions
# TYPE teid_registry_count gauge
teid_registry_count 150

# HELP address_registry_count The number of addresses registered
to sessions
# TYPE address_registry_count gauge
address_registry_count 150

# HELP s5s8_inbound_messages_total The total number of messages
received from S5/S8 peers
# TYPE s5s8_inbound_messages_total counter
s5s8_inbound_messages_total{message_type="create_session_request"}
1523
s5s8_inbound_messages_total{message_type="delete_session_request"}
1487
```

□□□□

OmniPGW □□□□□□□□

□□□□

□□□□□□

메트릭	유형	설명
teid_registry_count	Gauge	S5/S8 TEID 개수
seid_registry_count	Gauge	PFCP SEID 개수
gx_session_id_registry_count	Gauge	Gx Diameter Session-ID 개수
gy_session_id_registry_count	Gauge	Gy Diameter Session-ID 개수
session_registry_count	Gauge	IMSI, EBI 개수
address_registry_count	Gauge	UE IP 개수
charging_id_registry_count	Gauge	ID 개수 CDR 개수 CDR 개수
sxb_sequence_number_registry_count	Gauge	PFCP sequence number 개수
s5s8_sequence_number_registry_count	Gauge	S5/S8 sequence number 개수
sxb_peer_registry_count	Gauge	PFCP peer 개수

예제

```
# 메트릭 이름
teid_registry_count

# 5분 평균
rate(teid_registry_count[5m])

# 1시간 최대
max_over_time(teid_registry_count[1h])
```

출력 예시

PDN

pgw_session_create_total	Counter	result, cause, pdn_type, rat_type	/ GTP PDN RAT PDN
pgw_session_delete_total	Counter	pdn_type	PDN PDN
pgw_session_modify_total	Counter	-	
pgw_bearer_create_total	Counter	type, qci	/ QCI
pgw_bearer_delete_total	Counter	type	/

```
# 5
sum(rate(pgw_session_create_total{result="success"}[5m])) /
sum(rate(pgw_session_create_total[5m]))

#
sum(rate(pgw_session_create_total{result="failure"}[5m])) by
(cause)
```

UE IP

UE IP

메트릭	유형	태그	설명
ue_pool_addresses_allocated	Gauge	pool, ip_version	현재 UE IP 주소가 할당된 개수
ue_pool_addresses_total	Gauge	pool, ip_version	현재 UE IP 주소의 총 개수
ue_pool_addresses_utilization_ratio	Gauge	pool, ip_version	UE IP 주소의 사용률 (0.0-1.0)
ue_ip_allocation_failures_total	Counter	ip_version, reason	UE IP 할당 실패 횟수 (이전 버전과 비교)

예제

```
# UE IP 주소가 80% 이상 사용됨
ue_pool_addresses_utilization_ratio > 0.8

# UE IP 할당 실패 횟수
sum(rate(ue_ip_allocation_failures_total[5m])) by (reason)
```

참고

S5/S8 (GTP-C) 관련

计数器	类型	消息类型	描述
s5s8_inbound_messages_total	Counter	message_type	统计 S5/S8 消息
s5s8_outbound_messages_total	Counter	message_type	统计 S5/S8 消息
s5s8_inbound_errors_total	Counter	message_type	S5/S8 消息错误
s5s8_outbound_errors_total	Counter	message_type	S5/S8 消息错误

消息类型

- create_session_request
- create_session_response
- delete_session_request
- delete_session_response
- create_bearer_request
- delete_bearer_request

Sxb (PCFP) 消息

计数器	类型	参数	说明
sxb_inbound_messages_total	Counter	message_type	PFCP 消息接收总数
sxb_outbound_messages_total	Counter	message_type	PFCP 消息发送总数
sxb_inbound_errors_total	Counter	message_type	PFCP 消息接收错误总数
sxb_outbound_errors_total	Counter	message_type	PFCP 消息发送错误总数
sxb_outbound_responses_total	Counter	message_type, cause_code_class, peer_ip	PFCP 消息发送响应总数，按消息类型、原因码类和对端 IP 地址统计

消息类型

- association_setup_request
- association_setup_response
- heartbeat_request
- heartbeat_response
- session_establishment_request
- session_establishment_response
- session_modification_request

- session_deletion_request

Gx (Diameter) 表

項目名	型	注	備考
gx_inbound_messages_total	Counter	message_type	表 Diameter 表
gx_outbound_messages_total	Counter	message_type	表 Diameter 表
gx_inbound_errors_total	Counter	message_type	Diameter エラー
gx_outbound_errors_total	Counter	message_type	Diameter エラー
gx_outbound_responses_total	Counter	message_type, result_code_class, diameter_host	表 Diameter エラー エラー 表

表

- gx_CCA (Credit-Control-Answer)
- gx_CCR (Credit-Control-Request)
- gx_RAA (Re-Auth-Answer)
- gx_RAR (Re-Auth-Request)

表 gx_outbound_responses_total 表

- 2xxx - 2001 DIAMETER_SUCCESS
- 3xxx - 3001 DIAMETER_COMMAND_UNSUPPORTED

- 4xxx - 4001 DIAMETER_AUTHENTICATION_REJECTED
- 5xxx - 5012 DIAMETER_UNABLE_TO_COMPLY

00000

```

# Gy Gx 00000
sum(rate(gx_outbound_responses_total{result_code_class="2xxx"}[5m]))
sum(rate(gx_outbound_responses_total[5m])) * 100

# PCRF 00000
rate(gx_outbound_responses_total{result_code_class!="2xxx"}[5m]) by (

# 00000 Re-Auth-Answer 0000
gx_outbound_responses_total{message_type="gx_RAA",result_code_class='

# 0000 PCRF 000000000
rate(gx_outbound_responses_total{result_code_class=~"4xxx|5xxx",diame
[5m])) > 0.1
  
```

Gy (Diameter) 000

Gy 00000OCS0000000000 Diameter Gy 000000

0000	00	00	00
gy_inbound_messages_total	Counter	message_type	000 Gy 00
gy_outbound_messages_total	Counter	message_type	000 Gy 00
gy_inbound_errors_total	Counter	message_type	Gy 000000
gy_outbound_errors_total	Counter	message_type	Gy 000000

00000

- gy_CCA (Credit-Control-Answer)
- gy_CCR (Credit-Control-Request)

Gn/Gp (GTPv1-C) 000

Gn/Gp GGSN 2G/3G SGSN Gn

gn_inbound_messages_total	Counter	message_type	Gn/Gp SGSN
gn_outbound_messages_total	Counter	message_type	Gn/Gp SGSN
gn_inbound_errors_total	Counter	message_type	Gn/Gp
gn_outbound_errors_total	Counter	message_type	SGSN Gn/Gp

rescues_total	Counter	module, function	

項目名	形式	単位	注
s5s8_inbound_handling_duration	Histogram	request_message_type	5 [[
gn_inbound_handling_duration	Histogram	request_message_type	(([[[
sxb_inbound_handling_duration	Histogram	request_message_type	F [[
gx_inbound_handling_duration	Histogram	request_message_type	(([[[
gy_inbound_handling_duration	Histogram	request_message_type	(([[[

項目名

項目名	種類	単位
s5s8_outbound_transaction_duration	Histogram	request_message_type
gn_outbound_transaction_duration	Histogram	request_message_type
sxb_outbound_transaction_duration	Histogram	request_message_type
gx_outbound_transaction_duration	Histogram	request_message_type
gy_outbound_transaction_duration	Histogram	request_message_type

項目名

- 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1.0, 5.0
- 100μs, 500μs, 1ms, 5ms, 10ms, 50ms, 100ms, 500ms, 1s, 5s

単位


```
# 95th 百分位 S5/S8 值
histogram_quantile(0.95,
  rate(s5s8_inbound_handling_duration_bucket[5m])
)

# 平均 PFCP 值
rate(sxb_inbound_handling_duration_sum[5m]) /
rate(sxb_inbound_handling_duration_count[5m])
```

UPF 网络

UPF 网络

메트릭	유형	필터	설명
<code>upf_peers_total</code>	Gauge	-	현재 UPF 개수
<code>upf_peers_healthy</code>	Gauge	-	현재 UPF 정상 상태 개수 + 연결 상태
<code>upf_peers_unhealthy</code>	Gauge	-	현재 UPF 비정상 상태 개수
<code>upf_peers_associated</code>	Gauge	-	현재 PCF와 연결된 UPF 개수
<code>upf_peers_unassociated</code>	Gauge	-	현재 PCF와 연결되지 않은 UPF 개수
<code>upf_peer_associated</code>	Gauge	<code>peer_ip</code>	현재 UPF 개수 (PCF 연결 상태 1/0)
<code>upf_peer_healthy</code>	Gauge	<code>peer_ip</code>	현재 UPF 정상 상태 개수 (1=정상, 0=비정상)
<code>upf_peer_missed_heartbeats</code>	Gauge	<code>peer_ip</code>	현재 UPF 미수신된 하트비트 개수
<code>upf_heartbeat_rtt</code>	Histogram	<code>peer_ip</code>	PCF 하트비트 응답 시간 분포

메트릭

```
# 检查 UPF 节点
upf_peers_healthy / upf_peers_total

# 检查 UPF 节点是否健康
upf_peers_unhealthy > 0

# 检查 UPF 节点
upf_peer_healthy{peer_ip="10.98.0.20"}

# 检查 UPF 节点是否健康
upf_peer_missed_heartbeats > 2
```

检查

```

# UPF Peer Down
- alert: UPF_Peer_Down
  expr: upf_peer_healthy == 0
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "UPF {{ $labels.peer_ip }} Down"
    description: "UPF {{ $labels.peer_ip }} PFCP Down"

# UPF Pool Degraded
- alert: UPF_Pool_Degraded
  expr: (upf_peers_healthy / upf_peers_total) < 0.5
  for: 2m
  labels:
    severity: critical
  annotations:
    summary: "UPF Pool Degraded"
    description: "{{ $value | humanizePercentage }} % UPF Peers Healthy"

# UPF Heartbeat Issues
- alert: UPF_Heartbeat_Issues
  expr: upf_peer_missed_heartbeats > 2
  for: 30s
  labels:
    severity: warning
  annotations:
    summary: "UPF {{ $labels.peer_ip }} Heartbeat Issues"
    description: "{{ $value }} Missed Heartbeats"

```

P-CSCF

P-CSCF

名称	タイプ	単位	説明
<code>pcscf_fqdns_total</code>	Gauge	-	登録済みの P-CSCF FQDN の数
<code>pcscf_fqdns_resolved</code>	Gauge	-	DNS 解決済みの P-CSCF FQDN の数
<code>pcscf_fqdns_failed</code>	Gauge	-	DNS 解決失敗の P-CSCF FQDN の数
<code>pcscf_servers_total</code>	Gauge	-	登録済みの P-CSCF サーバの数
<code>pcscf_servers_healthy</code>	Gauge	fqdn	健康な FQDN の P-CSCF サーバの数
<code>pcscf_servers_unhealthy</code>	Gauge	fqdn	不健康な FQDN の P-CSCF サーバの数

この P-CSCF 情報は IMS からも提供される

Diameter 接続

この Diameter 接続

名称	タイプ	単位	説明
<code>diameter_peers_connected</code>	Gauge	application	登録済みの Diameter サーバの接続数 単位: gx/gy/all

例

```
# この Gx (PCRF) 接続を確認
diameter_peers_connected{application="gx"} == 0
```

確認方法

確認方法

項目	型	値
license_status	Gauge	有効状態1 = 0 = 有効

有効

```
# 有効状態
license_status == 1

# 無効状態
license_status == 0
```

有効

```
- alert: PGW_C_License_Invalid
  expr: license_status == 0
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "PGW-C 有効状態"
    description: "有効状態 - 有効状態"
```

有効

有効状態有効状態有効状態有効状態有効状態有効状態 GTP-C 有効 “有効状態”73有効状態有効状態有効状態有効状態

メトリック名	タイプ	説明
<code>vm_memory_total</code>	Gauge	VM メモリ使用量
<code>vm_memory_processes</code>	Gauge	プロセスメモリ使用量
<code>vm_memory_system</code>	Gauge	システムメモリ使用量
<code>vm_system_process_count</code>	Gauge	Erlang プロセス数
<code>vm_system_port_count</code>	Gauge	ポート数

Prometheus

インストール

OmniPGW に Prometheus `prometheus.yml` を追加

```
# prometheus.yml
global:
  scrape_interval: 15s
  evaluation_interval: 15s

scrape_configs:
  - job_name: 'omnipgw'
    static_configs:
      - targets: ['10.0.0.20:9090']
        labels:
          instance: 'omnipgw-01'
          environment: 'production'
          site: 'datacenter-1'
```


❏❏ OmniPGW ❏❏

```
scrape_configs:
  - job_name: 'omnipgw'
    static_configs:
      - targets:
          - '10.0.0.20:9090'
          - '10.0.0.21:9090'
          - '10.0.0.22:9090'
        labels:
          environment: 'production'
```

❏❏❏❏

Kubernetes❏

```
scrape_configs:
  - job_name: 'omnipgw'
    kubernetes_sd_configs:
      - role: pod
    relabel_configs:
      - source_labels: [__meta_kubernetes_pod_label_app]
        action: keep
        regex: omnipgw
      - source_labels: [__meta_kubernetes_pod_ip]
        target_label: __address__
        replacement: '${1}:9090'
```

❏❏

❏❏❏❏❏

```
# ❏❏ Prometheus ❏❏
curl http://prometheus:9090/api/v1/targets

# ❏❏❏❏❏❏
curl 'http://prometheus:9090/api/v1/query?
query=teid_registry_count'
```

Grafana 安装

安装步骤

1. 安装 Prometheus 服务

安装 → 配置 → 启动 → Prometheus
URL: <http://prometheus:9090>

2. 配置 Grafana

配置 Grafana 的 JSON 数据源

配置项

配置 1

```
# 名称  
teid_registry_count  
  
# 数据类型Gauge  
# 单位  
# 范围 < 5000  
# 范围 5000-8000  
# 范围 > 8000
```

配置 2

```
# 名称  
rate(s5s8_inbound_messages_total{message_type="create_session_request"  
[5m])  
  
# 数据类型  
# 单位/秒
```

例 3 IP 割合

```
# 10.0.0.0 /24 の IP 数
(address_registry_count / 254) * 100

# Gauge
# 0-100
#
# < 70%
# 70-85%
# > 85%
```

例 4 95th 分位数

```
#
histogram_quantile(0.95,

rate(s5s8_inbound_handling_duration_bucket{request_message_type="create"}
[5m])
)

#
#
```

例 5

```
#
rate(s5s8_inbound_errors_total[5m])

#
# /
# > 0.1
```

例 6 Gx

```
# 計算 Gx の割合
sum(rate(gx_outbound_responses_total{result_code_class="2xxx"}
[5m])) /
sum(rate(gx_outbound_responses_total[5m])) * 100

# Gauge
# 0-100
#
# > 95%
# 90-95%
# < 90%
```

図 1 - 計算結果

```
# 計算結果
sum(rate(gx_outbound_responses_total[5m])) by (result_code_class)

# 計算結果
# {{ result_code_class }}
```

図 2 - **PCRf** の計算結果

```
# PCRf の計算結果
sum(rate(gx_outbound_responses_total[5m])) by (diameter_host,
result_code_class)

# 計算結果
# {{ diameter_host }} - {{ result_code_class }}
```

図 3 - **UPF** の計算結果

```
# 仪表盘百分比
(upf_peers_healthy / upf_peers_total) * 100

# 仪表盘Gauge
# 仪表盘范围0-100
# 单位
# 100%
# 50-99%
# < 50%
```

UPF - 仪表盘

```
# 仪表盘 UPF 仪表盘
upf_peer_healthy

# 仪表盘
# 单位
# 1 = "UP"仪表盘
# 0 = "DOWN"仪表盘
```

Dashboard

```
{
  "dashboard": {
    "title": "OmniPGW - Dashboard",
    "panels": [
      {
        "title": "Registry Count",
        "targets": [
          {
            "expr": "teid_registry_count",
            "legendFormat": "Count"
          }
        ],
        "type": "graph"
      },
      {
        "title": "Session Rate",
        "targets": [
          {
            "expr":
"rate(s5s8_inbound_messages_total{message_type=\"create_session_request\"}[5m])",
            "legendFormat": "Rate/Sec"
          }
        ],
        "type": "graph"
      },
      {
        "title": "IP Address Usage",
        "targets": [
          {
            "expr": "(address_registry_count / 254) * 100",
            "legendFormat": "Usage %"
          }
        ],
        "type": "gauge"
      },
      {
        "title": "Histogram p95",
        "targets": [
          {
            "expr": "histogram_quantile(0.95,
```

```
rate(s5s8_inbound_handling_duration_bucket[5m]))",
    "legendFormat": "S5/S8 p95"
  },
  {
    "expr": "histogram_quantile(0.95,
rate(sxb_inbound_handling_duration_bucket[5m]))",
    "legendFormat": "PFCP p95"
  }
],
"type": "graph"
}
]
```

□□

□□□□

□□ `omnipgw_alerts.yml` □

```

groups:
- name: omnipgw
  interval: 30s
  rules:
    # 会话数
    - alert: OmniPGW_HighSessionCount
      expr: teid_registry_count > 8000
      for: 5m
      labels:
        severity: warning
      annotations:
        summary: "OmniPGW 会话数高"
        description: "{{ $value }}"
    - alert: OmniPGW_SessionCountCritical
      expr: teid_registry_count > 9500
      for: 2m
      labels:
        severity: critical
      annotations:
        summary: "OmniPGW 会话数临界"
        description: "{{ $value }}"

    # IP 池
    - alert: OmniPGW_IPPoolUtilizationHigh
      expr: (address_registry_count / 254) * 100 > 80
      for: 10m
      labels:
        severity: warning
      annotations:
        summary: "OmniPGW IP 池利用率高"
        description: "IP 池 {{ $value }}% 利用率"
    - alert: OmniPGW_IPPoolExhausted
      expr: address_registry_count >= 254
      for: 1m
      labels:
        severity: critical
      annotations:
        summary: "OmniPGW IP 池耗尽"
        description: "IP 池耗尽"

    # 地址数

```



```

- alert: OmniPGW_HighErrorRate
  expr: rate(s5s8_inbound_errors_total[5m]) > 0.1
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "OmniPGW S5/S8 S5/S8"
    description: "{{ $value }}"

- alert: OmniPGW_GxErrorRate
  expr: rate(gx_inbound_errors_total[5m]) > 0.05
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "OmniPGW Gx S5/S8"
    description: "{{ $value }}" Diameter S5/S8

# Gx S5/S8
- alert: OmniPGW_GxResponseFailureRate
  expr: |

sum(rate(gx_outbound_responses_total{result_code_class!="2xxx"}
[5m])) /
    sum(rate(gx_outbound_responses_total[5m])) > 0.1
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "OmniPGW Gx S5/S8"
    description: "{{ $value | humanizePercentage }}" Gx S5/S8
    "2xxx S5/S8"

- alert: OmniPGW_GxPCRFFailures
  expr:
rate(gx_outbound_responses_total{result_code_class=~"4xxx|5xxx"}
[5m]) by (diameter_host) > 0.05
  for: 3m
  labels:
    severity: warning
  annotations:
    summary: "PCRF {{ $labels.diameter_host }}"
    description: "{{ $value }}" PCRF {{
$labels.diameter_host }}"

```

```

# UPF 告警
- alert: OmniPGW_UPF_PeerDown
  expr: upf_peer_healthy == 0
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "UPF 告警 {{ $labels.peer_ip }} 故障"
    description: "UPF 告警 PFCP 告警"

- alert: OmniPGW_UPF_PoolDegraded
  expr: (upf_peers_healthy / upf_peers_total) < 0.5
  for: 2m
  labels:
    severity: critical
  annotations:
    summary: "UPF 告警"
    description: "{{ $value | humanizePercentage }} % UPF 告警
告警< 50%"

- alert: OmniPGW_UPF_HeartbeatFailures
  expr: upf_peer_missed_heartbeats > 2
  for: 30s
  labels:
    severity: warning
  annotations:
    summary: "UPF {{ $labels.peer_ip }} 告警"
    description: "{{ $value }} 告警"

- alert: OmniPGW_UPF_AllDown
  expr: upf_peers_healthy == 0 and upf_peers_total > 0
  for: 30s
  labels:
    severity: critical
  annotations:
    summary: "告警 UPF 告警"
    description: "告警 UPF 告警"

# 告警
- alert: OmniPGW_HighLatency
  expr: |
    histogram_quantile(0.95,
      rate(s5s8_inbound_handling_duration_bucket[5m])

```

```

    ) > 100000
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "OmniPGW 高延迟"
    description: "p95 延迟 {{ $value }}μs (> 100ms)"

# 内存使用
- alert: OmniPGW_HighMemoryUsage
  expr: vm_memory_total > 2000000000
  for: 10m
  labels:
    severity: warning
  annotations:
    summary: "OmniPGW 高内存使用"
    description: "VM 内存 {{ $value | humanize }}B 高"

- alert: OmniPGW_HighProcessCount
  expr: vm_system_process_count > 100000
  for: 10m
  labels:
    severity: warning
  annotations:
    summary: "OmniPGW 高进程数"
    description: "{{ $value }} Erlang 高"

```

AlertManager ☐☐

```
# alertmanager.yml
global:
  resolve_timeout: 5m

route:
  receiver: 'ops-team'
  group_by: ['alertname', 'instance']
  group_wait: 10s
  group_interval: 10s
  repeat_interval: 12h

routes:
  - match:
      severity: critical
    receiver: 'pagerduty'

  - match:
      severity: warning
    receiver: 'slack'

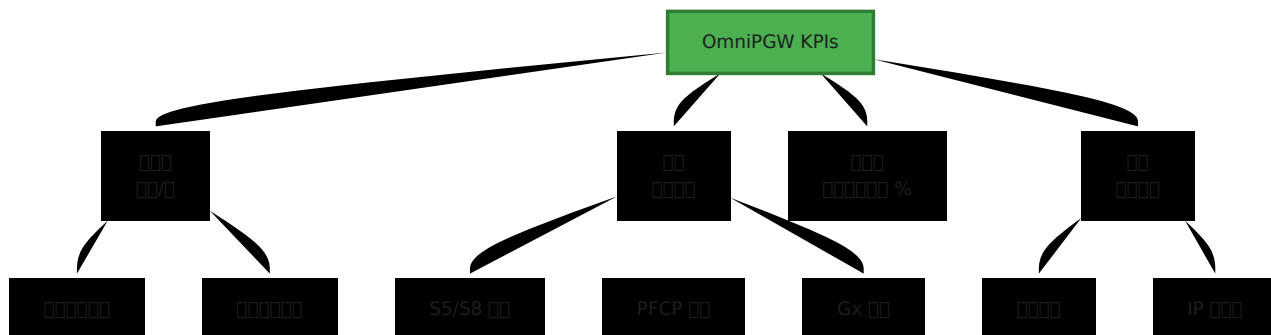
receivers:
  - name: 'ops-team'
    email_configs:
      - to: 'ops@example.com'

  - name: 'slack'
    slack_configs:
      - api_url:
'https://hooks.slack.com/services/YOUR/SLACK/WEBHOOK'
        channel: '#omnipgw-alerts'
        title: 'OmniPGW ☐☐☐{{ .GroupLabels.alertname }}'
        text: '{{ range .Alerts }}{{ .Annotations.description }}{{
end }}'

  - name: 'pagerduty'
    pagerduty_configs:
      - service_key: 'YOUR_PAGERDUTY_KEY'
```

□□□□

□□□□□□ (KPIs)



□□□□□

□□□□□□

```
rate(s5s8_inbound_messages_total{message_type="create_session_request"
[5m] )
```

□□□□□□

```
rate(s5s8_inbound_messages_total{message_type="delete_session_request"
[5m] )
```

□□□□□□

```
rate(s5s8_inbound_messages_total{message_type="create_session_request"
[5m] ) -
rate(s5s8_inbound_messages_total{message_type="delete_session_request"
[5m] )
```

□□□□

□□□□□□□□□□□□

```
# p50
histogram_quantile(0.50,
  rate(s5s8_inbound_handling_duration_bucket[5m])
)

# p95
histogram_quantile(0.95,
  rate(s5s8_inbound_handling_duration_bucket[5m])
)

# p99
histogram_quantile(0.99,
  rate(s5s8_inbound_handling_duration_bucket[5m])
)
```

スケーリング

```
histogram_quantile(0.95,
  rate(s5s8_inbound_handling_duration_bucket[5m])
) by (request_message_type)
```

スケーリング

スケーリング **24**時間

```
teid_registry_count -
teid_registry_count offset 24h
```

スケーリング

```
# スケーリング 10,000 回
10000 - teid_registry_count
```

スケーリング

```
# 1 10000000000
(10000 - teid_registry_count) /
(rate(teid_registry_count[1h]) * 86400)
```

10000000000

10000

10000000000

1000

```
rate(s5s8_inbound_errors_total[5m]) by (message_type)
```

1000

- 10000000000
- 100 PCRF 10000Gx 1000
- 100 IP 1000

10000000000

1000

```
histogram_quantile(0.95,
rate(s5s8_inbound_handling_duration_bucket{request_message_type="crea
[5m])
)
```

1000

- 100 Gx 1000PCRF 1000000
- 100 PFCP 1000PGW-U 1000000

- 配置参数

配置**PCRF** 参数

配置

```
# 配置 Gx 参数
sum(rate(gx_outbound_responses_total{result_code_class!="2xxx"}
[5m])) /
sum(rate(gx_outbound_responses_total[5m])) * 100

# 配置 PCRF 参数
sum(rate(gx_outbound_responses_total[5m])) by (diameter_host,
result_code_class)

# 配置参数
rate(gx_outbound_responses_total{result_code_class="5xxx"}[5m]) by
(diameter_host)
```

配置

- 配置 PCRF 参数
- 配置 PCRF 参数5xxx 参数
- 配置 Diameter 参数
- 配置 PCRF 参数
- 配置 5012DIAMETER_UNABLE_TO_COMPLY Re-Auth-Request 参数

配置参数

配置

```
# 配置参数
rate(vm_memory_total[1h])

# 配置参数
rate(vm_memory_processes[1h])

# 配置参数
rate(vm_system_process_count[1h])
```


□□□

- □□□□□□
- □□□□□□
- □□□□□□□□□□

□□□□

□□□□□□□□□□

```
max_over_time(teid_registry_count[24h])
```

□□□□□□□□

```
teid_registry_count /  
avg_over_time(teid_registry_count[7d])
```

□□□□□

```
abs(  
    teid_registry_count -  
    avg_over_time(teid_registry_count[1h])  
) > 100
```

□□□□

□□□□

1. □□□□□ 15-30 □□□□□□□□□□
2. □□□ 15 □□□□□□□□□□
3. □□□ □□□□□□□□□□□□□□□□□□□

Network

1. Network - NOC KPI
2. Network - Performance
3. Network - Security

Cloud

1. Cloud - Infrastructure
2. Cloud - VM → Container
3. Cloud - Kubernetes

Security

Network

- **Network** - Prometheus OAM REST API
- **Network** - Security

Cloud

- **PFCP** - PFCP UPF
- **Diameter Gx** - Gx PCRF
- **Diameter Gy** - Gy OCS
- **S5/S8** - GTP-C SGW-C

Security

- **P-CSCF** - P-CSCF IMS
- **Network** - Security
- **UE IP** - IP

Network

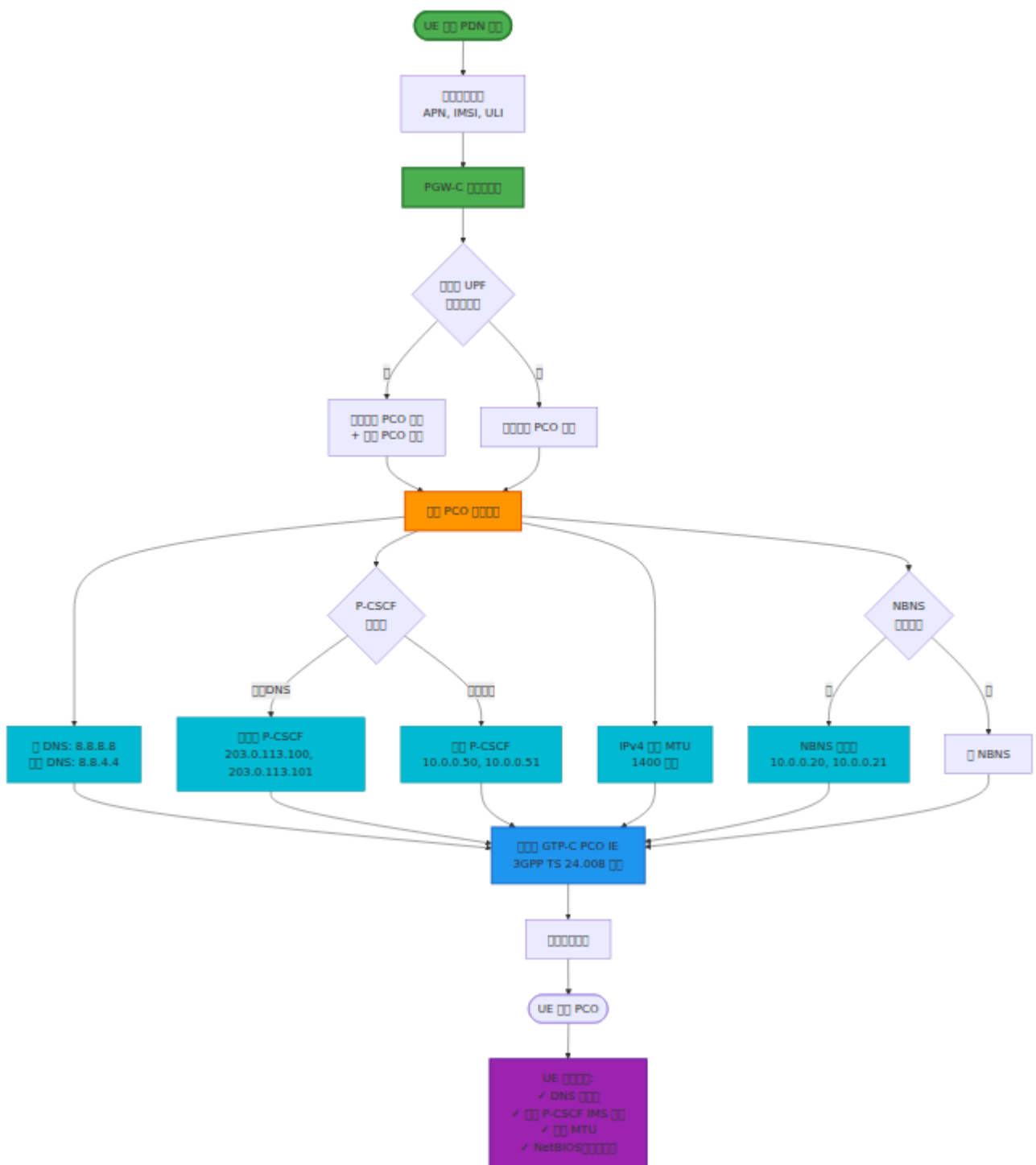
OmniPGW □□□□ - □ *Omnitouch* □□□□□□

PCO (PCO)

UE UE

OmniPGW OmniTouch

PCO (PCO) PDN UE (UE) UE DNS
IMS



OmniPGW PCO (MS → MS)

IE 名称	IE ID	描述	类型
DNS 服务器 IPv4 地址	0x000D	IP DNS	必需
DNS 服务器 IPv4 地址	0x000D	IP DNS	必需
DNS 服务器 IPv6 地址	0x0003	IP/IPv6 DNS (IPv6)	必需
P-CSCF IPv4 地址	0x000C	IMS 中 P-CSCF (IPv4 地址)	必需 (IMS)
P-CSCF IPv6 地址	0x0001	IMS 中 P-CSCF (IPv6 地址)	必需 (IMS)
IPv4 地址 MTU	0x0010	MTU 值	必需

IPCP 中 DNS/NBNS 的 UE 使用 IPCP (0x8021) 消息。DNS 服务器 DNS (NBNS/WINS) 消息。OmniPGW 使用 IPCP 消息。ms_dns1/ms_dns2 和 ms_wins1/ms_wins2 (RFC 1877) 消息。NBNS/WINS 消息。IPCP 消息——NBNS PCO 消息 (0x0011) 消息。PGW 消息。UE 使用 0.0.0.0 消息。IPCP 消息 (RFC 1661 §5)。

IPv4 和 IPv6 P-CSCF / DNS 消息 E-CSCF 消息。OmniPGW 使用 IPv4 和 IPv6 P-CSCF 消息。IPv4 消息 0x000C 和 IPv6 消息 0x0001。IPv6 P-CSCF 消息 (p_cscf_ipv6_address_request) 消息。p_cscf_ipv6_address_list (DNS/AAAA 消息 IPv6 消息) 消息。IPv6 DNS 消息 (dns_server_ipv6_address_request) 消息 0x0003 消息。primary_dns_server_ipv6_address / secondary_dns_server_ipv6_address 消息。E-CSCF 消息。APCO (PCO) 消息。

□□

□□□□

```
# config/runtime.exs
config :pgw_c,
  pco: %{
    # DNS □□ (□□)
    primary_dns_server_address: "8.8.8.8",
    secondary_dns_server_address: "8.8.4.4",

    # IPv6 DNS □□ (□□□□□□ 0x0003)
    primary_dns_server_ipv6_address: nil,
    secondary_dns_server_ipv6_address: nil,

    # NBNS □□ (□□□□□□ Windows □□)
    primary_nbns_server_address: nil,
    secondary_nbns_server_address: nil,

    # IMS/VoLTE □ P-CSCF □□ (□□)
    p_cscf_ipv4_address_list: [], # □□ 0x000C
    p_cscf_ipv6_address_list: [], # □□ 0x0001

    # P-CSCF □□□□ (□□□□□□ A □ AAAA □□)
    p_cscf_discovery_enabled: false,
    p_cscf_discovery_dns_server: nil,
    p_cscf_discovery_timeout_ms: 5000,

    # IPv4 MTU □□ (□□)
    ipv4_link_mtu_size: 1400
  }
```

PCO □□

DNS □□□□□

□ **DNS** □□□ **DNS**□

```
pco: %{\n  primary_dns_server_address: "8.8.8.8",\n  secondary_dns_server_address: "8.8.4.4"\n}
```

DNS Servers

Server	Primary	Secondary
Google	8.8.8.8	8.8.4.4
Cloudflare	1.1.1.1	1.0.0.1
Quad9	9.9.9.9	149.112.112.112
OpenDNS	208.67.222.222	208.67.220.220

DNS

```
pco: %{\n  primary_dns_server_address: "10.0.0.10",\n  secondary_dns_server_address: "10.0.0.11"\n}
```

P-CSCF (IMS)

IMS/VoLTE


```
pco: %{
  p_cscf_ipv4_address_list: [
    "10.0.0.50", # 1 P-CSCF (IPv4, 0x000C)
    "10.0.0.51" # 2 P-CSCF
  ],
  # IPv6 P-CSCF 1 (0x0001) 1 IPv6/IPv4v6 IMS PDN
  p_cscf_ipv6_address_list: [
    "2001:db8::50",
    "2001:db8::51"
  ]
}
```

UE 1—IPv4 P-CSCF 0x000C 1IPv6 P-CSCF 0x0001 1 IMS PDN

P-CSCF ()

- IMS
- VoLTEVoWiFiRCS
- UE SIP

P-CSCF

DNS P-CSCF

OmniPGW DNS P-CSCF 3GPP TS 23.003 TS 24.229 PGW-C DNS P-CSCF

```
pco: %{
  # 开启 P-CSCF 发现
  p_cscf_discovery_enabled: true,

  # P-CSCF 发现 DNS 服务器 (IP 地址)
  p_cscf_discovery_dns_server: {10, 179, 2, 177},

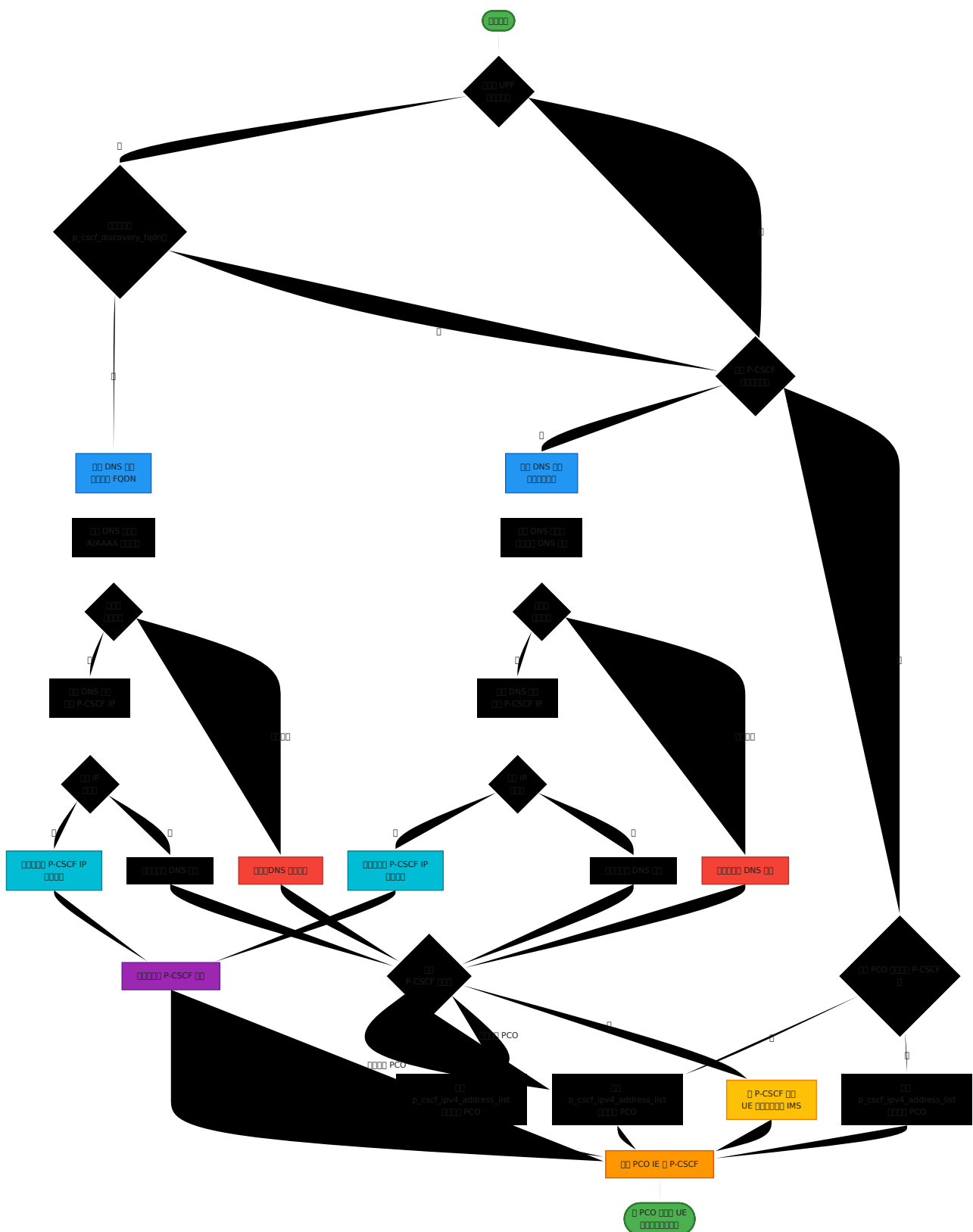
  # DNS 超时时间 (ms)
  p_cscf_discovery_timeout_ms: 5000,

  # 指定 P-CSCF IP (指定 DNS 服务器)
  p_cscf_ipv4_address_list: ["10.0.0.50"]
}
```

配置说明

1. 将 `p_cscf_discovery_enabled: true` 配置到 PGW-C 的 DNS 配置中，开启 P-CSCF 发现
2. DNS 服务器地址配置 `p_cscf_discovery_dns_server`
3. 指定 DNS 服务器地址，P-CSCF 通过 PCO 通知 UE
4. 指定 DNS 服务器地址列表 `p_cscf_ipv4_address_list`
5. 指定 P-CSCF IP 地址列表

P-CSCF ☐☐☐☐



□ □ □ □ □ □

1. **FQDN** (完全限定域名) - `p_cscf_discovery_fqdn` 是 UPF 的 FQDN
2. **DNS** (域名系统) - `p_cscf_discovery_enabled: true` 表示 PCO 中启用了 DNS

3. **PCO** `p_cscf_ipv4_address_list` PCO
4. **PCO** () - `p_cscf_ipv4_address_list` PCO

P-CSCF

- DNS /
-
-
-

P-CSCF

<code>p_cscf_discovery_enabled</code>		<code>false</code>	DNS P-CSCF
<code>p_cscf_discovery_dns_server</code>	(IP)	<code>nil</code>	DNS IP 4 ({10, 179, 2, 177})
<code>p_cscf_discovery_timeout_ms</code>		<code>5000</code>	DNS

- IMS** - P-CSCF DNS
- DNS P-CSCF
- DNS P-CSCF
- P-CSCF

DNS IMS

```
pco: %{
  primary_dns_server_address: "10.0.0.10",
  secondary_dns_server_address: "10.0.0.11",

  # P-CSCF
  p_cscf_discovery_enabled: true,
  p_cscf_discovery_dns_server: {10, 179, 2, 177}, # IMS DNS
  p_cscf_discovery_timeout_ms: 3000,

  # P-CSCF (DNS)
  p_cscf_ipv4_address_list: [
    "10.0.0.50", #
    "10.0.0.51" #
  ],

  ipv4_link_mtu_size: 1400
}
```

P-CSCF

P-CSCF UPF APN DNS P-CSCF

```
# upf_selection
rules: [
  %{
    name: "IMS",
    priority: 20,
    match_field: :apn,
    match_regex: "^ims",
    upf_pool: [...],

    # P-CSCF
    p_cscf_discovery_fqdn: "pcscf.mnc001.mcc001.3gppnetwork.org"
  }
]
```

UPF P-CSCF

P-CSCF P-CSCF

NBNS 配置 (NetBIOS)

Windows 配置

```
pco: %{\n  primary_nbns_server_address: "10.0.0.20",\n  secondary_nbns_server_address: "10.0.0.21"\n}
```

配置

- Windows 配置
- NetBIOS 配置

MTU 配置

配置

```
pco: %{\n  ipv4_link_mtu_size: 1400 # \n}
```

MTU 配置

MTU	配置
1500	以太网 (以太网)
1400	以太网 GTP 配置
1420	以太网
1280	IPv6 配置 MTU
1360	VPN/以太网

□□□ □□ LTE □□ **1400** □□□ GTP-U □□□

□□□□

Internet APN

```
pco: %{  
  primary_dns_server_address: "8.8.8.8",  
  secondary_dns_server_address: "8.8.4.4",  
  ipv4_link_mtu_size: 1400  
}
```

IMS APN

```
pco: %{  
  primary_dns_server_address: "10.0.0.10",  
  secondary_dns_server_address: "10.0.0.11",  
  p_cscf_ipv4_address_list: [  
    "10.0.0.50",  
    "10.0.0.51"  
  ],  
  ipv4_link_mtu_size: 1400  
}
```

□□ **P-CSCF** □□ □□□ IMS □□□□□□ P-CSCF □□□□

APN

```
pco: %{
  primary_dns_server_address: "10.100.0.10",
  secondary_dns_server_address: "10.100.0.11",
  primary_nbns_server_address: "10.100.0.20",
  secondary_nbns_server_address: "10.100.0.21",
  ipv4_link_mtu_size: 1400
}
```

PCO □ GTP-C □□□□□□

□ □ □ □ □ □

OmniPGW □ □□□□□ □□□□□ PCO□

```

| | | | |
├─ IP: | | | | |
├─ UE IP | |: 100.64.1.42
├─ PC0 (| | | | |)
│   ├─ DNS | | IPv4 | |: 8.8.8.8
│   ├─ DNS | | IPv4 | |: 8.8.4.4
│   ├─ P-CSCF IPv4 | |: 10.0.0.50
│   ├─ P-CSCF IPv4 | |: 10.0.0.51
│   └─ IPv4 | | MTU: 1400

```

UE ☐ ☐

UE □□ PCO □□

1. 配置 DNS 参数
2. 配置 P-CSCF 和 IMS 参数
3. 配置 MTU 参数

背景

UE 网络 DNS

背景

- UE 的 IP 地址由 DHCP 服务器分配
- DNS 服务器

网络

1. PCO 中的 DNS 服务器地址
2. 从 UE IP 地址的 DNS 服务器
3. 从 DNS 服务器

网络

```
# 检查 DNS 服务器地址
ping 8.8.8.8

# 检查 UE 中的 DNS 服务器
nslookup google.com 8.8.8.8

# 检查 PCO 中的 DNS 服务器
grep "primary_dns_server_address" config/runtime.exs
```

IMS 网络

背景

- VoLTE 网络
- UE 中的 IMS 网络

网络

1. 检查 P-CSCF 地址
2. P-CSCF IP 地址
3. P-CSCF 网络

配置

```
# P-CSCF 配置
pco: %{
  p_cscf_ipv4_address_list: ["10.0.0.50"] # 配置
}
```

配置MTU

配置

- 配置
- 配置
- 配置

配置

- MTU 配置
- MTU 配置

配置

```
# GTP 配置 1400
pco: %{
  ipv4_link_mtu_size: 1400
}

# 配置
pco: %{
  ipv4_link_mtu_size: 1360
}
```

□□□□

DNS □□

1. □□□□ **DNS** □□
 - □□□Google (8.8.8.8)□Cloudflare (1.1.1.1)
 - □□□□□ DNS
2. □□□□□
 - □□□□
 - □□□□□
3. □□ **DNS** □□□
 - □□ DNSSEC □□□□
 - □□□□□ DNS □□

IMS □□

1. □□□□ **P-CSCF**
 - □□ 2 □□□□□□
 - □□□□□□□□□□
2. □□□□□
 - P-CSCF □□□□□ UE IP □□□
 - □□ SIP □□□

MTU □□

1. □□□□
 - GTP-U□36 □□ (IPv4)
 - IPsec□□□ (50-100 □□)

2. **LTE MTU**

- **1400**
-

3.

- **MTU**
-

- **runtime.exs** - **UPF** **PCO**
- **UE IP** - IP **APN**
- **P-CSCF** - P-CSCF

- **PDN**
- **S5/S8** - GTP-C **PCO**
- **PFCP** -

IMS VoLTE

- **Diameter Gx** - IMS
- **PCO**

OmniPGW PCO - *Omnitouch*

P-CSCF 部署方案

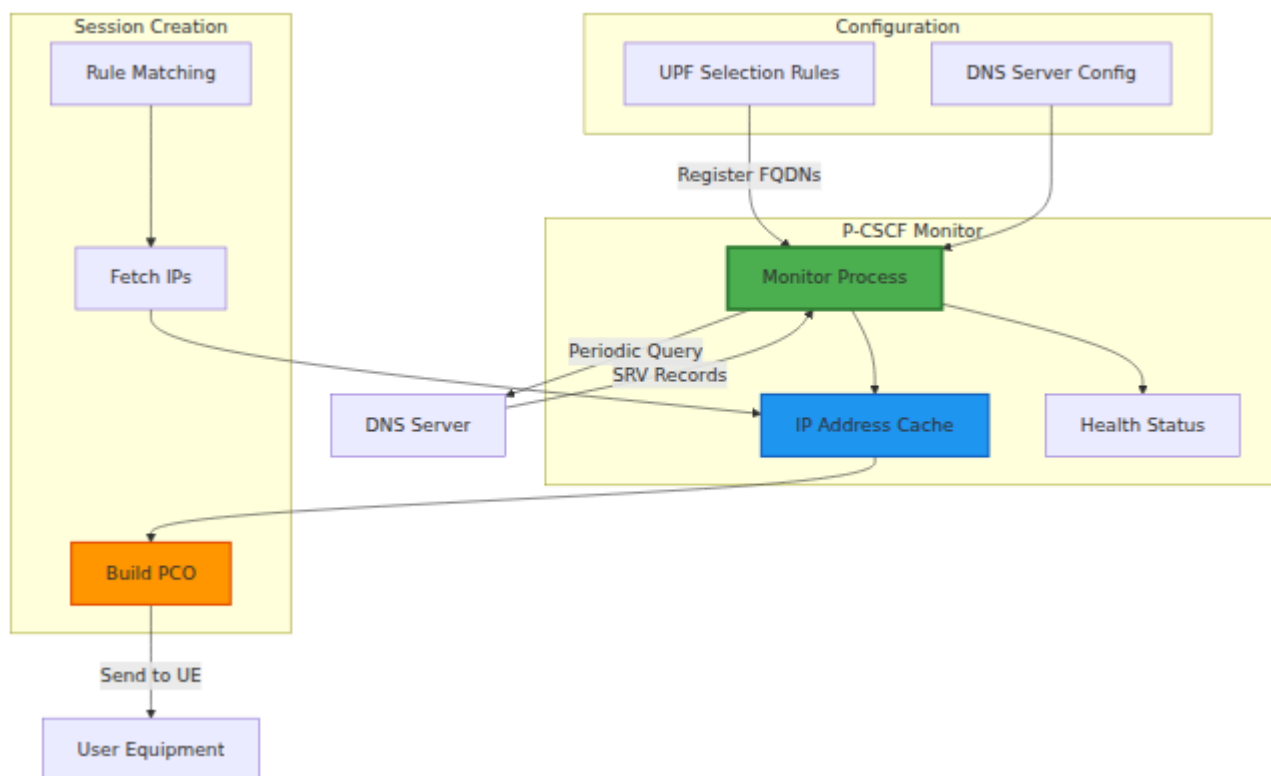
部署 P-CSCF 部署方案

OmniPGW 与 Omnitouch 部署方案

部署

P-CSCF (部署方案) 部署方案 部署 DNS SRV 部署 SIP OPTIONS 部署 IMS P-CSCF 部署方案

- 部署 **P-CSCF** 部署方案 P-CSCF 部署
- 部署方案 DNS 部署 60 部署
- SIP OPTIONS** 部署 SIP OPTIONS ping 部署 P-CSCF 部署
 - 部署 **TCP**部署 TCP 部署 SIP OPTIONS部署
 - UDP** 部署 TCP 部署 UDP
 - 部署 :up 部署 :down
- 部署 OAM API 部署 IP 部署
- 部署
- Prometheus** 部署 Prometheus 部署



□□

1. □□□□
2. □□
3. □□□□
4. OAM API □□
5. □□□□□□□□
6. □□□□
7. DNS □□
8. □□□□
9. □□□□

□□□□

□□□□

```
# config/runtime.exs
```

```
# □□ PCO □□□□ P-CSCF □□ DNS □□□□
```

```
config :pgw_c,
```

```
  pco: %{
```

```
    p_cscf_discovery_dns_server: "192.0.2.177",
```

```
    p_cscf_discovery_enabled: true,
```

```
    p_cscf_discovery_timeout_ms: 5000
```

```
  },
```

```
  upf_selection: %{
```

```
    rules: [
```

```
      # IMS □□ - □□ P-CSCF □□
```

```
      %{
```

```
        name: "IMS Traffic",
```

```
        priority: 20,
```

```
        match_field: :apn,
```

```
        match_regex: "^ims",
```

```
        upf_pool: [
```

```
          %{remote_ip_address: "10.100.2.21", remote_port: 8805,
```

```
weight: 80}
```

```
        ],
```

```
      # P-CSCF □□ FQDN□□□□□ UPF □□□□□□□□□□□□
```

```
      p_cscf_discovery_fqdn:
```

```
"pcscf.mnc001.mcc001.3gppnetwork.org",
```

```
      # □□□□□□□□ PCO □□□□□
```

```
      pco: %{
```

```
        p_cscf_ipv4_address_list: ["203.0.113.100",
```

```
"203.0.113.101"]
```

```
      }
```

```
    }
```

```
  ]
```

```
}
```

□□□ □□□□ □□□□□□ UPF □□□□□□□□□□ PCO □□ □□□□□ P-CSCF □□□□□

準備

1. 準備 OmniPGW
 2. 準備 OAM API `curl -k https://localhost:8443/api/pcscf_monitor`
 3. 準備 DNS 伺服器 IP
-

設定

準備 P-CSCF 設定

準備 PCO 設定 P-CSCF DNS 設定

```
pco: %{
  # 準備 P-CSCF DNS 設定 UE 的 DNS 設定
  p_cscf_discovery_dns_server: "192.0.2.177",

  # 準備 P-CSCF DNS 設定
  p_cscf_discovery_enabled: true,

  # DNS SRV 設定
  p_cscf_discovery_timeout_ms: 5000,

  # 準備 P-CSCF IP 地址列表
  p_cscf_ipv4_address_list: ["203.0.113.146"]
}
```

準備 P-CSCF FQDN

準備 UPF 設定 P-CSCF FQDN


```

upf_selection: %{
  rules: [
    # IMS - IMS P-CSCF
    %{
      name: "IMS Traffic",
      match_field: :apn,
      match_regex: "^ims",
      upf_pool: [...],
      p_cscf_discovery_fqdn:
"pcscf.ims.mnc001.mcc001.3gppnetwork.org",
      pco: %{
        p_cscf_ipv4_address_list: ["203.0.113.100"] #
      }
    },

    # - P-CSCF
    %{
      name: "Enterprise Traffic",
      match_field: :apn,
      match_regex: "^enterprise",
      upf_pool: [...],
      p_cscf_discovery_fqdn: "pcscf.enterprise.example.com",
      pco: %{
        p_cscf_ipv4_address_list: ["192.168.1.50"] #
      }
    },

    # - P-CSCF
    %{
      name: "Internet Traffic",
      match_field: :apn,
      match_regex: "^internet",
      upf_pool: [...]
      # p_cscf_discovery_fqdn - PCO
    }
  ]
}

```

--	--	--	--

1.

- P-CSCF → GenServer →
- → UPF → P-CSCF FQDN

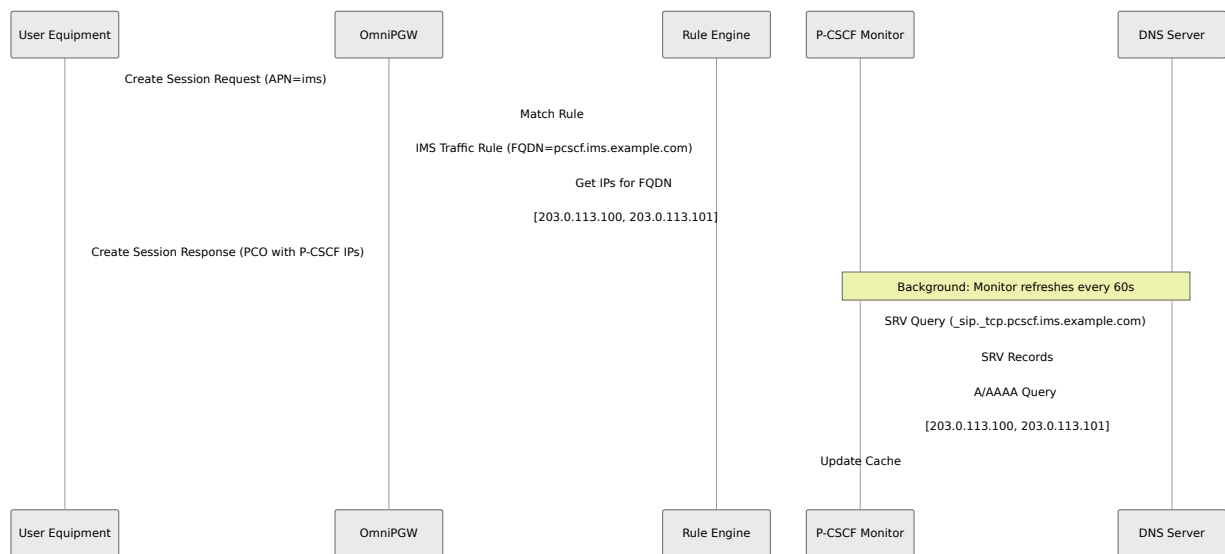
2. FQDN ☐

- 完全 FQDN 指定
- 部分 FQDN 指定 DNS SRV 記法
- **SIP OPTIONS** 拡張オプション
 - TCP SIP/2.0/TCP 5060
 - TCP 指定 UDP SIP/2.0/UDP 5060
 - 上向き :up 下向き :down 指定
- IP 指定

3. 每箱裝 60 罐

- 例題 FQDN
- DNS 例題
- 例題
 - TCP 例題 SIP OPTIONS 例題 5 例題
 - TCP 例題 UDP 例題 5 例題
 - 例題
- 例題 DNS 例題

Sequence Diagram



DNS

DNS SRV records are used to discover P-CSCF addresses

1. **SRV** records are used to discover P-CSCF addresses
2. SRV records are used to discover P-CSCF addresses
3. SRV records are used to discover P-CSCF addresses
4. SRV records are used to discover P-CSCF addresses
5. SRV records are used to discover P-CSCF addresses

P-CSCF

P-CSCF is used to discover P-CSCF addresses

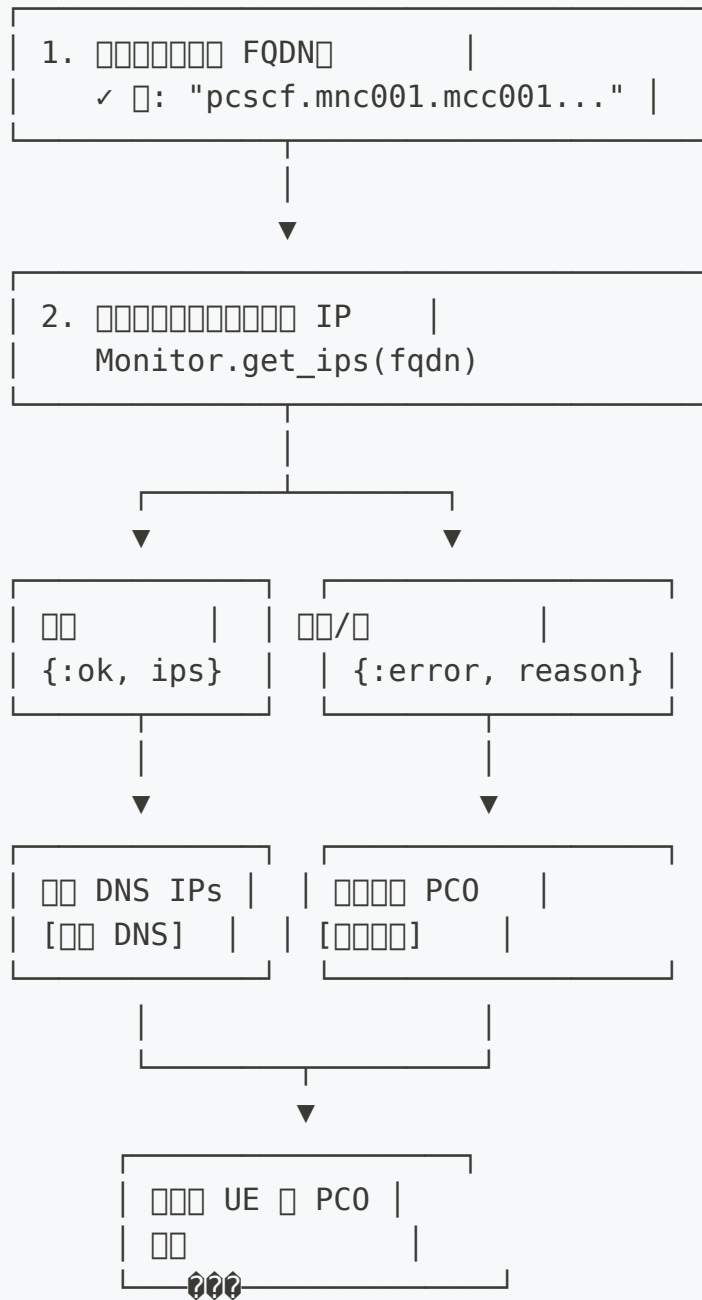
```
%{
  name: "IMS Traffic",
  p_cscf_discovery_fqdn: "pcscf.mnc001.mcc001.3gppnetwork.org", #
  ←
  pco: %{
    p_cscf_ipv4_address_list: ["203.0.113.100", "203.0.113.101"]
  }
  # ←
}
```

Table 1

Field	P-CSCF Type	IP Address	Value
FQDN Name	DNS Record	DNS Record IP	"FQDN pcscf.example.com P-CSCF Name"
FQDN Name	PCO Record	PCO pco.p_cscf_ipv4_address_list Record IP	"FQDN P- CSCF IP"
FQDN Name	PCO Record	PCO pco.p_cscf_ipv4_address_list Record IP	
IP Address	PCO Record	PCO pco.p_cscf_ipv4_address_list Record IP	
IP Address FQDN	PCO Record	IP Address	

Table 2

IMS 鉴权流程



鉴权

1. DNS 鉴权

####

```
p_cscf_discovery_fqdn: "pcscf.ims.example.com"
pco.p_cscf_ipv4_address_list: ["203.0.113.100"]
```

DNS ####[203.0.113.150, 203.0.113.151]

UE ####[203.0.113.150, 203.0.113.151] ← #### DNS

DNS ##### PC0

2####DNS ##### ⚠

####

```
p_cscf_discovery_fqdn: "pcscf.ims.example.com"
pco.p_cscf_ipv4_address_list: ["203.0.113.100"]
```

DNS ##### :no_naptr_records

UE ####[203.0.113.100] ← ##### PC0

DNS

3##### FQDN

####

```
# #### p_cscf_discovery_fqdn
pco.p_cscf_ipv4_address_list: ["192.168.1.50"]
```

UE ####[192.168.1.50] ← ##### PC0

DNS

#####

1. #####DNS #####
2. ##### DNS #####
3. #####
4. #####

FQDN #### PC0

```
# ✓ 000000000000
%{
  p_cscf_discovery_fqdn: "pcscf.ims.example.com", # 
  pco: %{
    p_cscf_ipv4_address_list: ["203.0.113.100"] # 
  }
}

# △ 000000000000 PC0
%{
  p_cscf_discovery_fqdn: "pcscf.ims.example.com"
  # 
}

# ✓ 000000000000 DNS 
%{
  pco: %{
    p_cscf_ipv4_address_list: ["192.168.1.50"]
  }
}
```

OAM API

P-CSCF

OAM REST API HTTPS/TLS 8443 /api P-CSCF

```
GET /api/pcscf_monitor
```

```
curl -k https://localhost:8443/api/pcscf_monitor
```

API Swagger UI

<https://localhost:8443/api/docs>

- 設定
 - 設定 FQDN 設定
 - 設定 FQDN 設定
 - 設定
 - 設定 P-CSCF IP 設定
- 設定 **FQDN** 設定
 - 設定 FQDN
 - 設定 / 設定 / 設定
 - 設定 IP 設定
 - 設定 IP 設定
 - 設定
 - 設定
 - IP 設定
 - 設定 DNS SRV 設定
 - 設定 / 設定

- 設定
 - 設定 **FQDN** 設定 FQDN 設定
 - 設定 DNS 設定 FQDN
 - 設定 **DNS** 設定 FQDN
 - 設定 **P-CSCF** 設定 FQDN 設定
 - 設定 **SIP OPTIONS** 設定 SIP OPTIONS 設定
 - 設定 **SIP OPTIONS** 設定 SIP OPTIONS 設定
 - **DNS** 設定 DNS 設定
 - 設定 SIP OPTIONS 設定 60 設定 5 設定

設定 DNS 設定 SIP OPTIONS 設定 P-CSCF 設定 GET
 /api/pcscf_monitor — 設定

UPF 設定

UPF 設定 GET /api/upf_selection 設定 P-CSCF 設定


```
❑ IMS 配置 20
    配置APN 为 ^ims
    配置UPF-IMS-Primary (10.100.2.21:8805)

❑ P-CSCF 配置
    FQDN: pcscf.mnc001.mcc001.3gppnetwork.org
    检查: ✓ 配置2 个 IP
    配置 IP: 203.0.113.100, 203.0.113.101

⚙️ PCO 配置
    DNS: 198.51.100.195
    P-CSCF配置为203.0.113.100, 203.0.113.101
```

配置配置配置

Prometheus 配置

P-CSCF 配置 Prometheus 配置 42069 配置配置

Gauge 配置

```
# FQDN 配置
pcscf_fqdns_total          # 配置 FQDN 配置
pcscf_fqdns_resolved       # 配置 FQDN 配置 DNS 配置
pcscf_fqdns_failed        # 配置 FQDN 配置 DNS 配置

# 配置配置配置配置
pcscf_servers_total       # 配置 DNS SRV 配置 P-CSCF 配置
#
pcscf_servers_healthy     # 配置 SIP OPTIONS 配置配置
pcscf_servers_unhealthy   # 配置 SIP OPTIONS 配置配置

# 配置配置配置 FQDN 配置
pcscf_servers_healthy{fqdn="..."} # 配置 FQDN 配置配置
pcscf_servers_unhealthy{fqdn="..."} # 配置 FQDN 配置配置
```

配置配置配置

- **healthy** 成功した SIP OPTIONS ping (TCP or UDP)
- **unhealthy** 失敗した SIP OPTIONS (5 分間隔)

まとめ

DNS 監視

```
# 監視対象 FQDN
pcscf_fqdns_resolved

# DNS 成功率
(pcscf_fqdns_resolved / pcscf_fqdns_total) * 100

# 監視対象数
pcscf_servers_total
```

SIP OPTIONS 監視

```
# SIP FQDN 監視対象
pcscf_servers_healthy

# 監視対象数
pcscf_servers_unhealthy

# 監視対象数
(pcscf_servers_healthy / pcscf_servers_total) * 100

# SIP FQDN 監視対象
pcscf_servers_healthy{fqdn="pcscf.mnc001.mcc001.3gppnetwork.org"}

# 監視対象数
pcscf_servers_healthy == 0 AND pcscf_servers_total > 0
```

Prometheus 監視

```

# [] P-CSCF []
- alert: AllPCSCFServersDown
  expr: pcscf_servers_healthy == 0 AND pcscf_servers_total > 0
  for: 5m
  labels:
    severity: critical
  annotations:
    summary: "[] P-CSCF []"
    description: "{{ $value }}" - [] SIP OPTIONS []

# [] 50% []
- alert: MajorityPCSCFServersDown
  expr: (pcscf_servers_healthy / pcscf_servers_total) < 0.5
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "[] P-CSCF []"
    description: "[] {{ $value }}% [] SIP OPTIONS"

# [] DNS []
- alert: PCSCFDNSResolutionFailed
  expr: pcscf_fqdns_failed > 0
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "P-CSCF DNS []"
    description: "{{ $value }}" FQDN []

```

[] []

[][] [] []

```
[info] P-CSCF Monitor started
[info] Registering 2 unique P-CSCF FQDNs for monitoring:
["pcscf.ims.example.com", "pcscf.enterprise.example.com"]
[info] P-CSCF Monitor: Registering FQDN pcscf.ims.example.com
[debug] P-CSCF Monitor: Successfully resolved
pcscf.ims.example.com to 2 IPs
[warning] P-CSCF Monitor: Failed to resolve
pcscf.enterprise.example.com: :nxdomain
[debug] Using P-CSCF addresses from FQDN pcscf.ims.example.com:
[{10, 101, 2, 100}, {10, 101, 2, 101}]
```

配置

配置 P-CSCF 发现 FQDN

1 DNS 配置

```
p_cscf_discovery_fqdn: "pcscf.ims.example.com"
```

- 配置 DNS 服务器 IP
- 配置 DNS 服务器 IP
- 配置 DNS 服务器 IP

2 配置 PCO

```
pco: %{
  p_cscf_ipv4_address_list: ["203.0.113.100", "203.0.113.101"]
}
```

- 配置 DNS 服务器 IP
- 配置 DNS 服务器 IP
- 配置 DNS 服务器 IP

3. PCO 配置

```
# PCO 配置
pco: %{
  p_cscf_ipv4_address_list: ["203.0.113.146"]
}
```

- DNS 配置
- P-CSCF 配置
- 其他配置

配置

```
配置 "IMS 配置" 配置

1. DNS 配置 "pcscf.ims.example.com"
   ├── IP → IP [203.0.113.100, 203.0.113.101] ✓
   └── IP → 配置

2. 配置 PCO 配置
   ├── IP → IP [203.0.113.100, 203.0.113.101] ✓
   └── IP → 配置

3. 配置 PCO 配置
   └── IP [203.0.113.146] ✓配置
```

DNS 配置

DNS 配置

DNS 配置 P-CSCF 配置 SRV 和 A/AAAA 配置

```

; P-CSCF SRV _sip._tcp
_sip._tcp.pcscf.mnc001.mcc001.3gppnetwork.org. IN SRV 10 50 5060
pcscf1.example.com.
_sip._tcp.pcscf.mnc001.mcc001.3gppnetwork.org. IN SRV 20 50 5060
pcscf2.example.com.

; A
pcscf1.example.com. IN A 203.0.113.100
pcscf2.example.com. IN A 203.0.113.101

```

OmniPGW FQDN `_sip._tcp.` `p_cscf_discovery_fqdn:`
`"pcscf.mnc001.mcc001.3gppnetwork.org"`
`_sip._tcp.pcscf.mnc001.mcc001.3gppnetwork.org`

SRV

SRV

```

_service._proto.domain. IN SRV priority weight port target.

```

- 10 ~ 20
- =
- SIP TCP 5060 UDP 5060
- IP

🔍 DNS 🔍

```
# 🔍 SRV 🔍 🔍 _sip._tcp 🔍  
dig SRV _sip._tcp.pcscf.mnc001.mcc001.3gppnetwork.org @192.0.2.177  
  
# 🔍 🔍  
# _sip._tcp.pcscf.mnc001.mcc001.3gppnetwork.org. 300 IN SRV 10 50  
5060 pcscf1.example.com.  
  
# 🔍 P-CSCF 🔍 🔍 IP  
dig A pcscf1.example.com @192.0.2.177  
  
# 🔍 🔍  
# pcscf1.example.com. 300 IN A 203.0.113.100
```

🔍 🔍 🔍 🔍

🔍 🔍 **FQDN** 🔍 “🔍” 🔍

🔍 🔍

- GET /api/pcscf_monitor 🔍 🔍 🔍 🔍
- 🔍 :nxdomain 🔍 :timeout 🔍 :no_naptr_records

🔍 🔍 🔍 🔍

1. DNS 🔍 🔍 🔍 🔍
2. FQDN 🔍 DNS 🔍 🔍 🔍
3. 🔍 🔍 NAPTR 🔍
4. DNS 🔍 🔍 🔍 🔍

🔍 🔍 🔍 🔍

```
# 1. DNS 테스트
ping 192.0.2.177

# 2. NAPTR 테스트
dig NAPTR pcscf.mnc001.mcc001.3gppnetwork.org @192.0.2.177

# 3. OmniPGW 로그
grep "P-CSCF" /var/log/pgw_c.log

# 4. 테스트
grep "p_cscf_discovery_dns_server" config/runtime.exs

# 5. OAM API 테스트
curl -k https://localhost:8443/api/pcscf_monitor
```

테스트 IP

테스트

- GET /api/pcscf_monitor 테스트 "0 IP"
- 테스트 결과 확인

테스트

1. NAPTR 테스트 결과 FQDN 확인
2. 테스트 IMS/SIP 테스트
3. 테스트 A/AAAA 테스트

테스트

```
# 테스트 NAPTR 테스트
dig NAPTR pcscf.example.com @192.0.2.177

# 테스트 "SIP" 테스트 "IMS":
# 테스트 "SIP+D2U", "x-3gpp-ims:sip"
# 테스트 "HTTP", "FTP"

# 테스트 A/AAAA 테스트
dig pcscf1.example.com A @192.0.2.177
```


🔍🔍🔍🔍🔍🔍 P-CSCF

🔍🔍

- UE 📶📶 P-CSCF 📶
- 📶📶📶📶📶📶 IP

🔍🔍🔍🔍

1. DNS 📶📶📶📶📶🔍
2. 📶📶📶🔍
3. FQDN 📶🔍

🔍🔍🔍🔍

```
# 1. 📶 P-CSCF 📶🔍
# 📶 FQDN 📶📶📶📶🔍

# 2. 📶📶📶🔍
grep "Using P-CSCF addresses from FQDN" /var/log/pgw_c.log

# 3. 📶 UPF 📶🔍
# 📶📶📶📶📶📶 FQDN 📶📶🔍

# 4. 📶📶📶🔍
# 📶📶 APN 📶📶📶📶📶📶🔍
```

🔍🔍🔍 DNS 📶🔍

🔍🔍

- 📶📶📶🔍
- 📶📶🔍 `pcscf_discovery_query_duration_seconds`

🔍🔍🔍🔍

1. DNS 📶📶📶🔍
2. 📶 DNS 📶📶📶📶🔍
3. 📶📶📶🔍

配置

```
# 配置
pco: %{
  p_cscf_discovery_timeout_ms: 2000 # 5000ms
}

# 配置 DNS
pco: %{
  p_cscf_discovery_dns_server: "10.0.0.10" # DNS
}
```

配置

1. DNS 配置

配置 DNS

```
pco: %{
  # P-CSCF DNS UE DNS
  p_cscf_discovery_dns_server: "192.0.2.177",

  # UE DNS
  primary_dns_server_address: "8.8.8.8",
  secondary_dns_server_address: "8.8.4.4"
}
```

配置

- UE DNS IMS DNS
-
-

2. 配置 P-CSCF

```
%{
  p_cscf_discovery_fqdn: "pcscf.ims.example.com", # 配置
  pco: %{
    p_cscf_ipv4_address_list: ["203.0.113.100"] # 配置
  }
}
```

配置

- 配置 DNS 记录
- 配置
- 配置 SLA

3. 配置 P-CSCF FQDN

```
rules: [
  # IMS
  %{
    name: "IMS",
    match_regex: "^ims",
    p_cscf_discovery_fqdn:
"pcscf.ims.mnc001.mcc001.3gppnetwork.org"
  },

  # 配置
  %{
    name: "Enterprise",
    match_regex: "^enterprise",
    p_cscf_discovery_fqdn: "pcscf.enterprise.example.com"
  }
]
```

配置

- 配置 P-CSCF
- 配置

- 配置

4. 配置 DNS 告警

```
# 配置 P-CSCF 告警
alert: HighPCSCFQueryLatency
expr: histogram_quantile(0.95,
pcscf_discovery_query_duration_seconds_bucket) > 2
for: 5m
labels:
  severity: warning
annotations:
  summary: "P-CSCF DNS 第95百分位 > 2s"
```

5. 配置 DNS 监控

- **OAM API** 配置 GET /api/pcscf_monitor
- 配置 pcscf_monitor_fqdns_failed 告警
- 配置 DNS 告警
- 配置 DNS 告警

6. 配置告警

```
# 配置告警
pco: %{
  p_cscf_discovery_timeout_ms: 5000 # 5 秒
}

# 配置告警
pco: %{
  p_cscf_discovery_timeout_ms: 2000 # 2 秒
}
```

7. 配置 DNS 告警

配置 DNS 告警

```
# 1 P-CSCF DNS
pcscf.mnc001.mcc001.3gppnetwork.org. IN NAPTR 10 50 "s" "SIP+D2U"
"" _sip._udp.pcscf1.example.com.

# 2 P-CSCF DNS
pcscf.mnc001.mcc001.3gppnetwork.org. IN NAPTR 20 50 "s" "SIP+D2U"
"" _sip._udp.pcscf2.example.com.
```

Network

- **PCO** - DNS to P-CSCF
 - - OmniPGW
 - -
 - - PCO
 - **PFCP** -
-

Network

OmniPGW P-CSCF - *OmniTouch*

PFCP/Sxb

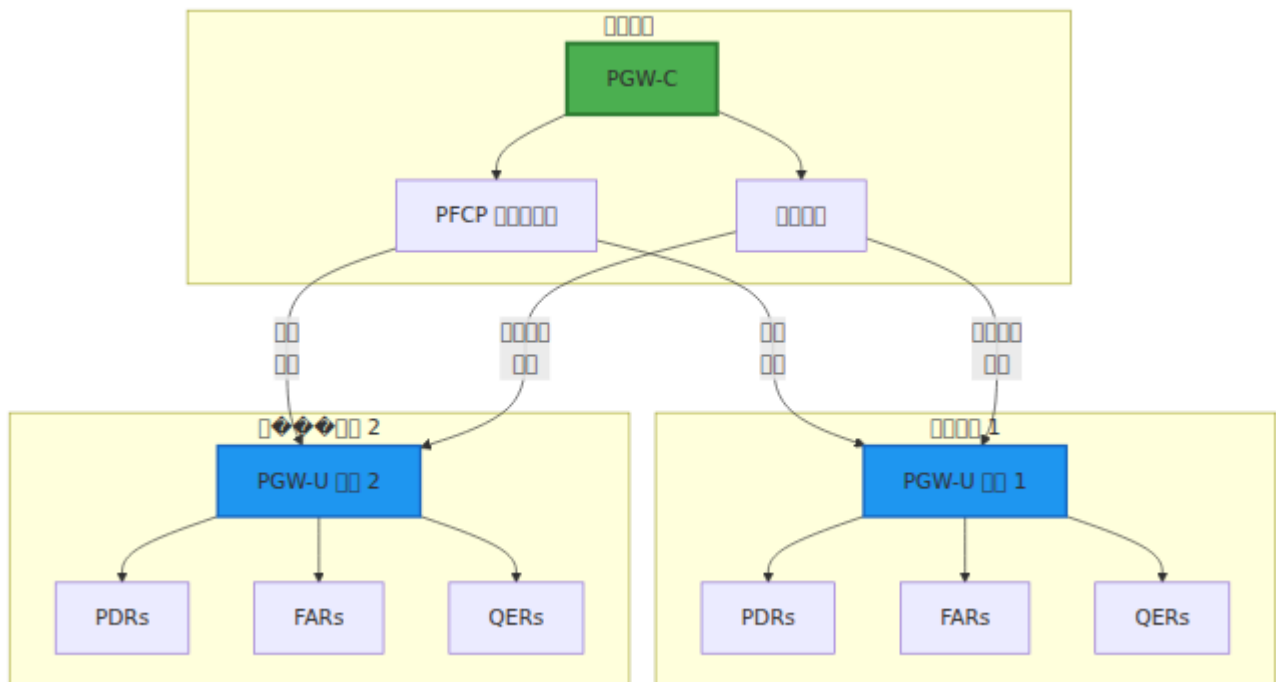
PGW-C - PGW-U

-
-
- PFCP
- PFCP
-
-
- DNS UPF
-
-
- PFCP
-

Sxb PGW-C PGW-U

- PGW-C
- PGW-U

PFCP



PFCP

PGW-C is **PFCP** 1 3GPP TS 29.244

- UDP
- 8805
- PFCP

ID

PFCP ID

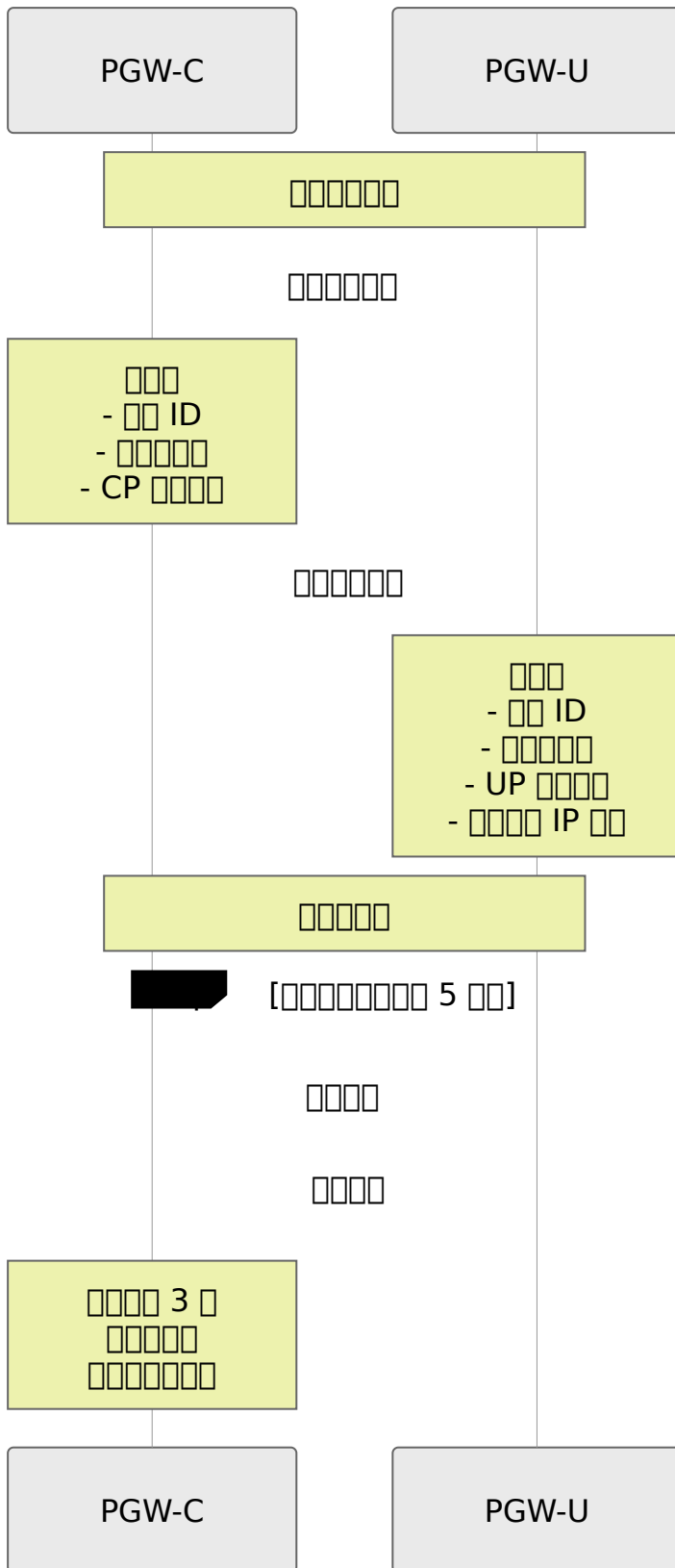
- **IPv4** -
- **IPv6**

- FQDN□□□□□□□□

PFCP □□□□

□□□□□□□□□□ PGW-C □ PGW-U □□□□ PFCP □□□

□□□□□□



配置参数

PFCP 配置参数

参数	说明
is_associated	是否关联
remote_node_id	远端节点 ID/IP 或 FQDN
remote_ip_address	远端 IP 地址
remote_port	UDP 端口 8805
heartbeat_period_ms	心跳周期
missed_heartbeats_consecutive	连续丢失心跳次数
up_function_features	UP 功能特性
up_recovery_time_stamp	UP 恢复时间戳

配置

配置 PFCP 参数

配置

```
# config/runtime.exs
sxb: %{
  local_ip_address: "10.0.0.20"
},
upf_selection: %{
  fallback_pool: [
    %{remote_ip_address: "10.0.0.21", remote_port: 8805, weight:
100}
  ]
}
# UPF 5
```

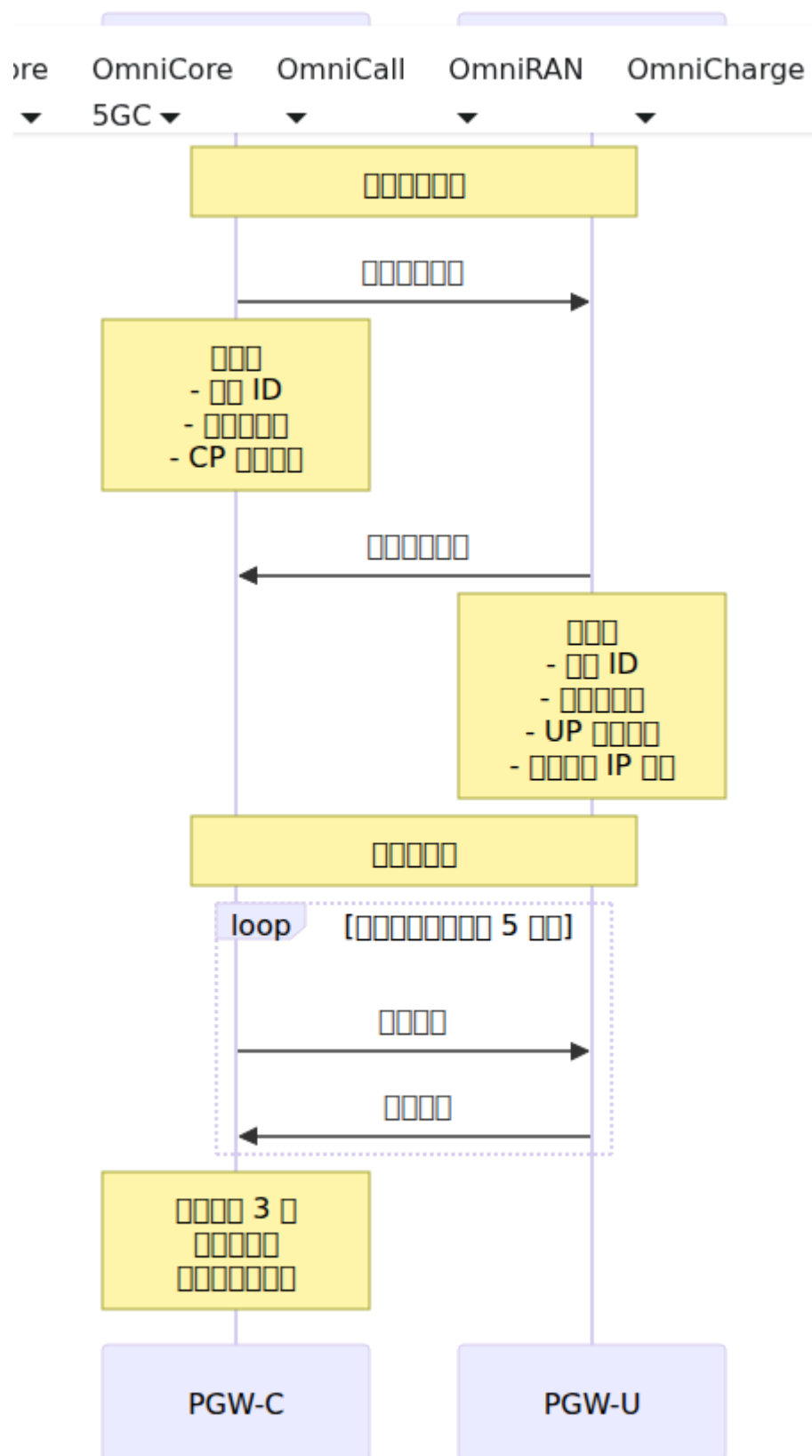
UPF

- missed_heartbeats_consecutive
- 3
-

PFCP

PFCP UE PDN

□□□□□□



PGW-C

UE PDN

PGW-C PGW-U

PGW-C

- **SEID** - ID -
- **ID** - PGW-C ID
- **F-SEID** - SEID IP + SEID
- **PDRs** - 2 +
- **FARs** - 2 +
- **QERs** - QoS
- **BAR** -

PGW-U

PGW-U

- -
- **F-SEID** - PGW-U
- **PDRs** -
- **F-TEID** - S5/S8 TEID

PGW-U

QoS

PGW-U

- PDRs FARs QERs
-
-

□□□□

UE PDN

111

1. PGW-C 0000000000 SEID
2. PGW-U 0000000000
3. PGW-U 00000000

F-TEID ☐

F-TEID 5G Core Network 5G Core Network GTP-U 5G Core Network PF-PCP 5G Core Network F-TEID 5G Core Network 5G Core Network UPF 5G Core Network

□□ F-TEID □□

□□□□□ F-TEID □□□□□

- **TEID**□□□□□□□□ - 32 □□□□□□□□
- **IP** □□ - □□ GTP-U □□□□□□□□UPF □ IP □□□

□□□ □□□ TEID □□

1UP

- PGW-C “TEID” CHOOSE
- UPF TEID

2 PGW-C

- PGW-C 検索 TEID 検索 UPF “検索 TEID”
- UPF 検索 TEID 検索

UPF 000000 - 0000

111

```
sxb: %{
    allocate_uplink_f_teid: false # []
}
```

□□□□□

1. PGW-C 選擇 F-TEID 並選擇 PCFP 地址
2. UPF 選擇 TEID
3. UPF 選擇 F-TEID + IP 地址
4. PGW-C 選擇 F-TEID

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

1 2 3 4 5 6

- UPF 1000000 = UPF 1000000000
- PGW-C 10000 UPF 1000 TEID
- 1000000000000000

PGW-C

- PGW-C 通過 UPF 通過
- PGW-C 通過 TEID 通過
- UPF 通過

□ □ **3GPP** □ □

- CHOOSE 3GPP TS 29.244
- UPF
- “”

□ □ □ □ □ □ □ □

- PGW-C 선택UPF 선택 TEID 선택
- 선택 TEID 선택
- UPF 선택 TEID

--	--	--	--	--

- 网络 UPF 鉴权
- 网络 UPF 鉴权 PGW-C 鉴权
- 网络鉴权/鉴权
- 网络鉴权 3GPP PFCP 鉴权

鉴权

- △ 网络鉴权 UPF 鉴权 CHOOSE 鉴权
- △ 网络鉴权/鉴权“鉴权 IE”鉴权UPF 鉴权 CHOOSE

PGW-C 鉴权

鉴权

```
sxb: %{
  allocate_uplink_f_teid: true
}
```

鉴权

1. PGW-C 鉴权 TEID
2. PGW-C 鉴权 TEID 鉴权 PFCP 鉴权
3. UPF 鉴权 TEID鉴权
4. PGW-C 鉴权 UPF 鉴权 TEID 鉴权

鉴权

网络 UPF 鉴权 CHOOSE

- 网络 UPF 鉴权/鉴权
- UPF 鉴权 PFCP 鉴权 TEID
- 鉴权

网络 TEID 鉴权

- 鉴权 PGW-C 鉴权 TEID
- 鉴权PGW-C 鉴权 TEID 鉴权
- 鉴权 TEID 鉴权

□ □□□□□

- □□□□□□□□ TEID □□□□
- □□□□□□□□□□□□□□ TEID □□

□□□□

⚠ □ **PGW-C** □□□□

- □□ UPF □□□ PGW-C □□□□□□ TEID □□
- □□
 - □□ PGW-C □□□ TEID □□□□□□□□
 - □□ TEID □□□□□□□□□□□□
 - □□□□□□□□□□□□□□□□

⚠ □□□□

- PGW-C □□□□□□□□ TEID □□□□□□
- PGW-C □□□ TEID □□□□□□□□□□□□□□
- □□□□□□□□□□

⚠ □□□□□□

- □□□□□□ PFCP □□□□□□
- □□□□□□□□ CHOOSE □ UPF □□□□□□

□□□□□□

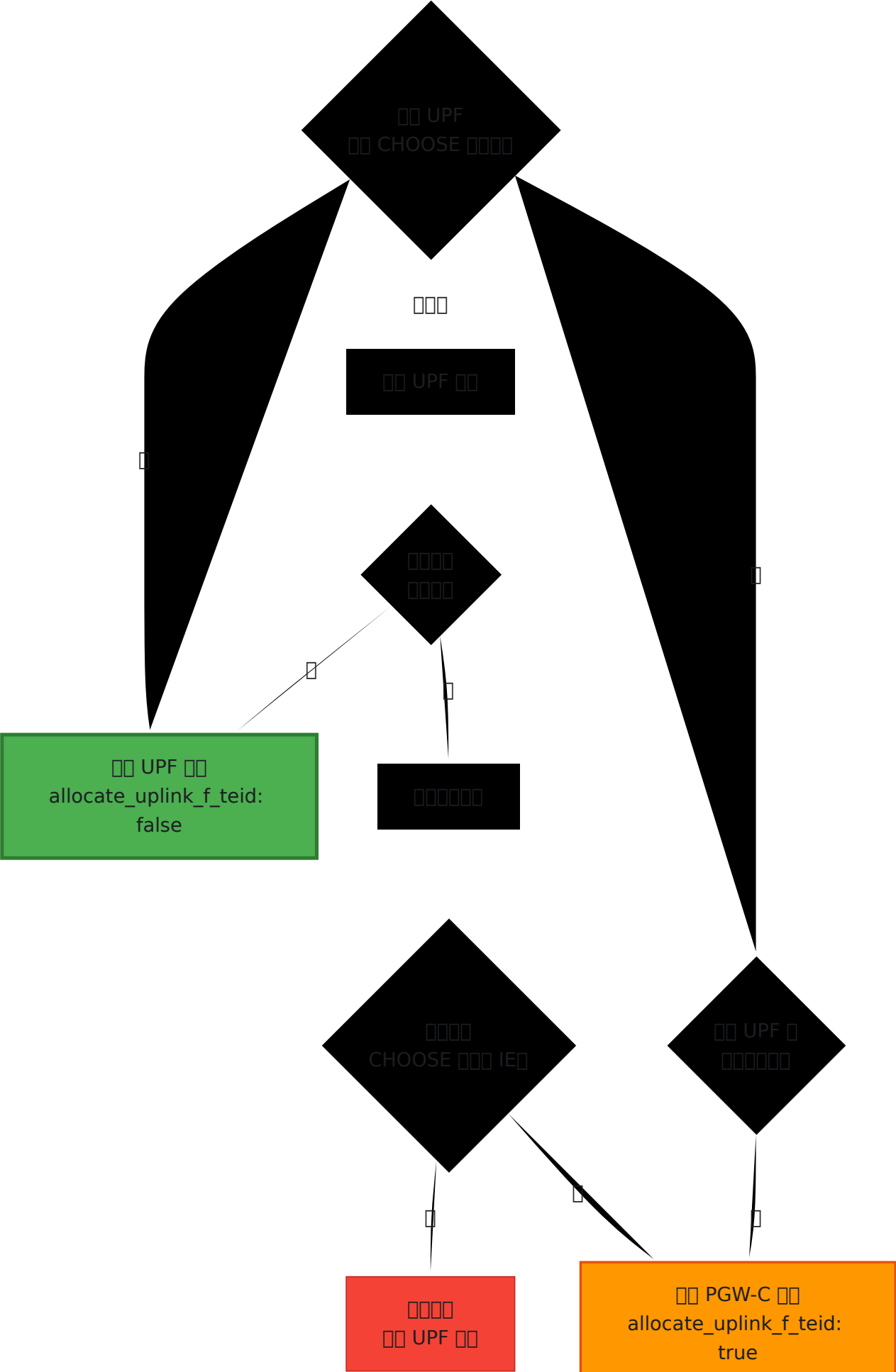
- ⚠ □□ **UPF** □□□ **CHOOSE** □□□
- ⚠ □□ UPF □□□□□□□□□□□□□□
- ⚠ □□□□□□□□□□
- ⚠ □□ PGW-C TEID □□□□□□□□

TEID □□□□□□ PGW-C □□□□□□□□□□□□

- TEID □□□1 □ 0xFFFFFFFF□42 □□□□□□
- □□□□□□□□ 100 □□□□□□□□ 0.023%
- □□□□□□□□□□□□□□□□

- 000000000000 TEID

0000



□□□□

□□□□□□□□□□

□□ PFCP □□□

```
# □□□ CHOOSE □□□□□
grep -i "choose\|mandatory.*missing" /var/log/pgw_c.log

# □□ PFCP □□□□□□□□□□
grep "Session Establishment Response" /var/log/pgw_c.log
```

□□ **UPF** □□ **CHOOSE** □□□

- □□□□□□“□□□□ IE”□“□□ IE”
- UPF □□□□ F-TEID□□□□ CHOOSE
- □□□□□ □□ `allocate_uplink_f_teid: true`

□□ **PGW-C** □□□□□□□

- □□□□ - TEID □□□□□40 □□□□
- □□ TEID □□□□□□□□□□□□□□□□

```
# □□□□□□□
grep "registered_teid_count" /var/log/pgw_c.log
```

□□□□□□□□

```
# □□ config/runtime.exs
sxb: %{
  local_ip_address: "10.0.0.20",
  allocate_uplink_f_teid: false # □□ UPF □□□ CHOOSE□□□□□ true
}
```

□□□□ PGW-C□

```
systemctl restart pgw_c
```

確認する必要があるのは PFCP のインストール

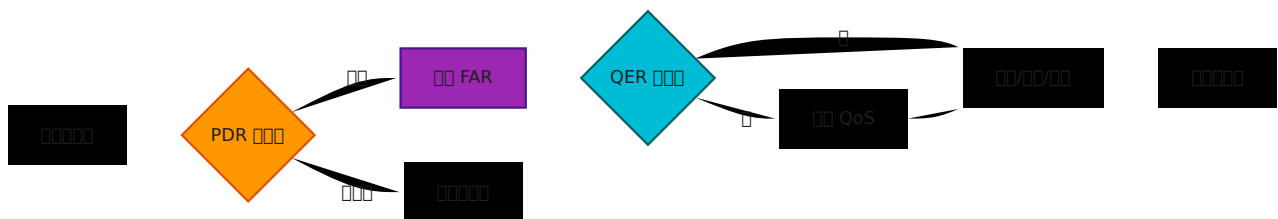
```
# PFCP のインストール
tcpdump -i any -n port 8805 -w pfcp.pcap

# Wireshark でパケットキャプチャ
# F-TEID から "CHOOSE" する UPF を指定
# F-TEID から TEID を PGW-C に指定
```

確認する必要があるのは

PFCP のインストール

確認する



PDRのインストール

確認する必要があるのは

PGW-C のインストール

PDR #1 - 確認

PDR ID: 1
QoS: 100
PDI: 0000000000000000
- 0000CORE00000000
- UE IP 000100.64.1.42/32
FAR ID: 1000000000

PDR #2 - 000

PDR ID: 2
QoS: 100
PDI: 0000000000000000
- 0000ACCESS0SGW 00
- F-TEID: <S5/S8 0000>
FAR ID: 2000000000
QER ID: 10QoS 000

00 PDR 000

- **PDR ID** - 0000000000000000
- 000 - 0000000000000000
- **PDI** - 00000000IP0TEID 00
- 00000 - 000000 GTP-U 0
- **FAR ID** - 00000000
- **QER ID** - 000 QoS 000000

FAR0000000000

000 0000000000000000

FAR #1 - 000000 → UE00

- ACCESS GW
- GTP-U/UDP/IPv4
- F-TEID: <SGW S5/S8 >

- 4 COREs
- 4 IP blocks

- **FAR ID** - 00000000
- 0000 - 000000000000
- 00 00
 - 000000ACCESS/CORE0
 - 0000000000 GTP-U 0000
 - 000000VRF/00000

□□ QER□

QER ID: 1

□□□□□□

□□□□□□

- □□□100 Mbps

- □□□50 Mbps

□□□□□□□□□□□□ GBR □□□

- □□□10 Mbps

- □□□10 Mbps

□□ **QER** □□□

- **QER ID** - □□□□□□□
- □□□ - □□□□□□□□□□□□□□
- **MBR** - □□□□□□□□/□□□
- **GBR** - □□□□□□□□□□□□□□
- **QCI** - QoS □□□□□□□□□□

BAR□□□□□□□□

□□□ □□ UE □□□□□□□□□□□□

□□ **BAR**□

BAR ID: 1

□□□□□□□□□100ms

□□□□□□□□□10

□□□ □□□□ DRX□□□□□□□□□□

□□

□□ **Sxb** □□

□□ `config/runtime.exs`□


```

config :pgw_c,
  sxb: %{
    # PCF IP
    local_ip_address: "10.0.0.20",

    # PCF port 8805
    local_port: 8805,

    # PCF timeout 500ms
    # PCF UPF
    # PCF UPF request timeout 500ms
    request_timeout_ms: 500,

    # PCF request attempts 3
    # request_timeout_ms = request_timeout_ms * request_attempts
    request_attempts: 3,

    # PCF F-TEID
    # false PCF UPF F-TEID CHOOSE
    # true PGW-C PCF F-TEID
    # PCF UPF CHOOSE
    allocate_uplink_f_teid: false
  },

  # UPF - PCF UPF
  upf_selection: %{
    fallback_pool: [
      %{
        # PGW-U IP
        remote_ip_address: "10.0.0.21",

        # PCF port 8805
        remote_port: 8805,

        # PCF 100 = 1000 = 100
        weight: 100
      }
    ]
  }
}

```

1. 背景

PGW-C と UPF の 間 の 通 信

2. 概要

パラメータ	単位	値	説明
request_timeout_ms	ms	500	PGW-C が UPF に 送 り 出 す 通 信 の 時 間 限 制
request_attempts	回	3	PGW-C が UPF に 送 り 出 す 通 信 の 試 行 回 数

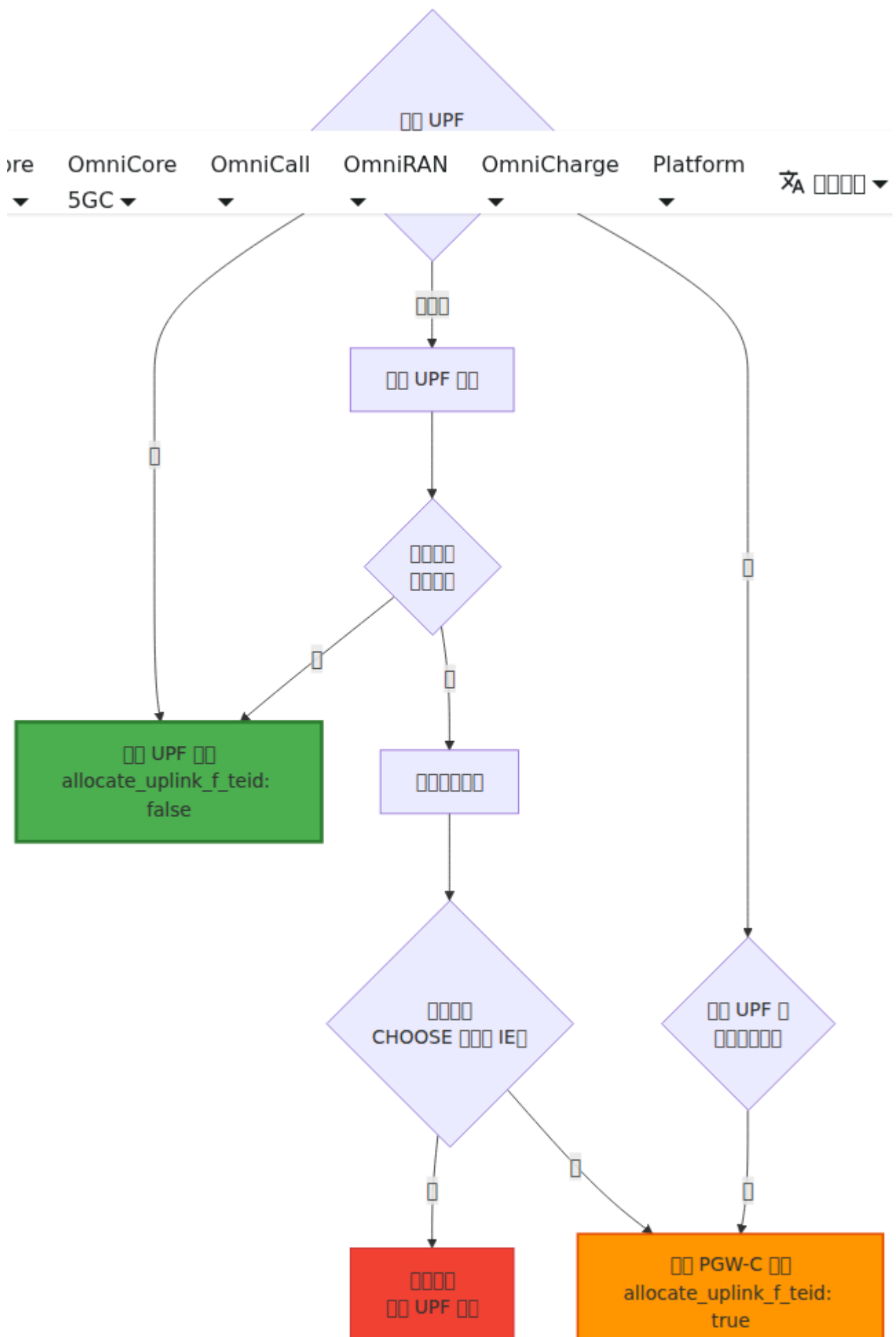
3. 計算式: $\text{request_timeout_ms} \times \text{request_attempts}$

4. 計算結果: $500\text{ms} \times 3 = 1.5 \text{ 秒}$

5. 結論

6. request_timeout_ms が UPF に 送 り 出 す 通 信 の 時 間 限 制

- PGW-C が 通 信 を 開 始
- PGW-C が UPF に 通 信 を 送 り 出 す
- PGW-C が 通 信 を 受 け 取 る
- UPF が 通 信 を 受 け 取 る **PF** 通 信
- PGW-C が 通 信 を 受 け 取 る ID
- PGW-C が UPF に 通 信 を 送 り 出 す



0000

UPF 0000	000 request_timeout_ms	00000
000<100ms	200-300ms	600-900ms3 0000
000100-300ms	500ms0000	1.5s3 0000
00300-500ms	750-1000ms	2.25-3s3 0000
0000>500ms	1500-2000ms	4.5-6s3 0000

000 0 request_timeout_ms 00000 2x 000 UPF 000000000000000000000000

00 - 0 UPF

```

sxb: %{
  local_ip_address: "10.0.0.20",
  request_timeout_ms: 1000, # 0000 1 0
  request_attempts: 3      # 00000 3 0
}
```

000000

000000000000

- UPF 000 PFCP 0000000000
- 00000 UPF 0000000
- 00000“0000000000”0000000000

000000

1. 00 UPF API 00000000 UPF 00000
2. 0 PGW-C 000000000000
3. 00 UPF 000000000000000000000000

000000

1. `request_timeout_ms` `UPF`
2. `PGW-C`
3. `UPF` `UPF`

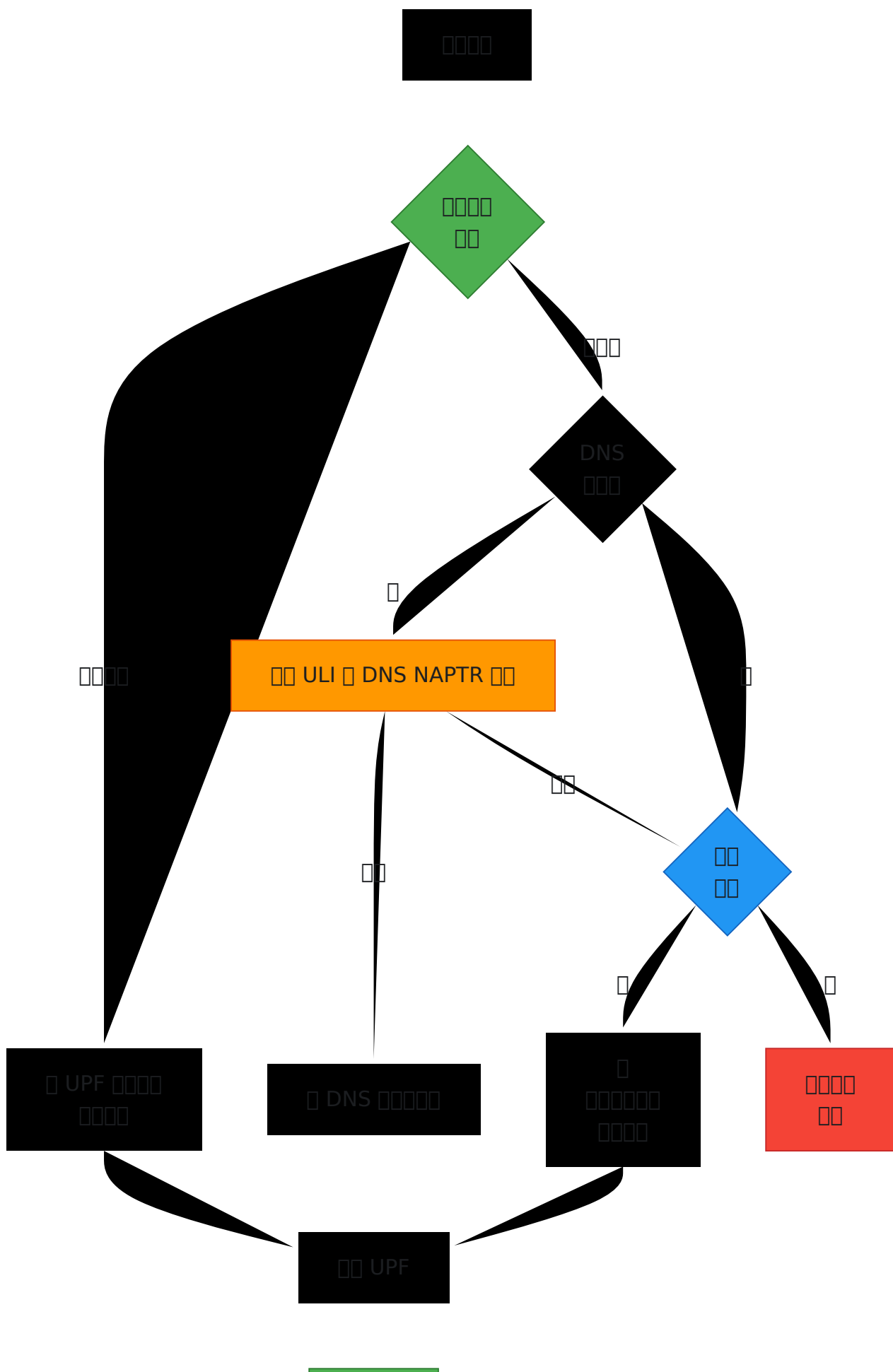
`PGW-U`

```
config :pgw_c,  
  sxb: %{  
    local_ip_address: "10.0.0.20"  
  },  
  upf_selection: %{  
    fallback_pool: [  
      %{remote_ip_address: "10.0.1.21", remote_port: 8805, weight:  
50}, # 50%  
      %{remote_ip_address: "10.0.2.21", remote_port: 8805, weight:  
50} # 50%  
    ]  
  }  
# UPF 5
```

UPF 四四四

PGW-C 四四 **UPF** 四四四四四四四四四四四四

1. 四四四四四四四四 - 四四四四四四四四
2. 四四 **DNS** 四四四四四四四四 - 四四 DNS NAPTR 四四四四四四四四
3. 四四四四四四四四 - 四四四四四四四四四四 UPF 四





☐☐☐ **UPF** ☐☐☐☐


```

config :pgw_c,
  # PFCP []
  sxb: %{
    local_ip_address: "10.0.0.20"
  },

  # UPF [] UPF []
  upf_selection: %{
    #
=====

    # [] DNS []
    #

=====

    # []ULI[] DNS
    # [] UPF []
    dns_enabled: false,
    dns_query_priority: [:ecgi, :tai, :rai, :sai, :cgi],
    dns_suffix: "epc.3gppnetwork.org",
    dns_timeout_ms: 5000,

    #
=====

    # []
    #

=====

    # []
    # [] UPF []
    rules: [
      # [] 1[]IMS [] - []
      %{
        name: "IMS []",
        priority: 20,
        match_field: :apn,
        match_regex: "^ims",
        upf_pool: [
          %{remote_ip_address: "10.100.2.21", remote_port: 8805,
weight: 80},
          %{remote_ip_address: "10.100.2.22", remote_port: 8805,
weight: 20}
        ],
        # [] PCO []
        pco: %{
          p_cscf_ipv4_address_list: ["203.0.113.100",

```

```

"203.0.113.101"]
    }
},

# 2 APN - 
%{
    name: "",
    priority: 15,
    match_field: :apn,
    match_regex: "^(enterprise|corporate)\\.apn",
    upf_pool: [
        %{remote_ip_address: "10.100.3.21", remote_port: 8805,
weight: 100}
    ],
    pco: %{
        primary_dns_server_address: "192.168.1.10",
        secondary_dns_server_address: "192.168.1.11",
        ipv4_link_mtu_size: 1500
    }
},

# 3 - 
%{
    name: "",
    priority: 10,
    match_field: :serving_network_plmn_id,
    match_regex: "^(310|311|312|313)", # 
    upf_pool: [
        %{remote_ip_address: "10.100.4.21", remote_port: 8805,
weight: 100}
    ]
},

# 4 - 
%{
    name: "",
    priority: 5,
    match_field: :apn,
    match_regex: "^internet",
    upf_pool: [
        %{remote_ip_address: "10.100.1.21", remote_port: 8805,
weight: 33},
        %{remote_ip_address: "10.100.1.22", remote_port: 8805,
weight: 33},

```

```
        %{remote_ip_address: "10.100.1.23", remote_port: 8805,
weight: 34}
    ]
}
],

#
=====

# 
#
=====

#  DNS 
fallback_pool: [
    %{remote_ip_address: "127.0.0.21", remote_port: 8805, weight:
100}
]
}
```

Parameters

Parameter	Description	Value
:imsi	Subscriber IMSI	"310260123456789"
:apn	Access Point Name	"internet" or "ims"
:serving_network_plmn_id	Serving Network PLMN ID (MCC+MNC)	"310260" or other PLMN
:sgw_ip_address	SGW IP Address	"10.0.1.50"
:uli_tai_plmn_id	ULI TAI PLMN ID	"310260"
:uli_ecgi_plmn_id	E-UTRAN TAI PLMN ID	"310260"

UPF Parameters

UPF Parameters

```
upf_pool: [  
  {%remote_ip_address: "10.100.1.21", remote_port: 8805, weight:  
50},  
  {%remote_ip_address: "10.100.1.22", remote_port: 8805, weight:  
30},  
  {%remote_ip_address: "10.100.1.23", remote_port: 8805, weight:  
20}  
]
```

负载均衡

1. 权重 $50 + 30 + 20 = 100$
2. 权重 $0.0 \sim 100.0$
3. 权重与 UPF
 - 0-50 UPF-1 50% 权重
 - 50-80 UPF-2 30% 权重
 - 80-100 UPF-3 20% 权重

权重

- 权重 33 33 34
- 80 20
- UPF 权重

PCO

PCO

```
%{
  name: "IMS  ",
  match_field: :apn,
  match_regex: "^ims",
  upf_pool: [...],
  pco: %{
    # 
    p_cscf_ipv4_address_list: ["203.0.113.100", "203.0.113.101"],
    #  PCO 
  }
}
```

PCO

- primary_dns_server_address
- secondary_dns_server_address
- primary_nbns_server_address
- secondary_nbns_server_address
- p_cscf_ipv4_address_list
- ipv4_link_mtu_size

DNS

PGW-C DNS NAPTR

```
upf_selection: %{
  dns_enabled: true,
  dns_query_priority: [:ecgi, :tai, :rai, :sai, :cgi],
  dns_suffix: "epc.3gppnetwork.org",
  dns_timeout_ms: 5000
}
```


1. **ECGI**E-UTRAN -
2. **TAI** -
3. **RAI** - 3G/2G
4. **SAI** - 3G

5. CGI - 2G

DNS

```
# ECI
eci-1a2b3c.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org

# TAI
tac-lb64.tac-hb00.tac.epc.mnc999.mcc999.epc.3gppnetwork.org
```

DNS

- ECGI TAI
- DNS
- UPF
- DNS DNS

DNS UPF

DNS UPF

DNS UPF UE ULI DNS NAPTR

3GPP TS 23.003 - UPF DNS

- UPF
-
-
- UPF

ECI - LTE ECI

ECI

```
eci-<HEX-ECI>.ecgi.epc.mnc<MNC>.mcc<MCC>.<dns_suffix>
```

ECI

```
# ECI ID: 0x1A2B3C1,715,004 ECI
# PLMN: MCC=999, MNC=999
eci-1a2b3c.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org
```

ECI LTE 4G ECI

2. TAI

ECI - TAI

ECI

```
tac-lb<LB>.tac-hb<HB>.tac.epc.mnc<MNC>.mcc<MCC>.<dns_suffix>
```

ECI

```
# TAC: 0x0064100 TAC
# TAC 0x64100000x00
tac-lb64.tac-hb00.tac.epc.mnc999.mcc999.epc.3gppnetwork.org
```

ECI LTE 4G ECI

3. RAI

3G/2G ECI

ECI


```
rac<RAC>.lac-lb<LB>.lac-hb<HB>.lac.raimnc<MNC>.mcc<MCC>.  
<dns_suffix>
```

□□□

```
# RAC: 0x0A□10 □□□□  
# LAC: 0x1234□4660 □□□□  
rac0a.lac-lb34.lac-hb12.lac.raimnc999.mcc999.epc.3gppnetwork.org
```

□□□□□ 3G/2G UMTS/GPRS □□

4. SAI□□□□□□□□□□

3G □□□□

□□□

```
sac<SAC>.lac-lb<LB>.lac-hb<HB>.lac.saimnc<MNC>.mcc<MCC>.  
<dns_suffix>
```

□□□

```
# SAC: 0x0001  
# LAC: 0x1234  
sac0001.lac-lb34.lac-hb12.lac.saimnc999.mcc999.epc.3gppnetwork.org
```

□□□□□ 3G UMTS □□□□

5. CGI□□□□□□□□□□

2G □□□

□□□

```
ci<CI>.lac-lb<LB>.lac-hb<HB>.lac.cgi.mnc<MNC>.mcc<MCC>.  
<dns_suffix>
```

例

```
# CI: 0x5678  
# LAC: 0x1234  
ci5678.lac-lb34.lac-hb12.lac.cgi.mnc999.mcc999.epc.3gppnetwork.org
```

例 2G GSM

DNS

NAPTR

DNS NAPTR UPF IP

```
eci-1a2b3c.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org.  
IN NAPTR 10 50 "a" "x-3gpp-upf:x-s5-gtp:x-s8-gtp" ""  
upf1.epc.mnc999.mcc999.3gppnetwork.org.  
  
upf1.epc.mnc999.mcc999.3gppnetwork.org.  
IN A 10.100.1.21
```

PGW-C

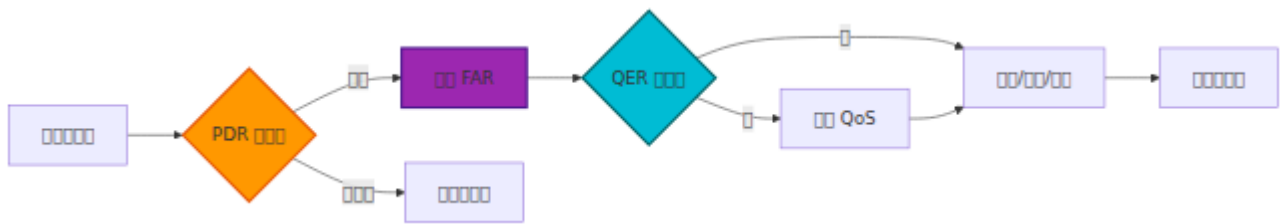
1. NAPTR UPF IP
2. DNS
- 3.

例

DNS [10.100.1.21, 10.100.5.99]

10.100.1.21
upf_selection

□□□□□□□□



□□

1. □□□□□□

□□□ □□□□□□□□□□ UPF

DNS □□□

```
# □□□□□□
eci-aaa.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org → UPF-□□□
□10.1.1.21□

# □□□□□□
eci-bbb.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org → UPF-□□
□10.2.1.21□

# □□□□□□
eci-ccc.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org → UPF-□□□
□10.3.1.21□
```

□□□ □□□□□□□□□□ UPF□□□□□□□□□□□□

2. □□□□□

□□□ □□□□□□□□ MEC□□□□□□□□□□UPF

DNS □□□

```
# □□□□□□□□□□□□ UPF
eci-*.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org → □□□□ UPF
```

如何 验证配置是否正确

3. 验证配置

如何 验证配置是否正确UPF 配置是否正确

如何 验证 DNS 配置是否正确 PGW-C 配置

如何 验证 DNS 配置

DNS 配置

如何

- 如何“DNS UPF 配置: nxdomain”
- 如何配置 DNS

如何

1. DNS 配置是否正确
2. DNS 配置是否正确 ID
3. GTP-C 配置是否正确 ULI

如何

```
# 如何 DNS 配置
dig eci-1a2b3c.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org NAPTR

# 如何 PGW-C 配置 DNS 配置
grep "DNS UPF selection: querying" /var/log/pgw_c.log

# 如何配置 ULI
# 如何配置 "uli" 配置
```

DNS 配置 UPF

如何

- DNS 配置 UPF 配置 `upf_selection` 配置
- 如何配置 DNS

- DNS PFCP 10.99.1.50 UPF
- DNS PFCP 10.100.1.21, 10.100.1.22

10.99.1.50

DNS 10.99.1.50
upf_selection: [10.100.1.21, 10.100.1.22]

10.99.1.50

- DNS PFCP
- 10.100.1.21
- 10.100.1.22

10.99.1.50

1. upf_selection 10.100.1.21, 10.100.1.22

```
upf_selection: %{
  fallback_pool: [
    %{remote_ip_address: "10.99.1.50", remote_port: 8805, weight:
100}
  ]
}
```

2. DNS 10.99.1.50 UPF IP

3. 10.100.1.21, 10.100.1.22 MEC/10.99.1.50

10.99.1.50

10.99.1.50

- “DNS UPF 10.99.1.50”
- 10.100.1.21

10.99.1.50

```
upf_selection: %{\n  dns_timeout_ms: 10000 # 1000ms 10 s\n}
```

🔍 DNS 🔍

🔍

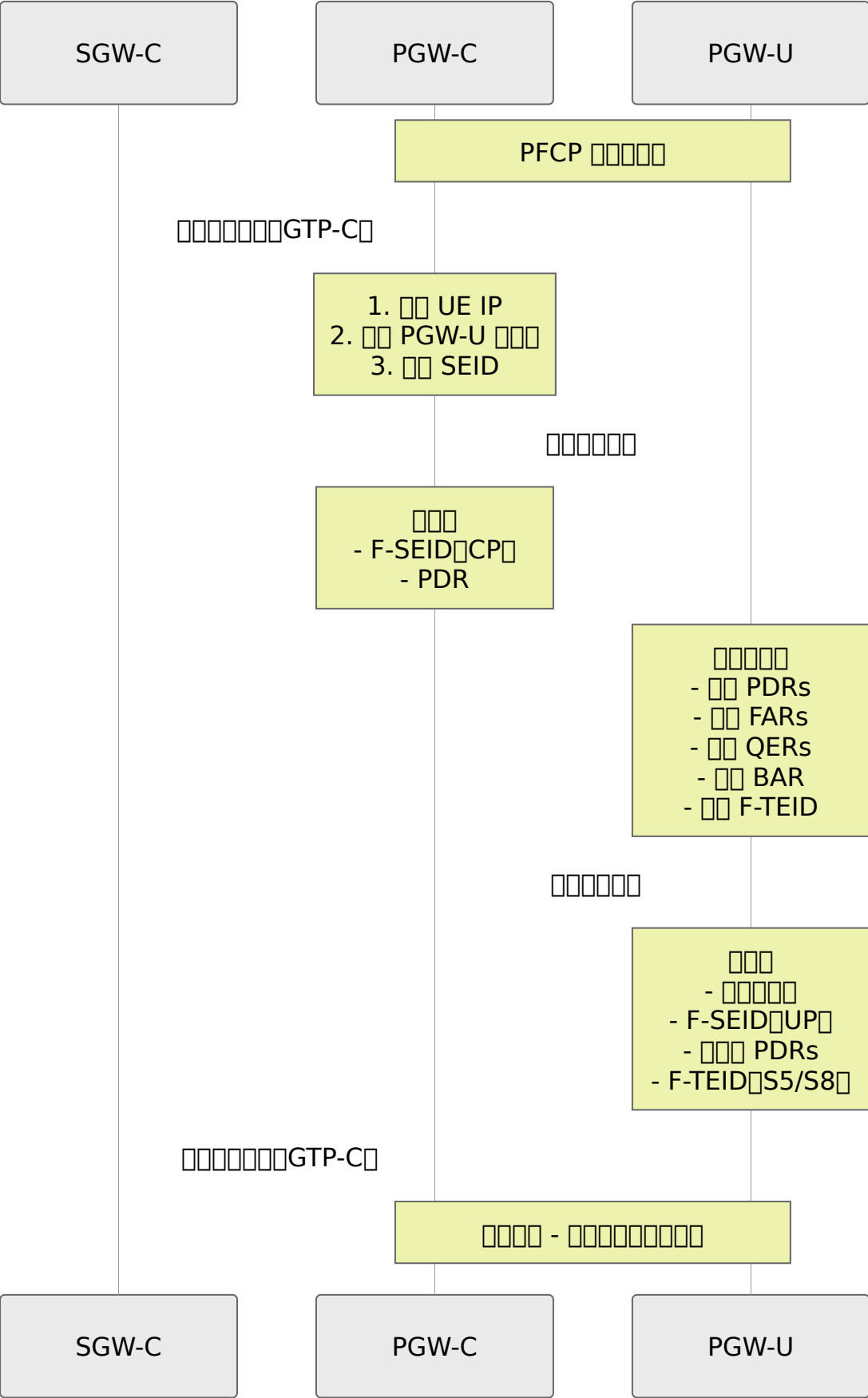
```
# DNS 🔍\nrate(upf_selection_dns_success_total[5m]) /\nrate(upf_selection_dns_attempts_total[5m])\n\n# DNS 🔍\nhistogram_quantile(0.95,\nrate(upf_selection_dns_duration_seconds_bucket[5m]))\n\n# 🔍 DNS 🔍\nrate(upf_selection_fallback_used_total[5m])
```

🔍🔍🔍🔍

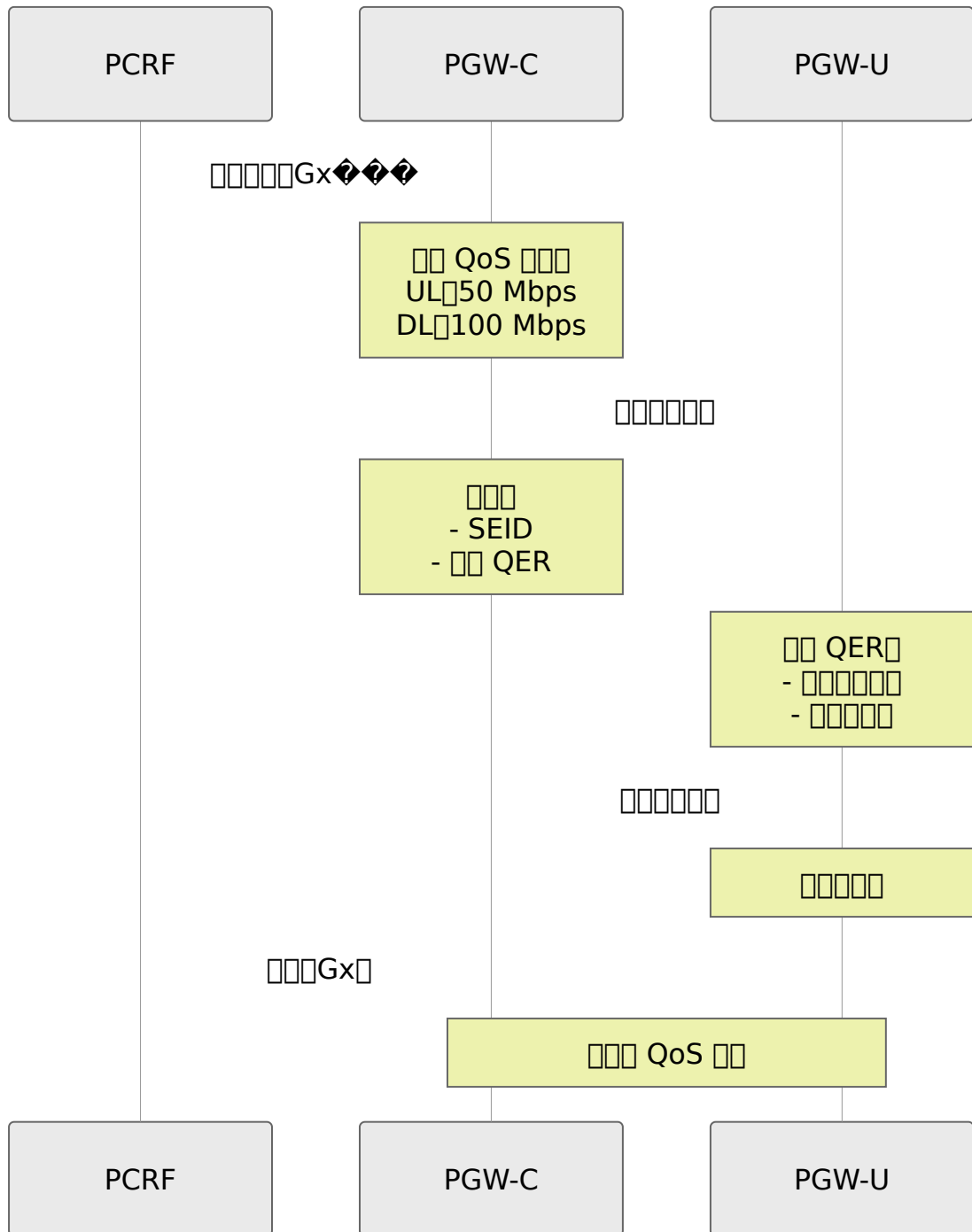
```
[debug] DNS UPF 🔍 eci-\n1a2b3c.ecgi.epc.mnc999.mcc999.epc.3gppnetwork.org\n[debug] DNS UPF 🔍 DNS 🔍 2 🔍\n[info] DNS UPF 🔍 10.100.1.21
```

□□□

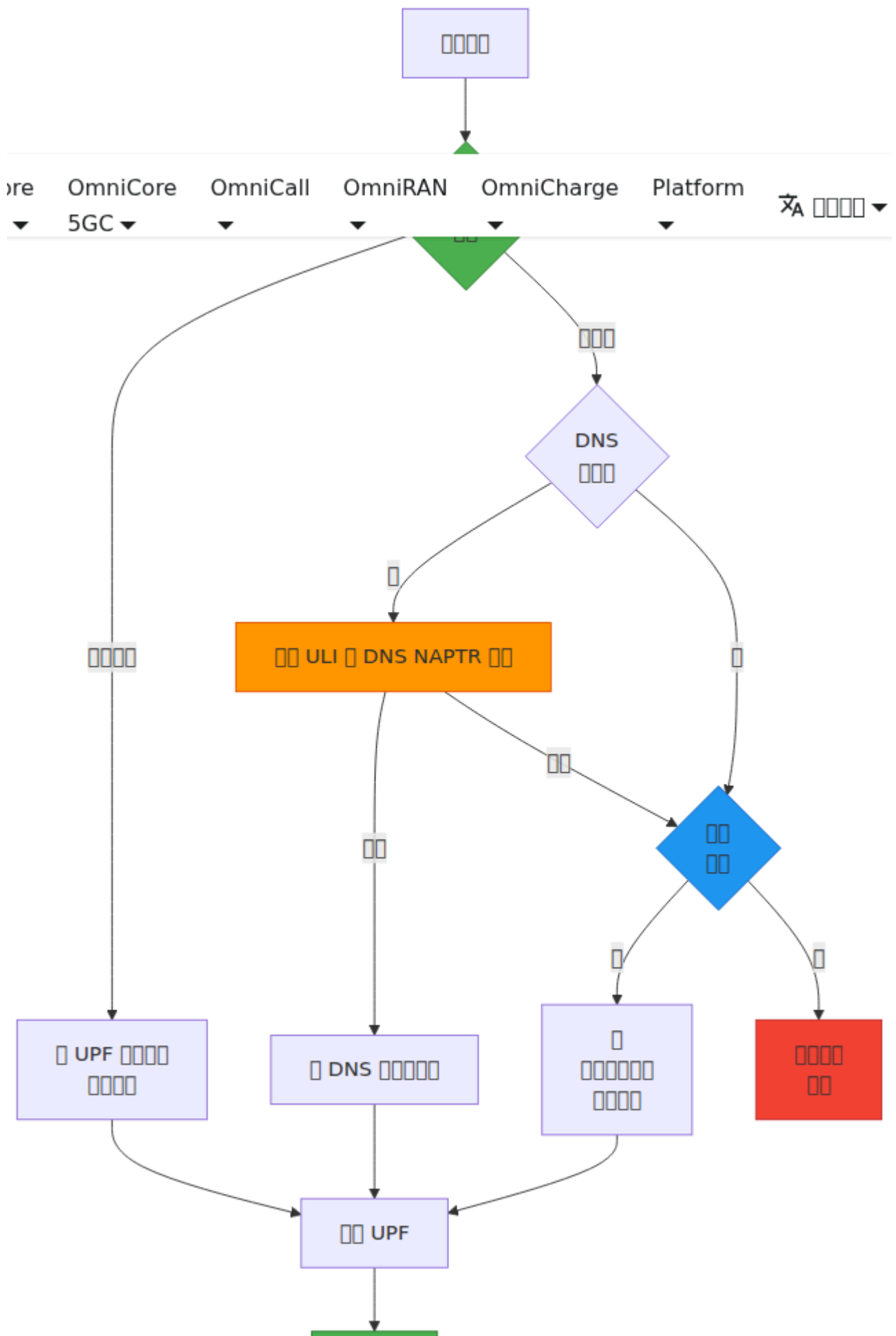
□□□□□□□



□□□□□



□□□□□□



0000

0000

1. 000000

000

- 000000“PFCP 000000”
- 000000000000

00000

- PGW-U 0000000000
- PGW-U 000
- 000000 UDP 00 8805
- 0000 remote_ip_address 000

00000

```
# 00000
ping <pgw_u_ip_address>

# 00 UDP 00
nc -u -v <pgw_u_ip_address> 8805

# 00000
iptables -L -n | grep 8805
```

2. 0000

000

- 000“000000003”

- 00000000

000000

- 000000000000
- PGW-U 00
- 00000000

000000

00000000 5 0000000000 3 00000000

3. 000000

000

- 00000000000000
- 000“PFCP 000000”

000000

- 000000 PGW-U 000
- PGW-U 000000
- 00000000

000

1. 000000000000 `is_associated = true`
2. 00 PGW-U 00000000
3. 00 SEID 0000

4. 00 SEID 00

000

- 0000000000“0000000000”

000

- SEID 0000000000

- PGW-U 向 PGW-C 发送

配置

- 配置 PFCP 参数
- PGW-C 向 PGW-U 发送

配置 PFCP 参数

配置

```
# PFCP 配置
pfcpeer_associated{peer="PGW-U Primary"} 1

# PFCP 配置
seid_registry_count 150

# PFCP 配置
rate(sxb_inbound_messages_total[5m])

# PFCP 配置
rate(sxb_inbound_errors_total[5m])

# 配置
pfcpeer_consecutive_heartbeat_failures{peer="PGW-U Primary"} 0
```

配置

```
# PFCPPeerDown
- alert: PFCPPeerAssociationDown
  expr: pfcpeer_associated == 0
  for: 1m
  annotations:
    summary: "PFCP {{ $labels.peer }} 断联"

# PFCPSessionEstablishmentFailureHigh
- alert: PFCPSessionEstablishmentFailureHigh
  expr:
rate(sxb_inbound_errors_total{message_type="session_establishment_res
[5m]) > 0.1
  for: 5m
  annotations:
    summary: "PFCP 会话建立失败"
```

PFCP 简介

PFCP/Sxb 接口 UPF 和 PGW-U 之间的 OAM REST API 接口，API 使用 HTTPS/TLS 加密，端口 8443。接口地址为 `/api`，可以通过 Swagger UI 访问，地址为 `https://<omnipgw-ip>:8443/api/docs`。

UPF / PFCP 接口

```
GET /api/upf          # 获取 UPF / PFCP 接口 + 版本
GET /api/upf/<ip>      # 获取 IP 地址的 UPF
```

通过 PGW-U 接口 PFCP 接口

接口

```
curl -k https://<omnipgw-ip>:8443/api/upf
curl -k https://<omnipgw-ip>:8443/api/upf/10.0.0.1
```

通过 PGW-U 接口

項目	項目
名前	名前空間
IP アドレス	PGW-U IP
タイプ	タイプ
ID	PFCP ID
名前	名前空間
タイプ	タイプ
名前空間	名前空間
UP 名前	PGW-U 名前

GET /api/upf/<ip> 名前空間 PGW-U IP 名前空間

PFCP ID

PGW-U PFCP ID

GET /api/sessions/<imsi>

PGW-U PDRs FARs QERs BARs F-SEIDs

名前

PGW-U PFCP ID

1. GET /api/upf
2. 返回数据中“imsi”
3. 返回数据中 = 0
4. 返回数据中“imsi”
 - 返回 IP 地址
 - 返回数据中
 - 返回UDP 8805

返回数据中

1. 返回数据中
2. GET /api/sessions/<imsi> - 返回数据
3. 返回数据 PFCP 数据 - PFCP 数据
4. 返回 PFCP 数据
 - GET /api/upf - 返回数据
 - 返回 PFCP 数据
5. 返回 PFCP 数据
 - 返回 PDRs/FARs 数据
 - 返回PGW-U 数据

返回数据中

1. 返回 PGW-U 数据
2. GET /api/upf 返回数据
3. 返回数据
4. 返回数据

返回数据中

- GET /api/upf 返回数据“imsi”
- 返回数据中
- 返回 OAM API 返回数据

返回

- 返回 - 返回数据 GET /api/upf
- 返回 - 返回数据/返回
- 返回 - 返回数据

- **SMF** - **SMF** PDRs/FARs/QERs
-

SMF

SMF

- **SMF** - UPF **SMF**PFCP **SMF**
- **SMF** - PDN **SMF**

SMF

- **Diameter Gx** **SMF** - **SMF** PFCP QoS **SMF** PCC **SMF**
- **Diameter Gy** **SMF** - **SMF** URR **SMF**
- **SMF CDR** **SMF** - **SMF** PFCP **SMF** CDR
- **SMF** - PFCP **SMF**UPF **SMF**

SMF

- **S5/S8** **SMF** - **SMF**
 - **UE IP** **SMF** - **SMF** PFCP **SMF** UE **SMF**
-

SMF

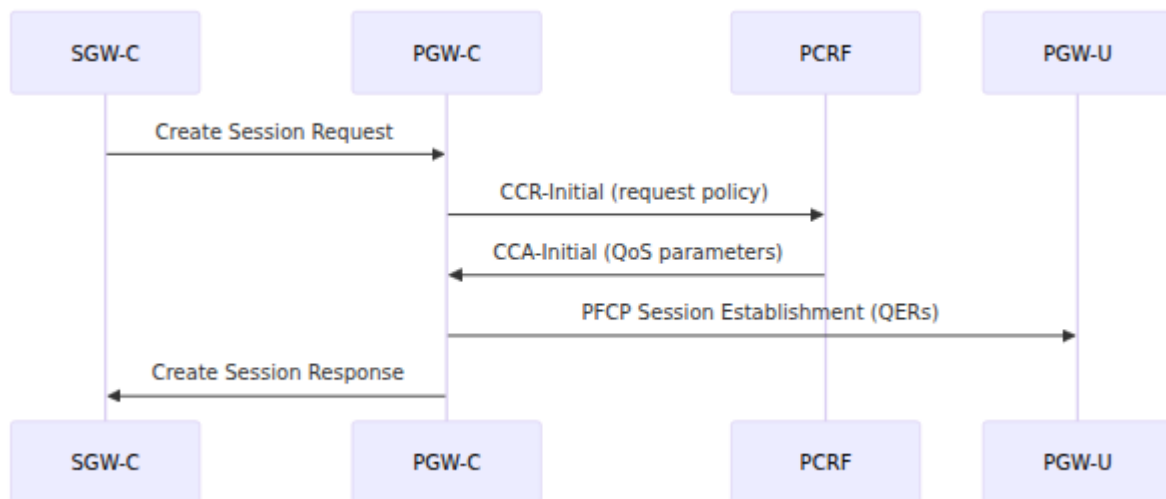
QoS

QoS

PGW-C QoS

- **Gx (Diameter)** - PCRF QoS
- **S5/S8 (GTP-C)** - SGW-C
- **Sxb (PFCP)** - QoS PGW-U

QoS



QoS

- UE PDR/FAR/QER/BAR AMBR
- EBI (EPS ID) PDR/FAR QER
- **QER (QoS)** MBR/GBR
- PDN
- PCRF QoS

□□

□□□□□□ **QoS** □□

□□ QoS □□□□□□ Diameter Gx □□□□□□ PCRF□□□ PCRF □□□□□□□□□□□□□□
OmniHSS□□

□□□□ `config/runtime.exs` □□□ **PCRF** □□□

```
config :pgw_c,  
  diameter: %{  
    listen_ip: "0.0.0.0",  
    host: "omni-pgw_c.epc.mnc999.mcc999.3gppnetwork.org",  
    realm: "epc.mnc999.mcc999.3gppnetwork.org",  
    peer_list: [  
      %{  
        host: "pcrf.epc.mnc999.mcc999.3gppnetwork.org",  
        realm: "epc.mnc999.mcc999.3gppnetwork.org",  
        ip: "192.168.1.100",  
        initiate_connection: true  
      }  
    ]  
  }  
}
```

QoS □□□□□□□□□□□□ **PCRF** □□□□□□□□ **PGW-C** □□□□□□

□□□□□□□□

□□□□□□□□

□□□□□□ PDN □□□□□□□□□□



Create Session Request

AllocateIP

UE IP assigned

RequestPolicy

CCR-Initial sent to PCRF

CreateBearer

CCA-Initial received
with QoS

ProgramUPF

PFCP Session
Establishment

Active

Delete Session Request



□□□□□

1. SGW-C  Create Session Request
2. PGW-C □□□□□□□□ UE IP □□
3. PGW-C □□ CCR-Initial □ PCRF□□□ IMSI□APN□IP □□
4. PCRF □□ CCA-Initial□□□ QoS □□□
 - Default-EPS-Bearer-QoS (QCI, ARP)
 - QoS-Information (AMBR □□)

5. PGW-C 消息交互

- 消息 ID: PDR=1, PDR=2, FAR=1, FAR=2, QER=1, BAR=1
- QER 消息, QoS 消息, MBR 消息

6. PGW-C 向 PGW-U 发送 PFCP Session Establishment Request

7. PGW-C 向 SGW-C 发送 Create Session Response

消息交互

- PDN 地址
- QCI 5 或 QCI 9 的 GBR
- EBI 值
- 承载类型

消息交互

消息: PCRF 消息

消息: PCRF 消息 RAR 消息 Charging-Rule-Install

消息

1. PCRF 向 RAR 消息 Charging-Rule-Definition 消息

- Charging-Rule-Name 消息
- Flow-Information 消息
- QoS-Information 消息 QCI, MBR, GBR, ARP
- Precedence 消息

2. PGW-C 向 PGW-U 发送 PFCP 消息

- Flow-Information 消息 → PDR 消息 SDF 消息
- QoS-Information → QER 消息 MBR/GBR 消息
- Flow-Description → IP 5 消息

3. PGW-C 向 PGW-U 发送 PFCP Session Modification Request 消息 PDRs/FARs/QERs

4. PGW-C 向 SGW-C 发送 Create Bearer Request

5. SGW-C 向 PGW-C 发送 Create Bearer Response 消息

消息 **Charging-Rule-Definition** 消息

```
Charging-Rule-Name: "video_streaming"
Flow-Information:
  - Flow-Description: "permit in ip from any to 10.0.0.1 5000-6000"
    Flow-Direction: 1 (downlink)
QoS-Information:
  QoS-Class-Identifier: 7
  Max-Requested-Bandwidth-UL: 5000000 (5 Mbps)
  Max-Requested-Bandwidth-DL: 10000000 (10 Mbps)
  Guaranteed-Bitrate-UL: 1000000 (1 Mbps)
  Guaranteed-Bitrate-DL: 2000000 (2 Mbps)
Precedence: 100
Flow-Status: 2 (ENABLED)
```

□□□□

□□ QoS □□□□□□□□□□

- **Gx RAR**□□□ Charging-Rule-Definition
- **PFCP Session Modification**□□□□□ QERs□□□□□□□□□□ FARs□□□□□□□□ PDRs□□□□□□□□□□

□□□□

□□□□

- **Delete Session Request**□SGW □□□ - □□□□□□□□□□□□
- **Re-Auth Request with Charging-Rule-Remove**□PCRF □□□ - □□□□□□□

□□□□□

1. □□□□□□□□□□□□
2. □□□□□ PDRs/FARs/QERs
3. □ SGW-C □□ Delete Bearer Request□□□□□ PCRF □□□
4. □□ PFCP Session Modification□□□□□□□□□□ Session Deletion□□□□□□□□□□

QoS

QCI (QoS Class Identifier)

PCRF → Gx → **QoS-Class-Identifier** AVP

QCI

- **QCI 1** GBR 100ms
- **QCI 2** GBR 150ms
- **QCI 3** GBR 50ms
- **QCI 4** GBR 300ms
- **QCI 5** IMS GBR 100ms -
- **QCI 6** TCP GBR 300ms
- **QCI 7** GBR 100ms
- **QCI 8** TCP YouTube GBR 300ms
- **QCI 9** GBR 300ms

QCI

- QCI → PCRF → QoS IE → SGW-C
- PGW-C → QCI → QERs → MBR/GBR
- QCI
- QCI

ARP (Allocation-Retention-Priority)

PCRF → **Allocation-Retention-Priority** AVP

ARP

- 1 15
- - 0 =
 - 1 =
-

- 0 = 00000000
- 1 = 00000001

0000

- 00000001
- 000000000
- 0000000001

00000000

- ARP → SGW-C → 000000 eNodeB
- **PGW-C** 00000 - 00000 eNodeB 0000000000
- 00000000000000000000
- 000000000000 100000000000

MBR (000000)

000 PCRF 00 Max-Requested-Bandwidth-UL 0 Max-Requested-Bandwidth-DL
AVP

000 0000000000 kbps bytes / 1000

0000 0000000000

00000

- PGW-C 00 QER00 mbr: %Bitrate{ul: kbps_ul, dl: kbps_dl}
- QER 00 PFCP 000 PGW-U
- **PGW-U** 000000000000
- 00 MBR 00000000

000

Max-Requested-Bandwidth-UL: 5000000 (5 Mbps)
Max-Requested-Bandwidth-DL: 10000000 (10 Mbps)

- QER `qer` mbr: {ul: 5000, dl: 10000} kbps
- PGW-U `pgw-u` 5 Mbps `pgw-u`
- PGW-U `pgw-u` 10 Mbps `pgw-u`

GBR (Guaranteed Bit Rate)

PCRF `Guaranteed-Bitrate-UL` `Guaranteed-Bitrate-DL` AVP

GBR `gb` kbps

GBR `gb` GBR `gb`

GBR

- Charging-Rule-Definition `Charging-Rule-Definition` GBR `gb` **GBR** `gb`
- PGW-U `pgw-u` QER `qer`
- eNodeB `eNodeB`
- GBR `gb` - `gb`

GBR

Guaranteed-Bitrate-UL: 1000000 (1 Mbps)
Guaranteed-Bitrate-DL: 2000000 (2 Mbps)

- QER `qer` gbr: {ul: 1000, dl: 2000} kbps
- `pgw-u` 1 Mbps `pgw-u` 2 Mbps `pgw-u`
- `VoIP`

GBR

- GBR `gb`
- `Charging-Rule-Definition` GBR `gb`
- `eNodeB` GBR `gb`

AMBR (Aggregate-Max-Bitrate)

PCRF から APN-Aggregate-Max-Bitrate-UL と APN-Aggregate-Max-Bitrate-DL AVP

この AMBR は GBR APN に対して

適用される

- AMBR は GBR に対して
- Create Session Response から SGW-C
- eNodeB/SGW から
- PGW-C から AMBR を SGW-C

例

APN-Aggregate-Max-Bitrate-UL: 50000000 (50 Mbps)
APN-Aggregate-Max-Bitrate-DL: 100000000 (100 Mbps)

- GBR は 50 Mbps / 100 Mbps
- MBR
- AMBR は UE/APN に対して

例

- HSS/PCRF から
- 10 Mbps から 100 Mbps
- GBR

例

(Gx) (PFCP)

PCRF から Charging-Rule-Definition から Flow-Status AVP

Flow-Status (Gx)	Gate-Status (PFCP QER)	備考
0 = ENABLED-UPLINK	ul: OPEN, dl: CLOSED	アップリンクのみ許可
1 = ENABLED-DOWNLINK	ul: CLOSED, dl: OPEN	ダウンリンクのみ許可
2 = ENABLED	ul: OPEN, dl: OPEN	アップリンクとダウンリンクの両方許可
3 = DISABLED	ul: CLOSED, dl: CLOSED	アップリンクとダウンリンクの両方禁止
4 = REMOVED	ul: CLOSED, dl: CLOSED	アップリンクとダウンリンクの両方禁止

備考

- **DISABLED**アップリンクとダウンリンクの両方禁止
- **ENABLED-UPLINK**アップリンクのみ許可
- **ENABLED-DOWNLINK**ダウンリンクのみ許可
- **ENABLED**アップリンクとダウンリンクの両方許可

セッション登録

Prometheus 監視

メトリクス

```
session_registry_count      # IMSI EBI 数
address_registry_count      # UE IP
charging_id_registry_count  # 充電ID
```

Gx 監視

```
gx_inbound_messages_total{message_type="gx_RAR"}      # [] PCRF []  
[]  
gx_outbound_messages_total{message_type="gx_CCR"}    # [] PCRF []  
[]  
gx_outbound_transaction_duration_bucket              # [] PCRF []
```

PFCP []

```
sxb_outbound_messages_total{message_type="pfc_session_establishment_  
sxb_outbound_messages_total{message_type="pfc_session_modification_  
sxb_outbound_transaction_duration_bucket
```

[][]

```
s5s8_inbound_messages_total{message_type="create_session_request"}  
# []  
s5s8_outbound_messages_total{message_type="create_bearer_request"}  
# []
```

[] OAM API []

[][] QoS [] OAM REST API [] HTTPS/TLS [] 8443 [] /api []
[] Swagger UI [] <https://<omnipgw-ip>:8443/api/docs> []

[][] QoS []

```
GET /api/sessions/<imsi>      # [] QCI/5QI []
```

[][]

```
curl -k https://<omnipgw-ip>:8443/api/sessions/999000123456789
```

[][] QoS [] QCI/5QI [] MBR [] GBR [] AMBR [] QER
[]

PCRF

```
[debug] Sending Credit Control Request: ... # CCR to PCRF
[debug] Handling Credit Control Answer: ... # CCA from PCRF
QoS
[debug] Handling Re-Auth Request # RAR from PCRF
PCRF
[debug] Sending Session Establishment Request # PFCP to PGW-
U-plane QERs
[debug] Sending Session Modification Request # PFCP to PGW-
U-plane QERs
```

QoS

QoS

1. GET /api/sessions/<imsi> curl -k https://<omnipgw-ip>:8443/api/sessions/999000123456789
2. QoS
3. QoS
4. QoS **qer_map** QoS

```
qer_id: 1
gate_status: {ul: OPEN, dl: OPEN}
mbr: {ul: 50000, dl: 100000} # kbps
gbr: {ul: 10000, dl: 20000} # kbps nil GBR
```

5. QoS PCRF QoS

QoS

QoS QoS

QoS

1. QoS **PCRF** QoS

- PCRF status = "connected" Diameter/Gx interface
- Diameter peer

2. CCR/CCA

- OmniPGW sends "Credit Control Answer"
- CCA contains QoS-Information AVP
- CCA contains Result-Code 2001 = SUCCESS

3. PFCP

- "PFCP Session Establishment Request"
- QER
- PGW-U sends PFCP

4. PCRF

- PCRF
- APN
- PCRF

Prometheus

```
# /
rate(s5s8_inbound_messages_total{message_type="create_session_request"
[5m])

# 
rate(s5s8_outbound_messages_total{message_type="create_bearer_request"
[5m])

# PCRF 
rate(gx_inbound_messages_total{message_type="gx_RAR"}[5m])
```

구현

구현 방법

```
# UE IP 분포율 계산
(address_registry_count / <configured_pool_size>) * 100

# 세션 수
session_registry_count

# PCRF P95 퍼센타일
histogram_quantile(0.95, gx_outbound_transaction_duration_bucket)
```

참고

- `config/runtime.exs`에 `ue.subnet_map` 설정
- TEID 32비트, 40비트 지원
- PCRF P95 퍼센타일

참고

- IP 주소 - 80% 분포
- PCRF - PCRF
- PCRF P95 퍼센타일

구현

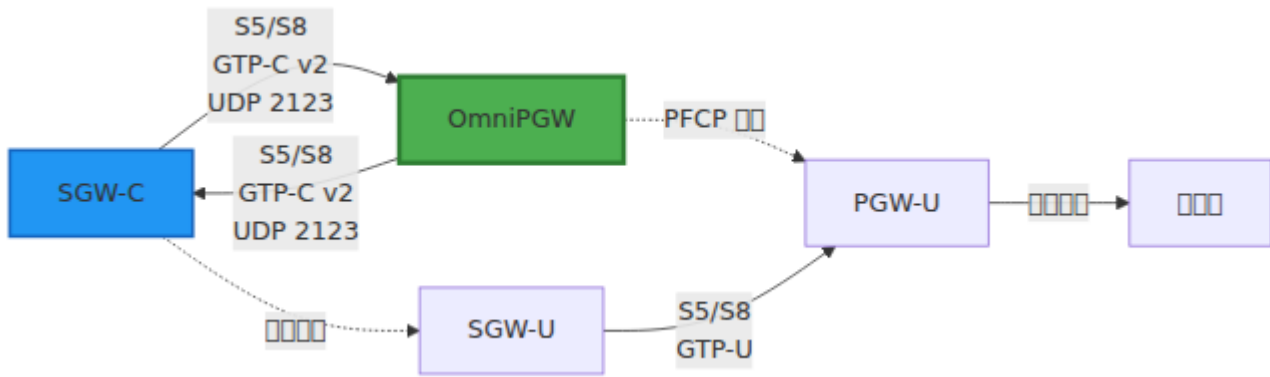
- PDN - PDN
- Diameter Gx - PCRF
- PFCP - PCRF
- PCRF - PCRF
- PCRF - PCRF

S5/S8

SGW-C GTP-C

OmniPGW Omnitouch

S5/S8 GTP-C v2 GPRS - OmniPGW SGW-C



GTP-C 2

- GTP-C v2 (3GPP TS 29.274)
- UDP
- 2123
-

TEID

TEID

- **TEID** - OmniPGW  
- **TEID** - SGW-C 

TEID

SGW-C → OmniPGW: TEID = OmniPGW TEID

OmniPGW → SGW-C: TEID = SGW-C TEID

TEID

TEID

```
# config/runtime.exs
config :pgw_c,
  s5s8: %{
    # S5/S8  IPv4 
    local_ipv4_address: "10.0.0.20",

    #  IPv6 
    local_ipv6_address: nil,

    # 
    local_port: 2123,

    # GTP-C  500ms
    #  GTP-C 
    request_timeout_ms: 500,

    # GTP-C  3
    #  = request_timeout_ms * request_attempts
    #  500ms * 3  = 1500ms 
    request_attempts: 3
  }
```

TEID

S5/S8  GTP-C 

□□□

- `request_timeout_ms` - □□□□□□□□□□□□□□□□500ms□
- `request_attempts` - □□□□□□□□□□□□□□□□3□

□□□□□□ `request_timeout_ms × request_attempts`

□□□□□ 500ms × 3 □□□ = □□□□□ **1.5** □

□□□□□

◆◆□□□	□□ <code>request_timeout_ms</code>	□□□□□
□□□ (<50ms)	200-300ms	600-900ms□3 □□□□
□□□50-150ms□	500ms□□□□	1.5s□3 □□□□
□□□ (>150ms)	1000-2000ms	3-6s□3 □□□□
□□□/□□	2000-3000ms	6-9s□3 □□□□

□□ - □□□□□□

```
s5s8: %{  
  local_ipv4_address: "10.0.0.20",  
  request_timeout_ms: 1500, # □□□□ 1.5 □  
  request_attempts: 3      # □□□□□ 4.5 □  
}
```

□□□□□□□

- OmniPGW □□□□□ "□□□□□□□□"
- □□□□□ PCRF□□□□□□□□□□5012 UNABLE_TO_COMPLY□
- □□□□□□□□□□□□□□□□□□□□ Charging-Rule-Remove □□□□

□□□□□

□□□□□□

```
# SGW-C GTP-C
iptables -A INPUT -p udp --dport 2123 -s <sgw_network>/24 -j
ACCEPT

# SGW-C GTP-C
iptables -A OUTPUT -p udp --dport 2123 -d <sgw_network>/24 -j
ACCEPT
```

```
# SGW-C
ip route add <sgw_network>/24 via <gateway_ip> dev eth0
```

S5/S8 PDN GTP-C

SGW-C → OmniPGW

PDN

IEs

IE 名称	长度	说明
IMSI	13	国际移动用户识别码
MSISDN	13	移动用户号码
APN	16	接入点名称，如“internet”
RAT 类型	1	无线接入技术，如EUTRAN
QoS 特性	1	服务质量特性
UE 名称	16	用户设备名称
ULI	13	用户本地标识符，包含TAI和ECGI
PLMN	3	公共陆地移动网络，包含MCC/MNC

示例

```
ULI
├─ IMSI: 310260123456789
├─ MSISDN: 14155551234
├─ APN: internet
├─ RAT 类型: EUTRAN (6)
├─ QoS 特性
│   ├─ EBI: 5
│   └─ QoS (QCI 9, ARP, 1)
├─ S5/S8 F-TEID (SGW-U 地址)
└─ ULI
    ├─ TAI: MCC 310, MNC 260, TAC 12345
    └─ ECGI: MCC 310, MNC 260, ECI 67890
```

网络名称

网络 OmniPGW → SGW-C

网络 网络名称

UE Parameters

IE Name	Length	Value
QoS	1	QoS
QoS	1	QoS
PDN Type	IP	UE IP address UE IP address
APN	1	APN
PCO	1	PCO PCO

QoS

QoS

- QoS: QoS (16)
- PDN Type
 - IPv4: 100.64.1.42
- QoS
 - EBI: 5
 - QoS: QoS
 - S5/S8 F-TEID (PGW-U QoS PFCP)
- APN Name: Public-1 (1)
- PCO
 - DNS Name: 8.8.8.8
 - DNS Name: 8.8.4.4
 - QoS MTU: 1400

PDN Type: PDN Type TS 29.274 §8.14 4 QoS IPv4 QoS

16 QoS IPv6 QoS IPv4v6 QoS 16 QoS IPv6 QoS 4 QoS IPv4 QoS

QoS IPv6 QoS SOS APN QoS IPv4v6 PDN PGW-C QoS IPv4 QoS PAA APN QoS

QoS—QoS QoS APN QoS IPv6 QoS IPv4v6 QoS

QoS

QoS SGW-C → OmniPGW

PDN

IE	
EBI	EPS ID
EBI	

OmniPGW → SGW-C

IE	

OmniPGW → SGW-C

PCRF

- PCRF PCC
- OmniPGW SGW-C

OmniPGW → SGW-C → SGW-C → OmniPGW

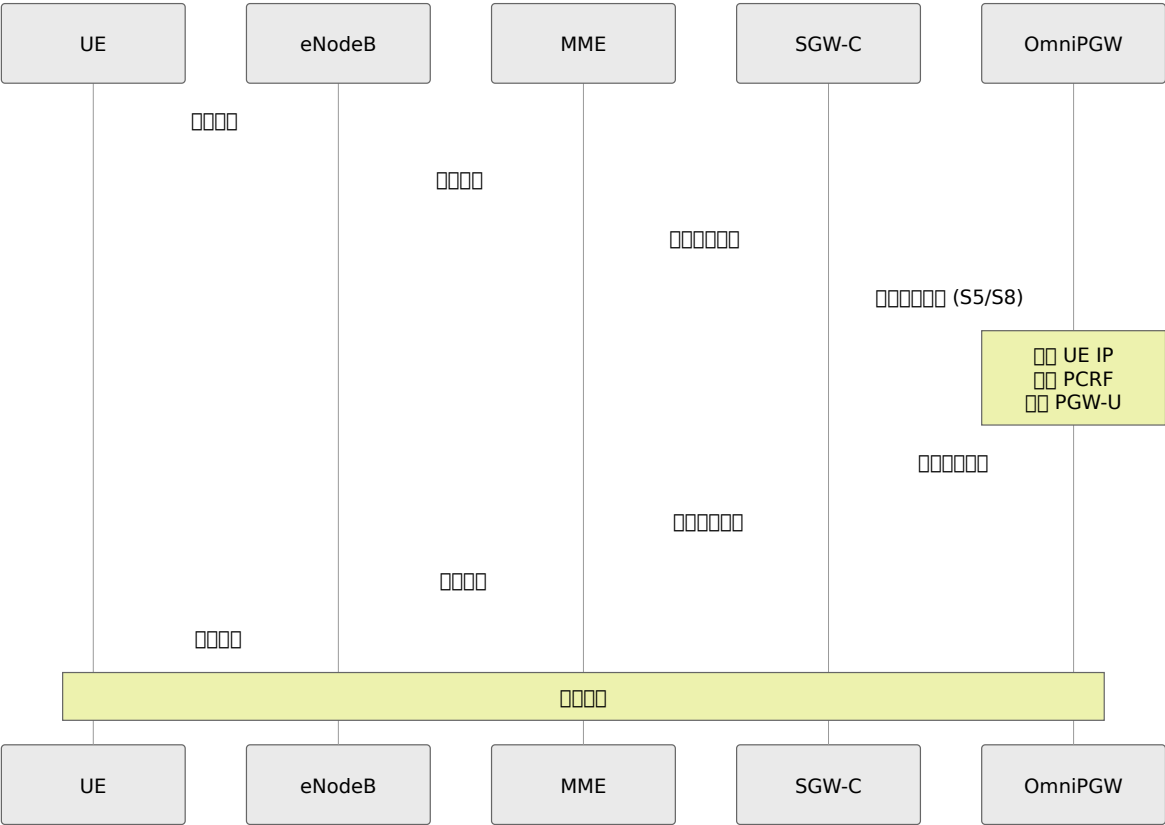
PGW 和 PCRF 的交互

PGW

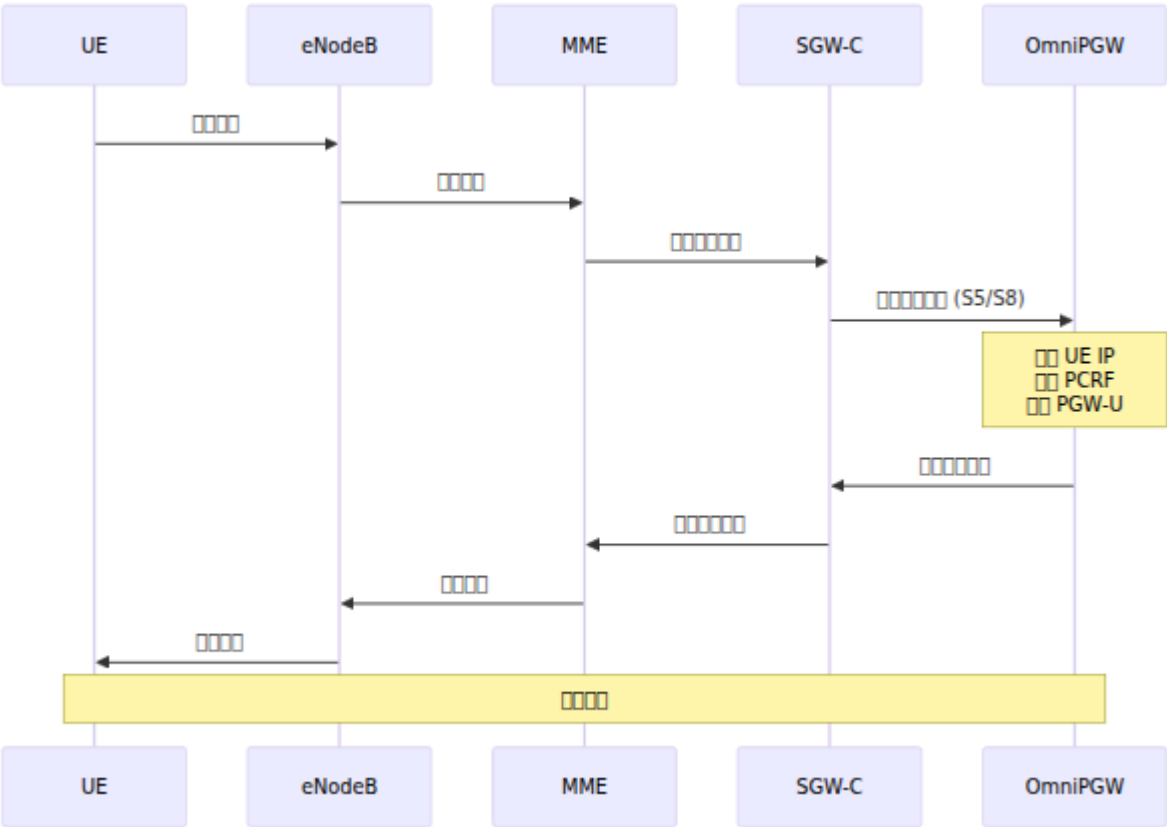
- PGW 和 PCRF 的交互
- SGW 和 PGW 的交互

PGW

PGW



□□□□



□□□□

□□

□□	□□	□□
16	□□□□□	□□□□

00000000

00	00	0000
65	0000	PCRF 000000 IMSI
66	00000	IP 000
93	00000	000 APN
94	TFT 000000	00000000

00000000

00	00	0000
72	00000000	PCRF/PGW-U 00
73	000000000	0000

□□

S5/S8 □□

```
# □□□□□
s5s8_inbound_messages_total{message_type="create_session_request"}
s5s8_inbound_messages_total{message_type="delete_session_request"}

# □□□□□
s5s8_inbound_errors_total

# □□□□□□
s5s8_inbound_handling_duration_bucket

# □□ TEID
teid_registry_count
```

□□□□□

□□□□□

```
rate(s5s8_inbound_messages_total{message_type="create_session_request"}[5m])
```

□□□□

```
rate(s5s8_inbound_errors_total[5m])
```

□□□**p95**□□

```
histogram_quantile(0.95,
rate(s5s8_inbound_handling_duration_bucket{request_message_type="create_session_request"}[5m])
)
```

環境構築

環境構築環境 OmniPGW 構築

環境構築

- SGW-C 環境構築
- 環境構築
- SGW-C 構築

環境構築

1. 環境構築
2. OmniPGW 環境構築 IP 構築
3. 環境構築 UDP 2123
4. 環境構築 TEID 構築

環境構築

```
# 環境構築 OmniPGW 環境構築
netstat -ulnp | grep 2123

# 環境構築
tcpdump -i any -n port 2123

# 環境構築
grep "local_ipv4_address" config/runtime.exs

# 環境構築
iptables -L -n | grep 2123
```

環境構築環境構築

環境構築

- 環境構築環境構築
- 環境構築

□□□□□

□□ 65□□□□□□□□

→ PCRF □□□□

→ □□ HSS/SPR □□ IMSI

□□ 66□□□□□□

→ IP □□□

→ □□: curl http://pgw:9090/metrics | grep address_registry_count

→ □□ IP □

□□ 72□□□□□□□□□□□□

→ PCRF □□□ PGW-U □□

→ □□ Gx □□

→ □□ PFCP □□

□□□TEID □□

□□□

- □□□□□□□□□□
- □□□□

□□□

- TEID □□□□□□□□
- TEID □□□□□□

□□□□□

- □□ TEID □□□□
 - □□ TEID □□□□□□□□
-

□□□□

□□□□

1. □□□□□□

- □ S5/S8 □□□□ VLAN
- □□□□□□

2. **MTU** □□

- □□ MTU □□ GTP □
- □□ MTU□1500 □□□1464 □□□□ + 36 GTP□

3. □□

- □□ OmniPGW □□
- □ SGW-C □□□□ DNS □□□□□

□□

1. **UDP** □□□□□

- □□□□□□□□□□□□
- □□□□□□□□ 4-8 MB

2. □□□□

- □□□□□□□□□□□□
- □□ TEID □□□□□

□□

1. **IP** □□

- □□□□□□□ SGW-C IP □ GTP-C
- □□ iptables □□□ ACL

2. 詳細

- OmniPGW 詳細
- 4G/LTE 詳細 GTP-C 詳細

概要

機能

- S5/S8 - S5/S8 詳細 IP 詳細
- PDN - PDN 詳細
- UE IP 詳細 - UE IP 詳細
- PCO 詳細 - GTP-C 詳細 PCO 詳細

インターフェース

- Gn/Gp 詳細 - 2G/3G GGSN 詳細 GTP-C v1
- PFCP 詳細 - S5/S8 詳細
- Gx 詳細 - 詳細
- Gy 詳細 - 詳細

データ

- S5/S8 GTP-C 詳細
- CDR 詳細 - GTP-C 詳細 CDR

詳細

OmniPGW S5/S8 詳細 - *OmniPGW S5/S8 詳細*



11

```
graph TD; Start(( )) -- "5 white squares" --> Creating[Creating]; Creating -- "3 white, 1 orange" --> Active[Active]; Active -- "3 white" --> Modifying[Modifying]; Active -- "2 white, 1 orange" --> Terminating[Terminating]; Modifying -- "3 white" --> Active; Terminating -- "3 white, 1 orange" --> Active; Terminating -- "4 white" --> End((( ))) Creating -- "5 white" --> End
```

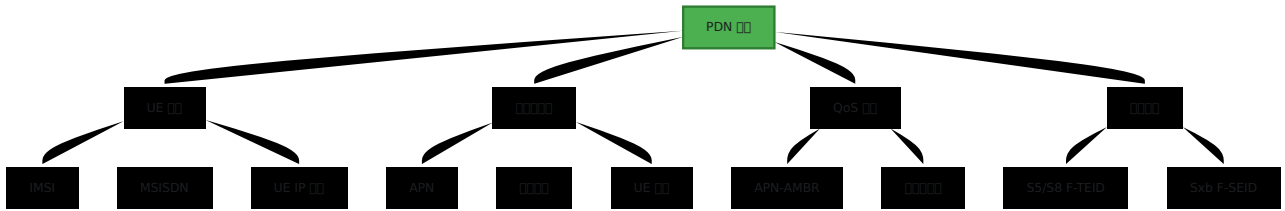

□□□□

□□□□□

□□□□□□□□□□□□□□□□

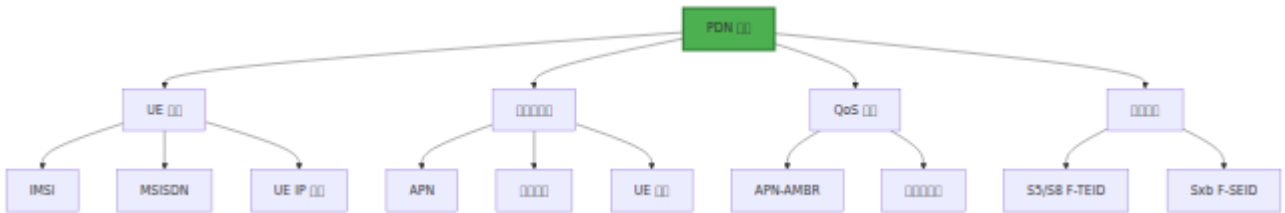
□□□	□□	□□
TEID	S5/S8 (GTP-C)	SGW-C □□□□□□□ ID
SEID	Sxb (PFCP)	PGW-U □□□□□□□ ID
Session-ID	Gx (Diameter)	PCRF □□□ Diameter □□
Charging-ID	□□	□□□□□ ID

□□□□



□□□□

□□□□



□□

1. □□□□□□ (S5/S8)

□□□□□ S5/S8 □□□ GTP-C □□□□□□□□ GTP-C □□□□□□□□□□ S5/S8 □□□

□□:

- IMSI, MSISDN, IMEI
- APN (□□□"internet")
- RAT □□ (EUTRAN)
- UE □□ (TAI, ECGI)
- □□□□□ (QoS, F-TEID)

2. □□□□

- □ APN □□□□ UE IP
- □□□□ ID
- □□ Gx □□-ID
- □□ S5/S8 TEID
- □□ PGW-U □□□

3. □□□□ (Gx)

□ PCRF □□□□□

- □□ CCR-□□
- □□□□ QoS □ PCC □□□ CCA-□□

4. □□□□□□ (PFCP)

□□□□□□□ PGW-U□

- □□□□□□□□
- □□ PDRs, FARs, QERs, BAR
- □□ S5/S8 □□□ F-TEID

5. □□ SGW-C

UE Parameters

- UE IP
 - S5/S8 F-TEID (PGW-U)
 - PCO (DNS, P-CSCF, MTU)
 - QoS
-

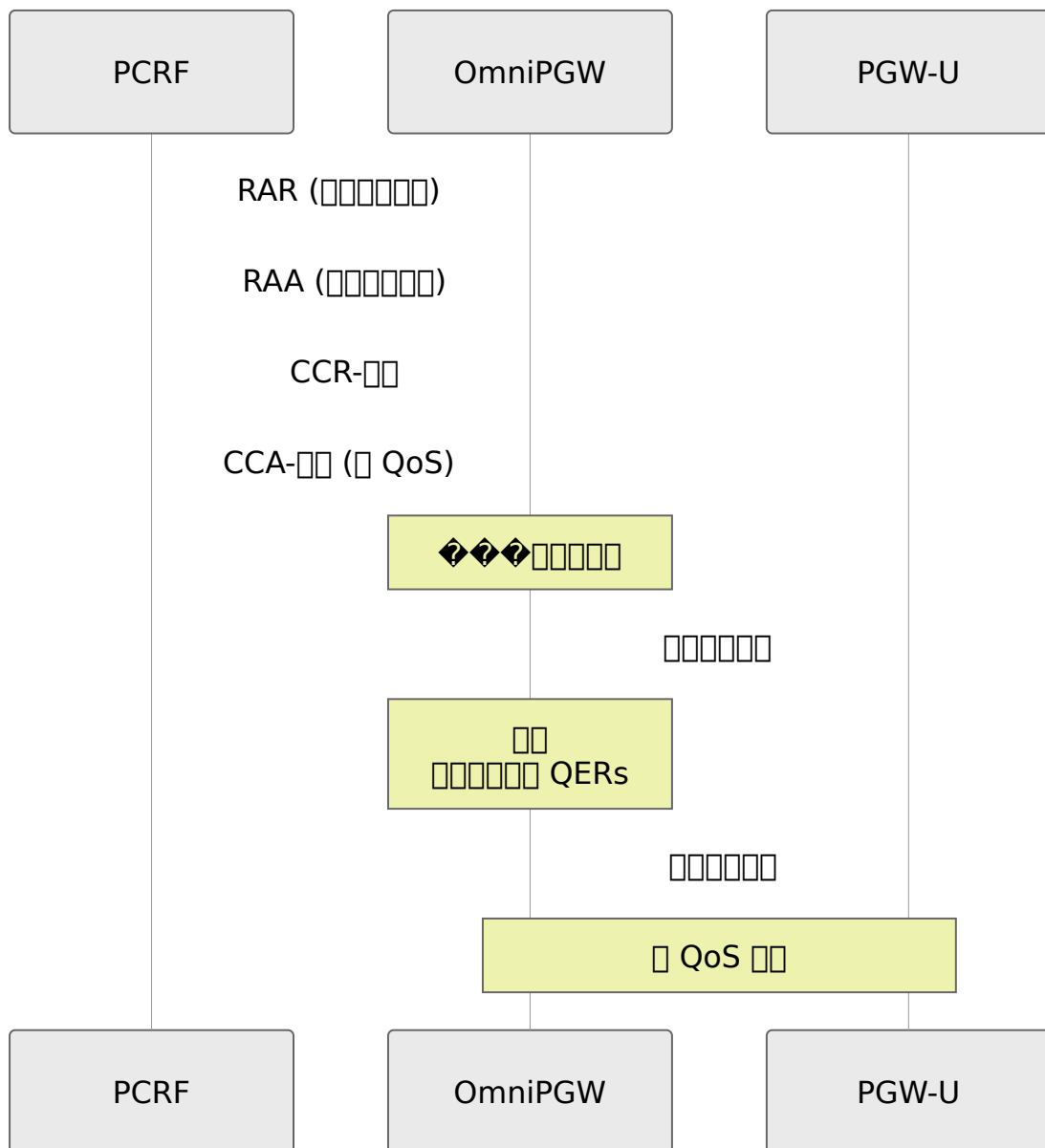
Network

QoS

QoS Parameters

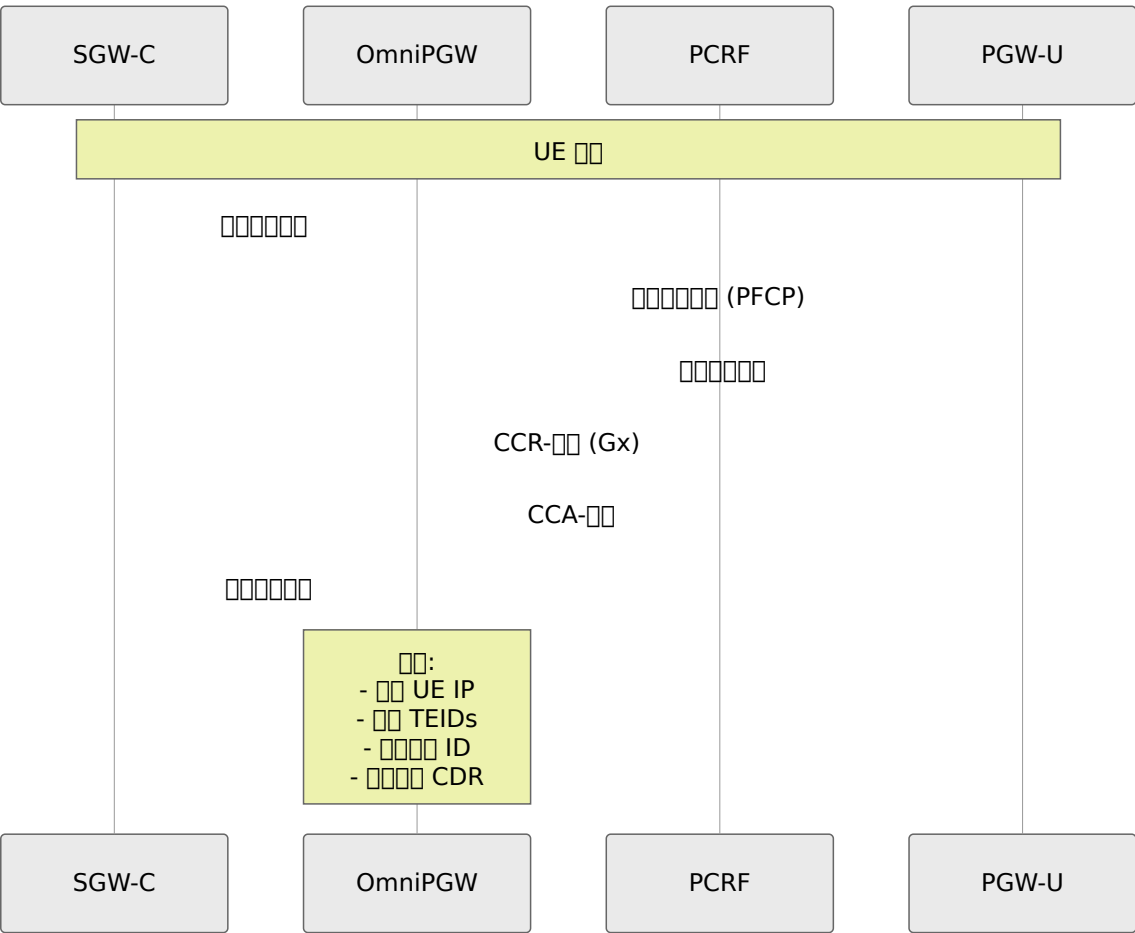
- **QoS** - PCRF
- - /
- - SGW
- - PCRF PCC

QoS



□□□□

□□□□



□□□□

□□□□□:

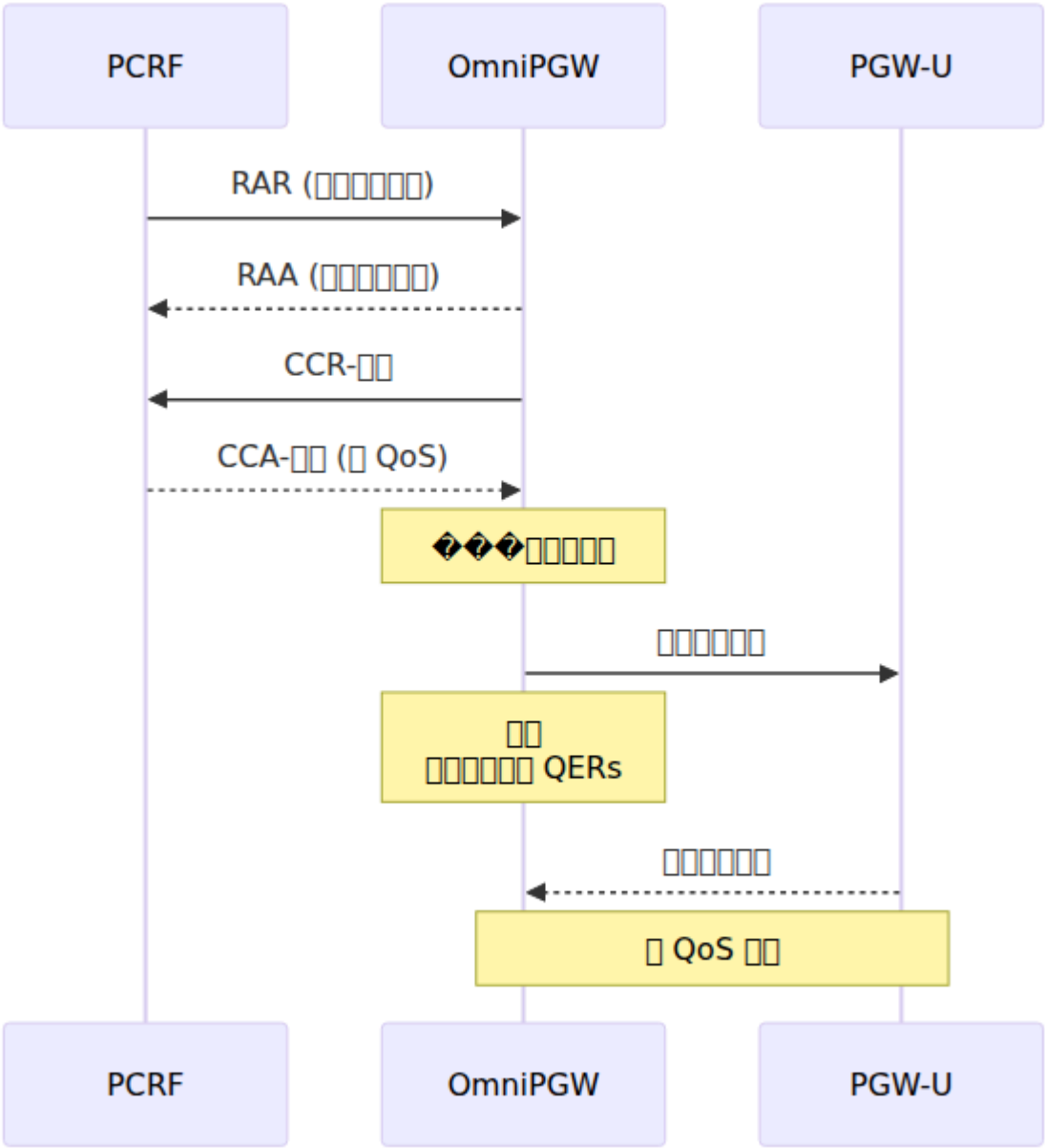
1. UE IP □□ → □□□□
2. TEID → □□□□□□
3. SEID → □□□□□□
4. □□-ID → □□□□□□
5. □□-ID → □□
6. □□□□□□

□□□□□□□:

- 3GPP CDR (3GPP) - 3GPP CDR 3GPP

3GPP

3GPP



3GPP

3GPP:

- ☐ TEID (S5/S8):
TEID 0x12345678 → ☐☐ PID
- ☐ SEID (Sxb):
SEID 0xABCDEF → ☐☐ PID
- ☐ ☐☐-ID (Gx):
"pgw.example.com;123;456" → ☐☐ PID
- ☐ UE IP:
100.64.1.42 → ☐☐ PID
- ☐ IMSI + EBI:
"310260123456789" + EBI 5 → ☐☐ PID

☐☐☐☐

☐☐☐☐☐☐

```
# ☐☐☐☐☐  
teid_registry_count  
  
# PFCP ☐☐  
seid_registry_count  
  
# Gx ☐☐  
session_id_registry_count
```

Queries

```
# Session creation rate
rate(s5s8_inbound_messages_total{message_type="create_session_request"})

# Session deletion rate
rate(s5s8_inbound_messages_total{message_type="delete_session_request"})

# Session creation (p95)
histogram_quantile(0.95,

rate(s5s8_inbound_handling_duration_bucket{request_message_type="create_session_request",
[5m])
)
```

Queries

Queries

Queries:

1. **IP** address - Session IP
2. **PCRF** address - Gx address
3. **PGW-U** address - Session PFPCP address
4. **PCRF** address - Session PFPCP address

Queries:

```
# IP address
curl http://pgw:9090/metrics | grep address_registry_count

# PCRF address
# Session Gx address

# PGW-U address
# PFPCP address
```


設定/項目

項目:

- 項目1
- 項目2
- 項目3

項目:

- 項目1
- 項目2
- 項目3

項目:

```
# 00 OmniPGW (項目1)
# 項目2
```

UE 項目

項目:

- UE 項目
- 項目3

項目:

項目	項目	項目
項目1	PCRF 項目 (IMSI 項目)	項目
項目2	IP 項目	項目 IP 項目
項目3	PCRF/PGW-U 項目	項目
項目4	項目 APN	項目 APN 項目

部署

部署

部署:

节点数: 10,000

节点内存: ~10KB RAM

节点 RAM: ~100MB

Erlang VM 配置:

- 节点数: 262,144 (节点)
- 节点数: 节点数

部署

部署:

1. 节点数
2. 节点数
3. 节点数

部署

部署:

- 节点数 (节点数)
 - 节点数 HA (节点)
 - DNS/节点数
-

□□□□□□

□□□□□□□□□□

□□□□ PDN □□□□□□□□

UE □□:

- IMSI: "310260123456789" (□□□□)
- MSISDN: "14155551234" (□□□□)
- MEI/IMEI: □□□□□

PDN □□□□□□:

- APN: "internet" (□□□□)
- UE IP □□: 100.64.1.42 (□□□ IP)
- PDN □□: IPv4, IPv6, □ IPv4v6

□□□□□:

- ID: □□□□□□□□
- EBI: EPS □□□□□ (□□□ 5)

QoS □□:

- APN-AMBR: □□□□□□□
 - : 100 Mbps
 - : 50 Mbps

□□□□:

- PDRs (□□□□□□□): □□□□□
- FARs (□□□□□□): □□/□□□□
- QERs (QoS □□□□): □□□□
- BAR (□□□□□□): □□□□

□□□□□:

- ☐ ☐ ☐ ☐ ☐ ☐

```
GET      /api/sessions                # 取得セッション
GET      /api/sessions?search=<imsi> # IMSI / MSISDN / IP 取得
GET      /api/sessions/<imsi>      # 取得セッション詳細
PUT      /api/sessions/<imsi>      # 更新セッション (セッションID)
DELETE   /api/sessions/<imsi>      # 削除セッション (セッションID)
```

```
# curl -k https://<omnipgw-ip>:8443/api/sessions

# curl -k "https://<omnipgw-ip>:8443/api/sessions?search=310170123456789"
```

```
GET /api/sessions?search=<imsi|msisdn|ip> # 
GET /api/sessions/<imsi> # 
```

search

- **MSISDN** (IMEI)
- **IP** 地址 (例如 "100.64.1.42")

```
# 创建会话
curl -k https://<omnipgw-ip>:8443/api/sessions/310170123456789
```

返回的响应如下：

a) 会话信息

- 会话标识符
- IMSI, MSISDN, UE IP 地址
- APN, RAT 类型
- PGW TEID, SGW TEID

b) 会话参数 和 会话管理信息

- **TAC** (跟踪区域码) - UE 所在区域
- **ECI ID (ECI)** - E-UTRAN 小区标识
- **ECGI** - E-UTRAN 小区标识 (PLMN + ECI)
- **MCC/MNC** - 移动国家码 / 移动网络码

c) 会话参数 和 会话管理信息 QoS 参数

返回的响应：

- EBI (EPS 承载标识)
- QCI (QoS 承载标识)
- 会话速率
- APN-AMBR (上行/下行)

返回的响应 (会话管理)：

- EBI, QCI, 会话速率
- MBR UL/DL (上行/下行)
- GBR UL/DL (上行/下行)

d) Gy (Gy)

- Gy ID
- ,
-

e) Gx (Gx)

- Gx ID
- PCRF /
- CC
- (PCC)

f)

- ()
- / /

:

-
-
- IP
-

:

GET	/api/sessions	# PDN
GET	/api/sessions?search=<term>	#
PUT	/api/sessions/<imsi>	#
DELETE	/api/sessions/<imsi>	#

: PDN GET /api/sessions

```
# 创建会话
curl -k https://<omnipgw-ip>:8443/api/sessions

# 查询 IMSI 列表, UE IP, MSISDN 和 APN 列表
curl -k "https://<omnipgw-ip>:8443/api/sessions?search=100.64"

# 更新会话
curl -k -X PUT https://<omnipgw-ip>:8443/api/sessions/310170123456789 # 更新
curl -k -X DELETE https://<omnipgw-ip>:8443/api/sessions/310170123456789 # 删除
```

返回结果 (返回结果):

- **IMSI** - IMSI
- **UE IP** - UE IP
- **SGW TEID** - SGW S5/S8 TEID
- **PGW TEID** - OmniPGW S5/S8 TEID
- **APN** - APN

过滤 (过滤 ?search=):

- IMSI (过滤 "310260")
- UE IP 列表 (过滤 "100.64")
- MSISDN / 列表
- APN 列表

返回结果 (GET /api/sessions/<imsi>):

- 返回结果 (IMSI, MSISDN, IMEI)
- 返回结果 (RAT 列表, 列表 MCC/MNC)
- QoS 列表 (AMBR 列表/列表)
- 返回结果 (列表 TEIDs)
- 返回结果 ID
- 返回结果 (列表)

API 호출 순서

OAM API 호출 순서: PFCP / UPF, Diameter (Gx/Gy)

```
GET /api/upf          # PFCP / UPF 호출
GET /api/diameter      # Diameter (Gx/Gy) 호출
```

```
# Diameter 호출
curl -k https://<omnipgw-ip>:8443/api/upf
curl -k https://<omnipgw-ip>:8443/api/diameter
```

결과:

- UPF 정보 (UPF ID, PCRF/OCS ID)
- Diameter (Gx/Gy) 정보
- Session ID

API 호출 순서

결과:

```
GET /api/session_history[?type=&search=&page=&page_size=]
```

결과: Session History

결과:

- `type` - Session Type (Session ID, Session ID)
- `search` - IMSI, MSISDN, IP Address, TEID
- `page`, `page_size` - 페이지 정보

```
# Session History 호출
curl -k "https://<omnipgw-ip>:8443/api/session_history?
type=session_deleted&search=310170123456789&page=1&page_size=50"
```


API endpoints:

- Sessions
- Sessions
- Sessions
- Sessions

API:

- Sessions (Sessions)
- Sessions
- Sessions
- Sessions
- Sessions

API

API:

1. Sessions
2. GET /api/sessions?search=<imsi|msisdn> Sessions
3. GET /api/sessions/<imsi> Sessions UE IP Sessions
4. Sessions QoS Sessions
5. Sessions

API:

- GET /api/sessions (teid_registry_count Sessions)
- Sessions
- APN Sessions (search=<apn>)

API:

- Sessions search=
- Sessions (GET /api/sessions/<imsi>) Sessions SSH/IE
- Sessions SGW PGW TEIDs Sessions
- Sessions PCRF Sessions AMBR Sessions

- DELETE /api/sessions/<imsi> 删除会话

返回结果:

- 返回结果 (JSON)
 - GET ?search= 搜索会话
 - 返回结果
 - 返回结果 (GET HTTPS 返回 JSON) 返回结果
-

接口

接口

- **PFCP** 接口 - PFCP 接口, PDRs, FARs, QERs, URRs
- **UE IP** 接口 - IP 地址, APN 接口
- **PCO** 接口 - 返回 UE 的 DNS, P-CSCF, MTU 接口
- **接口** - UPF 接口, 接口

接口

- **Diameter Gx** 接口 - PCRF 接口, PCC 接口, QoS 接口
- **Diameter Gy** 接口 - OCS 接口, 接口
- **接口 CDR** 接口 - 接口

接口

- **S5/S8** 接口 - GTP-C 接口, SGW-C 接口
- **QoS** 接口 - 接口 QoS 接口

接口

- **接口** - 接口, 接口, 接口
 - **P-CSCF** 接口 - IMS 接口
-



OmniPGW □□□□ - □ *Omnitouch* □□□□□□

OmniPGW

OmniPGW 架构图

OmniPGW 架构图

架构

- 核心网
- 接入网
- 用户设备
- PFCP / 会话管理
- Diameter (Gx/Gy) 接口
- IP 地址
- 网络

部署

OmniPGW 部署架构图

- 核心网
- 接入网
- 用户设备
- 网络
- 网络

部署

- 部署 - Prometheus 部署
- 部署 - 部署

□□□□□□

OAM API

□□ **URL:** `https://<omnipgw_ip>:8443` (HTTPS/TLS□□□□□□ `/api` ◆◆□□)

□□□ **API □□ (Swagger UI):** `https://<omnipgw_ip>:8443/api/docs`

□□□□□

- `GET /api/sessions[?search=<imsi|msisdn|ip>]`, `GET /api/sessions/<imsi>` - □□□□□□□□ IMSI□IP□MSISDN □□□
- `GET /api/diameter[?status=connected|disconnected]`, `GET /api/diameter/<origin-host>` - Diameter □□□□ (Gx PCRF, Gy OCS)
- `GET /api/charging[?search=<imsi>]`, `GET /api/charging/<imsi>` - □□□□ (Gy) □□
- `GET /api/upf`, `GET /api/upf/<ip>` - UPF / PFCP □□□□ (PGW-U □□□)
- `GET /api/ip_pools`, `GET /api/ip_pools/<name>` - IP □□□□
- `GET /api/session_history[?type=&search=&page=&page_size=]` - □□□□
- `GET /api/status` - □□□□ (`{"result":"ok"}`)

□□□

```
curl -k https://<omnipgw_ip>:8443/api/sessions
```

□□□□□□□□□□□□□□□□ (journalctl/□□□□) □□□□□□□

Prometheus □□

□□: `http://<omnipgw_ip>:9090/metrics`

□□□□□

- `teid_registry_count` - □□□□
- `address_registry_count` - □□□ UE IP
- `sxb_inbound_errors_total` - PFCP □□

- `gx_inbound_errors_total` - Diameter Gx
- `gy_inbound_errors_total` - Diameter Gy

이제 이걸로 테스트해보겠습니다

이제 ? ? ?

이제 이걸로 (`journalctl -u omnigw_c` 이걸로) `grep` 이걸로
`GET /api/session_history[?type=&search=&page=&page_size=]` 이

이걸로 테스트

- "create_session_request" - 이걸로
- "Credit Control" - Gx/Gy
- "PFCP Session" - 이걸로
- "error" 이 "ERROR" - 이걸로
- "timeout" - 이걸로

이제 이걸로

이제 이걸로 "이걸로"

이제

- SGW-C 이걸로 "이걸로" (73)
- 이걸로
- 이걸로
- 이걸로 [PGW-C] 이걸로 - 이걸로

Wireshark *포스팅을 할 때는 "캡처"로*

캡처

- 캡처할 OmniPGW 선택
- 캡처할 인터페이스

필터

1. *이름*

license_status

- *이름* 0 *이름*

2. *이름*

- *이름* "license" *이름* "License"
- *이름* "이름" *이름*

3. *이름* *이름* *이름*

- *이름* config/runtime.exs *이름* :license_client *이름* URL
- *이름* https://localhost:10443/api

캡처

1. 確認ライセンスサーバーの接続

```
curl -k https://<license_server_ip>:10443/api/status
```

2. 設定ファイル `config/runtime.exs` の確認

```
config :license_client,  
  license_server_api_urls:  
  ["https://<license_server_ip>:10443/api"],  
  licensee: "XXXXXX"
```

3. 接続先の確認

- 接続先 `omnipgwc`
- 接続先 Omnitouch の確認

4. 接続先 **OmniPGW**

確認

- 接続先 `license_status` の確認
- 接続先 Omnitouch の確認
- 接続先 Omnitouch の確認

接続先 Omnitouch の確認

確認

- SGW-C の確認
- 接続先 PDN の確認
- 接続先 `s5s8_inbound_errors_total` の確認

接続先

1. IP の確認
2. PCRF (Gx) の確認

3. PGW-U (PFCP) 확인

4. 확인 APN 확인

확인

1. 확인 IP 확인

```
address_registry_count
```

- 확인

2. 확인 PCRF 확인

- GET /api/diameter (GET /api/diameter?status=disconnected)
- PCRF = "disconnected"
- "Credit Control Answer" 확인

3. 확인 PFCP 확인

- GET /api/upf (GET /api/upf/<ip> 확인 PGW-U)
- "Association: DOWN"
- pfcpeer_associated = 0

4. 확인 APN 확인

- config/runtime.exs ue.apn_map
- 확인 APN 확인

확인

확인 IP 확인

1. GET /api/sessions (GET /api/sessions?search=<imsi|msisdn|ip>) 확인
2. config/runtime.exs IP 확인

```
config :pgw_c,
  ue: %{
    subnet_map: %{
      "internet" => "10.0.0.0/23" # /24 /23
    }
  }
}
```

3. OmniPGW

4. `curl http://<ip>:9090/metrics | grep address_registry_count`

PCRf

1. `ping <pcrf_ip>`
2. PCRf Diameter `telnet <pcrf_ip> 3868`
3. `config/runtime.exs` Diameter
4. OmniPGW
5. `GET /api/diameter?status=connected` "connected"

PFCP

- PFCP /

- IP 80%
- PCRf Diameter
-

- `GET /api/sessions`
-
-

1. PFCP 註冊 S5/S8 註冊
2. PCRF CCR-Initial
3. 註冊
4. 註冊

註冊

1. OAM API 註冊

- GET /api/sessions/<imsi> (GET /api/sessions?search=<imsi>)
- pfcpc_seid 註冊 PFCP
- gx_session_id 註冊 Gx

2. IMSI 註冊

- IMSI 註冊
- "PFCP"
- "Gx"
- 註冊

3. 註冊

```
# TEID PFCP 註冊
teid_registry_count - seid_registry_count

# TEID Gx 註冊
teid_registry_count - session_id_registry_count
```

註冊

1. PFCP 註冊

- PGW-U 註冊
- PFCP GET /api/upf
- SGW-C 註冊

2. Gx 註冊

- PCRF `histogram_quantile(0.95, rate(gx_outbound_transaction_duration_bucket[5m]))`
- `config/runtime.exs` Gx
-

3. OmniPGW

- OmniPGW
- `teid_registry_count`

- PFCP Gx
- /
-

PFCP /

PFCP

- `GET /api/upf` "Association: DOWN"
-
- `pfcpeer_associated = 0`
- "PFCP" " "

1. PGW-U
2. PGW-U
3. PFCP IP
4. UDP 8805

1. ping

```
ping <pgw_u_ip>  
nc -u -v <pgw_u_ip> 8805
```

2. PFCP

- `config/runtime.exs` `upf.peer_list`
- IP address ID PGW-U

3. PGW-U

- PGW-U
- PGW-U `systemctl status omnipgw_u`

4.

```
#  
pfcpc_consecutive_heartbeat_failures  
  
# PFCP  
rate(sxb_inbound_errors_total[5m])
```


1.

- `traceroute <pgw_u_ip>`
- UDP 8805
-

2. PGW-U

- PGW-U
- 30
- `GET /api/upf` "Association: UP"

3.

- 在 `config/runtime.exs` 中配置 PFCP 参数
- 在 OmniPGW 中
- 配置相关参数

配置

- 在 `pfcp_peer_associated` 中配置相关参数
- 在 `pfcp_consecutive_heartbeat_failures` 中配置 `> 2` 次失败
- 配置 PGW-U 参数
- 配置 PFCP 参数/相关配置

配置 PFCP 参数

配置

- 配置相关参数
- QoS 参数配置 PCRf RAR 参数
- 配置 "配置相关参数"
- 配置


```
sxb_inbound_errors_total{message_type="session_modification_response"}
```

配置

1. 配置 PFCP 参数 PDR/FAR/QR 参数
2. PGW-U 参数
3. 配置 ID 参数
4. PGW-U 参数

配置

1. 配置
 - 配置 "配置" 中 SEID
 - 配置 PFCP 参数
 - 配置 "配置 ID 参数" 中 "配置"

2. PGW-U

- PFCP
- CPU

3. OAM API

- GET /api/sessions/<imsi> (GET /api/sessions?search=<imsi>)
- pdr_map far_map qer_map
- ID

1.

-
- UE

2. PGW-U

- PGW-U PDR
- PGW-U
- PGW-U

3.

- (GET /api/sessions/<imsi>)
- PFCP
-

- PGW-U
 -
 - `sxb_inbound_errors_total`
-

Diameter (Gx/Gy) 配置

配置PCRF 配置Gx

配置

- GET /api/diameter (或 GET /api/diameter?status=disconnected) 返回 PCRF 返回 "disconnected"
- 配置 QoS 配置 QCI=5
- 配置 "Diameter 配置" 为 "CER/CEA 配置"

配置

1. PCRF 配置
2. PCRF 配置
3. Diameter 配置 Origin-Host 和 Realm
4. 配置 TCP 3868

配置

1. 配置

```
ping <pcrf_ip>
telnet <pcrf_ip> 3868
```

2. 配置 **Diameter** 配置

- 配置 config/runtime.exs 配置 diameter.peer_list
- 配置 host 和 realm 配置 IP 配置 PCRF 配置
- 配置 origin_host 配置 PCRF 配置

3. 配置 **PCRF** 配置

- 配置 PGW-C 配置 CER 配置
- 配置

4. 配置


```
# Diameter 配置
diameter_peer_connected{peer="<pcrf_host>"}
```

配置

1. 配置

- 配置 PCRF 配置
- 配置 TCP 3868
- 配置 `nc -v <pcrf_ip> 3868`

2. 配置 PCRF 配置

- 配置 PCRF 配置
- 配置 30 配置
- 配置 `GET /api/diameter?status=connected` 配置

3. 配置

- 配置 `config/runtime.exs` Diameter 配置

```
config :pgw_c,
  diameter: %{
    host: "pgw-c.epc.mnc999.mcc999.3gppnetwork.org", # 配置
    PCRF 配置
    realm: "epc.mnc999.mcc999.3gppnetwork.org",
    peer_list: [
      %{
        host: "pcrf.epc.mnc999.mcc999.3gppnetwork.org",
        realm: "epc.mnc999.mcc999.3gppnetwork.org",
        ip: "192.168.1.100",
        initiate_connection: true
      }
    ]
  }
```

- 配置 OmniPGW
- 配置

配置

- Diameter 認證
- PCRF 認證
- Diameter 認證

CCR/CCA Gx 認證

認證

- 認證 > 5 秒
- 認證 "認證"
- 認證 `gx_outbound_transaction_duration` 認證 > 5s
- 認證 QoS 認證

認證

1. PCRF 認證
2. PCRF 認證
3. 認證
4. PCRF 認證

認證

1. 認證 **Gx** 認證

```
# P95 認證
histogram_quantile(0.95,
rate(gx_outbound_transaction_duration_bucket[5m]))

# P99 認證
histogram_quantile(0.99,
rate(gx_outbound_transaction_duration_bucket[5m]))
```

2. 認證 **PCRF** 認證

- 認證 PCRF 認證
- 認證 CPU 認證
- 認證 PCRF 認證

3. ping 测试

```
ping -c 100 <pcrf_ip> | tail -1 # 测试结果
```

4. 配置

- 配置 CCR/CCA 配置文件 "config"
- 配置 "配置 CCR" 和 "配置 CCA" 配置文件

配置

1. 配置 PCRF 配置

- 配置 PCRF 配置文件
- 配置 CCR 配置
- 配置 PCRF 配置/配置

2. 配置

- 配置配置配置配置配置
- 配置 PGW-C 和 PCRF 配置

3. 配置配置配置配置配置

- 配置 config/runtime.exs

```
config :pgw_c,  
  diameter: %{  
    transaction_timeout_ms: 10000 # 配置 5000 配置  
  }
```

- 配置 OmniPGW
- 配置 配置配置配置配置配置

配置

- 配置 Gx 配置配置配置配置 > 1s配置 > 5s
- 配置配置配置配置 PCRF 配置
- 配置配置 PCRF 配置

□□□OCS □□□□□Gy□

111

- GET /api/diameter (GET /api/diameter?status=disconnected) OCS
"disconnected" GET /api/charging (Gy)
-
- "Gy "

□ □ □ □ □ □ □ □

PCRF 0.0001 Gy

□ □ □ □ □

- [illegible]

□□□ Diameter Gy □□ □□□ Gy □□□□□□□□

IP

□□□IP □□□

111

- 000000000000 "000000"
- 000 address_registry_count 00000000
- GET /api/sessions 00000000 GET /api/ip_pools 0000
- 000 "IP 00000000"

□ □ □ □ □

1.

2. 計算 IP 数
3. 計算
4. IP 数

計算

1. 計算

```
# 10.0.0.0/24 254 個 IP  
(address_registry_count / 254) * 100
```

2. 計算

- `config/runtime.exs` の `ue.subnet_map`
- `"10.0.0.0/24" = 254 個 IP`

3. 計算 IP 数

```
# 計算  
teid_registry_count  
address_registry_count
```

4. 計算

- `GET /api/sessions` と `GET /api/ip_pools/<name>` の結果
- 計算
- 計算

計算

計算

1. `config/runtime.exs` の `APN` の `CIDR` 数

```
config :pgw_c,
  ue: %{
    subnet_map: %{
      "internet" => ["10.0.0.0/22"] # 1022 IP /24 = 254
    }
  }
}
```

2. OmniPGW

3.

1. GET /api/sessions
2. SGW-C
3. PCRF/SGW
4. address_registry_count

- IP
 - > 70%
 - > 85%
-
-
-

IP

- UE IP
- "IP"
- GET /api/sessions IP

1. 準備環境
2. 準備環境
3. 準備環境

準備

1. 準備 OAM API 準備 IP

- `GET /api/sessions?search=<ip>`
- 準備環境 IMSI 準備 IP

2. 準備環境

- 準備 IP 準備
- 準備 "IP 準備" 準備

準備環境

1. 準備環境

- 準備 IMSI 準備 IP

2. 準備環境

- 準備 SGW-C 準備環境 IMSI
- 準備環境

3. UE 準備環境

- UE 準備環境
- 準備環境 IP

4. 準備環境

- 準備 OmniPGW 準備 IP 準備
- 準備環境

準備

- 準備環境
- 準備環境

如何配置 IPv6 的 APN 的 IPv4v6 的

如何

- UE 的 **IPv4v6** PDN 的 APN 的 `ue.subnet_map` 的 IPv4 CIDR
- 的 (SOS) APN 的 **IPv4-only** 的 PAA 的 IPv6 的 "的 APN 的 IPv6 的...的 IPv4-only"
- 的 APN 的 的 "APN 的 IPv6 的...的 IPv4v6 的"

如何

1. APN 的 IPv6 的 IPv4-only 的
2. APN 的 `ue.subnet_map` 的 IPv6 CIDR

如何

- IPv6 的 IPv6 的
- 的 (SOS) APN 的 `"sos"` 的 IPv4-only — 的 IPv4 的 IPv6 的 PDN 的 IPv4 的 P-CSCF 的 PCO 的 IMS 的
- 的 APN 的 /的

如何

1. 的 APN 的 `ue.subnet_map` 的 IPv6 CIDR 的 OmniPGW
2. 的 IPv4-only 的 IPv4-only 的 SOS APN 的

如何 APN 的 IPv6 的 IPv4v6 的 的 UE IP 的 → 的 4

□□□□

□□ Prometheus □□

```
# □□□□
teid_registry_count

# □□□□□□□□□□
rate(s5s8_inbound_messages_total{message_type="create_session_request

# IP □□□□□□□□ /24 □□□
(address_registry_count / 254) * 100

# P95 □□□□□□
histogram_quantile(0.95,
rate(s5s8_inbound_handling_duration_bucket{request_message_type="crea
[5m]))

# □□□
rate(s5s8_inbound_errors_total[5m])

# PCRF □□
histogram_quantile(0.95, rate(gx_outbound_transaction_duration_bucket

# PFCP □□□□
pfcf_peer_associated
```

□□□□□□□□

□□□□□□ grep □□□□□ (journalctl -u omnipgw_c □□□□□□□□□□)□□□□□□□□□□□□□□ GET
/api/session_history[?type=&search=] □□□□□□□□□□□□□□□□

項目	値
IMSI	15桁の整数
"create_session"	セッション作成
"delete_session"	セッション削除
"Credit Control"	Gx PCRF からの信用制御
"PFCP Session"	PFCP セッション
"error"	エラー
"timeout"	タイムアウト
"Association"	PFCP アソシエーション

確認方法

```
# 確認方法
systemctl status omnigw_c

# OAM API 確認(HTTPS/TLS)
curl -k https://<omnigw_ip>:8443/api/status

# 確認方法
curl http://<omnigw_ip>:9090/metrics

# 確認方法
curl http://<omnigw_ip>:9090/metrics | grep teid_registry_count

# PFCP 確認
curl http://<omnigw_ip>:9090/metrics | grep pfcpeer_associated

# IP 確認
curl http://<omnigw_ip>:9090/metrics | grep address_registry_count
```

□□□□

- □□□□ - Prometheus □□□Grafana □□□□□
- □□□□ - □□□□□□
- □□□□ - □□□□□□□□□□
- **PFCP** □□ - PFCP □□□□□□□□
- **Diameter Gx** □□ - Gx □□□□□□
- **Diameter Gy** □□ - Gy □□□□□□
- **QoS** □□□□□ - □ QoS □□□□□

□□□□□□

OmniPGW □□□□□□ - *Omni*touch □□□□□□

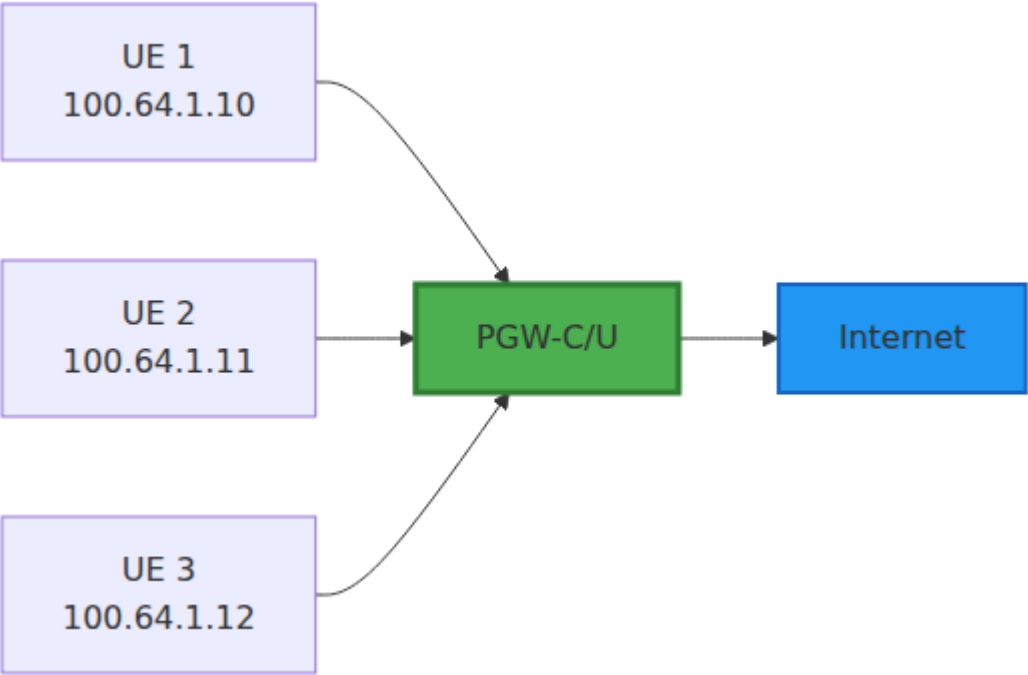
UE IP

IP

- 1.
2. IP
- 3.
- 4.
- 5.
- 6.
- 7.

PGW-C UE PDN IP

IP



UE PGW-C IP

-
-
-
- PDN

IP

IP		
IPv4		IPv4
IPv6		IPv6
IPv4v6		IPv4 IPv6

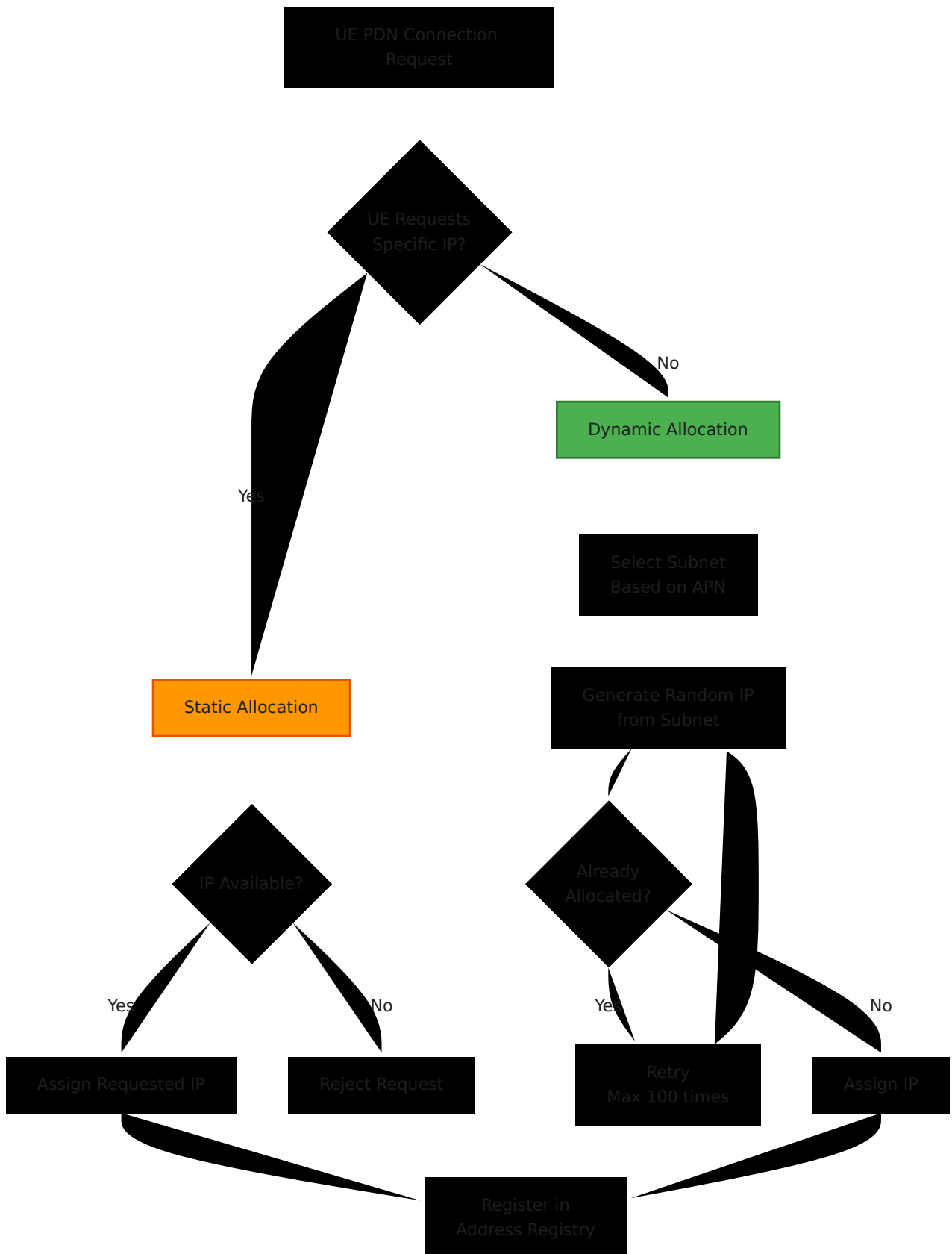
IP

PDN

UE PDN PDN

PDN		
IPv4	IPv4	IPv4
IPv6	IPv6	IPv6/64
IPv4v6		IPv4IPv6

PGW-C IP



1. IP Allocation

- PGW-C allocates IP
- IP Allocation

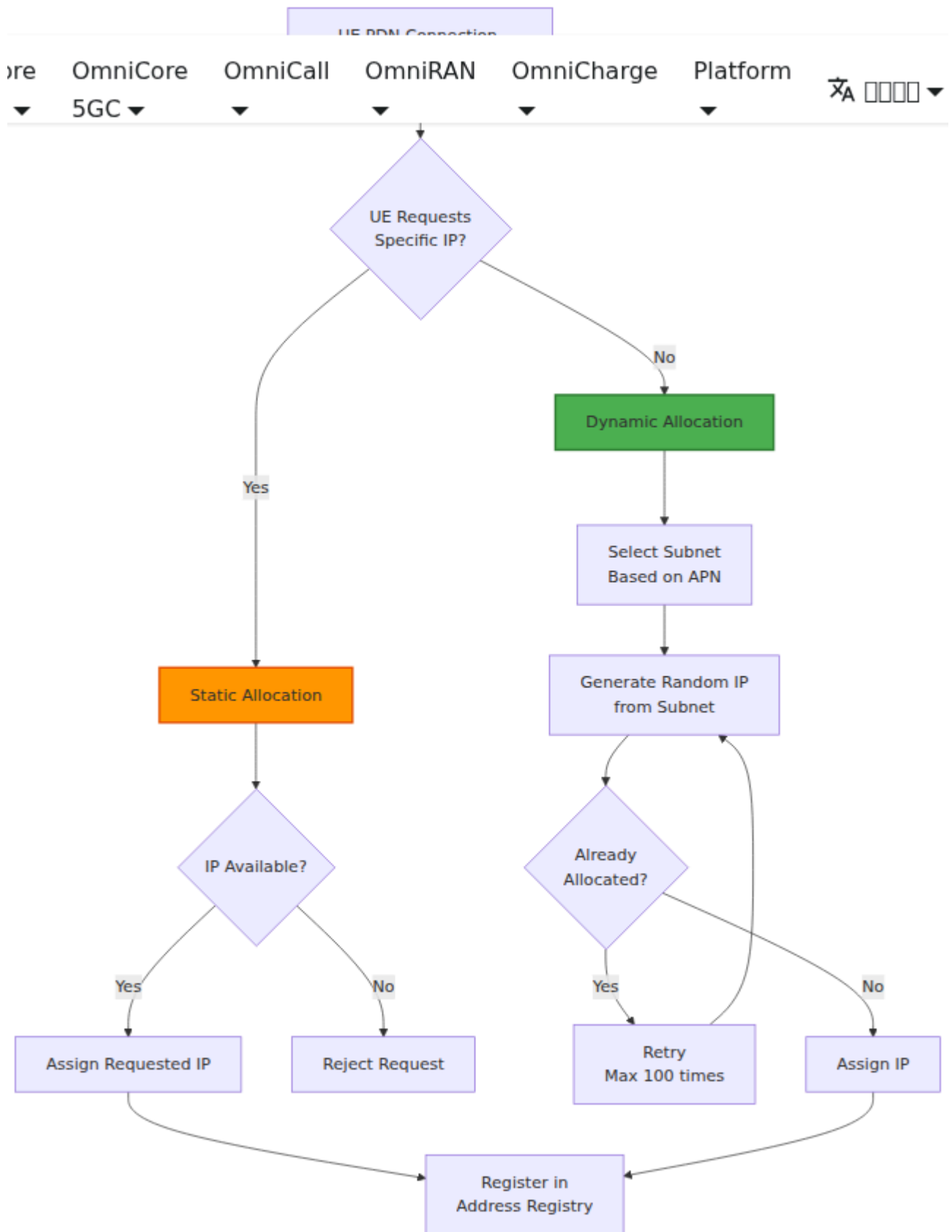
- 0000000000

2. 00000

- UE GTP-C 00000000IP
- PGW-C 00000
- 000000IP 000000000

00**APN**00000

000**APN0000000**0000000IP00



□□□

- □□□□ - □□APN□□□□□□□□
- □□□□□□ - □□APN□□□□□□□□

- 0000 - 0000000000000000
- 00 - 000000000000

000000

000000000000IP

00	00
00	00UE IP → 0000PID
00	00UE IP0000
00	00000000IP
0000	000000

00

0000

00config/runtime.exs

```

config :pgw_c,
  ue: %{
    subnet_map: %{
      # APN "internet"
      "internet" => [
        "100.64.1.0/24",    # 254 IP
        "100.64.2.0/24"    # 254 IP
      ],

      # APN "ims"
      "ims" => [
        "100.64.10.0/24"
      ],

      # Default APN
      default: [
        "42.42.42.0/24"
      ]
    }
  }
}

```

APN

APN is a configuration parameter that defines the network access point for a user equipment (UE) to connect to the internet or other services.

APN configuration

- APN configuration is defined in the UE configuration file.
- APN configuration is defined in the UE configuration file.
- APN configuration is defined in the UE configuration file.
- APN configuration is defined in the UE configuration file.

```

config :pgw_c,
  ue: %{
    subnet_map: %{
      # 物联网"ims"APN
      "ims" => ["ims.apn" => ["ims.something.else"
        "ims" => [
          "100.64.10.0/24"
        ],

        # 物联网"m2m."APN"m2m.test"m2m.prod"
        "m2m\" => [
          "100.64.20.0/24"
        ],

        # 企业 - "enterprise.corp"
        "enterprise.corp" => [
          "10.100.0.0/16"
        ],

        # APN
        default: [
          "42.42.42.0/24"
        ]
      }
    }
  }

```

物联网

- 物联网Elixir/Erlang
- Elixir \ \
- ^
-
- 企业 -

物联网

Regex	Pattern	Matches
^ims	"^ims"	ims, ims.apn, ims.anything
^.*\.corp\$	"^.*\.corp\$"	foo.corp, bar.corp
test	"^.*test.*"	test, foo.test.bar, testing
internet.apn	"^internet\.apn\$"	internet.apn

Regex

Regex: ^.*\.suffix\$

```
subnet_map: %{\n  # ".corp" APN\n  "^.*\\.corp$" => ["10.100.0.0/16"],\n\n  # ".iot" APN\n  "^.*\\.iot$" => ["10.200.0.0/16"],\n\n  default: ["42.42.42.0/24"]\n}
```

Regex

APN	Regex	IP Range
ims	^ims	100.64.10.0/24
ims.apn	^ims	100.64.10.0/24
ims.something.else	^ims	100.64.10.0/24
m2m.test	^m2m\.	100.64.20.0/24
m2m	default	42.42.42.0/24
enterprise.corp	enterprise.corp	10.100.0.0/16
foo.corp	^.*\.corp\$	10.100.0.0/16
unknown.apn	default	42.42.42.0/24

IP Address

CIDR <network>/<prefix_length>

CIDR	IP Count	IP Range
/24	254	100.64.1.1 - 100.64.1.254
/23	510	100.64.0.1 - 100.64.1.254
/22	1022	100.64.0.1 - 100.64.3.254
/20	4094	100.64.0.1 - 100.64.15.254
/16	65534	100.64.0.1 - 100.64.255.254

IP

- 100.64.1.0/24
- 100.64.1.255/24
- PGW-C $\langle \text{network} \rangle + 1 \langle \text{broadcast} \rangle - 1$

APN

```
config :pgw_c,
  ue: %{
    subnet_map: %{
      "internet" => [
        "100.64.1.0/24",
        "100.64.2.0/24",
        "100.64.3.0/24",
        "100.64.4.0/24"
      ]
    }
  }
}
```

- PGW-C
-
-

-
-
-

□□□□□□

```
config :pgw_c,  
  ue: %{  
    subnet_map: %{  
      # □□□□□□  
      "internet" => [  
        "100.64.0.0/20"      # 4094□IP□□□□□  
      ],  
  
      # IMS□LTE□□□  
      "ims" => [  
        "100.64.16.0/22"    # 1022□IMS IP  
      ],  
  
      # □□APN  
      "enterprise.corp" => [  
        "10.100.0.0/16"     # 65534□□□IP  
      ],  
  
      # IoT□□□□□□□□  
      "iot.m2m" => [  
        "100.64.20.0/22"   # 1022□IoT IP  
      ],  
  
      # □□□□  
      default: [  
        "42.42.42.0/24"    # 254□□□APN□IP  
      ]  
    }  
  }  
}
```


IPv6

```
config :pgw_c,  
  ue: %{  
    subnet_map: %{  
      "internet" => [  
        # IPv4  
        "100.64.1.0/24"  
      ],  
      "internet.ipv6" => [  
        # IPv6  
        "2001:db8:1::/48"  
      ],  
      default: [  
        "42.42.42.0/24"  
      ]  
    }  
  }  
}
```

IPv6

- UE/64
- UEIP
- UE 2001:db8:1:a::/64

IPv4v6

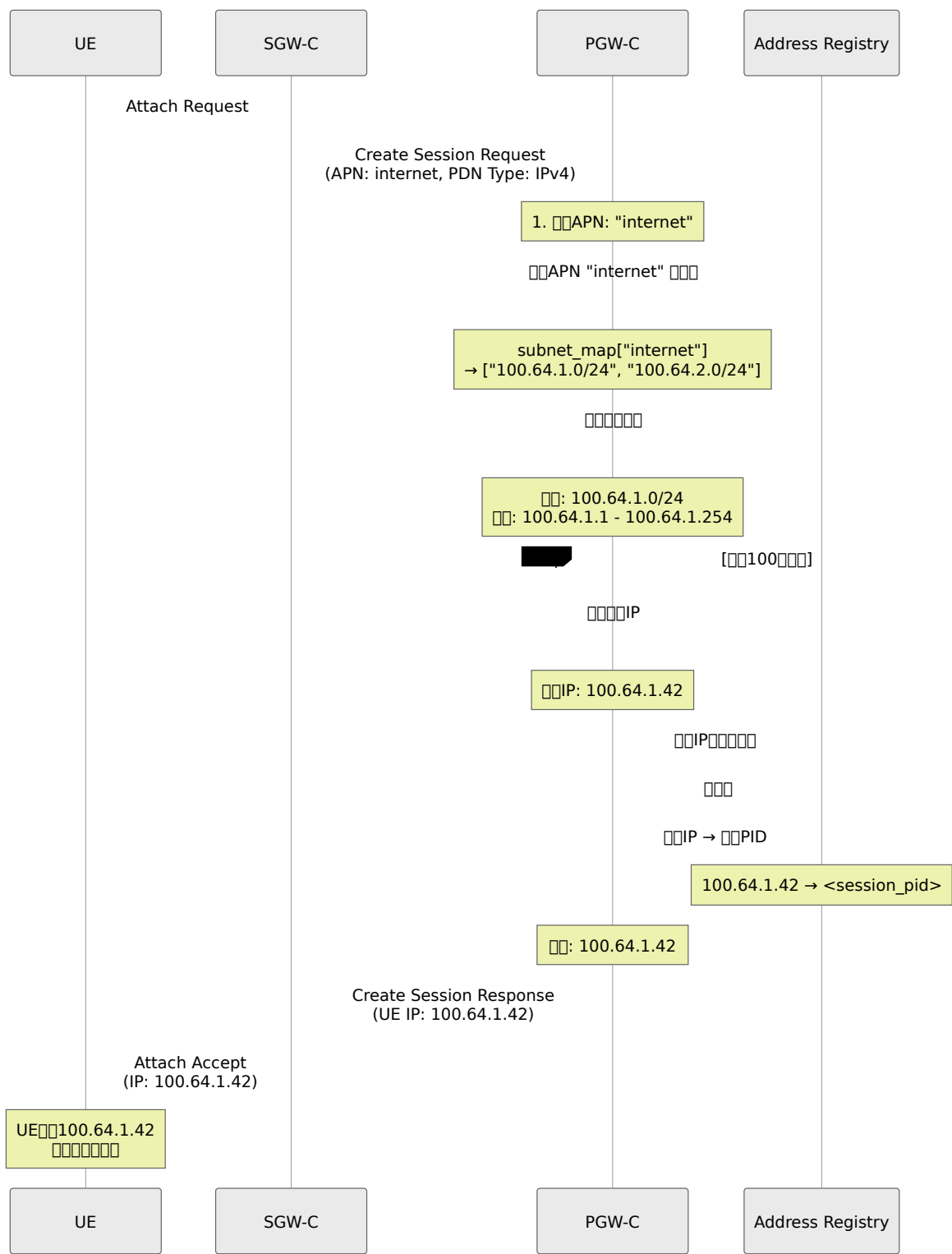
```
config :pgw_c,  
  ue: %{  
    subnet_map: %{  
      "internet" => [  
        "100.64.1.0/24",      # IPv4  
        "2001:db8:1::/48"    # IPv6  
      ],  
    }  
  }  
}
```

- UEのPDNはIPv4v6
 - PGW-CはIPv4とIPv6
 - 接続
-

接続

IPはPGW-CのS5/S8とGTP-CのS5/S8で接続

IPv4



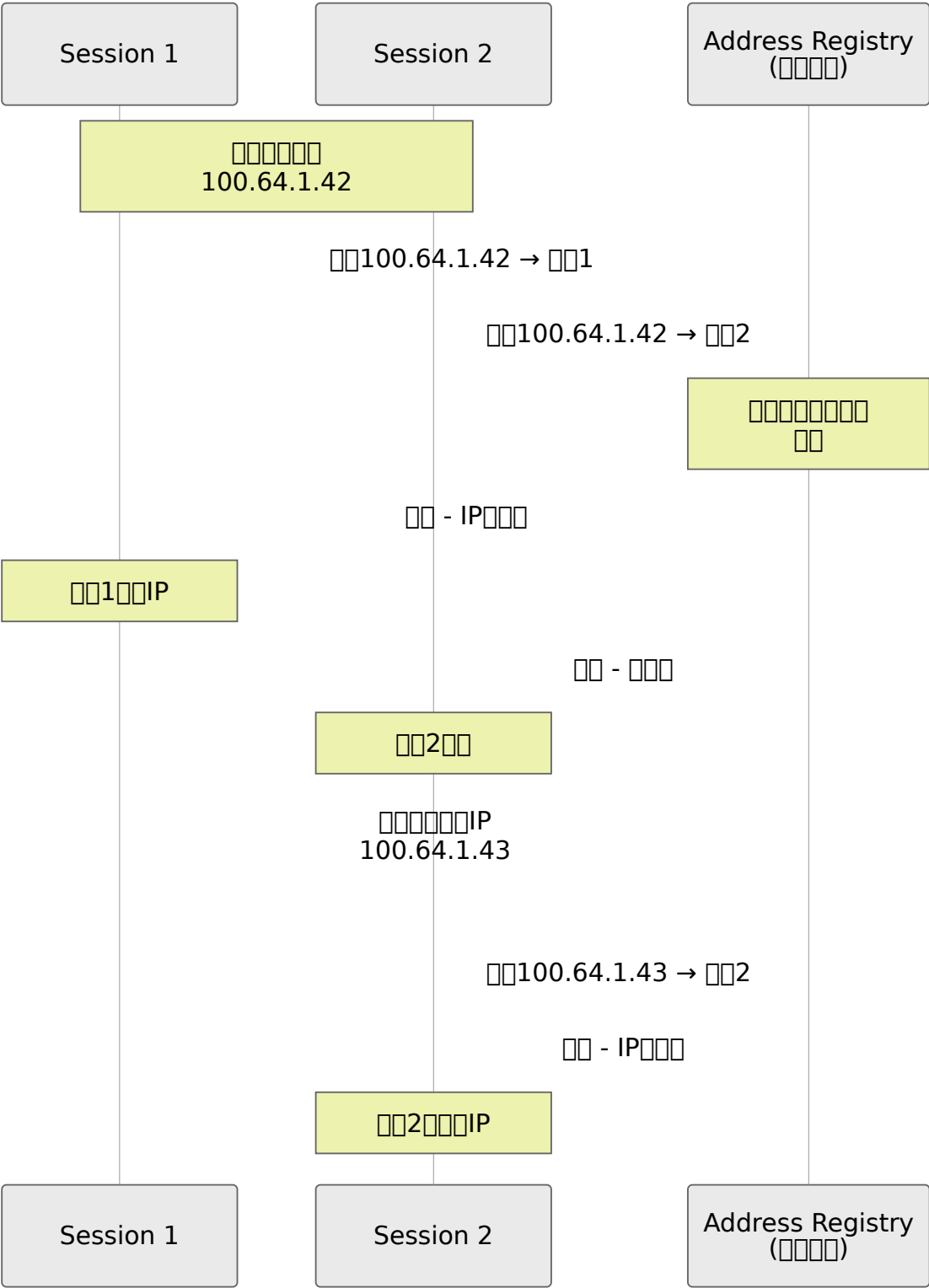
1. 設定した 設定項目のAPNを確認
2. 設定した 設定項目を確認
3. **IP**項目 設定項目を確認IP
4. 設定した 設定項目IPを確認
5. 設定した 設定項目を確認100設定項目を確認IP
6. 設定 設定項目IPを確認

設定項目を確認

- 設定**100**項目 設定項目を確認
- 設定した 設定項目IPを確認
- 設定した 設定項目を確認
- 設定した 設定項目を確認APNを確認

設定項目

設定 設定項目IP



100.64.1.42

- 100.64.1.42
- 100.64.1.42
- 100.64.1.42
- 100.64.1.42

- 100.64.1.0/24 100.64.1.1 IP

100.64.1.1

UE 100.64.1.1 APN

100.64.1.1

```
# 100
subnet_map: %{
  "internet" => ["100.64.1.0/24"],
  default: ["42.42.42.0/24"]
}
```

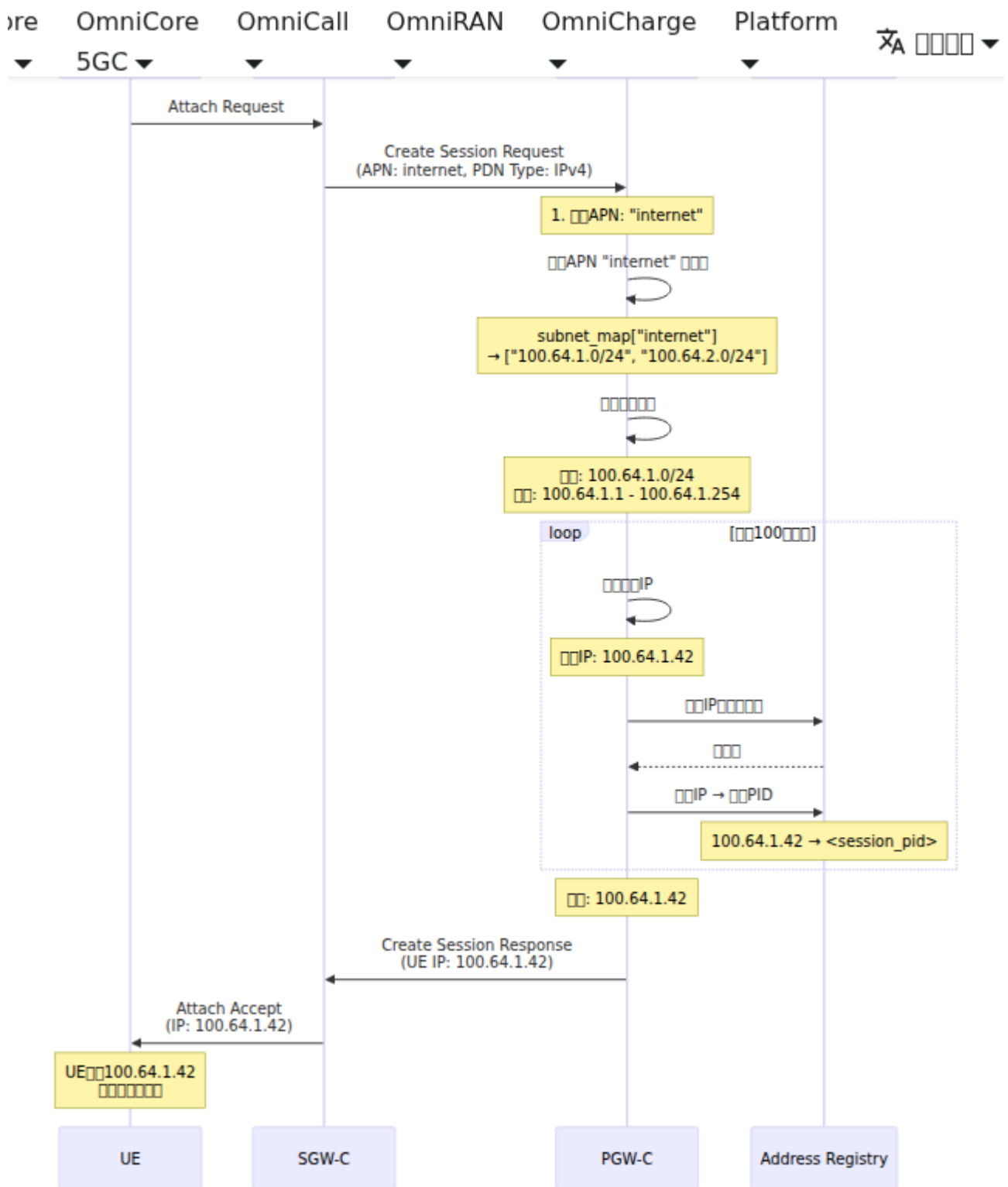
100

- UE 100 APN: "unknown.apn"
- 100 subnet_map 100 "unknown.apn"
- 100.64.1.1
- 42.42.42.0/24 100 IP

100.64.1.1

1. 100.64.1.1 APN 100
2. 100.64.1.1 default
3. 100.64.1.1

5G Core Network



Key Points

- The PGW-C is responsible for IP address allocation.
- The IP address is allocated from the subnet map.
- The IP address is allocated from the Address Registry.

IP Pool

Overview

IP Pool Management

IP: 100.64.1.0/24 (254 usable IP)

Pool: 254 IP

Pool ID → IP

Configuration

- PGW-C pool 100 IP
- IP Pool Management
- IP: `{:error, :ue_ip_address_allocation_failed}`
- IP Pool
- SGW-C pool 100 IP

Configuration

```
# IP Pool Management
address_registry_count / total_pool_size > 0.8 # 80% full

# IP Pool Configuration
"internet" => [
  "100.64.1.0/24",
  "100.64.2.0/24", # IP Pool
  "100.64.3.0/24"
]
```

IP Pool

IP Pool Management

GTP-C Pool

UE Context

└ IMSI: 310260123456789
└ APN: enterprise.corp
└ PDN Type: IE
| └ PDN Type: IPv4
| └ IPv4 Address: 10.100.0.50 ← UE IP

OmniPGW

1. UE IP → PDN Type
2. UE IP → APN
3. UE IP → PDN Type
4. UE IP
 - PDN Type
 - PDN Type

UE Context

- UE IP → PDN Type
- UE IP → PDN Type - IP
- UE IP → PDN Type - IP

IPv6

UE IPv6

UE Context

└ PDN Type: IPv6

PGW-C/64

UE: 2001:db8:1:a::/64

UE

- 2001:db8:1:a::1
- 2001:db8:1:a::2
- ...18

- UE IP
- SLAAC
- NAT

UE IPv4v6

PDN: IPv4v6

PGW-C

IPv4: 100.64.1.42

IPv6: 2001:db8:1:a::/64

- IPv4 IPv4
- IPv6 IPv6
-
- GTP

APN IPv6

IPv4v6 PDN APN IPv4 CIDR IPv6 IPv6 {error, :no_ipv6_pool} APN

APN	
SOS APN "sos"	IPv4 — IPv4IPv6PDNIPv4 PAAPFCPGxIPv4P-CSCFPCOIMS TS 29.274 / TS 23.401 / PDN
APN	—

IPv6APNIPv6APNIPv6IPv4v6

IP

IPRFC 1918

```
#  
subnet_map: %{\n  "internet" => [\n    "10.0.0.0/8",\n    "172.16.0.0/12",\n    "192.168.0.0/16"\n  ]\n}
```

PGW-UNAT

IP

```
#  
subnet_map: %{\n  "internet" => [\n    "203.0.113.0/24" # IP\n  ]\n}
```

NAT -

- 使用IP RFC 6598 100.64.0.0/10 NAT
- 使用私有IP

二

IP

IP OAM REST API HTTPS/TLS 8443 /api Swagger UI https://<omnipgw-ip>:8443/api/docs

GET /api/ip_pools # +
GET /api/ip_pools/<name> #

三

curl -k https://<omnipgw-ip>:8443/api/ip_pools
curl -k https://<omnipgw-ip>:8443/api/ip_pools/Internet

四

五	六
七	/APN “default” “ims.something.else” “Internet”
八	IP
九	IP
十	IP
十一%	

五

- 1000000000000
- 1000000000
- 100000
- 1APN1000000

10000

100000000

```
# 100000000IP
address_registry_count

# 1000000000000
address_registry_count / <total_pool_size> * 100
```

1000

```
1: 100.64.1.0/24 (254IP)
1000: 150IP
1000: 150 / 254 = 59%
```

00

```
# 0000000000
- alert: UEIPPoolUtilizationHigh
  expr: address_registry_count > 200 # 00/240
  for: 10m
  annotations:
    summary: "UE IP00000080%"
    description: "00: {{ $value }} / 2540IP000"

# 00000000
- alert: UEIPPoolExhausted
  expr: address_registry_count >= 254 # 00/240
  for: 1m
  annotations:
    summary: "UE IP000 - 0000IP"

# 0000000000
- alert: UEIPAllocationFailures
  expr: rate(ue_ip_allocation_failures_total[5m]) > 0
  for: 5m
  annotations:
    summary: "UE IP00000000"
```

Grafana000

0010IP0000

```
# 0000000000
(address_registry_count / 254) * 100
```

00200000000000IP

```
# 0000
address_registry_count
```

00300000

```
# 计算速率
rate(address_registry_count[5m])
```

每4分钟更新一次

```
# 计算地址注册量占总数量的比例
(254 - address_registry_count) / rate(address_registry_count[1h])
```

地址注册量

每1分钟更新一次地址注册量IP

地址注册量

- 地址注册量“地址注册量”
- 地址“UE IP地址注册量”

地址注册量

1. 地址注册量

```
# 计算速率
curl http://<pgw_c_ip>:42069/metrics | grep
address_registry_count
```

2. 地址注册量

```
# 配置
config :pgw_c,
  ue: %{
    subnet_map: %{
      "internet" => [
        "100.64.1.0/24" # 公网CIDR
      ]
    }
  }
}
```

3. APN配置

```
# 配置APN
# 配置
subnet_map: %{
  default: ["42.42.42.0/24"]
}
```

配置

- 配置 配置
- 配置 PGW-C 配置 IP
- 配置 `runtime.exe` 配置

2 IP

配置

- UE 配置 IP
- 配置

配置

- 配置

配置


```
# 確認IP
grep "already_registered" /var/log/pgw_c.log
```

確認

- 確認
- 確認

3 IP

確認

- UE IP
- APN "internet" "ims" IP

確認

- subnet_map

確認

```
# APN
subnet_map: %{
  "internet" => [...],    #
  "Internet" => [...],    # APN
}
```

確認

- APN
- 確認

4 IPv6

確認

- UE IPv6

- UE가 **IPv4v6** PDN을 APN을 IPv6로 설정하면 **IPv4**가 SOS APN로 설정되면 APN

다음과 같이

1. IPv6

```
# IPv6
subnet_map: %{
  "internet" => [
    "100.64.1.0/24" # IPv4
  ]
}
```

2. IPv6

```
# 2001::/48
"internet" => [
  "2001:db8::/128" # IPv6 - 2001::/48
]
```

IPv6를 IPv4v6로 설정하면

IPv4v6로 설정하면 IPv6를 APN에 register_new_ipv6/1 또는 {error, :no_ipv6_pool}로 설정하면

- **SOS APN**에 "sos"로 설정하면 **IPv4** — IPv4를 IPv6로 설정하면 PDN을 IPv4로 설정하면 P-CSCF PCO로 설정하면
- **APN**에 {error, :no_ipv6_pool} — IPv6로 설정하면

APN을 IPv6로 IPv4v6로

다음과

```
# IPv6-only APN
subnet_map: %{
  "internet" => [
    "100.64.1.0/24",
    "2001:db8:1::/48" # IPv6
  ]
}
```

IPv4-only APN IPv4-only — SOS PDN IPv6 CIDR

5

❓❓❓

-
- `address_registry_count`

1.

```
"internet" => [
  "100.64.1.0/24", # 
  "100.64.2.0/24", # 254 IP
  "100.64.3.0/24" # 254 IP
]
```

2.

```
# /24 /22
"internet" => [
  "100.64.0.0/22" # 1022 IP
]
```

3.

-
-

□□□□

□□□□

□□□□□□□□

□□□□□□10,000
□□□□30%□3,000□□□□□□
□□□□50%
□□IP□3,000 * 1.5 = 4,500□IP

□□□/20□4,094□□□IP□ - □□
□□□/19□8,190□□□IP□ - □□

□□□□

□□□

- □□ 100.64.0.0/10 □RFC 6598 - □□□□NAT□
- □□400□□IP
- □□□□□□□□NAT

□□□

- □□IP□□□□□□□□
- □□□VPN□□□□□□□□□□

配置

```
config :pgw_c,  
  ue: %{  
    subnet_map: %{  
      # 互联网APN - 互联网  
      "internet" => [  
        "100.64.0.0/18" # 16,382 IP  
      ],  
  
      # IMS - 互联网  
      "ims" => [  
        "100.64.64.0/22" # 1,022 IP  
      ],  
  
      # 企业 - 互联网  
      "enterprise.corp" => [  
        "100.64.68.0/22" # 1,022 IP  
      ],  
  
      # IoT - 互联网  
      "iot.m2m" => [  
        "100.64.72.0/20" # 4,094 IP  
      ],  
  
      # 默认 - 互联网  
      default: [  
        "100.64.127.0/24" # 254 IP  
      ]  
    }  
  }  
}
```

配置

配置

- **配置** - UE IP地址APN配置
- **PCO** - IP地址DNS P-CSCF MTU

- **N4** - 连接PDN的IP
- **PFCP** - 连接PFCP和UPF的UE IP

网络

- **S5/S8** - 连接GTP-C的IP
- **Diameter Gx** - IP地址

IP

- **N4** - IP地址
- **CDR** - 连接CDR的UE IP

网络