□级路由

□级路由允许 DRA 根据每个 Diameter 请求的内容做出路由决策。它不仅仅依赖于 目标域。规则 匹配消息的属性，例如发起对等体、应用程序 ID、命令代码和 AVP 值。当规则匹配时，它将请求引导到选定的对等体或特殊目的地。

规则还可以在多个对等体之间分配流量以实现 故障转移 和 负载均衡。它使用每个对等体的 `priority` 和 `weight`。对于状态保持应用程序，例如 Gx (PCRF) 和 Gy (OCS)，□级路由保持 会话亲和性。会话中的消息始终返回到拥有会话的对等体。

□级路由仅评估 入站 **Diameter** 请求数据包。响应数据包遵循已建立的会话路由返回到发起对等体。DRA 从不重新评估它们以符合这些规则。要重写消息内容，请参见 □级转换。有关在没有规则匹配时运行的基于域/主机的路由，请参见 标准路由。

工作原理

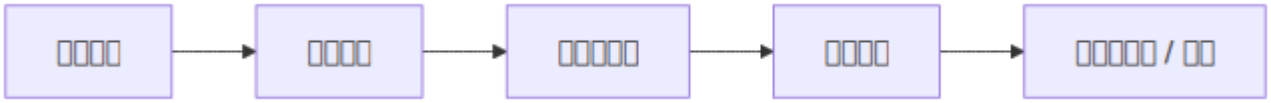
DRA 按照配置的规则 从上到下 逐个评估每个入站请求。第一个匹配的规则获胜并决定路由。DRA 不会评估后面的规则。如果没有规则匹配，请求将继续使用标准 目标主机 / 目标域 路由。

匹配的规则只能解析到 当前未连接 的对等体。在这种情况下，DRA 用 **DIAMETER_UNABLE_TO_DELIVER (3002)** 回答。DRA 从不将请求交还给标准路由。标准路由会将其交付给错误的节点并返回误导性结果。

流水线位置

□级路由在漫游引导和订阅者查找之后运行。它的决策覆盖标准路由本来会选择的对等体。

□□□□	□□□□	□□□□□	□□□□	□□□□□ / □□



规则处理

DRA 从上到下评估规则。第一个匹配的规则获胜。每个规则根据其 `match` 模式组合其过滤器。一旦匹配，它就会应用其 `route` 操作。如果没有规则匹配，DRA 将回退到标准路由。

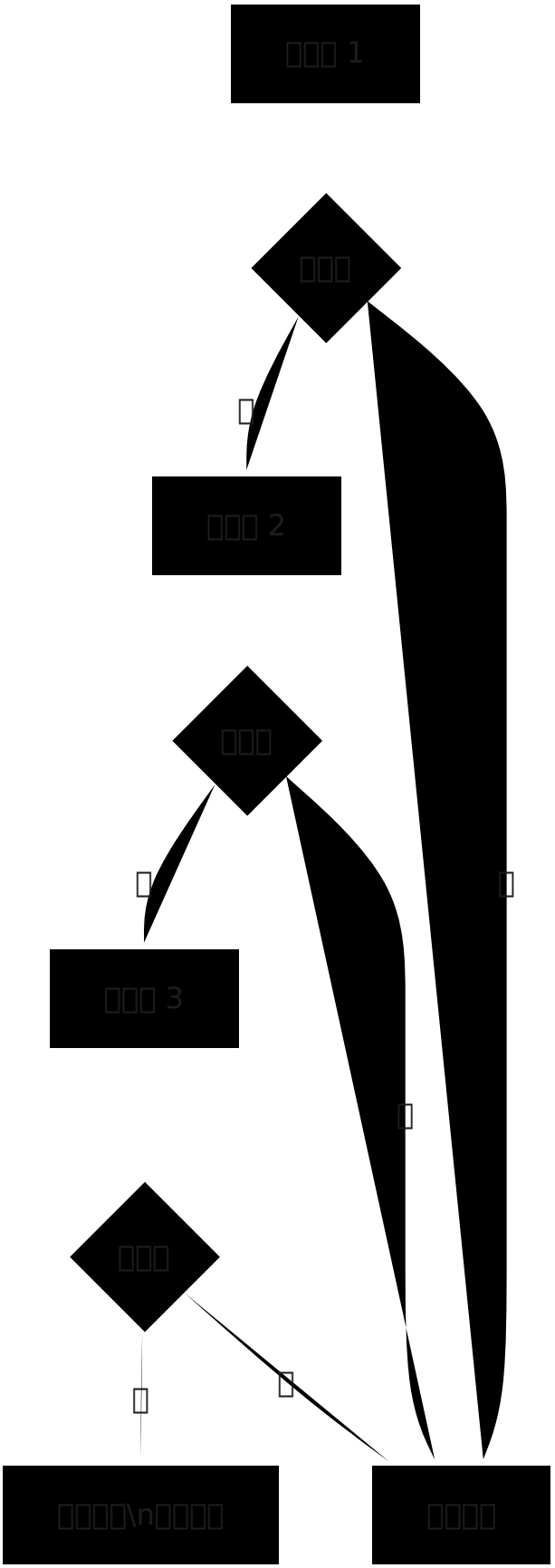
1. DRA 按照配置中定义的 从上到下 的顺序评估规则。
2. DRA 根据规则的 `match` 模式（`:all`，`:any`，或 `:none`）组合规则内的过滤器。
3. 第一个匹配的规则获胜。DRA 不会评估后面的规则。
4. 如果没有规则匹配，DRA 使用标准路由。

匹配模式

`match` 参数决定 DRA 如何组合规则的过滤器。

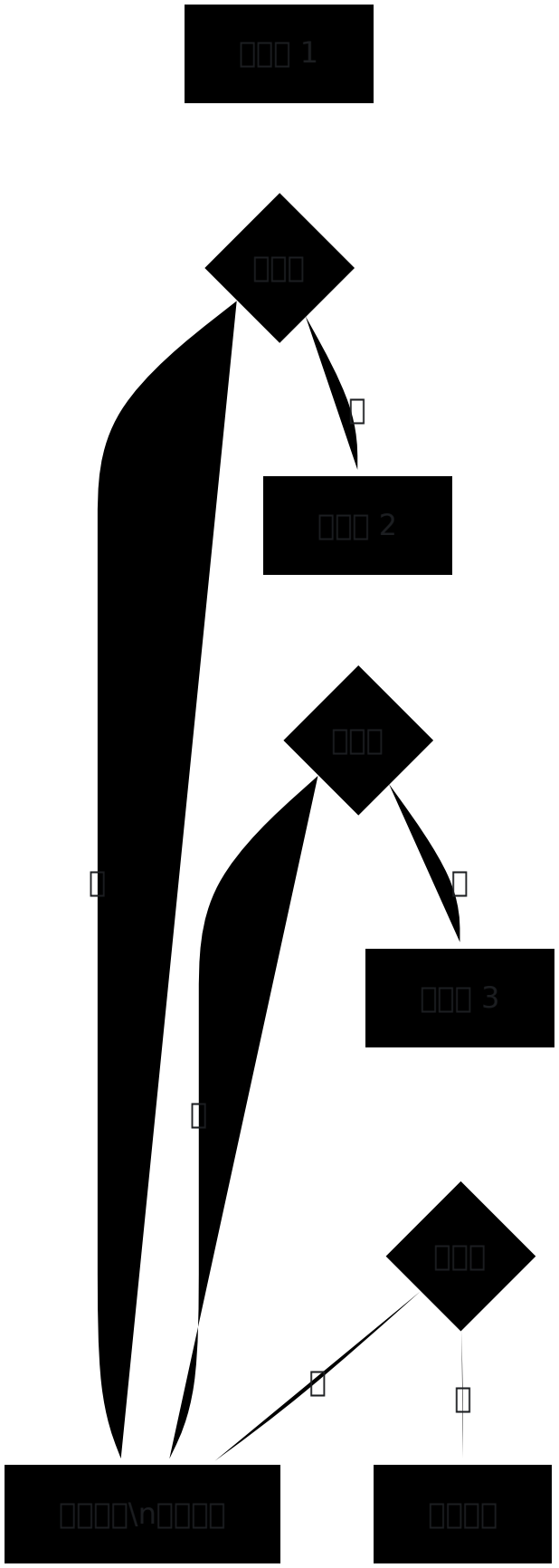
`:all` (与逻辑)

每个过滤器必须匹配才能使规则成功。对于三个过滤器，`filter1 AND filter2 AND filter3` 必须都为真。



是
是
是

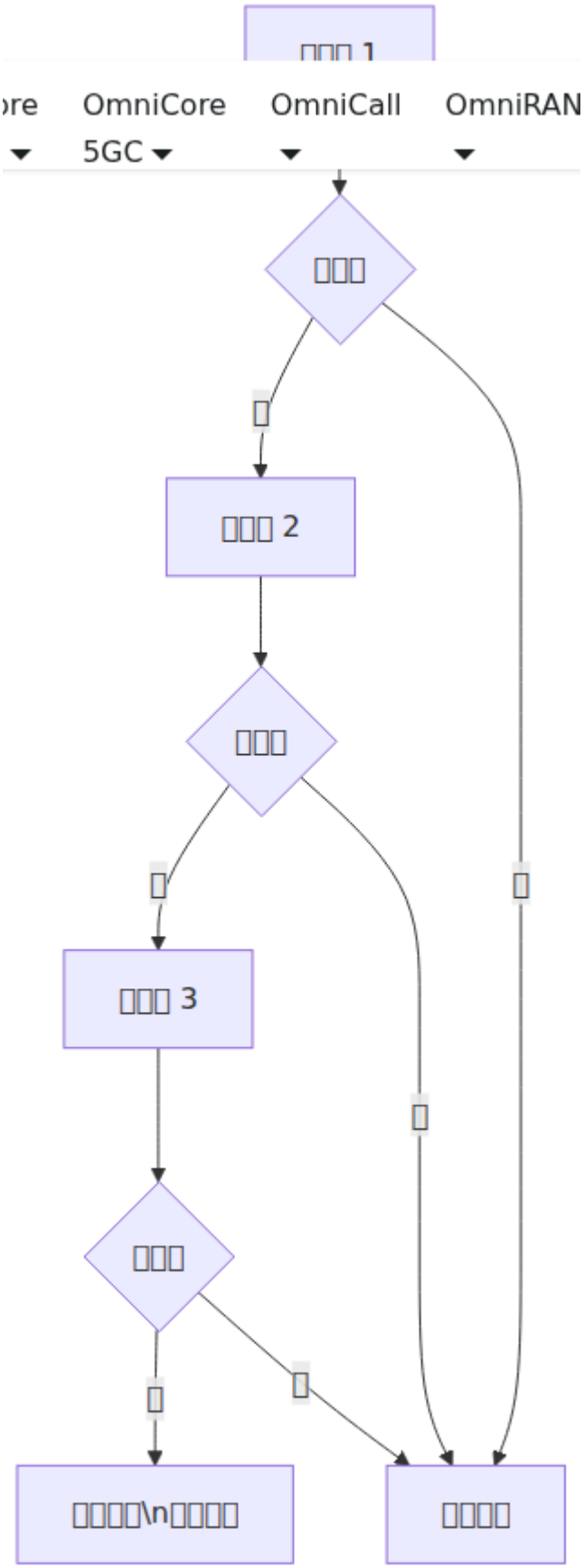
`:any` (或逻辑)



至少一个过滤器必须匹配。对于三个过滤器，`filter1 OR filter2 OR filter3` 如果任何一个通过则成功。

`:none`（非逻辑）

没有过滤器可以匹配。这是反向匹配。对于三个过滤器，`NOT filter1 AND NOT filter2 AND NOT filter3` 必须都为真才能使规则成功。每个过滤器必须失败。


□□□ 1
ore
OmniCore
5GC
OmniCall
OmniRAN
□□□
□
□□□ 2
□□□
□
□
□
□□□ 3
□
□□□
□
□
□□□□\n□□□□
□□□□

对等体选择

当规则路由到对等体列表时，DRA 仅在 当前连接 的对等体中进行选择。您编写对等体的方式决定了 DRA 如何在它们之间选择。

简单主机名

一组简单主机名跟随在 `diameter` 下配置的全局 `peer_selection_algorithm`（`:failover` 或 `:random`）之后。这是原始行为，不会改变。您必须在 DRA 的 Diameter 对等体配置中定义每个对等体。每个对等体的名称必须是精确 的主机名，区分大小写。每个对等体必须是当前连接的。DRA 会跳过未连接的对等体。

```
route: %{peers: ["hss01.example.com", "hss02.example.com"]}
```

- `:failover` - 始终选择列表中的第一个连接对等体。这提供了确定性的主对等体，然后是备用对等体。
- `:random` - 随机选择一个连接对等体。这均匀分配负载。

优先级和权重

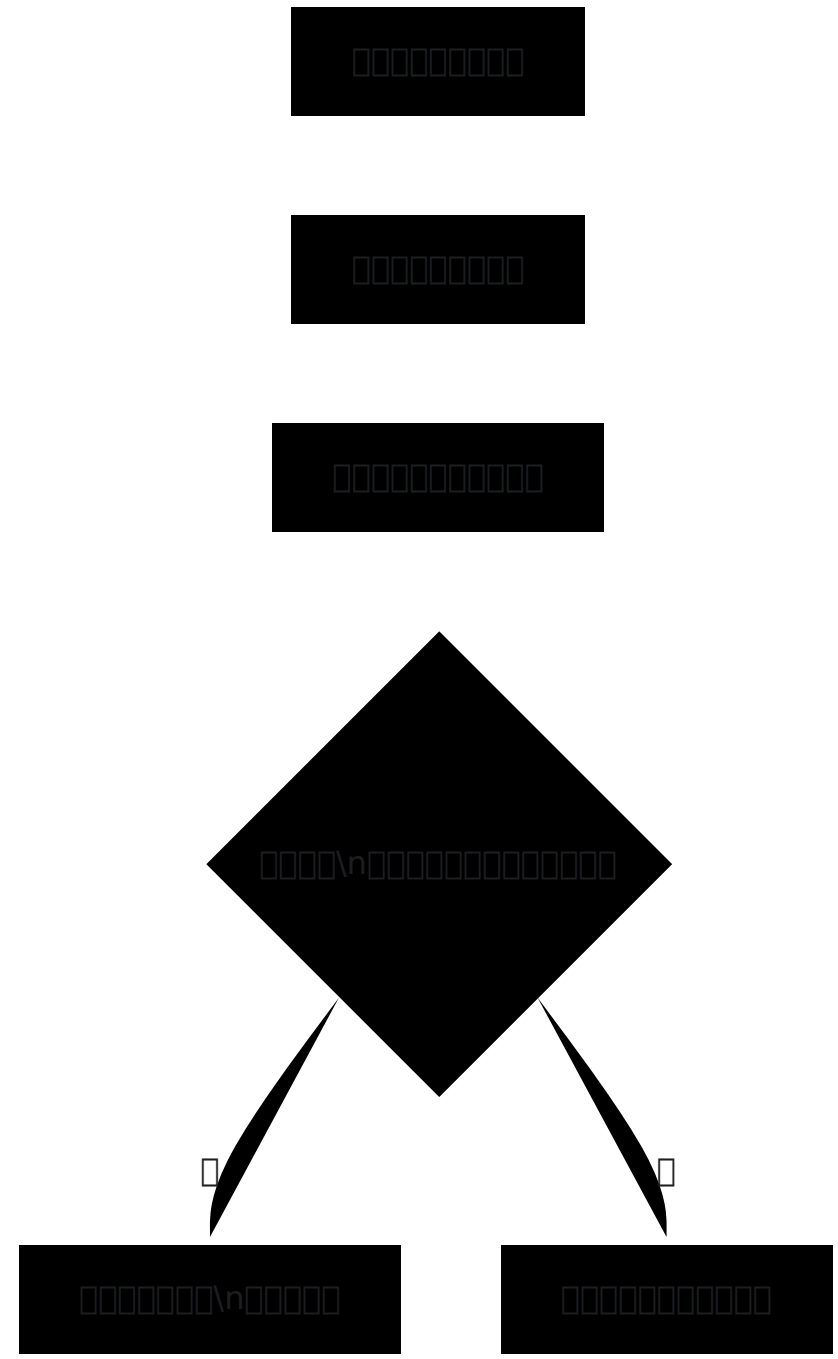
对于每个规则的故障转移和负载均衡，将对等体写为带有 `priority` 和 `weight` 的映射。这与全局算法无关。

```
route: %{
  peers: [
    %{host: "pcrf01.example.com", priority: 10, weight: 50},
    %{host: "pcrf02.example.com", priority: 10, weight: 50},
    %{host: "pcrf-dr.example.com", priority: 20, weight: 100}
  ]
}
```

选择是“优先级选择层级，权重分配负载”：

1. DRA 仅考虑连接的对等体。
2. 最低优先级数字 的对等体获胜。较低的数字更受欢迎，就像 DNS SRV 记录一样。较□编号的层级作为备份。DRA 仅在所有较低层级的连接对等体都不可用时使用备份层级。这提供了 故障转移。
3. 在获胜的层级内，DRA 随机选择一个对等体，按 **`weight`** 加权。相等的权重均匀分配负载。不等的权重则偏向分配。这提供了 负载均衡。

如果您省略 `priority` 或 `weight`，每个默认值为 `100`。



这个单一模型表达了每个常见拓扑：

目标	如何表达
纯故障转移	不同的递减优先级（例如 10, 20, 30）；权重无关
纯负载均衡	所有对等体的一个优先级；相等权重
加权负载均衡	所有对等体的一个优先级；权重按所需比例（例如 70 / 30）
带备份的负载均衡	两个优先级 10 的对等体，一个优先级 20 的对等体

会话亲和性 (Gx / Gy)

状态保持应用程序在会话生命周期内携带一个 会话 **ID**。会话中的消息必须到达处理会话发起请求的 同一 服务器。如果 DRA 将会话中的消息重新平衡到不同的对等体，该对等体将以 **DIAMETER_UNKNOWN_SESSION_ID (5002)** 拒绝它。

级路由通过标准 Diameter 机制，即 目标主机 **AVP (293)** 保持亲和性。当规则使用优先级/权重对等体时：

- 如果请求携带一个目标主机，命名规则的连接对等体，DRA 固定 请求到该对等体。这覆盖优先级和权重。
- 如果没有这样的目标主机，DRA 将正常负载均衡请求。

会话发起请求（例如 Gx CCR-Initial）不携带目标主机。DRA 在层级中进行负载均衡。响应返回所选服务器的 Origin-Host。客户端随后在所有后续请求（CCR-Update, CCR-Terminate）中将该值放入目标主机。这会自动将这些请求固定到该服务器。在 Gx/Gy 中，**CC-Request-Type AVP (416)** 区分请求类型。

这种亲和性适用于优先级/权重对等体列表。简单主机名列表遵循全局算法。这种行为不会影响它们。

配置

在 `config/runtime.exs` 中的 `module_advanced_routing` 下配置□级路由。

```
module_advanced_routing: %{
  # □用或禁用模块
  enabled: false,

  # 规则按顺序评估；第一个匹配的规则获胜
  rules: [
    %{
      rule_name: "route_s6a_air_to_hss",
      match: :all,
      filters: [
        {:application_id, 16_777_251},
        {:command_code, 318}
      ],
      route: %{
        peers: ["hss01.example.com"]
      }
    }
  ]
}
```

模块参数

参数	类型	必需	默认	描述
`enabled`	布尔	否	`false`	模块的主开关。当为 `false` 时，DRA 不评估任何规则，所有流量使用标准路由。
`rules`	列表	是	`[]`	有序的路由规则列表。DRA 从上到下评估它们。第一个匹配的规则获胜。见规则参数。

规则参数

参数	类型	必 ??	默认	描述
`rule_name`	字符串	是	-	规则的描述性唯一名称。DRA 在日志和管理 UI 中显示它。
`match`	原子	是	-	过滤器组合的方式：`:all`（逻辑与）、`:any`（逻辑或）或 `:none`（逻辑非）。见匹配模式。
`filters`	列表	是	-	消息必须满足的条件。见过滤器。
`route`	映射或原子	是	-	DRA 发送匹配流量的位置。这是一个对等体列表或特殊目的地。见路由目的地。

过滤器

过滤器是 `{filter_type, value}` 的元组。□级路由支持以下过滤器：

过滤器	示例	描述
`:via_peer`	`{:via_peer, "mme01.example.com"}`	发送请求的对等体的 Origin-Host（最近的跳跃）。
`:application_id`	`{:application_id, 16_777_251}`	Diameter 应用程序 ID。见应用程序 ID。
`:command_code`	`{:command_code, 318}`	Diameter 命令代码。
`:avp`	`{:avp, {1, "999999000000001"}}`	AVP 值，写为 `{avp_code, value}`。

过滤器值

- 列表（或）值。任何过滤器值都可以是列表。DRA 将该列表视为其成员的逻辑或。例如，`{:command_code, [317, 318]}` 匹配命令代码 317 或 318。`{:via_peer, ["dra01.example.com",`

`"dra02.example.com"]}` 匹配任一对等体。

- 正则表达式值。 AVP 过滤器值和列表成员通过正则表达式支持正则表达式，例如 `{:avp, {1, ~r"999001.*"}}`。您可以混合成员，例如 `{:avp, {1, ["505571234567", ~r"999001.*", ~r"505057.*"]}}`。
- 存在检查（`:any`）。值为 `:any` 时，当属性存在且具有任何值时匹配。例如，`{:avp, {264, :any}}` 在 Origin-Host AVP 存在时匹配。`{:via_peer, :any}` 匹配任何发起对等体。

AVP 过滤器限制

仅对 简单 AVP 使用 AVP 过滤器，例如用户名称、Origin-Host 和目标域。不支持分组 **AVP**，并且不匹配。复杂的二进制值不受支持。始终使用 `{:avp, {code, value}}` 形式。

路由目的地

`route` 参数可以是对等体列表映射或以下特殊目的地原子/元组之一。

目的地	示例	描述
对等体列表	`%{peers: [...]}`	路由到列出的对等体之一。条目可以是简单主机名或优先级/权重映射。见对等体选择。
目标主机	`:destination_host`	路由到消息的 目标主机 **AVP (293)** 中命名的对等体。如果 AVP 缺失，则规则不适用，DRA 使用标准路由。
丢弃	`:drop`	在不向请求对等体���应的情况下丢弃消息。使用此方法来黑洞不需要的流量，例如测试订阅者或不需要的范围。
错误	`{:error, 3004}`	使用给定的结果代码响应 Diameter 错误答案。DRA 自动填充 Origin-Host、Origin-Realm 和 Session-Id。DRA 不会将消息转发给任何对等体。

对于 转发 应用程序，只有 3xxx（协议）结果代码在生成的答案中严格有效（`3002`、`3003`、`3004`、`3007`）。某些部署还使用 `5012`（DIAMETER_UNABLE_TO_COMPLY）来拒绝特定命令。有关含义，请参见参考表。

对等体参数（优先级/权重形式）

参数	类型	必需	默认	描述
`host`	字符串	是	-	对等体的 Origin-Host。它必须与对等体的配置 Diameter 身份完全匹配，并区分大小写。
`priority`	整数	否	`100`	层级。DRA 更喜欢较低的数字。仅在较低层级的所有连接对等体不可用时使用较□编号的层级。
`weight`	整数	否	`100`	在层级内的流量份额。DRA 按照权重在层级的对等体之间分配流量。权重为 `0` 时排除对等体，除非该层级中的每个对等体都是 `0`。

配置示例

故障转移到备份对等体

将 S6a 认证信息请求路由到�� HSS。仅在主 HSS 关闭时回退到灾难恢复 HSS。

```
%{
  rule_name: "s6a_air_failover",
  match: :all,
  filters: [
    {:application_id, 16_777_251},
    {:command_code, 318}
  ],
  route: %{
    peers: [
      %{host: "hss01.example.com", priority: 10},
      %{host: "hss-dr.example.com", priority: 20}
    ]
  }
}
```

工作原理：当 `hss01` 连接时，每个请求都发送给它。如果 `hss01` 断开连接，请求将故障转移到 `hss-dr`。如果两者都关闭，DRA 将用 3002 回答请求。

均匀负载均衡

将 Gx 流量均匀分配到两个 PCRF。

```
%{
  rule_name: "gx_load_balance",
  match: :all,
  filters: [
    {:application_id, 16_777_238}
  ],
  route: %{
    peers: [
      %{host: "pcrf01.example.com", priority: 10, weight: 50},
      %{host: "pcrf02.example.com", priority: 10, weight: 50}
    ]
  }
}
```

工作原理： DRA 将新会话（CCR-Initial）大致 50/50 地分配到两个 PCRF。DRA 将会话中的消息固定到处理会话的 PCRF。它使用消息携带的目标主机。

带备份的加权负载均衡

将更多流量发送到较大的 PCRF，较少流量发送到较小的 PCRF。保持一个备份以防两者都不可用。

```
%{
  rule_name: "gx_weighted_with_backup",
  match: :all,
  filters: [
    {:application_id, 16_777_238}
  ],
  route: %{
    peers: [
      %{host: "pcrf01.example.com", priority: 10, weight: 70},
      %{host: "pcrf02.example.com", priority: 10, weight: 30},
      %{host: "pcrf-dr.example.com", priority: 20, weight: 100}
    ]
  }
}
```

工作原理：当任一优先级 10 的 PCRF 连接时，DRA 将新会话以 70/30 的比例分配给它们。仅在 两个 优先级 10 的 PCRF 都关闭时，流量才会转移到 `pcrf-dr`。已建立的会话在其生命周期内保持固定到其服务器。

按发起对等体路由

将来自特定上游 DRA 的 S6a 流量路由到本地 HSS 对。此方法在迁移或切换期间非常有用。

```
%{
  rule_name: "s6a_via_peer_to_local_hss",
  match: :all,
  filters: [
    {:application_id, 16_777_251},
    {:via_peer, ["dra01.example.com", "dra02.example.com"]},
    {:avp, {296, "epc.mnc001.mcc001.3gppnetwork.org"}}
  ],
  route: %{
    peers: ["hss01.example.com", "hss02.example.com"]
  }
}
```

工作原理： DRA 仅将来自 `dra01` 或 `dra02` 并携带匹配 Origin-Realm（AVP 296）的 S6a 请求路由到本地 HSS 节点。其他所有内容将通过后续规则或标准路由处理。

基于 IMSI 的模式匹配漫游

根据 IMSI（用户名称，AVP 1）模式���由入站漫游流量。

```
%{
  rule_name: "inbound_s6a_roaming",
  match: :all,
  filters: [
    {:application_id, 16_777_251},
    {:avp, {296, "epc.mnc001.mcc001.3gppnetwork.org"}},
    {:avp, {1, ["505571234567", ~r"999001.*"]}}
  ],
  route: %{
    peers: ["dra01.example.com", "dra02.example.com"]
  }
}
```

工作原理：该规则匹配来自给定 Origin-Realm 的 S6a 消息，其用户名称恰好为 `505571234567` 或以 `999001` 开头。DRA 将这些消息路由到指定的对等体。列表值在精确字符串和正则表达式之间应用或逻辑。

尊重目标主机 AVP

仅针对匹配流量路由到发送方命名的对等体。

```
%{
  rule_name: "honour_destination_host",
  match: :all,
  filters: [
    {:avp, {1, [~r"90199.*"]}}
  ],
  route: :destination_host
}
```

工作原理：当 IMSI 匹配时，DRA 将请求发送到其目标主机 AVP (293) 中命名的对等体。如果目标主机 AVP 缺失，DRA 将该匹配视为失败并应用标准路由。

丢弃不需要的流量

静默丢弃测试范围流量。

```
%{
  rule_name: "drop_test_subscribers",
  match: :all,
  filters: [
    {:application_id, 16_777_251},
    {:avp, {1, [~r"999999.*"]}}
  ],
  route: :drop
}
```

工作原理：DRA 丢弃 IMSI 以 `999999` 开头的 S6a 消息。它不发送响应。使用此方法过滤测试流量或阻止特定订阅者范围。

通过错误答案进行过载保护

向特定过载对等体返回 DIAMETER_TOO_BUSY。

```
%{
  rule_name: "rate_limit_high_volume_peer",
  match: :all,
  filters: [
    {:via_peer, "mme-overloaded-01.example.com"},
    {:application_id, 16_777_251}
  ],
  route: {:error, 3004}
}
```

工作原理：DRA 用 **DIAMETER_TOO_BUSY (3004)** 回答来自命名对等体的 S6a 流量。这向对等体发出退避信号。DRA 不会向上游转发任何内容。

阻止特定命令

用解释性错误拒绝特定命令类型，而不是静默丢弃。

```
%{
  rule_name: "block_purge_requests",
  match: :all,
  filters: [
    {:application_id, 16_777_251},
    {:command_code, 321}
  ],
  route: {:error, 5012}
}
```

工作原理：DRA 用 **DIAMETER_UNABLE_TO_COMPLY (5012)** 回答 S6a 清除 UE 请求（命令 321）。这禁用该操作并保持其他命令不变。

阅读活动配置

您可以通过管理 API 读取活动规则，路径为 `GET /api/rules`。响应包括路由规则、转换规则和全局 `routing_algorithm`。

对于每个路由到对等体列表的规则，API 报告选择行为：

- `route.selection` 在规则使用优先级/权重映射时为 `"priority_weight"`。否则，它是全局算法（`"failover"` 或 `"random"`）。
- 在 `priority_weight` 模式下，API 返回每个对等体及其有效的 `priority` 和 `weight`。它填充默认值以使分配明确。

```
{
  "status": "success",
  "response": {
    "routing_algorithm": "random",
    "routing_rules": [
      {
        "rule_name": "gx_weighted_with_backup",
        "match": "all",
        "route": {
          "selection": "priority_weight",
          "peers": [
            { "host": "pcrf01.example.com", "priority": 10,
"weight": 70 },
            { "host": "pcrf02.example.com", "priority": 10,
"weight": 30 },
            { "host": "pcrf-dr.example.com", "priority": 20,
"weight": 100 }
          ]
        }
      }
    ],
    "transform_rules": []
  }
}
```


最佳实践

1. 优先具体性。按从最具体到最一般的顺序排列规则。第一个匹配的规则获胜。
2. 常见匹配靠前。将流量匹配放在顶部。这减少评估开销。
3. 验证正则表达式。在部署之前测试正则表达式模式与捕获的消息。
4. 清晰命名。使用描述性的 `rule_name` 值。日志和管理 UI 将自我解释。
5. 优先考虑状态保持应用的优先级/权重。对于 Gx/Gy 使用优先级/权重对等体映射。DRA 将自动应用会话亲和性。

故障排除

会话被拒绝，原因 **UNKNOWN_SESSION_ID**

症状：Gx 或 Gy 会话正确建立，但服务器拒绝中间会话更新，结果代码为 5002。

可能原因：

- 客户端（PCEF/OCF）在中间会话请求中未设置目标主机，因此 DRA 无法固定它们。
- 规则使用简单主机名和全局 `:random` 算法路由此流量。该算法不应用会话亲和性。

解决方案：

1. 确认中间会话请求携带一个目标主机，命名处理初始请求的服务器。
2. 将规则的对等体列表转换为优先级/权重映射。DRA 将自动应用会话亲和性。

匹配流量回答为 **3002**

症状：规则应该路由某些请求，但 DRA 用 DIAMETER_UNABLE_TO_DELIVER 回答它们。

可能原因：

- 匹配规则中列出的对等体均未连接。
- 规则中的主机名与对等体的 Origin-Host 值不完全匹配。

解决方案：

1. 检查管理 UI 或日志中的对等体连接状态。
2. 确认规则中的 `host` 值与对等体的 Origin-Host 完全匹配。匹配是区分大小写的。

规则未生效

症状：应该匹配规则的流量通过标准路由而不是。

可能因:

- `enabled` 设置为 `false`。
- 列表中的早期规则首先匹配并赢得路由。

- 过滤器值不匹配。这可能是 AVP 值的大小写不匹配，或不支持的分组 AVP。

解决方案：

1. 确认 `enabled` 为 `true`。更改后重启 DRA。
2. 检查规则顺序。第一个匹配的规则获胜。
3. 确认过滤器值与捕获的消息相符。请记住，DRA 无法匹配分组 AVP。

参考

Diameter 结果代码

代码	名称	描述	参考
3002	DIAMETER_UNABLE_TO_DELIVER	没有可用的连接对等体来交付请求	RFC 6733 §7.1.3
3003	DIAMETER_REALM_NOT_SERVED	没有配置的路由服务请求的域	RFC 6733 §7.1.3
3004	DIAMETER_TOO_BUSY	节点过载，无法处理请求	RFC 6733 §7.1.3
5002	DIAMETER_UNKNOWN_SESSION_ID	服务器不识别会话 ID。例如，中间会话消息到达错误的服务器	RFC 6733 §7.1.5
5012	DIAMETER_UNABLE_TO_COMPLY	DRA 无法满足请求。用于拒绝特定命令	RFC 6733 §7.1.5

相关 AVP

AVP	代码	描述	参考
目标主机	293	命名必须接收请求的特定对等体。这是会话亲和性的基础	RFC 6733 §6.5
会话 ID	263	在其生命周期内标识会话	RFC 6733 §8.8
CC-请求类型	416	区分 Gx/Gy 中的 INITIAL、UPDATE 和 TERMINATION 请求	RFC 4006 §8.3

应用程序 ID

ID	接口	描述	参考
16777238	Gx	PCEF 和 PCRF 之间的策略控制	3GPP TS 29.212
16777251	S6a/S6d	MME/SGSN 到 HSS 的认证和订阅管理	3GPP TS 29.272
4	Gy	OCF 和 OCS 之间的在线计费	RFC 4006

相关文档

- 标准路由 - 当没有规则匹配时使用的目标主机 / 目标域路由。
- □级转换 - 请求和响应的内容重写，使用相同的过滤器引擎。
- 配置参考 - 完整的 `config/runtime.exs` 参数参考，包括全局 `peer_selection_algorithm`。

□级转换

□级转换模块重写符合运营商定义标准的Diameter消息中的属性值对（**AVP**）。使用它来修正、翻译或规范化通过OmniDRA传递的消息内容。您无需更改发送或接收网络元素。

概述

□级转换根据每个Diameter消息评估一系列规则。规则将一组过滤器（匹配标准）与转换（要应用的更改）结合在一起。当规则匹配时，DRA将其转换应用于消息并停止规则处理。

□级转换提供三种转换操作。**`:edit`更改现有**AVP**的值。`:remove`通过代码删除一个或多个**AVP**。`:overwrite`**通过名称重写一个或多个AVP，包括分组AVP。请参见转换操作。

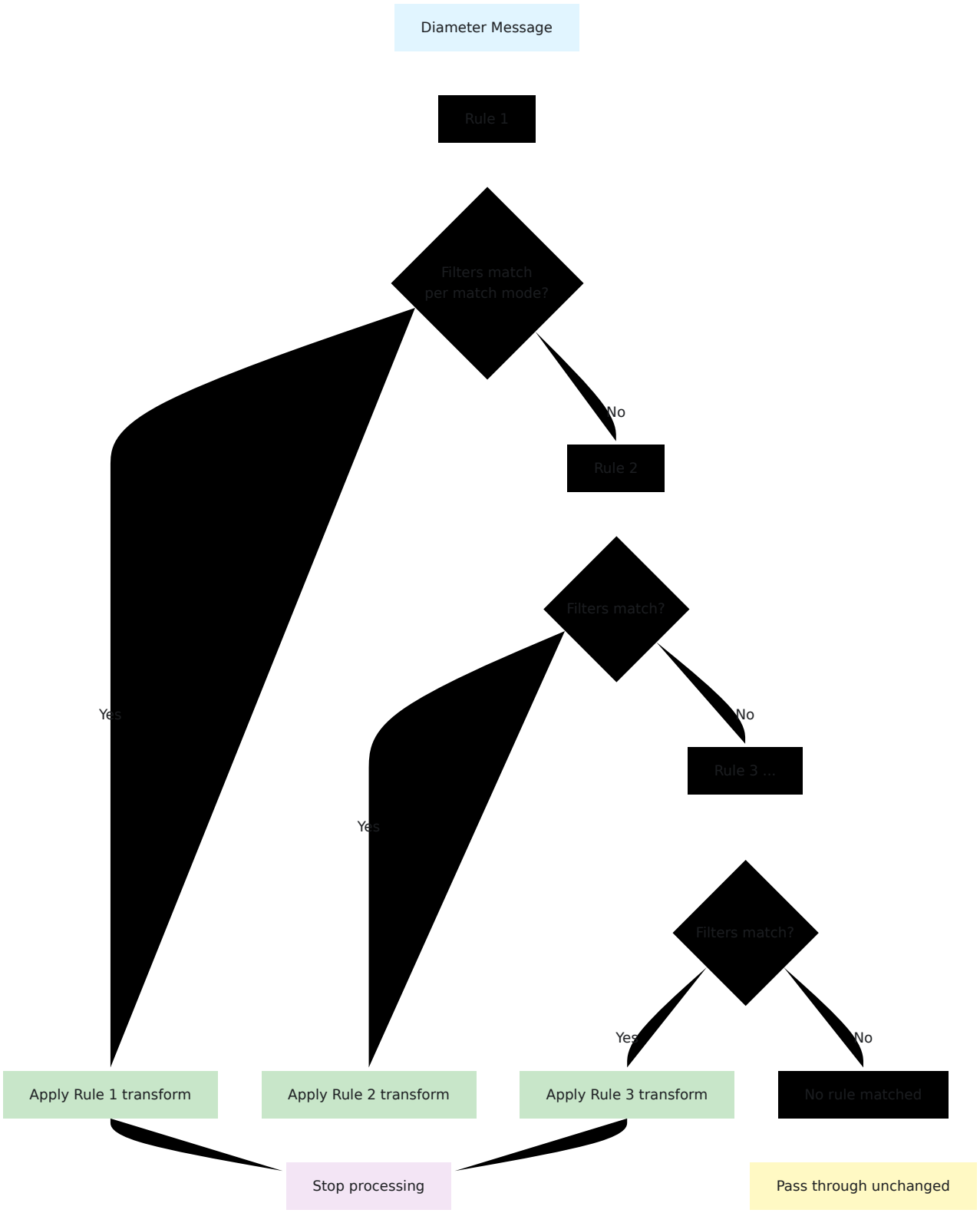
□级转换与路由引擎共享其规则处理模型。有关DRA如何做出路由决策，请参见□级路由。

规则处理方式

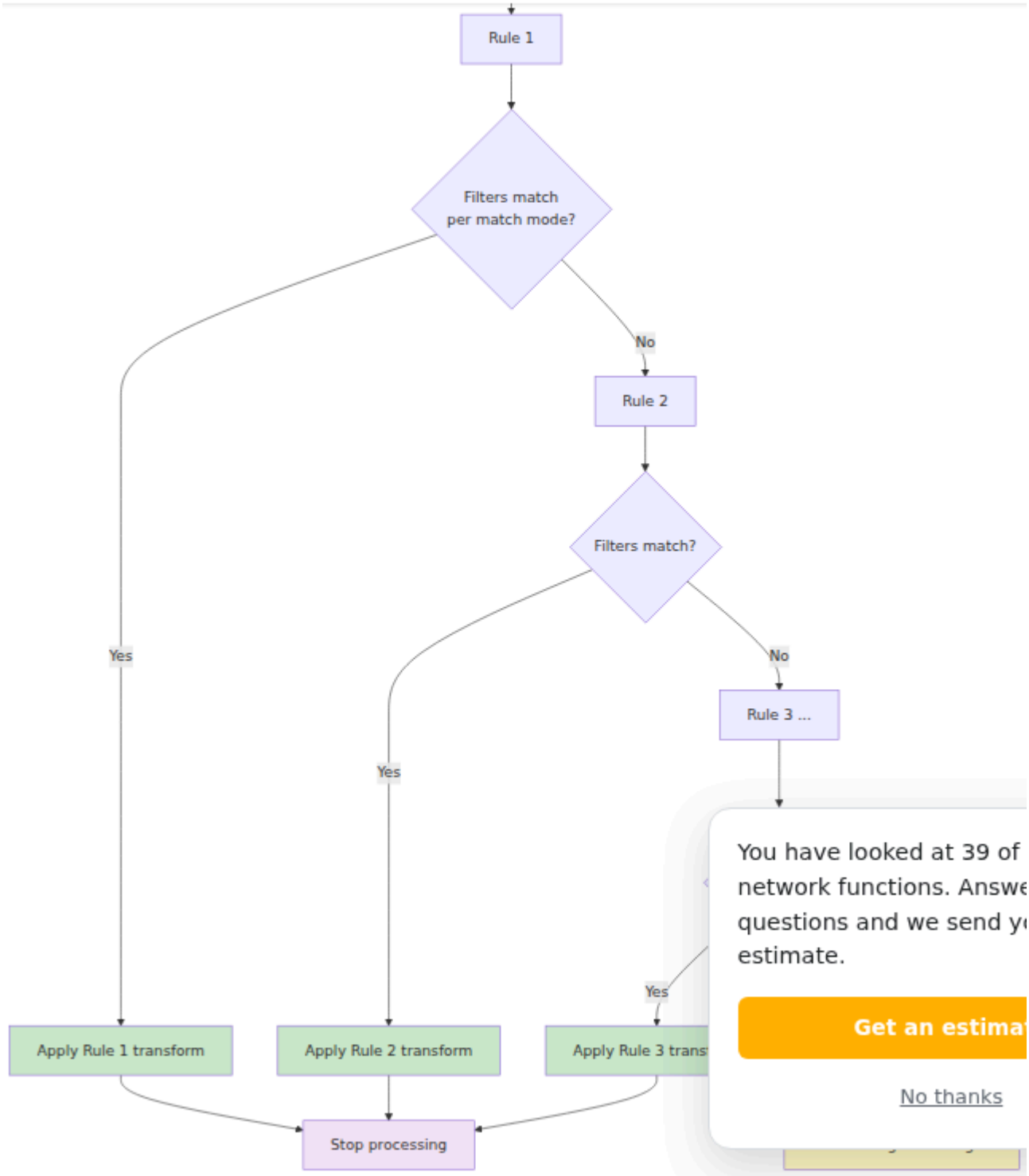
DRA按照配置中出现的顺序从上到下评估规则。第一个匹配的规则胜出。一旦规则的过滤器匹配，DRA将应用其转换并不再考虑其他规则。如果没有规则匹配，消息将透明地通过，不会进行修改。

在单个规则内，`match`参数控制单个过滤器的组合方式：

- `:all` - 逻辑**AND**。每个过滤器必须通过，规则才能匹配。
- `:any` - 逻辑**OR**。至少一个过滤器必须通过，规则才能匹配。
- `:none` - 逻辑**NOR**。没有过滤器可以通过，规则才能匹配（反向匹配）。


Diameter Message
Rule 1
Filters match per match mode?
No
Rule 2
Filters match?
Yes
No
Rule 3 ...
Yes
Filters match?
Yes
No
Apply Rule 1 transform
Apply Rule 2 transform
Apply Rule 3 transform
No rule matched
Stop processing
Pass through unchanged

□□□



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

□□□

□□□

过滤器	格式	描述
`:packet_type`	`{:packet_type, :request}`	匹配消息的方向。仅接受`:request`或`:answer`；不接受值列表。请参见数据包类型过滤。
`:application_id`	`{:application_id, <id>}`	匹配消息的Diameter应用程序ID（例如，S6a/S6d的`16777251`）。请参见应用程序ID。
`:command_code`	`{:command_code, <code>}`	匹配消息的Diameter命令代码（例如，AIR的`318`）。请参见命令代码。
`:avp`	`{:avp, {<code>, <value>}}`	按其数字代码和期望值匹配AVP。请参见AVP过滤。
`:to_peer`	`{:to_peer, "hss01.example.com"}`	匹配请求的解析目标对等体。仅匹配请求数据包，永远不匹配答案。接受主机名列表（或）。请参见对等体过滤。
`:from_peer`	`{:from_peer, "hss01.example.com"}`	匹配回答消息的对等体。仅匹配答案数据包，永远不匹配请求。接受主机名列表（或）。请参见对等体过滤。

数据包类型过滤

`:packet_type`过滤器将规则限制为一个方向的流量。它仅接受两个值之一：

- **`:request`** - 仅匹配Diameter 请求消息。
- **`:answer`** - 仅匹配Diameter 答案消息。

与`:application_id`、`:command_code`和`:avp`不同，`:packet_type`过滤器不接受值列表。

AVP过滤

`:avp`过滤器的结构为`{avp_id, avp_value}`。`avp_id`是数字AVP代码。`avp_value`是要比较的值，以字符串形式表示。

对于简单**AVP**（例如`User-Name`和`Origin-Host`），使用`:avp`过滤器。不支持分组**AVP**，并且不匹配。复杂的二进制值不匹配。

过滤器以多种方式匹配值：

- 精确字符串 - `{:avp, {1, "999999000000001"}}`仅匹配该精确值。
- 值列表 - `{:avp, {1, ["50557...", "505057..."]}}`如果AVP等于列表中的任何值，则匹配。这是在列表内的OR逻辑。
- 正则表达式 - `{:avp, {1, [~r"9999999.*"]}}`根据正则表达式模式匹配值。

使用`~r"pattern"`语法进行正则表达式匹配：

模式	匹配
`~r"999001.*"`	以`999001`开头的`User-Name`/IMSI值
`~r"^310[0-9]{3}.*"`	具有特定MCC/MNC前缀的值
`~r".*test$"`	以`test`结尾的值

对等体过滤

`:to_peer`和`:from_peer`过滤器匹配对等体主机名，而不是消息内容。它们是方向特定和互补的：

- `:to_peer`匹配**DRA**即将转发请求的解析目标对等体。仅匹配请求数据包。永远不匹配答案。
- `:from_peer`���配回答消息的对等体。仅匹配答案数据包。永远不匹配请求。

两个过滤器都接受单个主机名或主机名列表。使用列表时，如果对等体等于列表中的任何主机名，则过滤器匹配。这是在列表内的OR逻辑。

过滤器	示例	匹配
`:to_peer`	`{:to_peer, "hss01.example.com"}`	路由到对等体 `hss01.example.com` 的请求
`:to_peer`	`{:to_peer, ["hss01.example.com", "hss02.example.com"]}`	路由到任一列出的对等体的请求
`:from_peer`	`{:from_peer, "hss01.example.com"}`	从对等体 `hss01.example.com` 接收的答案

转换操作

`transform` 映射描述转换。其 `action` 键选择三种转换之一。消息的方向决定允许哪些操作：

操作	功能	请求	答案
`:edit`	重写现有AVP的值。	是	否
`:remove`	从消息中删除命名的AVP。	是	是
`:overwrite`	通过名称重写AVP，包括使用解码字典的分组AVP。	是	是

`:edit` 操作

```
transform: %{
  action: :edit,
  avps: [{:avp, {1, "999999000000002"}}]
}
```

`:edit` 的行为如下：

- 它修改Diameter消息中命名AVP的值为提供的值。
- 仅影响消息中已经存在的AVP。如果目标AVP不存在，DRA将不进行更改，并按接收到的内容传递消息。
- `avps`中的每个条目为`{:avp, {<code>, <new_value>}}`。它将该AVP设置为新值。
- `:edit`仅适用于请求。要在答案上重写AVP，或访问分组AVP，请使用`:overwrite`。

`:remove`操作

```
transform: %{
  action: :remove,
  avps: [{:avp, {264, :any}}]
}
```

`:remove`的行为如下:

- 它从消息中删除`avps`中命名的AVP。
- 仅按**AVP**代码匹配。它忽略每个`avps`条目的值以进行删除。使用`:any`作为值以明确这一点。
- 适用于请求和答案。

`:overwrite`操作

```
transform: %{
  action: :overwrite,
  dictionary: :diameter_gen_3gpp_s6a,
  avps: [{:avp, {1, "999999000000002"}}]
}
```

`:overwrite`的行为如下:

- 它通过名称重写命名的AVP。它使用提供的`dictionary`解码和重新编码消息。
- 它解码整个消息，因此`:overwrite`可以访问`edit`无法访问的分组/记录**AVP**。
- 它要求一个`dictionary`键。该键命名描述消息的Diameter字典（例如`:diameter_gen_3gpp_s6a`）。
- 适用于请求和答案。

转换映射参数

参数	类型	必需	默认	描述
`action`	原子	是	-	要应用的转换：`:edit`（重写现有AVP值 - □求），`:remove`（按代码删除AVP - 请求和或`:overwrite`（按名称重写AVP，包括□- 请求和答案）。
`avps`	列表	是	-	AVP目标列表。对于`:edit`/`:overwri`个条目为`{:avp, {<code>, <new_value>}}`并将该AVP设置为`<new_value>`。对于`:remove`，DR`<code>`并忽略值（使用`:any`）。
`dictionary`	原子	仅适用于`:overwrite`	`:not_used`	解码和重新编码消息的Diameter字典，以便D重写命名的（分组）AVP，例如`:diameter_gen_3gpp_s6a`。和`:remove`操作忽略此键。

配置

在`config/runtime.exs`中的`module_advanced_transform`下配置□级转换。该模块默认情况下是禁用的。在将`enabled`设置为`true`之前，它不会执行任何操作。

```
module_advanced_transform: %{
  enabled: true,
  rules: [
    %{
      rule_name: "transform_user_name",
      match: :all,
      filters: [
        {:application_id, 16_777_251},
        {:command_code, 318},
        {:avp, {1, "999999000000001"}}
      ],
      transform: %{
        action: :edit,
        avps: [{:avp, {1, "999999000000002"}}]
      }
    }
  ]
}
```

模块参数

参数	类型	必需	默认	描述
`enabled`	布尔	否	`false`	启用该模块。当为`false`时，DRA不评估任何规则，所有消息透明通过。
`rules`	列表	是（启用时）	-	转换规则的有序列表。DRA从上到下评估它们。第一个匹配的规则胜出。请参见规则参数。

规则参数

`rules`中的每个条目都是一个包含以下键的映射。

参数	类型	必需	默认	描述
`rule_name`	字符串	是	-	规则的唯一描述性标识符。它支持操作清晰度和监控。
`match`	原子	是	-	过滤器组合的方式：`:all`（AND）、`:any`（OR）或`:none`（NOR）。请参见规则处理方式。
`filters`	列表	是	-	消息必须满足的过滤条件列表，依据`match`模式。请参见过滤器。
`transform`	映射	是	-	当规则匹配时要应用的更改。请参见转换映射参数。

示例

示例1：重写S6a AIR请求中的User-Name

```
module_advanced_transform: %{
  enabled: true,
  rules: [
    %{
      rule_name: "transform_user_name",
      match: :all,
      filters: [
        {:application_id, 16_777_251},
        {:command_code, 318},
        {:avp, {1, "999999000000001"}}
      ],
      transform: %{
        action: :edit,
        avps: [{:avp, {1, "999999000000002"}}]
      }
    }
  ]
}
```

工作原理：使用`match: :all`，每个过滤器必须通过。该规则仅匹配S6a（`16777251`）身份验证信息请求（`318`）消息，其`User-Name`（AVP `1`）等于`999999000000001`。在匹配时，`:edit`操作将`User-Name`重写为`999999000000002`。如果消息不携带`User-Name` AVP，DRA将不进行更改。

用例：在DRA传输过程中修正或规范化特定的订阅者身份。例如，在迁移期间将遗留IMSI映射到其当前值。

示例2：根据IMSI范围重写领域

```
module_advanced_transform: %{
  enabled: true,
  rules: [
    %{
      rule_name: "rewrite_destination_realm_roaming_partner",
      match: :all,
      filters: [
        {:avp, {296, "epc.mnc057.mcc505.3gppnetwork.org"}},
        {:avp, {1, [~r"50557.*"]}}
      ],
      transform: %{
        action: :edit,
        avps: [{:avp, {283, "epc.mnc030.mcc310.3gppnetwork.org"}}]
      }
    }
  ]
}
```

工作原理：当`Origin-Realm`（AVP `296`）等于漫游合作伙伴的领域并且`User-Name`/IMSI（AVP `1`）以`50557`开头时，规则匹配。然后重写`Destination-Realm`（AVP `283`）。路由引擎随后将消息转发到正确的合作伙伴网络。第一个匹配的规则胜出，因此将更具体的IMSI范围放在更广泛的范围之上。

用例：将漫游或MVNO订阅者流量引导到正确的托管核心网络。DRA通过IMSI范围和原始领域选择网络。

示例3：从特定对等体的答案中删除AVP

```
module_advanced_transform: %{
  enabled: true,
  rules: [
    %{
      rule_name: "strip_origin_host_from_partner_answers",
      match: :all,
      filters: [
        {:packet_type, :answer},
        {:from_peer, "hss01.partner.example.com"},
        {:command_code, 318}
      ],
      transform: %{
        action: :remove,
        avps: [{:avp, {264, :any}}]
      }
    }
  ]
}
```

工作原理：`:packet_type`过滤器将规则限制为答案。`:from_peer`过滤器将其限制为来自`hss01.partner.example.com`的答案。`:command_code`过滤器将其限制为AIA（`318`）。当所有三个匹配时，`:remove`操作删除`Origin-Host`（AVP `264`），无论其值如何。删除仅按AVP代码匹配。

用例：从合作伙伴网络的答案中删除AVP��当合作伙伴错误填充AVP时，或当AVP必须不向下游传递时使用此方法。

示例4：在S6a请求上重写分组AVP

```
module_advanced_transform: %{
  enabled: true,
  rules: [
    %{
      rule_name: "overwrite_user_name_s6a",
      match: :all,
      filters: [
        {:packet_type, :request},
        {:application_id, 16_777_251},
        {:command_code, 318}
      ],
      transform: %{
        action: :overwrite,
        dictionary: :diameter_gen_3gpp_s6a,
        avps: [{:avp, {1, "999999000000002"}}]
      }
    }
  ]
}
```

工作原理：该规则匹配S6a（`16777251`）AIR（`318`）请求。`:overwrite`操作使用`:diameter_gen_3gpp_s6a`字典解码消息，重写命名的AVP，并重新编码消息。它解码整个消息，因此`:overwrite`可以访问；`edit`无法触及的嵌套在分组/记录结构中的AVP。

用例：修正位于分组AVP内部的AVP，例如订阅数据或特定于供应商组内的字段。简单的`:edit`将无法匹配这样的AVP。

参考表

应用程序ID

ID	接口	描述	参考
16777251	S6a/S6d	MME/SGSN ↔ HSS身份验证和订阅	3GPP TS 29.272

命令代码

代码	命令	接口	参考
318	身份验证信息请求/答案（AIR/AIA）	S6a/S6d	3GPP TS 29.272 §7.2.5
316	更新位置请求/答案（ULR/ULA）	S6a/S6d	3GPP TS 29.272 §7.2.3

常见**AVP**代码

代码	**AVP**	描述	参考
1	User-Name	订阅者身份（S6a上的IMSI）。	3GPP TS 29.272 / RFC 6733 §8.14
264	Origin-Host	消息发起者的Diameter身份。	RFC 6733 §6.3
283	Destination-Realm	消息的目标领域。DRA用于路由。	RFC 6733 §6.6
296	Origin-Realm	消息发起者的领域。	RFC 6733 §6.4

最佳实践

1. 特异性 - 按从最具体到最一般的顺序排列规则。第一个匹配的规则胜出。
2. 性能 - 将最常匹配的规则放在前面。这可以减少评估开销。
3. 测试 - 在部署之前验证正则表达式模式。
4. 命名 - 使用描述性的`rule_name`以提操作清晰度和监控。
5. 监控 - 跟踪规则匹配率。这可以确认规则按预期运行。

相关文档

- □级路由 - 使用相同规则处理模型进行目标选择。

DRA 架构概述

OmniDRA 是一个 **Diameter** 路由代理 **(DRA)**。它位于 Diameter 信令网络的中心。它在客户端（如 MME/SGSN 节点）和后端服务（如 HSS、PCRF 和 OCS）之间中继请求和应答消息。OmniDRA 是基于 Elixir/Erlang 在 OTP `:diameter` 库上构建的。它作为一个 **Diameter** 中继运行，并在对等方之间转发消息。它不终止消息所携带的应用程序。

目录

1. DRA 的定义
2. 网络上下文
3. 请求处理管道
4. 对等注册表
5. 消息中继
6. 参考表
7. 进一步阅读

DRA 的定义

一个 **Diameter** 中继 在一个监听套接字上接收 Diameter 消息。它决定哪个连接的对等方必须处理每条消息。然后，它转发消息，除了中继管理的路由 AVP 之外，消息保持不变。OmniDRA 作为中继运行，因此它不实现其携带的接口的应用逻辑。它明地路由 **S6a**、**Gx**、**Gy** 和任何其他 Diameter 应用程序。对等方必须宣传对该应用程序的支持。

主要特征：

- 基于标准的路由 - 请求路由遵循 Diameter 基本协议 (RFC 6733) 的优先级机制。它选择一个具有 **Destination-Host**、**Destination-Realm** 和 **Application-Id** 的对等方。
- 有状态应答关联 - 应答总是沿着其请求的反向路径返回。DRA 通过 **Hop-by-Hop Identifier** 关联每个应答。
- 可选处理模块 - 漫游引导、订阅者查找、级路由和转换模块扩展了标准行为。它们并不替代标准行为。
- 安全管道 - 速率限制、Diameter 防火墙、AVP 清理和拓扑隐藏在信令边缘保护网络。

网络上下文

在 接入侧 信令节点（如 MME 和 SGSN）与 核心侧 服务（如 HSS、PCRF 和 OCS）之间部署 DRA。每个对等方与 DRA 建立 Diameter 连接。DRA 从不发起自己的应用流量。

DRA 在 TCP/SCTP 端口 `3868` 上监听传入的 Diameter 连接。这是 RFC 6733 定义的标准端口。每个对等方在 能力交换（CER/CEA）期间宣传它所支持的领域和应用程序。DRA 将此记录在其 对等注册表 中，以便在路由决策中使用。

请求处理管道

每个传入请求都通过一个有序的管道。安全和引导阶段首先运行。接下来运行路由阶段。应答绕过此管道。请参见 消息中继。



传入 Diameter 请求

[redacted]

[redacted]

[redacted]

[redacted]

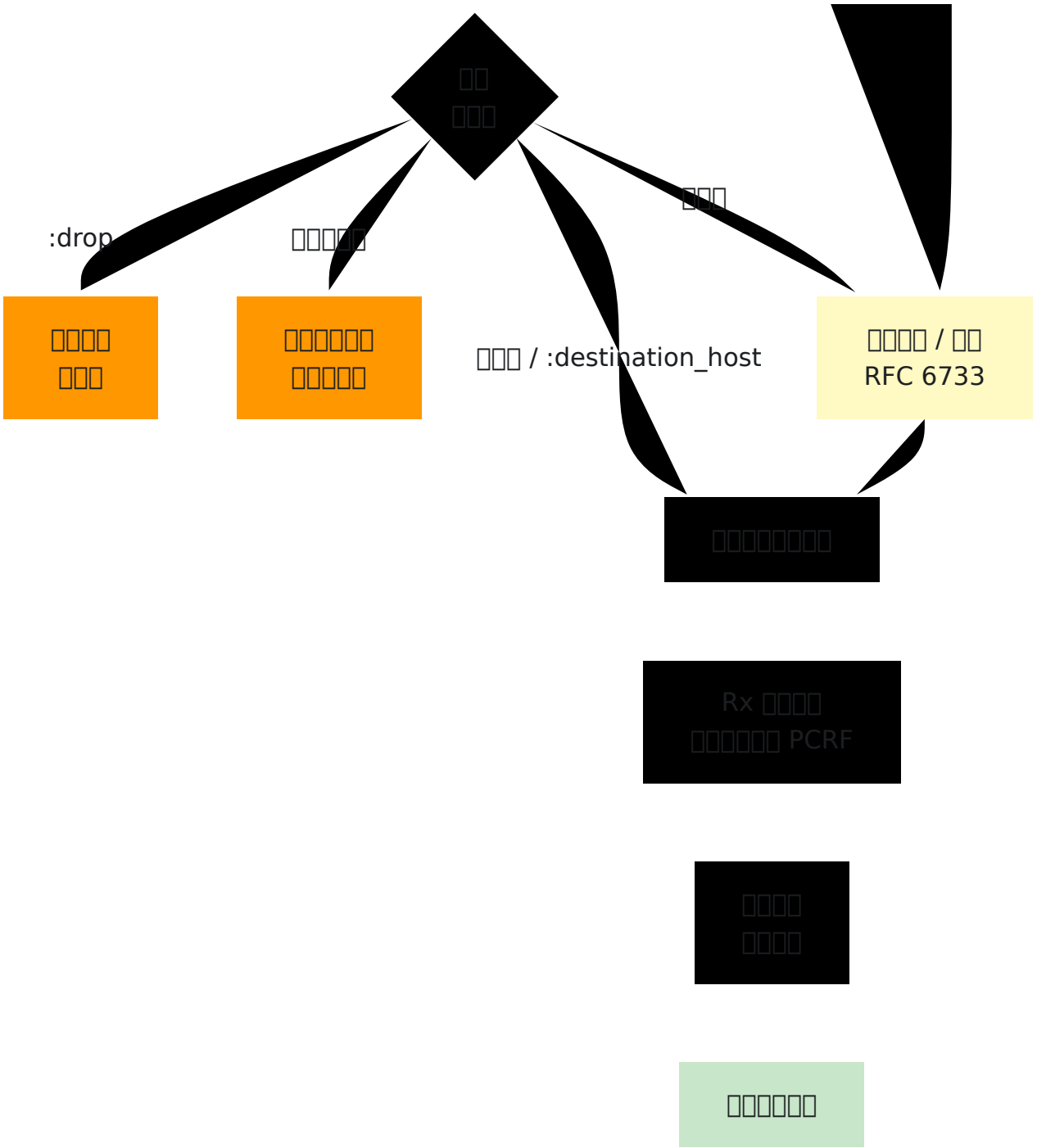
[redacted]

[redacted]

[redacted]

是

评估路由规则



:drop
错误,
配
丢弃消息
无应答
生成错误应答
带结果代码
对等方 / :destination_host
标准路由 / 中继
RFC 6733
Rx
PCRF
转发到对等方

管道阶段

阶段	目的
速率限制器	强制每个对等方的请求速率限制，以保护下游服务免受过载。
Diameter 防火墙	根据 GSMA FS.19 低层和类别 1/2/3 检查检查请求。请参见 安全。

阶段	目的
AVP 清理器	剥离不允许的供应商 AVP 并强制 AVP 嵌套深度限制。
漫游引导	应用运营商定义的引导，以影响漫游订阅者被引导到哪个网络。
订阅者查找	在规则评估之前解析特定于订阅者的路由上下文。
□级路□□	评估基于规则的路由，可以丢弃、错误或将请求定向到特定对等方。请参见 □级路由。
标准路由 / 中继	当没有适用的□级规则时，回退到 RFC 6733 优先级路由。请参见 标准路由。
Rx 会话绑定	将 Rx 请求重定向到持有其会话的 PCRF。这是唯一一个在□级路由之后运行的阶段。Rx `Session-Termination-Request` 不携带其他路由信息。请参见 Rx 会话绑定。
拓扑隐藏	在出站消息上重写或剥离标识 AVP（Route-Record、Origin-Host/Realm）。请参见 安全。

当□级路由产生 `:drop` 结果时，DRA 会丢弃请求而不作应答。`error` 结果生成一个携带配置的 **Result-Code** 的 Diameter 应答。否则，DRA 将匹配的请求转发到指定的对等方或 **Destination-Host** 对等方。未匹配的请求将进入标准路由。

Rx 会话绑定是□级路由的一个例外，它拥有最后的决定权。它的范围很窄。它只重定向已经要被中继的请求，并且仅适用于 Rx。它不会影响 `:drop` 结果、`error` 结果和显式的 `Destination-Host`。

标准路由优先级

当请求到达标准路由时，DRA 使用 RFC 6733 §6.1 的优先级机制选择一个对等方：

1. **Destination-Host AVP (293)** - 如果请求包含此 AVP，DRA 直接路由到指定的对等方。这是最□优先级的机制。如果该对等方未连接，则路由失败。
2. **Destination-Realm AVP (283)** - 如果缺少 Destination-Host，DRA 选择一个连接的对等方，该对等方宣传目标领域。DRA 在匹配的对等方之间进行负载均衡。
3. **Application-Id** - DRA 将候选对等方过滤为那些宣传支持消息的应用程序的对等方。对等方在能力交换期间宣传此支持。

完整的标准路由行为，包括对等方选择算法，记录在 标准路由 中。

对等注册表

DRA 维护一个内存中的 对等注册表。该注册表跟踪每个对等方连接和对等方在能力交换期间宣传的能力。该注册表是路由决策的权威来源。只有处于 连接 状态的对等方才有资格接收流量。DRA 会自动排除断开连接或故障的对等方。

对于每个连接的对等方，注册表记录：

- 对等方的 **Diameter** 身份（Origin-Host / FQDN）。
- 对等方宣传的 领域。
- 对等方支持的 **Application-Ids**。
- 当前的 连接状态。

DRA 从 CER/CEA 动态学习领域和应用程序支持。要为接口增加容量，请连接一个额外的对等方，该对等方宣传相关的领域和应用程序。DRA 会自动开始对新对等方进行负载均衡。

消息中继

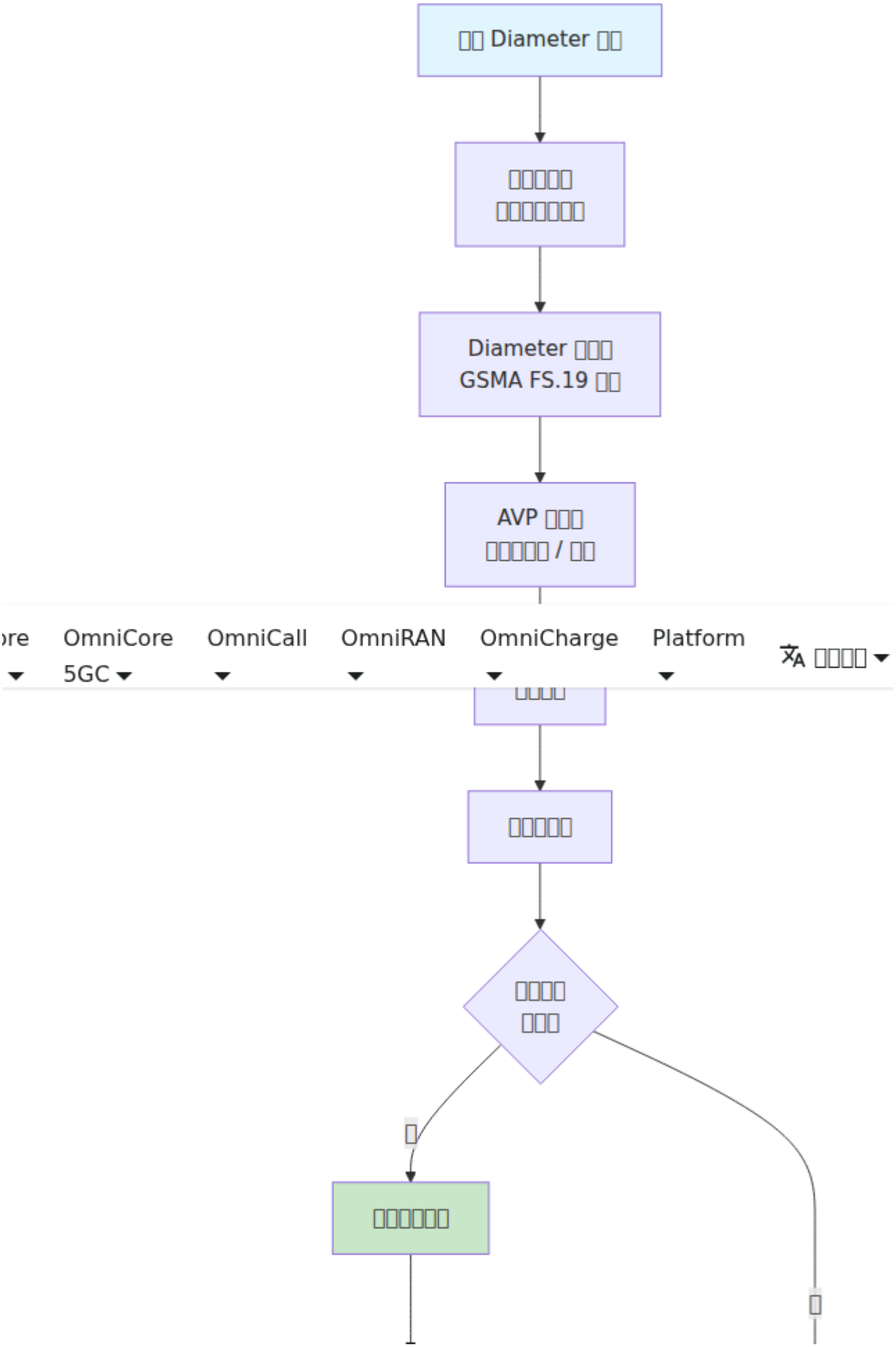
DRA 中继两类根本不同的消息，并以不同的方式处理它们。

请求

请求遍历完整的 处理管道。DRA 选择一个目标对等方，更新它管理的路由 AVP，并转发消息。DRA 是一个中继，因此它将应用负载原封不动地传递。

应答

应答从不重新进入路由管道。相反，DRA 将每个应答与其原始请求关联。然后，DRA 沿着反向路径返回应答。


□□ Diameter □□
□□□□□□
□□□□□□□□
Diameter □□□
GSMA FS.19 □□
AVP □□□
□□□□□ / □□
ore
OmniCore
5GC
OmniCall
OmniRAN
OmniCharge
Platform
□□□□□
□□□□□□
□□□□
□□□
□
□□□□□□□
□

应答的中继规则是：

- 基于会话的路由 - 应答总是沿着其请求的反向路径返回。
- 端到端 **ID** 保持 - 端到端标识符 在每个跳跃中保持不变。
- 逐跳路由 - 逐跳标识符 在每个跳跃时变化。DRA 使用它来维护路由状态并将应答匹配回发送请求的对等方。
- 无规则评估 - DRA 不会评估路由规则或检查应答的 AVP 内容。
- 有状态关联 - 内部路由表跟踪每个请求是由哪个对等方发送的。DRA 在中继应答后清理此状态。

应答路由是确定性的。Diameter 协议要求应答沿着已建立的请求路径返回到原始对等方。这保持了正确的会话管理并防止路由循环。有关应答消息路由的详细信息，请参见 RFC 6733 §6.2。

参考表

常见 3GPP 应用程序 ID

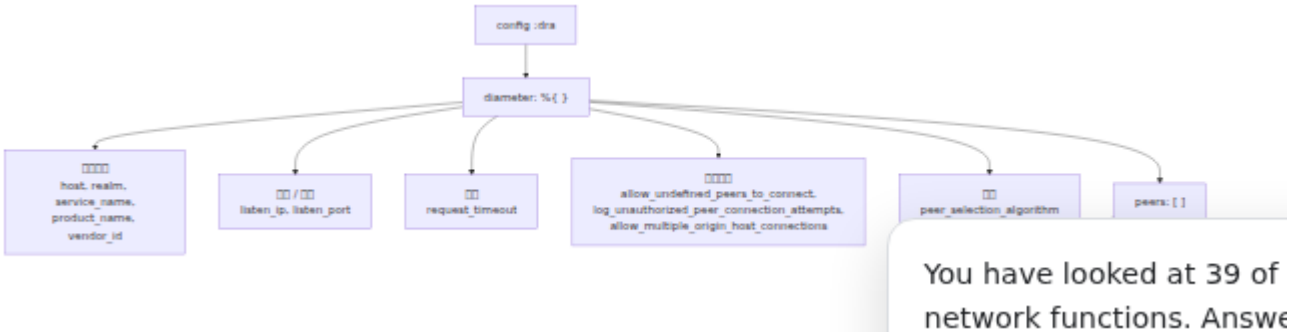
ID	接口	描述	参考
16777251	S6a/S6d	MME/SGSN ↔ HSS 认证和订阅	3GPP TS 29.272
16777238	Gx	PCEF ↔ PCRF 策略和计费控制	3GPP TS 29.212
4	Gy	在线计费（信用控制）↔ OCS	3GPP TS 32.299

常见路由 AVP

代码	名称	描述	参考
293	Destination-Host	显式主机级路由目标（最□优先级）	RFC 6733 §6.5
283	Destination-Realm	用于选择连接对等方的领域	RFC 6733 §6.4
282	Route-Record	记录消息经过的中继的身份	RFC 6733 §6.7.1

进一步阅读

- 标准路由 - RFC 6733 优先级路由、对等方选择和应答路由的详细信息。
- □级路由 - 基于规则的路由、丢弃/错误结果和对等方定位。
- 配置 - 基础 DRA 配置、传输和模块设置。
- 安全 - 速率限制、GSMA FS.19 Diameter 防火墙、AVP 清理和拓扑隐藏。



基础 DRA 配置

基础 **DRA** 配置 定义了 OmniDRA 的身份、网络监听器、超时、对等接入策略以及它连接的 Diameter 对等体列表。这个块是代理做出每个路由决策的基础。在 `config :dra, diameter: %{...}` 中声明它，位于 `config/runtime.exs`。

概述

所有基础设置都位于一个单一的 `diameter` 映射中。该映射包含一组描述 DRA 本身的顶级键。它还包含一个 `peers` 列表。列表中的每个条目描述一个 Diameter 对等体。

配置结构

```
config :dra,
  diameter: %{
    service_name: :omnitouch_dra,
    listen_ip: "127.0.0.2",
    listen_port: 3868,
    host: "dra01",
    realm: "production.omnitouch",
    product_name: "OmniDRA",
    vendor_id: 10415,
    request_timeout: 5000,
    allow_undefined_peers_to_connect: false,
    log_unauthorized_peer_connection_attempts: true,
    allow_multiple_origin_host_connections: false,
    peer_selection_algorithm: :random,
    peers: [
      # 对等体配置...
    ]
  }
```

顶级参数

这些键配置 DRA 自身的身份、网络监听器、超时以及对等接入和对等选择策略。每个键在 `diameter` 映射中出现一次。

□□	□□	□□	□□	
`service_name`	□□	□	-	Diameter □□□□□□ `:c`
`listen_ip`	□□□□□□□	□	-	DRA □□□□□ □□□□□□□□□ □ □□□□□□□ Multihomi
`listen_port`	□□	□	-	DRA □□□□□ `3868`□
`host`	□□□	□	-	DRA □ Dia `realm` □□□ `Host`□□□ `dra01.pro` □ CER/CEA
`realm`	□□□	□	-	DRA □ Dia `Origin-Re`
`product_name`	□□□	□	-	□ CER/CEA
`vendor_id`	□□	□	-	□□ RFC 67 □□ `Vendor` □□□□□□
`request_timeout`	□□	□	`5000`	□□/□□□□□□ □□DRA □□□

参数	类型	必需	默认	
`allow_undefined_peers_to_connect`	布尔	是	`false`	当为 `false` 以建立连接。当为 配置中列出的对等[`false`。
`log_unauthorized_peer_connection_attempts`	布尔	是	`true`	当为 `true` 时 接尝试以进行安全[
`allow_multiple_origin_host_connections`	布尔	是	`false`	当为 `false` `Origin-Ho` 时，DRA 允许[个并发连接。
`peer_selection_algorithm`	原子	是	-	在路由过程中选择[对等选择 和 标准[
`peer_settle_time_ms`	整数	否	`0`	新连接的对等体在 动的毫秒数。这可[CER/CEA [
`watchdog_timer_ms`	整数	否	`30000`	RFC 3539 隔）的可选覆盖。[参见 对等连接性[
`peers`	列表	是	-	对等体映射的列表[对等体连接。请参[

SCTP 套接字缓冲区大小

套接字缓冲区大小是一个 顶级 `:dra` 键。使用 `config :dra, diameter_socket_buffer_size:` 进行配置。它 不是 `diameter:` 映射的成员。

```
config :dra, diameter_socket_buffer_size: 4_194_304

config :dra,
  diameter: %{
    # ... 基础配置 ...
  }
```

参数	类型	必需	默认	描述
`diameter_socket_buffer_size`	整数	否	`4194304` (4 MB)	SCTP 套接字接收/发送缓冲区大小（字[`:dra` 键（`config :dra, diameter_socket_buffe` 不是 `diameter:` 映射的成员。仅[`:diameter_sctp` 对等体。请[Multihoming。

对等选择

当多个配置的对等体匹配请求的路由标准时，`peer_selection_algorithm` 决定哪个对等体接收请求。此设置影响 DRA 如何分配流量以及在对等体故障期间的行为。有关完整的路由决策流程，请参见 标准路由。有关基于规则的覆盖，请参见 级路由。

值	行为
`:random`	DRA 为每个请求随机选择一个匹配的对等体。这在所有合格对等体之间分散负载。
`:failover`	DRA 优先选择第一个匹配的对等体。仅在优先级较的对等体不可用时使用后续对等体。

对等配置

`peers` 列表中的每个条目是一个描述单个 Diameter 对等体的映射。该映射定义了其身份、网络地址、所使用的传输及 DRA 是拨打对等体还是等待被拨打。

```
%{
  host: "diameter-benchmark-server.production.omnitouch",
  realm: "production.omnitouch",
  ip: "127.0.0.3",
  port: 3868,
  transport: :diameter_tcp,
  initiate_connection: true
}
```

每个对等体参数

参数	类型	必需	默认	描述
`host`	字符串	是	-	对等体的 Diameter 身份（FQDN）。此值必须完全匹配，并且区分大小写。路由和 □级 路由 规则需要精确匹配。
`realm`	字符串	是	-	对等体的 Diameter 领域。
`ip`	字符串	是	-	用于连接的对等体的主要 IP 地址。
`additional_ips`	列表	否	-	用于多宿主 SCTP 关联的 IP 地址列表。当未设置时，DRA 使用 `ip`。此键仅适用于 `transport: :diameter_sctp`。请参见 SCTP Multihoming。
`port`	整数	是	-	对等体的 Diameter 端口（通常为 `3868`）。
`transport`	原子	是	-	连接的传输协议：`:diameter_tcp` 用于 TCP，或 `:diameter_sctp` 用于 SCTP。请参见下面的 传输 参考。
`source_ip`	字符串	否	`listen_ip`	DRA 拨打此对等体时要绑定的本地 IP 地址（仅限出站连接）。默认为服务的 `listen_ip`。
`source_port`	整数	否	-	DRA 拨打此对等体时要绑定的本地端口（仅限出站连接）。未设置时，操作系统分配一个临时端口。如果两个对等体共享相同的目标 IP 和端口，

参数	类型	必需	默认	描述
				请设置一个显式值。每个连接使用不同的本地套接字。
`initiate_connection`	布尔	是	-	当为 `true` 时，DRA 连接到对等体（客户端角色）。当为 `false` 时，DRA 等待对等体连接（服务器角色）。请参见 连接模式。
`origin_host`	字符串	否	服务 `host.realm`	DRA 向此对等体呈现的 `Origin-Host`。此键与上面的 `host` 分开，后者是此对等体的身份。当对等体期望 DRA 使用特定主机时设置此键（例如合作伙伴领域）。DRA 将覆盖应用于此连接的 `CER`/`CEA` 和基本协议消息。请参见 每个对等体身份。
`origin_realm`	字符串	否	服务 `realm`	DRA 向此对等体呈现的 `Origin-Realm`。此键与上面的 `realm` 分开，后者是此对等体的领域。与 `origin_host` 一起或单独设置此键。请参见 每个对等体身份。

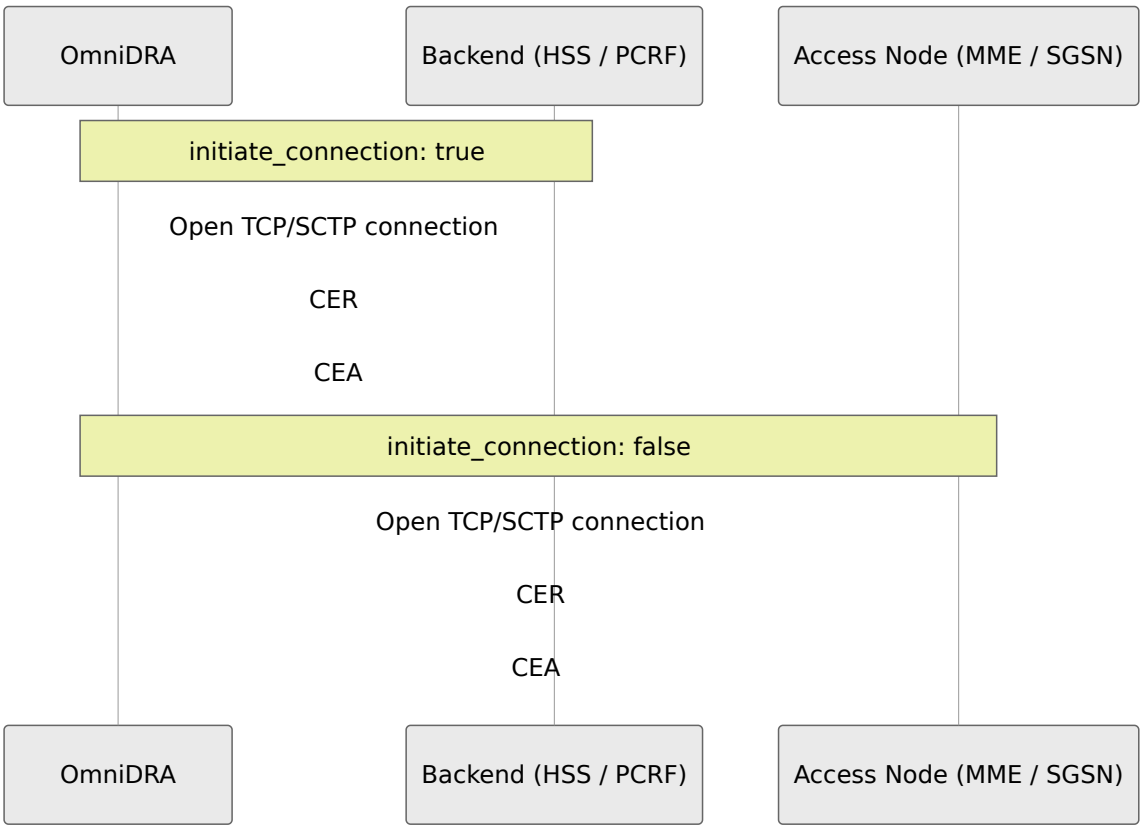
> **SCTP** 多宿主：使用 `:diameter_sctp` 的对等体可以绑定多个本地和远程地址以提□弹性。SCTP Multihoming 描述了多宿主设置和每个对等体地址列表。

传输

传输值	协议	参考
`:diameter_tcp`	TCP	RFC 6733 (Diameter Base)
`:diameter_sctp`	SCTP	RFC 4960 (SCTP) - 请参见 SCTP Multihoming

连接模式

`initiate_connection` 键控制 DRA 是拨打对等体还是等待被拨打。



发起连接（`initiate_connection: true`）

- DRA 作为 Diameter 客户端并发起与对等体的 TCP/SCTP 连接。
- 对于后端系统（如 HSS 或 PCRF），使用此模式。
- 如果对等体不可达，DRA 会重试连接。

接受连接（`initiate_connection: false`）

- DRA 作为 Diameter 服务器并等待对等体连接。
- 对于接入侧节点（如 MME、SGSN 或 P-GW），使用此模式。
- 对等体必须在 `peers` 中存在，除非您将 `allow_undefined_peers_to_connect` 设置为 `true`。
- DRA 通过它们在 CER 中广告的 `Origin-Host` 区分入站对等体。两个 Diameter 节点可以共享一个源 IP。DRA 在��个节点呈现不同的 `host` 时独立匹配它们。

每个对等体身份

DRA 具有一个服务范围的身份。它将 `host` 和 `realm` 组合成 `Origin-Host`（例如 `dra01.production.omnitouch`），并将 `realm` 广告为 `Origin-Realm`。默认情况下，DRA 向每个对等体呈现此身份。

某些对等体期望 DRA 呈现不同的身份。互联或漫游合作伙伴通常要求 DRA 使用合作伙伴命名方案中的主机和领域。设置 `origin_host` 和 `origin_realm` 以更改 DRA 在该连接上呈现的身份。

这两个键是 DRA 向对等体呈现的身份。它们与对等体自己的 `host` 和 `realm` 分开，后者是对等体向 DRA 广告的身份。DRA 仅将覆盖应用于该一个连接。每个其他对等体继续看到服务范围的身份。

```
peers: [
  %{
    # 此对等体的身份（它向 DRA 广告的内容）：
    host: "hss.partner.example",
    realm: "partner.example",
    ip: "10.0.0.9",
    port: 3868,
    transport: :diameter_sctp,
    initiate_connection: true,
    # DRA 向此对等体呈现的身份：
    origin_host: "dra-partner.epc.mnc001.mcc001.3gppnetwork.org",
    origin_realm: "epc.mnc001.mcc001.3gppnetwork.org"
  }
]
```

请参见 每个对等体参数 以获取 `origin_host` 和 `origin_realm` 参数定义。

下面的图表显示 DRA 向一个对等体呈现其默认身份，并向一个合作伙伴对等体呈现覆盖身份。

行为

覆盖更改了该连接上的 能力交换（`CER` / `CEA`）中的身份。它还更改了该连接上基本协议消息中的身份：看门狗消息（`DWR` / `DWA`）和断开消息（`DPR` / `DPA`）。

您可以单独设置任一键。未设置的键将回退到服务值。如果仅设置 `origin_realm`，DRA 仅更改领域并保持服务范围的 `Origin-Host`。

DRA 在该连接上使用覆盖的 `Origin-Host` 进行循环检测。如果请求已经携带与覆盖匹配的 `Route-Record` AVP，DRA 仍然会根据 RFC 6733 §6.1.8 检测循环。

DRA 在发送 `CER` 时固定身份。如果您在运行时更改覆盖并重新加载配置，DRA 将重新建立该连接以使新身份生效。与其他对等体的连接不受影响。

完整示例

以下示例配置了一个 DRA。DRA 接受来自 MME 的 SCTP 连接。它通过 TCP 拨打 HSS 和 PCRF。

```
config :dra,
  diameter: %{
    service_name: :mvno_dra,
    listen_ip: "10.100.1.10",
    listen_port: 3868,
    host: "dra01",
    realm: "mvno.example.com",
    product_name: "OmniDRA",
    vendor_id: 10415,
    request_timeout: 5000,
    allow_undefined_peers_to_connect: false,
    log_unauthorized_peer_connection_attempts: true,
    allow_multiple_origin_host_connections: false,
    peer_selection_algorithm: :random,
    peers: [
      # MME - DRA □□ MME □□
      %{
        host: "mme01.operator.example.com",
        realm: "operator.example.com",
        ip: "10.100.2.15",
        port: 3868,
        transport: :diameter_sctp,
        initiate_connection: false
      },
      # HSS - DRA □□□□□□□□□□□□
      %{
        host: "hss01.mvno.example.com",
        realm: "mvno.example.com",
        ip: "10.100.3.141",
        port: 3868,
        source_port: 13868,
        transport: :diameter_tcp,
        initiate_connection: true
      },
      # PCRF - DRA □□□□
      %{
        host: "pcrf01.mvno.example.com",
        realm: "mvno.example.com",
        ip: "10.100.3.22",
        port: 3868,
        transport: :diameter_tcp,
        initiate_connection: true
      }
```

```
    ]
  }
```

操作说明

- 主机名匹配 - DRA 精确且区分大小写地匹配对等体 `host` 值。□级路由 中的规则必须引用您在此处配置的相同 `host` 字符串。
- 能力交换 - 连接时，DRA 与每个对等体通过 CER/CEA 交换支持的应用程序和身份。DRA 在此阶段广告 `host`、`realm`、`product_name` 和 `vendor_id` 值。
- 供应商 **ID 10415** - 这是标准的 3GPP `Vendor-Id`。这是 3GPP Diameter 接口的预期值。
- 对等选择 - 当多个对等体匹配请求时，`peer_selection_algorithm` 决定 DRA 使用哪个。请参见 标准路由。

安全考虑

- ��生产环境中将 `allow_undefined_peers_to_connect` 设置为 `false`，以便仅允许已知对等体连接。
- 启用 `log_unauthorized_peer_connection_attempts: true` 以记录被拒绝的连接尝试以进行安全监控。
- 确保防火墙规则与配置的 `listen_ip` 和 `listen_port` 一致。

相关文档

- SCTP Multihoming - 多宿主 SCTP 传输和每个对等体地址配置。
- 标准路由 - DRA 如何选择对等体和路由请求，包括 `peer_selection_algorithm` 行为。
- □级路由 - 基于规则的路由覆盖，基于对等体 `host` 值。

指标与监控

OmniDRA 提供全面的 **Prometheus** 指标。这些指标涵盖对等连接、消息速率、成功和错误结果、响应延迟、丢失流量和未经授权的连接尝试。此页面记录每个指标、其标签和现成的 PromQL 查询。使用它们构建仪表板和警报。

可观察性流程

OmniDRA 在 `/metrics` 端点以 Prometheus 展示格式发布所有指标。Prometheus 定期抓取此端点并存储样本。然后，仪表板或警报工具查询生成的时间序列。

Prometheus 从 `/metrics` 抓取 标准 **Prometheus** 指标。扩展指标模块将基数、订阅者级遥测直接发布到 **InfluxDB**。这使得 Prometheus 系列计数保持可管理（请参见 扩展指标模块）。

> OmniDRA 还在 `/metrics` 上暴露 **BEAM/Erlang** 运行时指标。此页面不记录它们。所有计数器都是累积的。使用 `rate()` 查询它们以获取每秒速率。

核心直径指标

这些指标始终可用。它们描述对等健康状况和 DRA 中的消息流。

对等连接

对等状态

- 指标: `diameter_peer_status`

- 类型：Gauge
- 描述：对等方是否连接（`1`）或未连接（`0`）。

标签：

标签	描述
`origin_host`	对等方的 直径身份 (FQDN)。
`ip`	对等方的 IP 地址。

示例查询：

```
# 检查特定对等方是否连接
diameter_peer_status{origin_host="hss01.example.com"}

# 计算未连接的对等方
count(diameter_peer_status == 0)
```

有关诊断频繁变动或从未达到连接状态的对等方，请参见 故障排除。

支持的应用程序

- 指标：`diameter_peer_supported_applications`
- 类型：Gauge
- 描述：对等方在能力交换（CER/CEA）期间广告的应用程序（如已知）。

标签：

标签	描述
`origin_host`	对等方的直径身份 (FQDN)。
`supported_applications`	对等方广告的支持应用程序。

示例查询：

```
# 特定对等方广告的应用程序
diameter_peer_supported_applications{origin_host="hss01.example.com"}
```

消息速率

消息计数

- 指标：`diameter_peer_message_count_total`
- 类型：Counter
- 描述：与对等方交换的直径消息的总数。

标签：

标签	描述
`origin_host`	对等方的直径身份。
`received_from`	发送消息的对等方。
`application_id`	直径应用程序 ID。请参见 应用程序 ID。
`cmd_code`	直径命令代码。请参见 命令代码。
`application_name`	可读的应用程序名称（例如 `3GPP_S6a`）。
`cmd_name`	可读的命令名称（例如 `AIR`）。
`direction`	`request` 或 `response`。

示例查询：

```
# 从特定 MME 的 S6a AIR 请求速率
rate(diameter_peer_message_count_total{
  cmd_code="318",
  direction="request",
  origin_host="mme01.example.com"
}[5m])

# 按应用程序的总消息速率
sum by (application_name)
(rate(diameter_peer_message_count_total[5m]))
```

成功与错误率

响应结果代码

- 指标：`diameter_peer_message_result_code_count_total`
- 类型：Counter
- 描述：按结果代码统计的直径响应总数。

标签：

标签	描述
`origin_host`	原始请求者。
`routed_to`	发送答案的对等方。
`application_id`	直径应用程序 ID。
`cmd_code`	直径命令代码。
`application_name`	应用程序名称。
`cmd_name`	命令名称。
`result_code`	直径结果代码或实验结果代码。

示例查询：

```
# S6a AIR 请求的成功率
rate(diameter_peer_message_result_code_count_total{
  cmd_code="318",
  result_code="2001"
}[5m])

# 按结果代码的错误率
sum by (result_code) (
  rate(diameter_peer_message_result_code_count_total{
    result_code!="2001"
  }[5m])
)
```

常见结果代码：

代码	名称	参考
`2001`	DIAMETER_SUCCESS	RFC 6733 §7.1.1
`3002`	DIAMETER_UNABLE_TO_DELIVER	RFC 6733 §7.1.3
`3003`	DIAMETER_REALM_NOT_SERVED	RFC 6733 §7.1.3
`3004`	DIAMETER_TOO_BUSY	RFC 6733 §7.1.3
`5001`	DIAMETER_AUTHENTICATION_REJECTED	RFC 6733 §7.1.5
`5004`	DIAMETER_INVALID_AVP_VALUE	RFC 6733 §7.1.5
`5012`	DIAMETER_UNABLE_TO_COMPLY	RFC 6733 §7.1.5

延迟

响应延迟

- 指标：`diameter_peer_last_response_delay`
- 类型：Gauge
- 描述：最近的响应延迟（毫秒）（DRA → 对等方 → DRA）。

标签：

标签	描述
`origin_host`	原始请求者。
`routed_to`	发送答案的对等方。
`application_name`	应用程序名称。
`cmd_name`	命令名称。

示例查询：

```
# 从 HSS 的最近响应延迟
diameter_peer_last_response_delay{routed_to="hss01.example.com"}

# 跨 S6a 对等方的最高最近响应延迟
max(diameter_peer_last_response_delay{application_name="3GPP_S6a"})
```

> 这是一个保持每个对等方最近延迟的 Gauge。它不是直方图，因此诸如 `histogram_quantile` 的分位数函数不适用。

未回答的请求

未回答请求计数

- 指标：`diameter_peer_unanswered_request_count_total`
- 类型：Counter
- 描述：DRA 发送但对等方未在超时期间内回答的请求。

标签：

标签	描述
`origin_host`	原始请求者。
`routed_to`	未回答的对等方。
`application_id`	直径应用程序 ID。
`cmd_code`	直径命令代码。
`application_name`	应用程序名称。
`cmd_name`	命令名称。

示例查询：

```
# 未回答请求的速率
rate(diameter_peer_unanswered_request_count_total[5m])

# 识别问题对等方
topk(5, sum by (routed_to) (
  rate(diameter_peer_unanswered_request_count_total[5m])
))
```

未经授权的连接

未经授权的连接尝试

- 指标：`diameter_peer_unauthorized_connection_count_total`
- 类型：Counter
- 描述：来自未经授权对等方的连接尝试。

标签：

标签	描述
`origin_host`	未经授权对等方声称的身份。
`supported_applications`	对等方广告的应用程序。
`peer_ip`	连接尝试的 IP 地址。

示例查询：

```
# 未经授权的连接尝试
rate(diameter_peer_unauthorized_connection_count_total[5m])

# 对未经授权访问的警报
diameter_peer_unauthorized_connection_count_total > 0
```

有关安全模块指标（防火墙、速率限制、拓扑隐藏、AVP 清理）的更详细信息，请参见 安全。

□级路由模块指标

这些指标描述□级路由规则产生的结果。有关相关流量控制遥测的信息，请参见 安全 中的安全模块文档。

丢失流量

丢弃计数

- 指标：`diameter_advanced_routing_drop_count_total`
- 类型：Counter
- 描述：□级路由 `:drop` 操作丢弃的请求。DRA 不发送响应。

标签：

标签	描述
`application_id`	直径应用程序 ID。
`cmd_code`	直径命令代码。
`application_name`	应用程序名称。
`cmd_name`	命令名称。

示例查询：

```
# 按命令的丢弃速率
sum by (cmd_name) (
  rate(diameter_advanced_routing_drop_count_total[5m])
)

# 总丢失流量
sum(rate(diameter_advanced_routing_drop_count_total[5m]))
```

当 **DRA** 丢弃流量时：

- 级路由规则与 丢 操作匹配。
- DRA 丢弃消息。
- DRA 不向请求对等方发送答案消息。

速率限制与过载

错误响应

- 指标：`diameter_advanced_routing_error_count_total`
- 类型：Counter
- 描述：级路由错误操作生成的错误响应。

标签：

标签	描述
`result_code`	在错误响应中发送的直径结果代码。
`application_id`	直径应用程序 ID。
`cmd_code`	直径命令代码。
`application_name`	应用程序名称。
`cmd_name`	命令名称。

示例查询：

```
# 按结果代码的错误响应
sum by (result_code) (
  rate(diameter_advanced_routing_error_count_total[5m])
)

# DIAMETER_TOO_BUSY 响应
rate(diameter_advanced_routing_error_count_total{
  result_code="3004"
}[5m])

# 速率限制或过载检测
diameter_advanced_routing_error_count_total{
  result_code="3004"
} > 100
```

当 **DRA** 生成错误响应时：

- □级路由规则与指定的结果代码的 错误 操作匹配。
- DRA 生成带有指定结果代码的直径答案。
- DRA 将答案发送回请求对等方，而不转发答案。

常用错误代码：

代码	名称	含义
`3002`	DIAMETER_UNABLE_TO_DELIVER	路由失败。
`3003`	DIAMETER_REALM_NOT_SERVED	不支持的领域。
`3004`	DIAMETER_TOO_BUSY	过载保护。
`5012`	DIAMETER_UNABLE_TO_COMPLY	一般错误。

扩展指标模块

扩展指标模块 提供□级遥测和分析。使用它来分析超出标准指标的直径流量模式。它当前跟踪 LTE 订阅者附加尝试。它关联 S6a 认证信息请求（AIR）和答案（AIA）消息对。

这些指标携带 □基数 字段，如 IMSI。因此，该模块异步写入 **InfluxDB**，并不在 Prometheus `/metrics` 端点上暴露它们。这使得 Prometheus 系列计数保持在边界内，避免指标爆炸。

配置

该模块默认禁用。在 `config/runtime.exs` 中配置：

```
module_extended_metrics:
  enabled: true
  attach_attempt_reporting_enabled: true
```

参数	类型	必需	默认	描述
`enabled`	布尔值	否	`false`	设置为 `true` 以激活扩展指标模块。当为 `false` 时，DRA 跳过该模块。
`attach_attempt_reporting_enabled`	布尔值	否	`false`	启用 LTE 附加尝试（S6a AIR/AIA）的跟踪和报告。

InfluxDB 连接

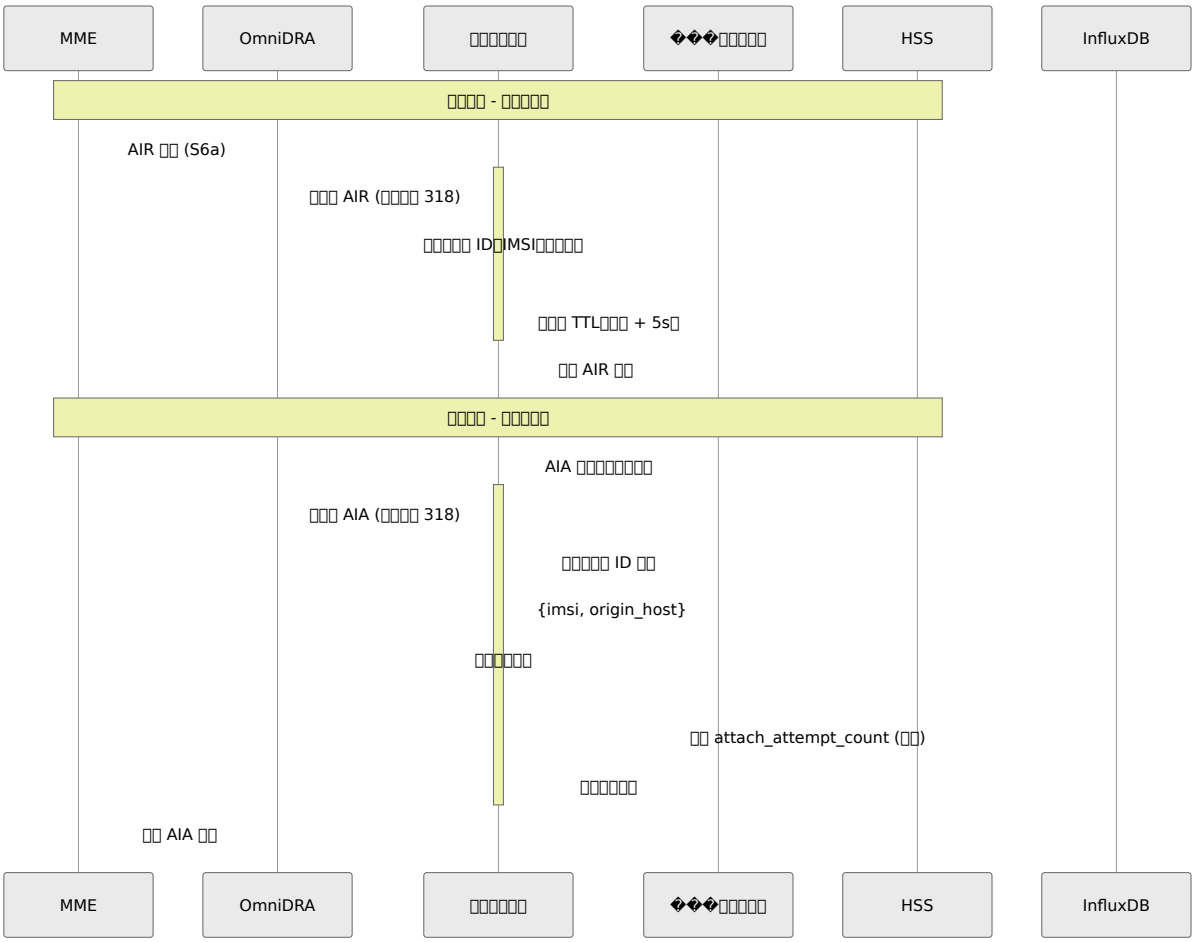
扩展指标模块通过 Instream 连接异步写入 InfluxDB（Instream）。在 `config :dra, DRA.Metrics.InfluxDB` 下配置目标。

```
config :dra, DRA.Metrics.InfluxDB,
  auth: [username: "example_user", password: "example_password"],
  database: "dra",
  host: "127.0.0.15",
  port_udp: 8089,
  writer: Instream.Writer.Line
```

参数	类型	必需	默认	描述
`auth`	关键字列表	否	-	InfluxDB 凭据，格式为 `[username: ..., password: ...]`。
`database`	字符串	是	-	接收指标测量的目标 InfluxDB 数据库。
`host`	字符串	是	-	InfluxDB 服务器的主机名或 IP 地址。
`port_udp`	整数	否	`8089`	将行协议数据写入 InfluxDB 的 UDP 端口。使用 InfluxDB 的 UDP 监听端口（默认 `8089`），而不是 HTTP API 端口（`8086`）。
`writer`	模块	否	`Instream.Writer.Line`	Instream 写入模块。`Instream.Writer.Line` 使用行协议进行写入。

附加尝试跟踪

该模块监控 S6a AIR/AIA 对。它通过直径端到端 ID 将每个请求与其答案关联。然后，它发出一个订阅者级指标。该指标携带原始对等方和 HSS 结果代码。



MME
OmniDRA
扩展指标模块
���存关联存储
HSS
InfluxDB
请求阶段 - 捕获元数据
AIR 请求 (S6a)
检测到 AIR (命令代码 318)
提取端到端 ID、IMSI、原始主机
存储带 TTL (超时 + 5s)
转发 AIR 请求
答复阶段 - 关联和报告
AIA 答复 (结果代码)
检测到 AIA (命令代码 318)
通过端到端 ID 查找
{imsi, origin_host}
提取结果代码
写入 attach_attempt_count (异步)
删除关联条目
转发 AIA 答复
MME
OmniDRA
扩展指标模块
���存关联存储
HSS
InfluxDB

测量：`attach_attempt_count`

字段：

字段	描述
`imsi`	订阅者 IMSI，取自用户名称 AVP。

标签：

标签	描述
`origin_host`	发起附加请求的对等方。
`result_code`	HSS 响应中的直径结果代码。

写入到此测量的结果代码是标准直径代码：

代码	名称
`2001`	DIAMETER_SUCCESS
`5001`	DIAMETER_AUTHENTICATION_REJECTED
`5004`	Diameter AVP unsupported

请参见 RFC 6733 以获取完整的结果代码列表。

用例

- 附加成功率分析 - 按结果代码跟踪成功与失败的附加尝试。
- **IMSI** 级故障排除 - 识别有附加失败的个别订阅者。
- 网络性能监控 - 按来源（MME/SGSN）监控附加尝试模式。
- 漫游分析 - 分析入境漫游附加成功率。

操作说明

- 附加尝试指标仅跟踪 S6a AIR/AIA 对（应用程序 ID `16777251`，命令代码 `318`）。
- 请求元数据在配置的请求超时加 5 秒后过期。
- 指标处理是异步的。它不会阻塞消息流。
- 该模块独立于路由和转换模块操作。

参考表

应用程序 ID 参考

ID	应用程序	描述	参考
16777251	S6a/S6d	MME/SGSN ↔ HSS 认证和订阅	3GPP TS 29.272
16777238	Gx	PCEF ↔ PCRF 策略和计费控制	3GPP TS 29.212
16777216	Cx	I-CSCF/S-CSCF ↔ HSS IMS 注册	3GPP TS 29.229

❓❓令代码参考

代码	命令	应用程序	参考
318	AIR/AIA (认证信息)	S6a	3GPP TS 29.272

注意事项

- 所有标准指标均在 `/metrics` 端点以 Prometheus 格式提供。
- 扩展指标模块 通过 InfluxDB 提供额外的 S6a 特定指标。
- OmniDRA 还暴露 **BEAM/Erlang** 运行时指标。此页面不记录它们。
- 所有计数器都是累积的。使用 `rate()` 查询它们以获取每秒速率。
- 只有扩展指标模块使用□基数标签，如 IMSI。这避免了指标爆炸。

相关文档

- 故障排除 - 诊断连接、路由和交付问题。
- 安全 - 有关安全模块指标的详细信息（防火墙、速率限制、拓扑隐藏、AVP 清理）。

对等连接性与稳定时间

该模块控制Diameter对等体在连接后变得可路由的速度。它还防止对频繁连接和断开的对等体的流量。频繁连接和断开的对等体会反复连接和断开。

背景

在OmniDRA路由到对等体之前，对等体必须完成能力交换（CER/CEA）并被提升为服务。默认情况下，RFC 3539看门狗将□□□□的对等体保持在REOPEN状态。对等体保持在该状态，直到它通过连续的看门狗（DWR/DWA）交换。看门狗定时器**Tw**以大约30秒的标准30000毫秒默认值来调节这些交换。即使CER/CEA已经成功，此保持仍然适用。

OmniDRA消除了这种隐式延迟。看门狗立即提升重新连接的对等体。OmniDRA用稳定时间替换了延迟。稳定时间是一个明确的、可配置的保持时间，适用于对等体连接后。在保持期间，对等体已连接但不可路由。如果它在窗口内断开，OmniDRA将取消保持并不向其发送流量。

配置

这两个键都是可选的，并位于`:diameter`配置映射中。如果您未设置任何键，对等体在连接后立即路由。

```
config :dra,
  diameter: %{
    # ...服务设置...
    peer_settle_time_ms: 5000,
    watchdog_timer_ms: 30_000
  }
```

参数	类型	必需	默认	描述
`peer_settle_time_ms`	整数	否	`0`	新连接的对等体在DRA路由流量之前必须保持多长时间（毫秒）。`0`表示立即路由。正值则在该持续时间内保持对等体。如果对等体在窗口内断开，则取消保持。
`watchdog_timer_ms`	整数	否	`30000`	RFC 3539看门狗定时器**Tw**的覆盖，DWR/DWA存活间隔。降低它以加快死对等体检测。这会增加更多的看门狗流量。

将稳定时间设置为不□于您必须忍受的最坏情况的波动间隔。较长的值增加了波动保护。但这也会延迟健康对等体在合法重启后的返回。

监控

在其稳定窗口中的对等体在`diameter_peer_status`仪表中报告**down**（`0` = down/尚未可路由，`1` = 可路由）。它在变得可路由之前一直报告为down。因此，该指标反映了DRA可以路由的内容。

启动时会产生以下日志行：

日志消息	何时
`peer connected, settling before routable`	对等体已连接，稳定窗口开始（包括`settle_time_ms`）。
`peer disconnected during settle window; never became routable`	对等体在稳定之前掉线。DRA未路由任何流量。
`destination peer is settling; request not yet routable...`	针对仍在稳定中的对等体的请求到达。DRA回答`DIAMETER_UNABLE_TO_DELIVER (3002)`。
`no suitable peer to route request...`	DRA无法将请求路由到不在稳定中的目标（down、未知或配置错误）。

最后两行区分了预期的瞬态稳定与真正缺失的目标。

参考

- RFC 3539 - Diameter看门狗状态机和看门狗定时器（Tw）。
- RFC 6733 - Diameter基础协议（CER/CEA, DWR/DWA）。

漫游引导 (SoR)

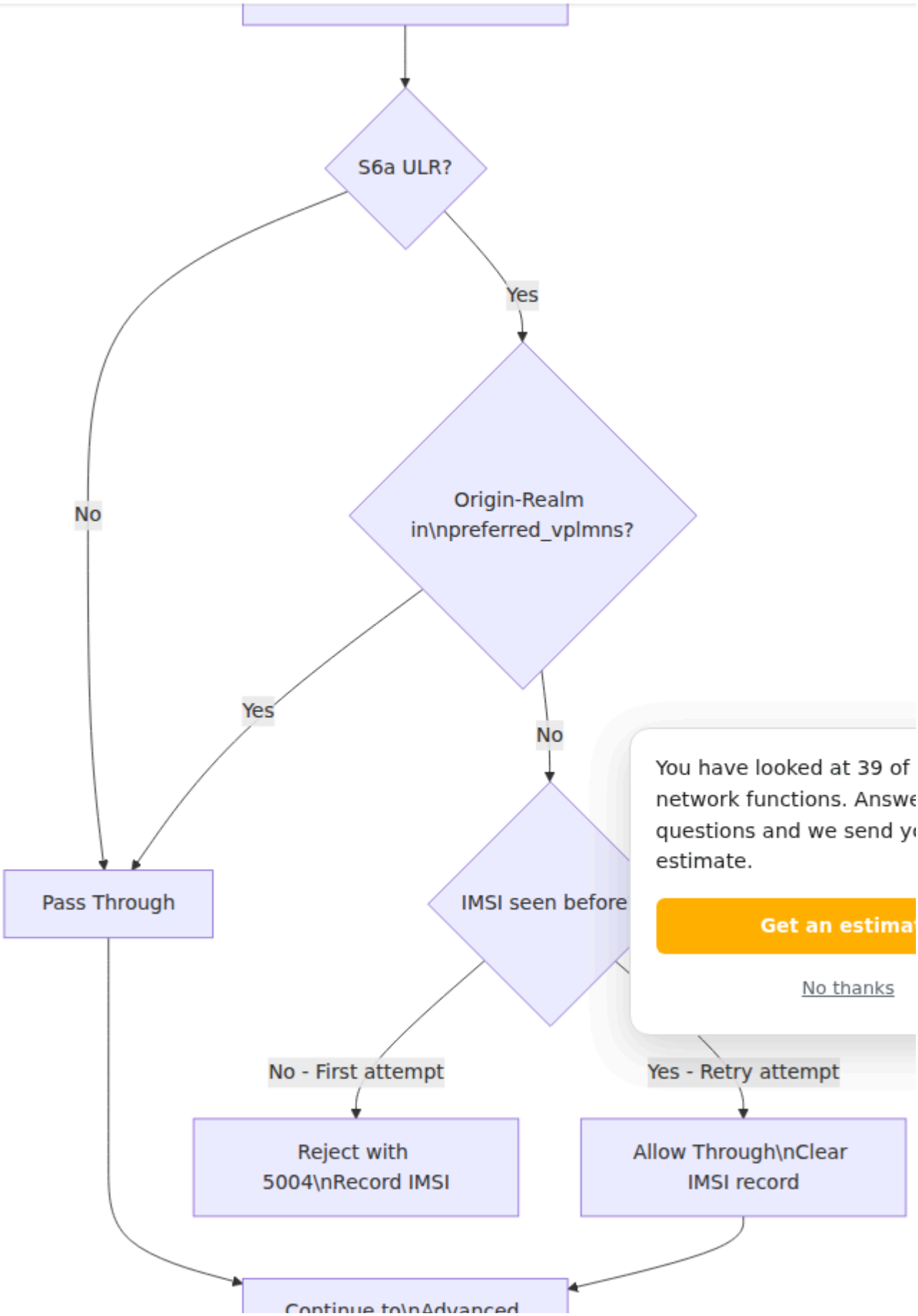
漫游引导 (SoR) 允许 HPLMN 影响漫游用户连接到哪个 VPLMN。当用户尝试通过非首选 VPLMN 注册时，DRA 拦截 S6a 更新位置请求 (ULR)。它用 **DIAMETER_ERROR_ROAMING_NOT_ALLOWED (5004)** 拒绝该 ULR。这导致 UE 断开连接并重新连接。如果有可用的首选 VPLMN，UE 将选择一个。

如果首选覆盖不可用，并且用户返回到相同的非首选 VPLMN，模块将在下次尝试时允许请求通过。

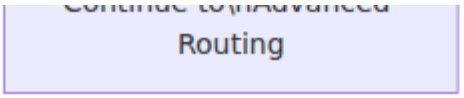
3GPP TS 29.272 第 5.2.1.1 节和 3GPP TS 23.122 定义了 SoR。

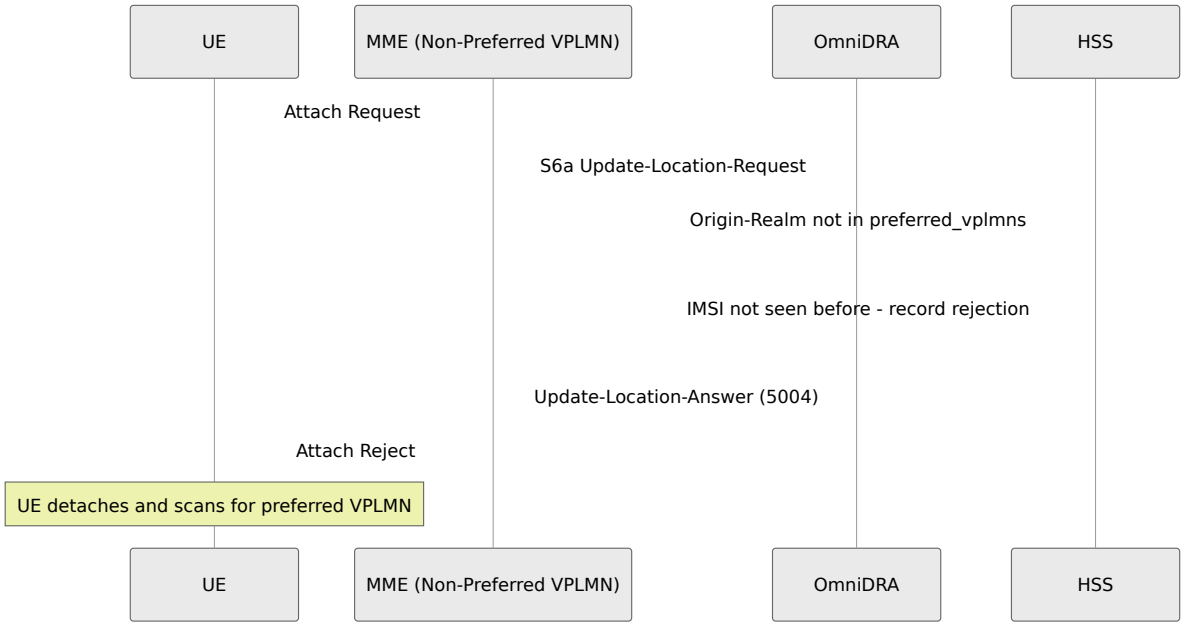
工作原理

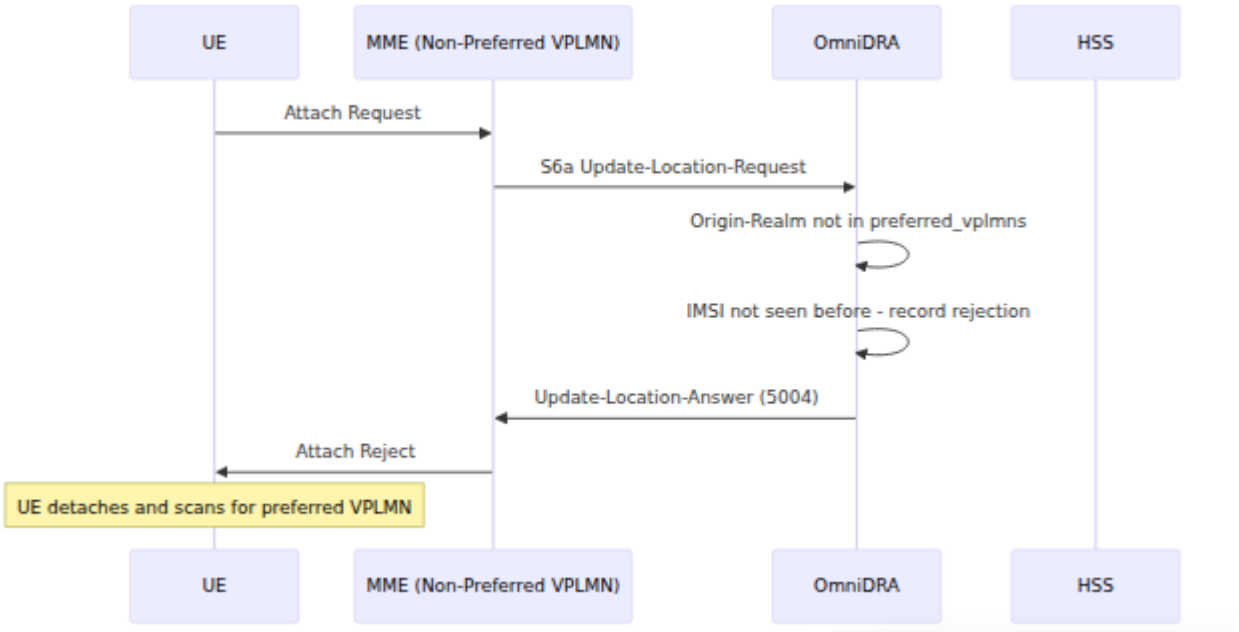
该模块在 DRA 请求处理管道中运行。它在□级路由之前运行。该模块仅评估 S6a 更新位置请求 (应用程序 ID `16777251`，命令代码 `316`)。所有其他 Diameter 消息均未修改地通过。



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







序列：第一次尝试（被拒绝）

序列：第二次尝试（允许）

管道位置

漫游引导在请求处理管道中在□级路由之前运行。如果 SoR 拒绝请求，该消息将永远不会到达□级路由。



用户跟踪

该模块通过 IMSI 跟踪每个被拒绝的用户。每条记录包括拒绝计数和时间戳。模块在配置的 TTL 到期后自动清理记录。

当被跟踪的 IMSI 达到 `max_rejections` 阈值并发送另一个 ULR 时，模块允许请求通过。然后它删除跟踪记录。

如果在 TTL 窗口内没有进一步的 ULR 到达，记录将过期。下一个清理周期将其删除。然后模块将来自同一 IMSI 的后续 ULR 视为第一次尝试并再次拒绝。

配置

在 `config/runtime.exs` 中的 `module_roaming_steering` 下配置该模块。

```
module_roaming_steering: %{
  # 启用或禁用该模块
  enabled: false,

  # 允许通过之前的拒绝次数
  max_rejections: 1,

  # 被跟踪的 IMSI 记录过期前的时间（秒）
  rejection_ttl_seconds: 300,

  # 拒绝答案中返回的 Diameter 结果代码
  rejection_result_code: 5004,

  # 被认为是首选 VPLMN 的 Origin-Realm 值列表
  # 来自这些领域的 ULR 始终被允许通过
  preferred_vplmns: [
    "epc.mnc001.mcc001.3gppnetwork.org",
    "epc.mnc002.mcc001.3gppnetwork.org"
  ]
}
```

参数

参数	类型	必需	默认	描述
`enabled`	布尔值	是	`false`	启用或禁用漫游引导模块。当为 `false` 时，所有请求均
`children`	列表	否	`[]`	DRA 主管在启用时启动的进程。将其设置为 `[DRA.Router.Modules.RoamingSt` IMSI 跟踪进程运行。
`max_rejections`	整数	否	`1`	在模块允许用户通过之前拒绝的次数。默认值为 `1`，模块拒[第二次。
`rejection_ttl_seconds`	整数	否	`300`	被跟踪的 IMSI 记录过期前的时间（秒）。清理然后将其删 未重试，则模块将下一个尝试视为第一次尝试。该值还控制清[
`rejection_result_code`	整数	否	`5004`	拒绝答案中的 Diameter 结果代码。`5004` 对应[DIAMETER_ERROR_ROAMING_NO 详见 3GPP TS 29.272 第 7.4.4 节。
`preferred_vplmns`	列表	否	`[]`	标识首选 VPLMN 的 Origin-Realm 字符串[具有匹配 Origin-Realm 的对等体的 ULR 通过

配置示例

拒绝一次，然后允许

这是默认行为。模块在非首选 VPLMN 上拒绝用户一次。如果用户返回，模块将 ULR 转发到 HSS。



```
module_roaming_steering: %{
  enabled: true,
  max_rejections: 1,
  rejection_ttl_seconds: 300,
  rejection_result_code: 5004,
  preferred_vplmns: [
    "epc.mnc001.mcc310.3gppnetwork.org",
    "epc.mnc002.mcc310.3gppnetwork.org"
  ]
}
```

工作原理：一个来自 `epc.mnc003.mcc310.3gppnetwork.org` 的 ULR 到达，该网络不在首选列表中。模块用结果代码 5004 拒绝它。UE 断开连接并尝试寻找首选网络。如果它在 300 秒内返回到相同的非首选 VPLMN，模块将正常转发 ULR。

用例：标准漫游引导。运营商与特定合作网络有首选漫游协议。

多次拒绝后允许

当运营商希望 UE 在回退到非首选 VPLMN 之前进行多次尝试时使用此选项。

```
module_roaming_steering: %{
  enabled: true,
  max_rejections: 3,
  rejection_ttl_seconds: 600,
  rejection_result_code: 5004,
  preferred_vplmns: [
    "epc.mnc001.mcc310.3gppnetwork.org"
  ]
}
```

工作原理：模块拒绝用户三次。然后在第四次尝试时允许用户通过。TTL 延长至 600 秒以进行额外的重试周期。

用例：首��覆盖存在的密集城市地区。UE 可能需要多次连接尝试才能获取它。

指标

拒绝

指标：`diameter.roaming_steering.reject.count` 类型：计数器 描述：每次 SoR 模块拒绝 ULR 时，DRA 增加此计数器。 标签：

- `origin_realm` - 非首选 VPLMN 的 Origin-Realm
- `result_code` - 拒绝中返回的 Diameter 结果代码
- `imsi` - 被拒绝用户的 IMSI

允许（首选 **VPLMN**）

指标：`diameter.roaming_steering.allow.count` 类型：计数器 描述：当模块允许 ULR 通过时，DRA 增加此计数器。 标签：

- `origin_realm` - 源 VPLMN 的 Origin-Realm
- `reason` - 模块允许请求的原因：`preferred_vplmn` 或 `max_rejections_reached`
- `imsi` - 用户的 IMSI。仅在原因是 `max_rejections_reached` 时存在

错误响应

指标：`diameter.roaming_steering.error.count` 类型：计数器 描述：当处理器返回 SoR 拒绝的错误答案消息时，DRA 增加此计数器。 标签：

- `result_code` - Diameter 结果代码
- `application_id` - Diameter 应用程序 ID（数字）
- `cmd_code` - Diameter 命令代码（数字）
- `application_name` - 可读的应用程序名称
- `cmd_name` - 可读的命令名称

故障排除

用户总是被拒绝（从未允许通过）

症状：模块反复拒绝非首选 VPLMN 上的用户。它们从未成功连接。

可能原因：

- `rejection_ttl_seconds` 太短。跟踪记录在用户重试之前过期。
- `max_rejections` 于 UE 在放弃之前执行的重试次数。

解决方案：

1. 增加 `rejection_ttl_seconds` 以适应 UE 的重试时机。
2. 确认 `max_rejections` 是 UE 在其重试周期内可以达到的值。

首选 **VPLMN** 上的用户被拒绝

症状：模块用 5004 拒绝来自首选合作网络的 ULR。

可能原因：

- 首选 VPLMN 的 Origin-Realm 不在 `preferred_vplmns` 中。
- Origin-Realm 字符串不完全匹配。匹配是区分大小写的。

解决方案：

1. 在 DRA 日志中检查对等体的 Origin-Realm 在连接时的情况。
2. 确认确切的 Origin-Realm 字符串在 `preferred_vplmns` 列表中。

模块未生效

症状：模块未拒绝地转发来自非首选 VPLMN 的 ULR。

可能原因：

- `enabled` 设置为 `false`。
- 模块进程未运行。检查主管。

解决方案：

1. 确认配置中的 `enabled: true`。
2. 确认 DRA 在配置更改后已重启。

参考

Diameter 代码

代码	名称	描述	参考
5004	DIAMETER_ERROR_ROAMING_NOT_ALLOWED	��户无法在此 VPLMN 漫游	3GPP TS 29.272 第 7.4.4 节

S6a 命令

命令代码	名称	描述	参考
316	更新位置请求/答案 (ULR/ULA)	MME 在附加期间发送此请求以更新用户的服务节点	3GPP TS 29.272 第 7.2.3 节

应用程序 ID

ID	接口	描述	参考
16777251	S6a/S6d	MME/SGSN 到 HSS 的身份验证和订阅管理	3GPP TS 29.272

Rx 会话绑定

Rx 会话绑定 将每个 Rx 会话的消息保留在拥有它的 PCRF 上。它的存在是因为 Rx 是唯一一个其会话终止请求不携带路由信息的接口。

问题

3GPP TS 29.214 §5.6.5 使得 `Destination-Host` 在 Rx `Session-Termination-Request` 中是 可选 的。它在消息中 没有 定义 `Subscription-Id` 和 没有 `Framed-IP-Address`。这一第二点是结构性的。这些 AVP 在实践中并非仅仅缺失。它们在消息定义中根本不存在，因此没有符合规范的 AF 可以提供它们。

因此，STR 是否携带 `Destination-Host` 取决于 AF。一些 AF 实现并不发送。一个代表性的捕获显示了这些 AVP：

`Session-Id`, `Origin-Host`, `Origin-Realm`, `Destination-Realm`, `Auth-Application-Id`, `Vendor-Specific-Application-Id`, `AF-Application-Identifier`, `Termination-Cause`。

`Session-Id` 是 AF 自己的透明句柄。根据 RFC 6733 §8.8，它在会话生命周期内是全局唯一且稳定的。它既不标识订阅者，也不标识服务器。

将其与打开会话的 `AA-Request` 进行比较。该请求携带 `Subscription-Id` 和 `Framed-IP-Address`，因此一个 □级路由 规则可以对其进行路由。还要将其与 **Gx** 进行比较。`CC-Request` ABNF (TS 29.212 §5.6.2) 包含 `[ Destination-Host ]`、`*[ Subscription-Id ]` 和 `[ Framed-IP-Address ]`。这正是 Gx 不需要该模块等效物的原因。

TS 29.213 §7.3.5 无论如何将负担放在 DRA 上。客户端“应能够将会话的每个消息发送到 DRA”。旁路仅仅是“可以配置的”。

在单个 PCRF 的情况下，这种不对称是不可见的。在 DRA 后面有多个 PCRF 时，它就不是了：

这种失败是无声的。 RFC 6733 §8.4.2 对于命名未知会话的 STR 没有规定结果代码。§8.1 状态机处理来自 □□ 状态的“接收到 STR”，并“发送 STA，清理”。因此，一个对它从未拥有的会话回答 `2001` 的 PCRF 是符合规范的，实际实现也这样做。不要指望结果代码能揭示这一点。(`5002 DIAMETER_UNKNOWN_SESSION_ID` 是为 **ASA** 规定的，§8.5.2，而不是 STA。)

同样的情况，较不明显地，适用于 □□□ *PCRF* □□。一个路由到 `peers: [a, b]` 的规则每条消息选择。一个 AAR 和它的 STR 因此可以落在同一池的不同成员上。

何时需要它

- 在 PCRF 之间迁移，或为每个订阅者试验并行运行两个 PCRF。
- 任何将 Rx 发送到多个对等体的□级路由规则。

在单个 PCRF 的情况下，您 不需要 它。默认情况下，它是禁用的。

它是如何工作的

消息	操作
`AA-Answer` (265, `Result-Code` 2001)	绑定 `Session-Id` → 回应的 PCRF
`AA-Request` (265)	路由到绑定的 PCRF
`Session-Termination-Request` (275)	路由到绑定的 PCRF
`Session-Termination-Answer` (275)	释放绑定，仅当它来自绑定的 PCRF 且其 `Result-Code` 是最终的 (2xxx 或 5xxx)

绑定遵循事实，而非意图

DRA 在 应答 上绑定，而不是在请求上。请求仅表示 DRA □□ 发送某些内容的位置。应答证明该节点上存在会话。这有三个后果：

- 用错误回答的 AAR，或从未回答的 AAR，不留下绑定。
- DRA 在 STA 到达时释放会话，而不是在路由 STR 时。 RFC 6733 §5.5.4 明确允许在故障转移后重传。过早释放会错误路由重试，这是该模块所防止的确切失败。
- 请求从不重写绑定。因此，瞬态对等体故障不能无声地将活动会话迁移到不同的 PCRF。

释放需要 两个 条件。STA 必须来自拥有绑定的对等体，并且其 `Result-Code` 必须是最终的。RFC 6733 §8.4.2 规定服务器在发送 STA 时释放其资源，因此 2xxx 和 5xxx 都是最终的。3xxx/4xxx 是瞬态的，AF 可能会重传，因此绑定在重试时存活。

所有权条件并非防御性整洁。没有它，该模块在其构建的确切场景中失败。所有者变得不可达。STR 落到另一个 PCRF。该 PCRF 对一个它从未拥有的会话回答 `2001`。然后 DRA 在承载仍然存在时销毁对所有者的绑定。这使您比没有模块更糟，因为拆除现在 □□□ 被固定。所有权条件是绑定侧 `insert_new` 的镜像。第二个 PCRF 既不能窃取绑定，也不能销毁绑定。

它不会覆盖的内容

该模块仅将路由重定向远离中继决策。其他所有内容都是故意的结果，模块对此保持不变：

结果	来源	被尊重
`:discard`	一个 `route: :drop` 规则	是
`{:answer_message, code}`	一个 `route: {:error, code}` 规则	是
`{:answer_message, 3002}`	□级路由在没有连接对等体的情况下失败关闭	是
显式 `Destination-Host`	AF 命名其服务器 (RFC 6733 §6.1.5)	是

代价是 DRA 不能在其规则指向的每个对等体都不可达时拆除会话。这是正确的权衡。排空节点的操作员，或故意关闭的操作员，不会要求绕过路由。

排序

这是唯一一个在 □级路由 之后运行的阶段。它必须在那里运行。STR 只能匹配一个捕获所有规则，而绑定必须能够击败它。

应答路径观察者在 □级转换 之前运行。□级转换可以在应答上重写或移除 `Session-Id` 和 `Result-Code`。绑定必须基于 AF 实际使用的 `Session-Id`。该模块故意对 拓扑隐藏 不敏感。它从其 Diameter 能力中读取应答对等体，而不是从 `Origin-Host` AVP 中读取。因此，AVP 重写不会误导它。

配置

```
config :dra,
  module_rx_session_binding: %{
    enabled: true,
    children: [DRA.Router.Modules.RxSessionBinding],
    ttl_seconds: 43_200
  }
```

键	目的
`enabled`	默认关闭。独立于 `module_subscriber_lookup`。这两个解决不同的问题，您可能只想要其中一个。
`ttl_seconds`	用于会话的泄漏保护，STA 从未到达。它 不是 会话计时器。

将 `ttl_seconds` 设置为 □于 AF 的 `Authorization-Lifetime`。这样 DRA 就不会在会话中间解绑一个长通话。一个广告 `Authorization-Lifetime: 36000` 的 P-CSCF 需要超过 10 小时。12 小时的默认值适合它。清理每 `ttl_seconds / 2` 运行（最小 60s）。DRA 在 1.5 × TTL 内回收泄漏的绑定。

可观察性

三个遥测事件，均携带 `session_id`：

事件	元数据
`[:diameter, :rx_session_binding, :bind]`	`serving_host`
`[:diameter, :rx_session_binding, :route]`	`serving_host`
`[:diameter, :rx_session_binding, :release]`	-

第四个事件 `[:diameter, :rx_session_binding, :reap]` 携带一个 `count`。DRA 仅在 TTL 清理发现过期绑定时发出它。非零的收割计数意味着 STA 正在丢失。对此计数发出警报。

DRA 暴露活动绑定的计数，并可以将其列出为 `{session_id, origin_host, inserted_at}`。

限制

- 绑定是内存中的并且是节点本地的。 DRA 重启或在 `:one_for_all` 监督下的兄弟崩溃会丢失它们。在飞行中的会话随后会回退到规则所说的内容。DRA 不在 DRAs 之间共享绑定。因此，AF 必须在会话的两个部分中都到达相同的 DRA。
- **DRA** 绑定相邻的对等体。在中继模式下，DRA 可以附加 Route-Record，但不能插入 `Destination-Host` (RFC 6733 §6.1.9)。因此，DRA 不能固定位于另一个代理后面的 PCRF。只有该代理可以。

- 迟到的 **AA-Answer** 可以复活已释放的会话。如果 DRA 在 STA 之后处理 AAR 更新的应答，`insert_new` 会重新创建该行。会话已经结束，因此这是一个泄漏的行，而不是错误路由。会话 ID 不会重用 (RFC 6733 §8.8)。`ttl_seconds` 值限制该行，并且它会出现在 `:reap` 计数器中。
- **DRA** 不能在□级路由因 **Rx** 失败关闭时释放会话。 `{:answer_message, 3002}` 结果与操作员故意的 `route: {:error, 3002}` 无法区分。将每个持有活动会话的 PCRF 保持在规则的对等体列表中。这样就不会出现这种情况。
- 配置中需要 **`children`**。`DRA.Supervisor` 仅在配置中携带 `children` 列表时启动模块的进程。启用 `enabled: true` 而没有 `children` 时，GenServer 永远不会启动。每个入口点降级为“没有绑定”，并且没有任何警告您。
- **DRA** 仅绑定 **AF → PCRF** 方向。 PCRF 发起的 `RAR` 和 `ASR` 携带强制性的 `Destination-Host` (RFC 6733 §8.3.1, §8.5.1) 并在此基础上路由。它们不需要绑定。
- **DRA** 仅绑定 **Rel-7+ Rx** 应用程序 **ID (16777236)**。它不绑定在遗留 Rel-6 Rx/Gq ID (16777229) 上的会话。
- **DRA** 发出遥测但不导出它。上述四个事件没有 `Telemetry.Metrics` 定义，因此今天没有任何内容到达 Prometheus。这与先前存在的 `subscriber_lookup` 事件具有相同的差距。在那被连接之前，请使用远程 shell 中的活动绑定计数。

SCTP 多宿主

SCTP 多宿主 为 Diameter 传输提供网络冗余。它允许每个端点在单个关联中绑定多个 IP 地址。如果主网络路径失败，SCTP 会自动切换到备用路径。这不会中断 Diameter 会话。

为什么多宿主很重要

RFC 6733 定义了基于 SCTP 的 Diameter。这里描述的多宿主行为是 **SCTP** 本身的原生能力。RFC 4960 对其进行了规范。TCP 将关联绑定到单个源/目标 IP 对。SCTP 允许一个关联跨越每一侧的多个 IP 地址。这为 OmniDRA 传输层提供了对单个网络路径、NIC、交换机或路由段丢失的弹性。Diameter 连接不会被拆除或重新建立。

主要特征：

- SCTP 心跳 持续监控关联中的所有网络路径。
- 如果主路径变得不可达，自动故障切换 会在内核级别发生。
- 路径切换不会中断 Diameter 会话。关联及其 Diameter 状态保持不变。
- 操作系统内核会自动选择路径。

工作原理

OmniDRA 和对等方建立一个多宿主关联。每个端点会广播多个 IP 地址。内核为数据传输选择一个主路径。它将其余路径视为备用路由。如果主路径失败，内核会自动提升一个备用路由。

□□□

□□□□□

□□□□

参数	类型	必需	默认	描述
`host`	字符串	是	-	对等方的 Diameter 身份（FQDN）。必须与对等方的配置身份完全匹配。
`realm`	字符串	是	-	对等方的 Diameter 领域。DRA 用于路由决策。
`ip`	字符串	是	-	对等方的主 IP 地址。始终必需，包括与单宿主配置的向后兼容性。
`additional_ips`	字符串列表	否	-	多宿主 SCTP 关联的所有 IP 地址。此字段是可选的。如果省略，OmniDRA 仅使用 `ip`。对于 SCTP，请在此列表中包含主 IP。对于 TCP，OmniDRA 会忽略此字段，因为 TCP 不支持多宿主。
`port`	整数	是	-	对等方的 Diameter 端口。标准端口为 3868，符合 RFC 6733。
`transport`	原子	是	-	对等[illegible]接的传输协议：`:diameter_sctp` 或 `:diameter_tcp`。多宿主需要 `:diameter_sctp`。
`initiate_connection`	布尔值	是	-	当为 `true` 时，OmniDRA 发起与对等方的连接。当为 `false` 时，OmniDRA 等待对等方连接。

向后兼容性：`ip` 字段单独描述单宿主对等方。如果省略 `additional_ips`，则对等方保持单宿主。因此，现有对等方配置无需更改。

完整示例

以下配置将 OmniDRA 绑定到两个本地地址。它连接到一个多宿主 HSS 和一个单宿主 MME。

```
config :dra,
  diameter: %{
    service_name: :omnitouch_dra,
    listen_ip: ["192.168.1.10", "10.0.0.10"],  # 多宿主 DRA
    listen_port: 3868,
    host: "dra01",
    realm: "example.com",
    product_name: "OmniDRA",
    vendor_id: 10415,
    request_timeout: 5000,
    peer_selection_algorithm: :random,
    allow_undefined_peers_to_connect: false,
    peers: [
      # 多宿主 HSS 连接
      %{
        host: "hss01.example.com",
        realm: "example.com",
        ip: "192.168.1.20",
        additional_ips: ["192.168.1.20", "10.0.0.20"],
        port: 3868,
        transport: :diameter_sctp,
        initiate_connection: true
      },
      # 单宿主 MME（向后兼容）
      %{
        host: "mme01.example.com",
        realm: "example.com",
        ip: "192.168.1.30",
        port: 3868,
        transport: :diameter_sctp,
        initiate_connection: false
      }
    ]
  }
```

上面的表格记录了 `listen_ip`、`listen_port` 和所有对等参数。其余顶级参数控制 OmniDRA 的 Diameter 身份和对等选择行为。配置参考 记录了它们：

□□	□□	□□	□□	□□
`service_name`	□□	□	-	OmniDRA Diameter □□□□□□ □□
`host`	□□□	□	-	OmniDRA □□□ Diameter □□ □FQDN□□
`realm`	□□□	□	-	OmniDRA □□□ Diameter □□□
`product_name`	□□□	□	-	□□□□□□□□□□□□□□□□ □□□□
`vendor_id`	□□	□	-	□□□□□□□□□□ ID□ `10415` □ 3GPP □□ □□□□□
`request_timeout`	□□	□	`5000`	□□□□□□□□□□□□□□□□ □□□□□□□□□□
`peer_selection_algorithm`	□□	□	-	□□□□□□□□□□□□□□□ □□
`allow_undefined_peers_to_connect`	□□□	□	`false`	□□ `false` □□□□□ `peers` □□□□□□□□ □□□□ `true` ���□DRA □□□□ □□□□□□□
`peers`	□□	□	-	□□□□□□□□□□□□□ □ □□□□□□

SCTP 套接字缓冲区

OmniDRA 在每个 SCTP 传输上设置显式的套接字接收和发送缓冲区。它使用顶级的 `diameter_socket_buffer_size` 键（默认 4 MB）。与 `listen_ip` 和 `peers` 不同，此键位于 `:dra` 应用配置的顶级。它不在 `diameter:` 映射内。

如果没有显式缓冲区，SCTP 套接字会继承一个小的默认 `SO_RCVBUF`。这将 SCTP 接收窗口固定在大约一个 MTU，并导致在负载下出现零窗口停滞。Linux 内核将 `SO_RCVBUF` 限制在 `net.core.rmem_max`。它将发送缓冲区限制在 `net.core.wmem_max`。要使用超过该上限的缓冲区大小，请在主机上提□这些 sysctl。

```
config :dra,
  # 顶级键 - 不在 diameter: 映射内。应用于每个 SCTP 传输。
  diameter_socket_buffer_size: 4 * 1024 * 1024,  # 4 MB（接收缓冲区/发送缓冲区）
  diameter: %{
    # ... 服务和对等方配置 ...
  }
```

参数	类型	必需	默认	描述
`diameter_socket_buffer_size`	整数	否	`4194304` (4 MB)	每个 SCTP 传输的套接字接收和发送缓[`recbuf` 和 `sndbuf`。它仅影响对等方，并且对 TCP 被忽略。超过 `net.core.rmem_max`/`ne` 的值需要您提□这些内核 sysctl。

需求

SCTP 多宿主需要在两个端点之间满足以下条件：

- 主机上必须加载 **SCTP** 内核模块（Linux 上的 `lksctp-tools` 包）。
- 所有配置的 IP 地址必须能够路由到对等方并从对等方路由。
- 防火墙规则必须允许在每个配置的 IP 地址上进行 SCTP 流量。
- 配置两个端点以实现多宿主，以实现完全冗余。连接到单宿主对等方的多宿主 OmniDRA 仅在 OmniDRA 侧获得路径冗余。

限制

- **TCP** 不支持多宿主。使用 `:diameter_tcp` 传输时，OmniDRA 仅使用主 IP。`listen_ip` 使用列表中的第一个 IP，OmniDRA 会忽略对等方的 `additional_ips`。
- 路径故障切换时机取决于内核 SCTP 参数，例如心跳间隔和重传阈值。您可以在操作系统级别调整这些参数，而不是在 OmniDRA 内部。

参考文献

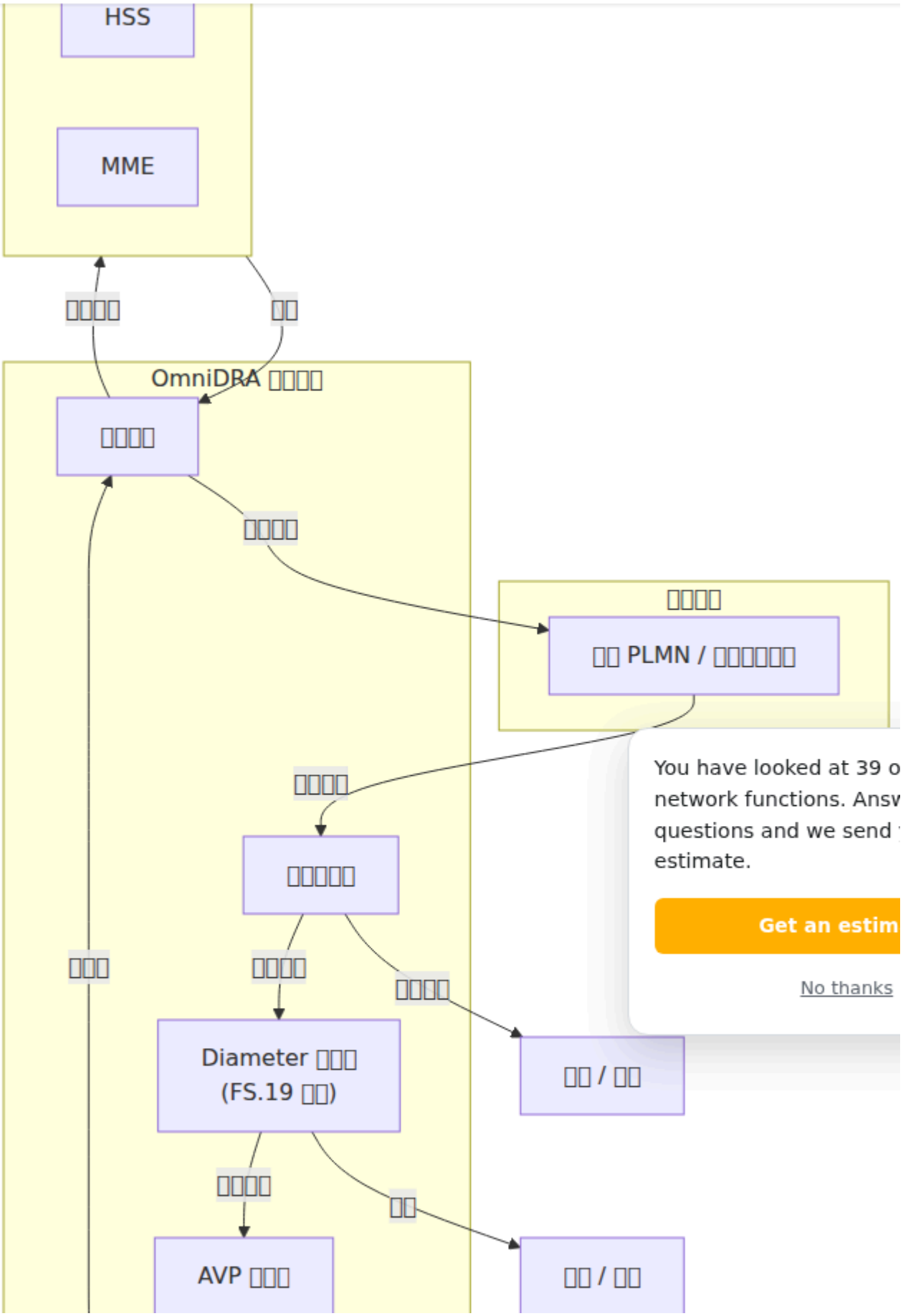
- RFC 4960 - 流控制传输协议 (SCTP)
- RFC 6733 - Diameter 基础协议
- 配置参考

Diameter 安全

OmniDRA 提供了一整套全面的 Diameter 安全模块。它们与 **GSMA FS.19**（Diameter 互联安全，v10.0）和 **GSMA FS.21**（互联信令安全建议，v12.0）保持一致。这些模块保护网络免受 Diameter 互联接口上的信令级攻击。

您可以独立配置每个模块。您可以启用或禁用每个模块，而不会影响其他模块。每个模块都会发出自己的遥测事件以进行监控和警报。

□□□□



You have looked at 39 o
network functions. Ansv
questions and we send
estimate.
Get an estim
No thanks

处理顺序

入站 Diameter 请求按以下顺序通过安全模块。如果某个阶段拒绝消息，DRA 将不会将其转发到后续阶段。

顺序	模块	方向	目的
1	速率限制器	入站	体积洪水保护
2	Diameter 防火墙	入站	FS.19 协议和内容过滤
3	AVP 清理器	入站	AVP 验证和剥离
4	路由引擎	-	标准 Diameter 路由
5	拓扑隐藏	出站	网络拓扑隐蔽

操作

所有安全模块在检测到违规时支持可配置的操作。您可以按模块和按类别为 Diameter 防火墙配置操作。

操作	行为	**Diameter** 结果
`{:error, 3002}`	以 Diameter 错误答复响应	DIAMETER_UNABLE_TO_DELIVER
`{:error, 3004}`	以 Diameter 错误答复响应	DIAMETER_TOO_BUSY
`{:error, 3007}`	以 Diameter 错误答复响应	DIAMETER_APPLICATION_UNSUPPORTED
`:drop`	丢弃消息	不发送响应
`:log_only`	记录违规但允许消息通过	消息继续

Diameter 防火墙

Diameter 防火墙实现了 **GSMA FS.19** 第 **3.3** 节中定义的四个过滤层。它还实现了 **FS.21** 第 **7** 节中的协议无关数据包分类。防火墙按顺序评估每一层消息。在任何一层发生违规时，进一步处理将停止。

低层格式过滤

GSMA 参考: FS.19 第 3.3.4 节, FS.21 第 7.3.1 节

低层过滤检测协议级违规和基本的反欺骗尝试。它不需要理解上层应用语义。此层在消息到达更深层检查之前捕获格式错误的消息。

执行的检查：

检查	**FS.19** 参考	描述
AVP 重复	第 3.3.4 节，4.8.1	检测必须最多出现一次的 AVP 的重复实例（例如 Origin-Host, Origin-Realm）。防止 AVP 重复规避攻击。
Session-Id 排序	第 3.3.4 节	确认 Session-Id（AVP 263）是消息中的第一个 AVP，符合 RFC 6733 第 8.8 节。
答复中的 Destination-Host	第 3.3.4 节	拒绝包含 Destination-Host AVP 的答复消息。这是一个协议违规，可能表明过滤规避。
强制 AVP 验证	第 3.3.5 节	确认每个命令代码所需的 AVP 存在。例如，ULR 必须包含 User-Name 和 Visited-PLMN-Id，符合 3GPP TS 29.272 第 7.2.3 节。

```
low_layer: %{
  enabled: true,
  action: {:error, 3002},
  # AVP 代码，不能出现多于一次（FS.19 第 3.3.4 节）
  # 264 = Origin-Host, 296 = Origin-Realm, 283 = Destination-
Realm, 293 = Destination-Host
  single_instance_avps: [264, 296, 283, 293],
  # Session-Id 必须是第一个 AVP（RFC 6733 第 8.8 节）
  enforce_session_id_first: true,
  # 答复消息不能包含 Destination-Host（FS.19 第 3.3.4 节）
  reject_destination_host_in_answers: true,
  # 每个命令代码的强制 AVP（3GPP TS 29.272）
  # 1 = User-Name (IMSI), 1407 = Visited-PLMN-Id, 264 = Origin-
Host, 296 = Origin-Realm
  mandatory_avps: %{
    316 => [1, 1407, 264, 296],
    318 => [1, 1407, 264, 296]
  }
}
```

类别 1 - 接口未授权数据包过滤

GSMA 参考：FS.19 第 3.3.5 节，FS.21 第 7.2.1 节

类别 1 过滤仅接受每个互联接口上授权的 Diameter 应用程序 ID 和命令代码。这防止外部访问仅限内部的接口。例如，它阻止通过 S6a 接口的 Sh 命令。它还强制漫游合作伙伴仅发送其漫游协议覆盖的消息类型。

白名单方法遵循 FS.19 的建议：阻止所有 **Diameter** 消息，除非该接口需要的消息。

您可以全局配置白名单，这适用于所有对等方，或按对等方配置。每个对等方的白名单允许不同的漫游合作伙伴拥有不同的允许消息集。

```
category_1: %{
  enabled: true,
  action: {:error, 3007},
  whitelists: %{
    # 每个对等方的白名单 - 限制此合作伙伴仅限 ULR 和 AIR（FS.19 第 3.3.5 节）
    "restricted-partner.roaming.com" => %{
      16_777_251 => [316, 318]
    },
    # 默认白名单 - 所有其他对等方允许的标准 S6a 命令
    all: %{
      # S6a/S6d (FS.19 第 3.3.5 节，表 2)
      16_777_251 => [316, 317, 318, 319, 320, 321, 323],
      # S13 - ME 身份检查 (FS.19 第 3.3.5 节)
      16_777_252 => [324],
      # S6c - 通过 HSS 的 SMS (FS.19 第 3.3.5.1 节)
      16_777_312 => [8388647, 8388648],
      # SGd - 通过 MME 的 SMS (FS.19 第 3.3.5.2 节)
      16_777_313 => [8388645, 8388646]
    }
  }
}
```

常见 **S6a/S6d** 白名单：

命令代码	名称	方向	参考
316	更新位置请求/答复	MME → HSS	3GPP TS 29.272 §7.2.3
317	取消位置请求/答复	HSS → MME	3GPP TS 29.272 §7.2.7
318	身份验证信息请求/答复	MME → HSS	3GPP TS 29.272 §7.2.5
319	插入用户数据请求/答复	HSS → MME	3GPP TS 29.272 §7.2.9
320	删除用户数据请求/答复	HSS → MME	3GPP TS 29.272 §7.2.11
321	清除 UE 请求/答复	MME → HSS	3GPP TS 29.272 §7.2.13
323	通知请求/答复	MME → HSS	3GPP TS 29.272 §7.2.15

漫游互联的常见应用程序 **ID**（FS.19 第 3.3.5 节）：

应用程序 **ID**	接口	参考
16777251	S6a/S6d	3GPP TS 29.272
16777252	S13	3GPP TS 29.272
16777312	S6c	3GPP TS 29.338
16777313	SGd	3GPP TS 29.338
16777255	SLg	3GPP TS 29.172
16777267	S9	3GPP TS 29.215

类别 **2 -** 家庭网络数据包过滤

GSMA 参考：FS.19 第 3.3.6 节，FS.21 第 7.2.2 节

□□ 2 □□□□

检查	**FS.19** 参考	描述
先前位置检查	第 3.3.7.1 节	比较当前 ULR 的 Origin-Realm 与该 IMSI 的最后一次看到的 Origin-Realm
速度/时间检查	第 3.3.7.2 节	标记在可配置的最小时间窗口内发生的领域变化

```
category_3: %{
  enabled: true,
  # 开始时使用 log_only 观察模式，然后再强制执行（FS.19 第 3.3.7 节）
  action: :log_only,
  # 最大合理速度（公里/小时）（FS.19 第 3.3.7.2 节）
  max_velocity_kmh: 1200,
  # 同一 IMSI 从不同领域更新之间的最小秒数
  min_time_between_updates_seconds: 2
}
```

配置

在顶层启用 Diameter 防火墙。独立配置每个过滤层。将每个类别部分中的配置片段组合在单个 `module_diameter_firewall` 键下：

```
config :dra,
  module_diameter_firewall: %{
    enabled: true,
    home_imsi_prefixes: ["31338", "31339"],
    home_msisdn_prefixes: ["+1313"],
    low_layer: %{ ... },      # 请参见上面的低层格式过滤
    category_1: %{ ... },     # 请参见上面的类别 1
    category_2: %{ ... },     # 请参见上面的类别 2
    category_3: %{ ... }      # 请参见上面的类别 3
  }
```

顶层参数

参数	类型	必需	默认	描述
`enabled`	布尔	是	`false`	启用或禁用整个 Diameter 防火墙模块。当为 `fals`
`children`	列表	否	`[]`	DRA 主管在启用模块时启动的进程。对于防火墙，将其设置[`[DRA.Router.Modules.DiameterFi` 类别 3 状态感知（速度）跟踪需要它。
`home_imsi_prefixes`	列表	否	`[]`	识别家庭用户的 IMSI 前缀字符串列表。类别 2 过滤使用[
`home_msisdn_prefixes`	列表	否	`[]`	识别家庭用户的 MSISDN 前缀字符串列表。类别 2 过[

□□□□

□□	□□	□□	□□	□□
`enabled`	□□	□	`true`	□□□□□□□□□□□□□□
`action`	□□	□	`{:error, 3002}`	□□□□□□□□□□□□□□□□□□□□
`single_instance_avps`	□□	□	`[264, 296, 283, 293]`	□□□□□□□□□□□□□□ AVP □□□□□□□□ Origin-Host (264)□ Origin-Realm (296)□ Destination-Realm (283)□ Destination-Host (293)□
`enforce_session_id_first`	□□	□	`true`	□□ Session-Id □□□□□□ AVP □□□□
`reject_destination_host_in_answers`	□□	□	`true`	□□□□ Destination-Host AVP □□□□□□
`mandatory_avps`	□□	□	`%{}`	□□□□□□□□ AVP □□□□□□□□□□□□□□ `%{316 => [1, 1407, 264, 296]}`

参数	类型	必需	默认	描述
				要求 ULR 消息中包含 User-Name、Visited-PLMN-Id、Origin-Host 和 Origin-Realm。

类别 1 参数

参数	类型	必需	默认	描述
`enabled`	布尔	否	`true`	启用或禁用类别 1 过滤。
`action`	操作	否	`{:error, 3007}`	当防火墙检测到类别 1 违规时采取的操作。默认响应 DIAMETER_APPLICATION_UNSUPPORTED。
`whitelists`	映射	是	-	对等主机名（或 `:all`）到允许的应用程序 ID / 命令代码组合的映射。`:all` 键提供默认白名单。特定对等方条目优先。

类别 2 参数

参数	类型	必需	默认	描述
`enabled`	布尔	否	`true`	启用或禁用类别 2 过滤。
`action`	操作	否	`{:error, 3002}`	当消息从互联针对家庭用户时采取的操作。
`protected_command_codes`	列表	否	`[]`	触发家庭用户身份检查的命令代码。通常是携带用户身份的 S6a 命令代码。

类别 3 参数

参数	类型	必需	默认	描述
`enabled`	布尔	否	`false`	启用或禁用类 3 过滤。它需要 LocationTracker 进程。主管在启用模块时自动启动此进程。
`action`	操作	否	`:log_only`	当位置合理性检查失败时采取的操作。开始时使用 `:log_only` 以调整阈值，然后再强制执行。
`max_velocity_kmh`	整数	否	`1200`	最大合理速度（公里/小时）。保留用于未来的地理距离计算。
`min_time_between_updates_seconds`	整数	否	`2`	同一 IMSI 从不同领域更新之间的最小时间（秒）。防火墙标记比此速度更快的领域变化。

遥测事件

事件	描述
`[:diameter, :firewall, :low_layer, :block]`	防火墙检测到并阻止了低层格式违规
`[:diameter, :firewall, :category_1, :block]`	防火墙检测到并阻止了类别 1 违规
`[:diameter, :firewall, :category_2, :block]`	防火墙检测到并阻止了类别 2 违规
`[:diameter, :firewall, :category_3, :block]`	防火墙检测到并阻止了类别 3 违规
`[:diameter, :firewall, :pass, :count]`	消息通过了所有防火墙检查

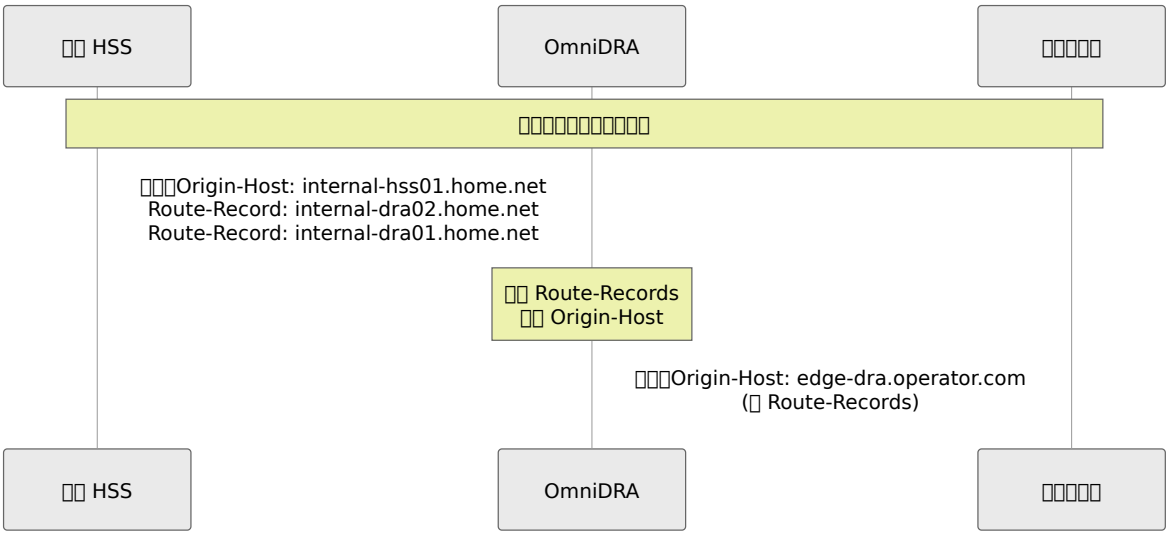
所有事件都包括以下元数据：`origin_host`、`application_id`、`command_code`、`application_name`、`command_name` 和 `reason`。

拓扑隐藏

GSMA 参考： FS.19 第 2.4 节，第 3.4 节；FS.21 第 3.6 节

拓扑隐藏模块防止将内部网络拓扑暴露给外部对等方。当 Diameter 消息穿越多个内部节点时，它们会累积 Route-Record AVP。它们还携带 Origin-Host/Origin-Realm 值，这些值揭示了内部主机名、网络结构和节点计数。攻击者可以利用这些信息来映射内部网络以进行针对性攻击。

拓扑隐藏在 出站路径 上操作。它在 DRA 做出路由决策后运行，并在将消息转发到目标对等方之前运行。



内部 HSS
OmniDRA
外部对等方
答复消息流（拓扑隐藏）
答复, Origin-Host: internal-hss01.home.net
Route-Record: internal-dra02.home.net
Route-Record: internal-dra01.home.net
剥离 Route-Records
重写 Origin-Host
答复, Origin-Host: edge-dra.operator.com
(无 Route-Records)
内部 HSS
OmniDRA
外部对等方

特性

特性	**FS.19 参考**	描述
Route-Record 剥离	第 2.4 节	移除所有揭示内部路由路径和节点主机名的 Route-Record AVP（282）
Origin-Host 重写	第 3.4 节	在答复消息中将 Origin-Host AVP 替换为 DRA 自身的身份（或自定义值）
Origin-Realm 重写	第 3.4 节	可选地替换 Origin-Realm AVP 以隐藏内部领域结构
每个对等方控制	-	有选择性地应用拓扑隐藏，仅对外部对等方或所有对等方

```
module_topology_hiding: %{
  enabled: true,
  # 剥离揭示内部路径的 Route-Record AVP（FS.19 第 2.4 节）
  strip_route_records: true,
  # 在答复中重写 Origin-Host 以隐藏内部节点名称（FS.19 第 3.4 节）
  rewrite_origin_host: %{enabled: true, replacement: :self},
  # 可选地隐藏内部领域结构
  rewrite_origin_realm: %{enabled: false, replacement: :self},
  # 应用于所有外部对等方，或列出特定主机名
  external_peers: :all
}
```

参数

参数	类型	必需	默认	描述
`enabled`	布尔	是	`false`	启用或禁用拓扑隐藏模块。
`strip_route_records`	布尔	否	`true`	在 DRA 将消息转发到外部对等方之前，从消息中删除所有 Route-Record AVP（代码 282）。
`rewrite_origin_host`	映射	否	见下文	控制答复消息中的 Origin-Host 重写。
`rewrite_origin_realm`	映射	否	见下文	控制答复消息中的 Origin-Realm 重写。
`external_peers`	`:all` 或 列表	否	`:all`	模块视为外部的对等��。模块仅对这些对等方的目标消息应用拓扑隐藏。对于互联部署，使用 `:all`。使用主机名列表进行选择性应用。

重写参数（适用于 `rewrite_origin_host` 和 `rewrite_origin_realm`）：

参数	类型	必需	默认	描述
`enabled`	布尔	否	varies	启用此 AVP 的重写。默认情况下，Origin-Host 为 `true`，Origin-Realm 为 `false`。
`replacement`	`:self` 或字符串	否	`:self`	替换值。`:self` 使用 DRA 的身份（来自 `diameter` 配置的 `host.realm`）。模块按原样使用字符串值。

遥测事件

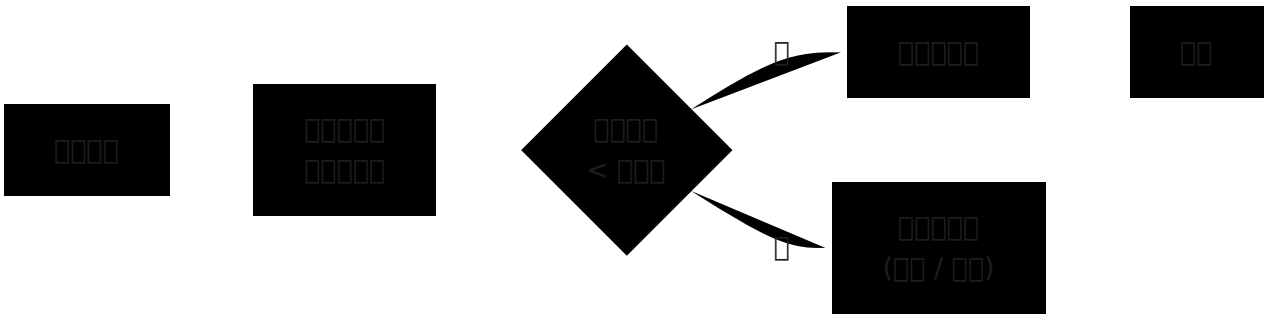
事件	描述
`[:diameter, :topology_hiding, :route_record, :stripped]`	模块移除了 Route-Record AVP。测量包括剥离的 AVP 数量。
`[:diameter, :topology_hiding, :origin_host, :rewritten]`	模块重写了 Origin-Host AVP
`[:diameter, :topology_hiding, :origin_realm, :rewritten]`	模块重写了 Origin-Realm AVP

速率限制器

GSMA 参考：FS.19 第 3.4 节（可用性 / DoS 保护）

速率限制器对每个对等方实施消息速率限制。它保护免受体积攻击和消息洪水。它作为管道中的第一个安全检查，在任何消息解析或内容检查之前运行。这尽早减轻了过载。

速率限制器通过滑动窗口计数器跟踪每个对等方的速率限制。每个对等方都有一个独立的计数器，每秒重置一次。



```
module_rate_limiter: %{
  enabled: true,
  # 所有对等方的默认限制（FS.19 第 3.4 节）
  default_max_requests_per_second: 1000,
  default_action: {:error, 3004},
  # 每个对等方的覆盖
  peer_limits: %{
    "high-volume-partner.roaming.com" => %
{max_requests_per_second: 5000, action: {:error, 3004}},
    "restricted-peer.roaming.com" => %{max_requests_per_second:
100, action: :drop}
  }
}
```

参数

参数	类型	必需	默认	[
`enabled`	布尔	是	`false`	启用或禁用速率限制器模块。
`children`	列表	否	`[]`	DRA 主管在启用时启动的进程[`[DRA.Router.Modu` 以便每个对等方的速率计数器进程[
`default_max_requests_per_second`	整数	否	`1000`	来自任何单个对等方的默认最大请[
`default_action`	操作	否	`{:error, 3004}`	当对等方超过其速率限制时的默认[响应 DIAMETER_TOO
`peer_limits`	映射	否	`%{}`	对等主机名到覆盖配置的映射。未[

每个对等方覆盖参数：

参数	类型	必需	默认	描述
`max_requests_per_second`	整数	否	继承默认值	此特定对等方的最大请求数。
`action`	操作	否	继承默认值	当此对等方超过其速率限制时的操作。

遥测事件

事件	描述
`[:diameter, :rate_limiter, :throttled]`	模块对消息进行了速率限制。测量包括 `current_rate` 和 `limit`。
`[:diameter, :rate_limiter, :allowed]`	消息在速率限制内。

AVP 清理器

GSMA 参考: FS.19 第 3.3.4 节, 4.8.1, 4.8.2; FS.21 第 3.6 节

AVP 清理器在互联边界验证和清理 Diameter AVP。它解决了 FS.21 第 3.6 节的建议。这些建议处理在 FS.19 过滤类别级别以下操作的协议级操控攻击。

特性

特性	**GSMA** 参考	描述
未知供应商 AVP 剥离	FS.21 第 3.6 节	移除来自非白名单供应商的 AVP。防止注入可能触发后端节点意外行为的专有 AVP。
分组 AVP 嵌套深度	FS.21 第 3.6 节	强制分组 AVP 的最大嵌套深度。防止使用深度嵌套 AVP 结构的堆栈溢出攻击。

```
module_avp_sanitizer: %{
  enabled: true,
  action: {:error, 3002},
  # 仅允许标准供应商 AVP 在互联上（FS.21 第 3.6 节）
  # 0 = IETF, 10415 = 3GPP, 13019 = ETSI, 5535 = 3GPP2
  allowed_vendor_ids: [0, 10415, 13019, 5535],
  strip_unknown_vendor_avps: true,
  # 防止通过深度嵌套的分组 AVP 导致堆栈溢出（FS.21 第 3.6 节）
  max_avp_nesting_depth: 10
}
```

参数

参数	类型	必需	默认	描述
`enabled`	布尔	是	`false`	启用或禁用 AVP 清理器模块。
`action`	操作	否	`{:error, 3002}`	当模块检测到嵌套深度违规时采取的操作。供应商剥离会移除违规的 AVP，而不会阻止消息。
`allowed_vendor_ids`	列表	否	`[0, 10415, 13019, 5535]`	允许在互联上的供应商 ID 列表。模块会剥离来自其他供应商的 AVP。
`strip_unknown_vendor_avps`	布尔	否	`true`	启用剥离不在 `allowed_vendor_ids` 列表中的供应商的 AVP。
`max_avp_nesting_depth`	整数	否	`10`	允许的分组 AVP 嵌套的最大深度。配置的 `action` 适用于超过此深度的消息。

常见供应商 **ID**:

供应商 ID	组织	备注
0	IETF	在 RFC 中定义的标准 Diameter AVP
10415	3GPP	所有 3GPP 定义的 AVP（S6a、Gx、Rx 等）
13019	ETSI	ETSI 定义的 AVP
5535	3GPP2	3GPP2 定义的 AVP（CDMA 互通）

遥测事件

事件	描述
`[:diameter, :avp_sanitizer, :unknown_vendor, :stripped]`	模块剥离了一个或多个来自未知供应商的 AVP
`[:diameter, :avp_sanitizer, :nesting_depth, :violation]`	消息超过了最大 AVP 嵌套深度
`[:diameter, :avp_sanitizer, :pass, :count]`	消息通过了所有清理检查

部署建议

推荐启用顺序

当您首次部署安全模块时，逐个启用它们。这可以避免对实时流量的干扰：

1. 速率限制器 - 从宽松的限制开始，监控流量模式。在了解基线速率后收紧限制。
2. **AVP** 清理器 - 误报风险低。启用剥离和嵌套检查。
3. **Diameter** 防火墙（类别 **1**） - 根据漫游协议定义白名单。从已知良好的应用程序 ID / 命令代码组合开始。
4. **Diameter** 防火墙（低层） - 启用协议一致性检查。
5. **Diameter** 防火墙（类别 **2**） - 配置家庭 IMSI/MSISDN 前缀。

6. 拓扑隐藏 - 首先启用 Route-Record 剥离，然后重写 Origin-Host。
7. **Diameter** 防火墙（类别 **3**） - 启用它并使用 `:log_only` 操作。在强制执行之前观察位置模式。

深度防御

这些模块实现了 FS.19 第 3.4 节中的 深度防御 原则。每一层针对不同类型的攻击：

攻击类别	主要防御	次要防御
体积型 DoS	速率限制器	Diameter 防火墙（所有类别）
接口滥用	类别 1	AVP 清理器
家庭用户针对	类别 2	类别 1（接口限制）
位置欺骗	类别 3	类别 2（家庭用户检查）
拓扑发现	拓扑隐藏	-
AVP 注入	AVP 清理器	低层过滤
协议规避（AVP 重复）	低层过滤	AVP 清理器

GSMA 文档交叉引用

模块 / 特性	FS.19 v10.0	FS.21 v12.0
低层格式过滤	第 3.3.4 节	第 7.3.1 节
类别 1 过滤	第 3.3.5 节，附录 B.3.3	第 7.2.1 节
类别 2 过滤	第 3.3.6 节，附录 B.3.4	第 7.2.2 节，第 16 节
类别 3 过�?	第 3.3.7 节，附录 B.3.5	第 7.2.3 节
拓扑隐藏	第 2.4 节，第 3.4 节	第 3.6 节
速率限制	第 3.4 节	-
AVP 清理	第 4.8.1 节，4.8.2 节	第 3.6 节
深度防御	第 3.4 节	第 3.15 节
过滤类别（协议无关）	附录 A	第 7 节

标准直径路由

标准直径路由是DRA的默认请求转发行为。当没有□级路由规则匹配传入请求时，DRA会应用此行为。它遵循直径基础协议（RFC 6733）中定义的基于优先级的路由机制。它从AVP内容和DRA的连接对等体集合中选择一个目标。

标准路由适用时机

DRA始终尝试标准路由作为其基本行为。如果启用了□级路由模块，并且其规则之一匹配请求，则该规则决定结果。该规则可以路由到命名的对等体，通过`Destination-Host`路由，生成错误答案，或丢弃请求。当□级路由被禁用，或没有规则匹配时，请求将按照此处描述的标准路由优先级进行处理。

有一个狭窄的例外适用于两个阶段之后。当启用时，Rx会话绑定将Rx请求重定向到持有其会话的PCRF。它仅覆盖中继决策。`:drop`结果、错误答案和显式的`Destination-Host`都保持有效。

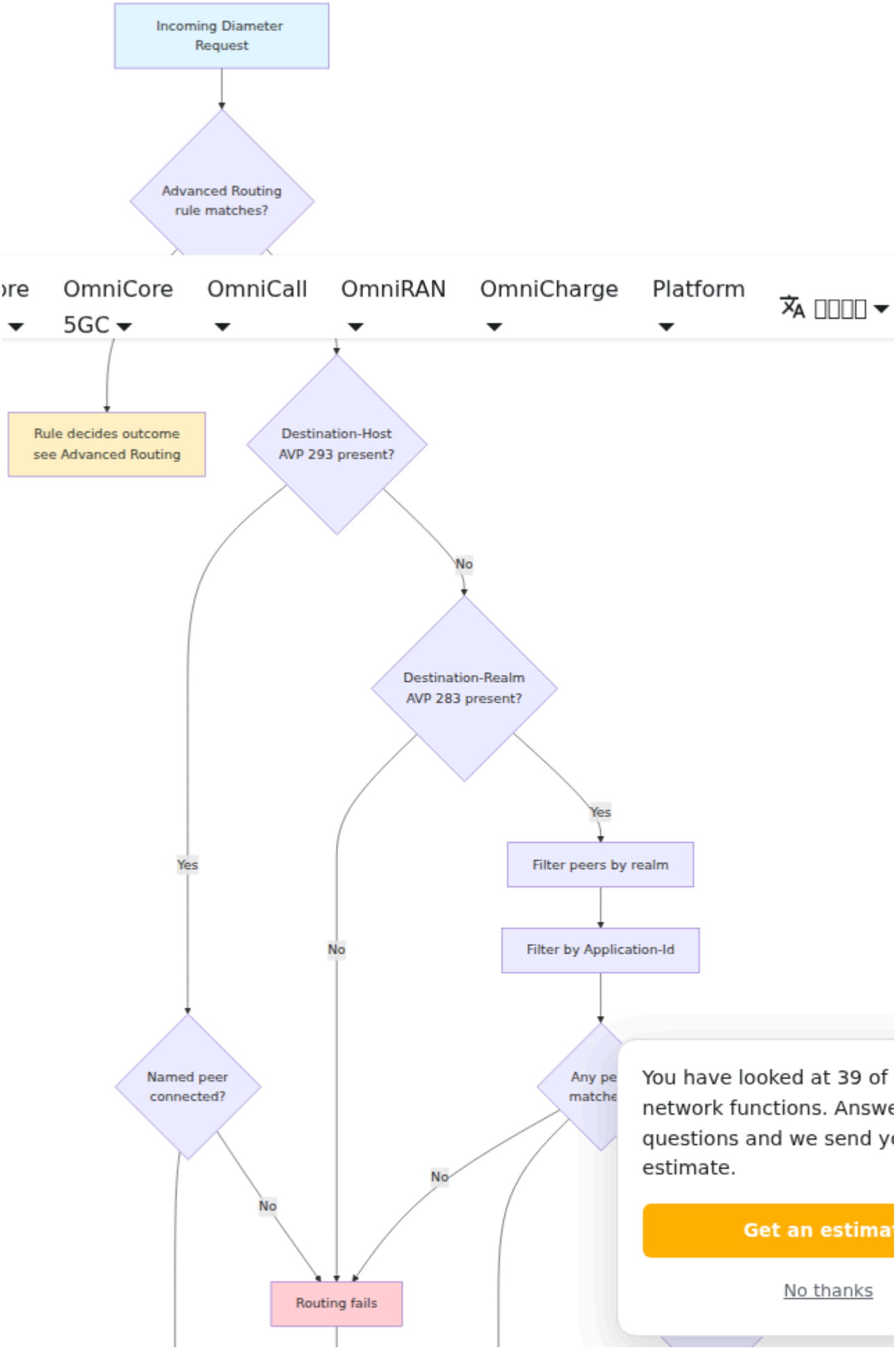
□级路由描述了基于内容的覆盖。这些覆盖基于AVP值，如IMSI模式、领域转换和发起对等体决策。该页面还详细说明了通过`Destination-Host` AVP实现的会话亲和性。

路由决策优先级

标准路由根据固定的优先级顺序评估每个请求。第一个适用的机制胜出。

1. **`Destination-Host` AVP (293)** - 这是优先级最□的机制。如果请求携带`Destination-Host` AVP，DRA将直接路由到该主机。这提供了显式的主机级路由控制。如果命名的对等体未连接，路由将失败。
2. **`Destination-Realm` AVP (283) + `Application-Id`** - 如果缺少`Destination-Host`，DRA将选择一个连接的对等体，该对等体声明支持目标领域。然后，DRA根据消息的**Application-Id**过滤候选对等体。只有那些声明支持该应用程序的对等体才会保留。这些对等体在能力交换期间声明此支持。当多个对等体匹配时，配置的对等体选择算法将在它们之间进行选择。
3. 中继/路由失败 - 如果没有可达的`Destination-Host`或匹配的领域加应用程序对等体可用，DRA将无法传递请求。DRA将返回`3002 DIAMETER_UNABLE_TO_DELIVER`答案。

以下流程图显示了完整的决策路径。���包括□级路由可以在标准路由运行之前拦截请求的点。


Incoming Diameter Request
Advanced Routing rule matches?
ore
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
Rule decides outcome see Advanced Routing
Destination-Host AVP 293 present?
No
Destination-Realm AVP 283 present?
Yes
Filter peers by realm
Filter by Application-Id
Yes
No
Named peer connected?
Any pe matche
No
No
Routing fails
You have looked at 39 of
network functions. Answe
questions and we send y
estimate.
Get an estima
No thanks

只有处于连接状态的对等体在任何阶段都是合格的。DRA会自动将断开连接或故障的对等体排除在领域和应用程序过滤之外。针对断开连接的对等体的`Destination-Host`请求会立即失败。

对等体选择

当多个连接的对等体匹配领域和Application-Id标准时，全局`peer_selection_algorithm`设置决定哪个对等体接收请求。这是一个服务范围的设置。您不需要按领域或对等体配置它。

```
config :dra, :service, %{
  # ... DRA identity and network settings ...
  peer_selection_algorithm: :random
}
```

参数	类型	必需	默认	描述
`peer_selection_algorithm`	Atom	否	`:random`	当多个连接的对等体匹配请求时的负载均衡策略。`:random`从匹配集合中选择一个随机对等体。`:failover`始终选择配置列表中的第一个对等体。仅在第一个对等体不可用时使用其余对等体。

- **`:random`** - 在所有匹配的对等体之间均匀分配负载。当对等体是可互换且成本相等时使用此选项。
- **`:failover`** - 优先选择一个主要对等体。它将有序的对等体列表视为优先级序列。对于主动/备用设计使用此选项。特定对等体在连接时承担流量。

请参见配置以获取完整的服务配置块和对等体定义。

当没有对等体可用时

优先级评估可能无法识别可达的目标。这发生在`Destination-Host`对等体断开连接时，当没有连接的对等体声明目标领域时，当没有匹配的对等体支持Application-Id时，或当请求缺少做出决策所需的AVP时。在这些情况下，DRA无法转发请求。

DRA将返回一个答案，`Result-Code`设置为`3002 DIAMETER_UNABLE_TO_DELIVER`，如RFC 6733 §7.1.5中定义。这告诉发起对等体，DRA已接收到请求，但无法将其传递给合适的下一跳。发起者可以应用自己的重试或故障转移逻辑。

参考

结果代码

代码	名称	描述	参考
2001	DIAMETER_SUCCESS	DRA成功完成请求	RFC 6733 §7.1.1
3002	DIAMETER_UNABLE_TO_DELIVER	DRA无法将请求传递给合适的下一跳。没有可用的匹配的连接对等体	RFC 6733 §7.1.3

常见**AVP**代码

代码	**AVP**名称	类型	用法	参考
1	User-Name	UTF8String	订阅者标识符（IMSI in 3GPP）	RFC 6733 §8.14
264	Origin-Host	DiameterIdentity	发起对等体主机名	RFC 6733 §6.3
268	Result-Code	Unsigned32	标准结果代码	RFC 6733 §7.1
283	Destination-Realm	DiameterIdentity	目标领域	RFC 6733 §6.6
293	Destination-Host	DiameterIdentity	目标主机（可选）	RFC 6733 §6.5
296	Origin-Realm	DiameterIdentity	源领域	RFC 6733 §6.4
297	Experimental-Result	Grouped	供应商特定结果代码	RFC 6733 §7.6

常见命令代码

命令代码是直径消息头的一部分，而不是AVP。

代码	命令名称	描述	参考
257	CER/CEA	能力交换请求/答案	RFC 6733 §5.3
258	RAR/RAA	重新认证请求/答案	RFC 6733 §8.3
274	ASR/ASA	中止会话请求/答案	RFC 6733 §8.5
275	STR/STA	会话终止请求/答案	RFC 6733 §8.4
280	DWR/DWA	��备监控请求/答案	RFC 6733 §5.5
282	DPR/DPA	断开对等体请求/答案	RFC 6733 §5.4
316	ULR/ULA	更新位置请求/答案 (S6a)	3GPP TS 29.272
317	CLR/CLA	取消位置请求/答案 (S6a)	3GPP TS 29.272
318	AIR/AIA	认证信息请求/答案 (S6a)	3GPP TS 29.272
321	PUR/PUA	清除UE请求/答案 (S6a)	3GPP TS 29.272

常见3GPP应用程序ID

对等体在能力交换期间声明其支持的Application-Ids。标准路由通过请求的Application-Id过滤候选对等体。

Application-Id	接口	描述	参考
16777251	S6a/S6d	MME/SGSN ↔ HSS 认证和订阅数据	3GPP TS 29.272
16777252	S13/S13'	MME ↔ EIR 设备身份检查	3GPP TS 29.272
16777238	Gx	PCEF ↔ PCRF 策略和计费控制	3GPP TS 29.212
16777267	S9	主PCRF ↔ 访��PCRF 漫游策略	3GPP TS 29.215
16777272	S6b	PDN-GW ↔ 3GPP AAA 用于非3GPP接入	3GPP TS 29.273
16777216	Cx	I-CSCF/S-CSCF ↔ HSS IMS 注册	3GPP TS 29.229
16777217	Sh	AS ↔ HSS IMS 用户数据	3GPP TS 29.329
16777255	SLg	MME/SGSN ↔ GMLC 位置服务	3GPP TS 29.172
16777291	SLh	GMLC ↔ HSS 位置订阅者信息	3GPP TS 29.173
16777302	Sy	PCRF ↔ OCS 会话绑定	3GPP TS 29.219
16777310	S6m	MTC-IWF ↔ HSS/HLR 用于M2M设备	3GPP TS 29.336

Application-Id	接口	描述	参考
16777312	S6c	SMS-SC/IP-SM-GW ↔ HSS SMS 路由	3GPP TS 29.338
16777345	S6t	SCEF ↔ HSS 监控事件	3GPP TS 29.336
16777236	Rx	AF ↔ PCRF 媒体授权	3GPP TS 29.214

计费应用程序 (Gy/Ro) 遵循3GPP TS 32.299。

相关文档

- 配置 - 完整的服务和对等体配置参考。
- □级路由 - 基于内容的路由规则，通过`Destination-Host`实现会话亲和性，以及优先于标准路由的规则驱动覆盖。

订阅者查找功能（SLF）

订阅者查找功能（SLF）动态学习每个订阅者由哪个网络元素提供服务。它观察通过 DRA 的 Diameter 信令流量。它使用学习到的绑定直接将后续请求（例如位置服务查询）路由到正确的服务节点。它不需要静态路由规则。

这对于 SLg/SLh 位置服务请求非常有用。GMLC 需要到达给定订阅者的服务 MME，但不知道是哪一个 MME。

3GPP TS 29.172 和 3GPP TS 29.173 描述了位置服务的 SLF 概念。3GPP TS 29.272 定义了订阅者注册。

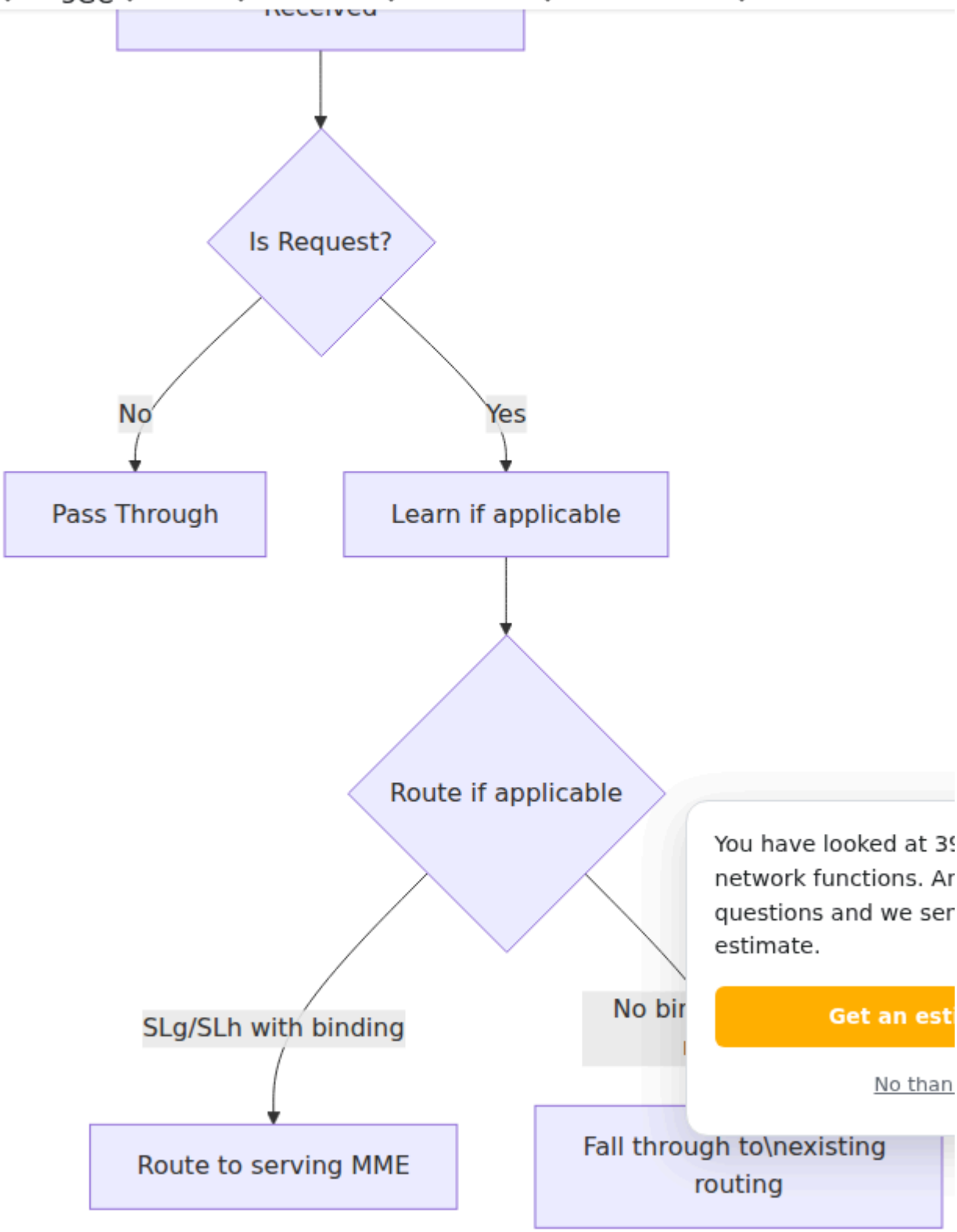
工作原理

该模块分为两个阶段：学习和路由。

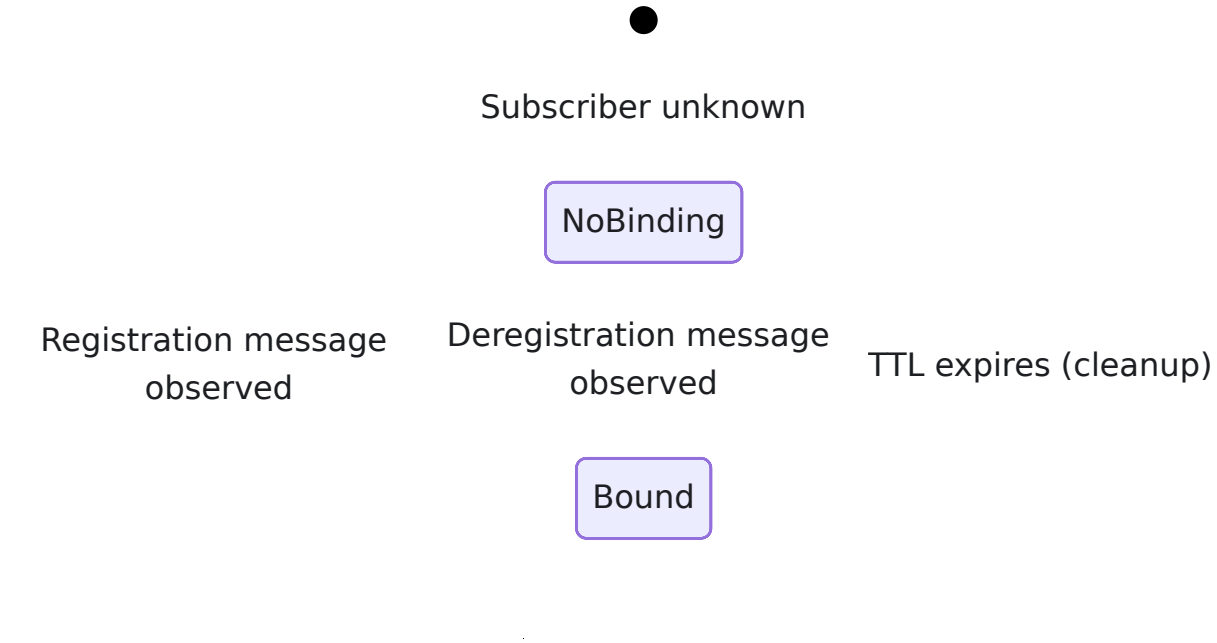
在 学习 阶段，模块观察通过 DRA 流动的注册和注销消息。当订阅者注册（例如通过 S6a Update-Location-Request）时，模块记录哪个网络元素现在为该订阅者提供服务。

在 路由 阶段，可能会到达一个请求，该请求必须��达订阅者的服务节点（例如 SLg Provide-Location-Request）。模块查找绑定并直接路由到正确的对等方。

如果不存在订阅者的绑定，请求将通过现有的路由逻辑，包括□级路由。



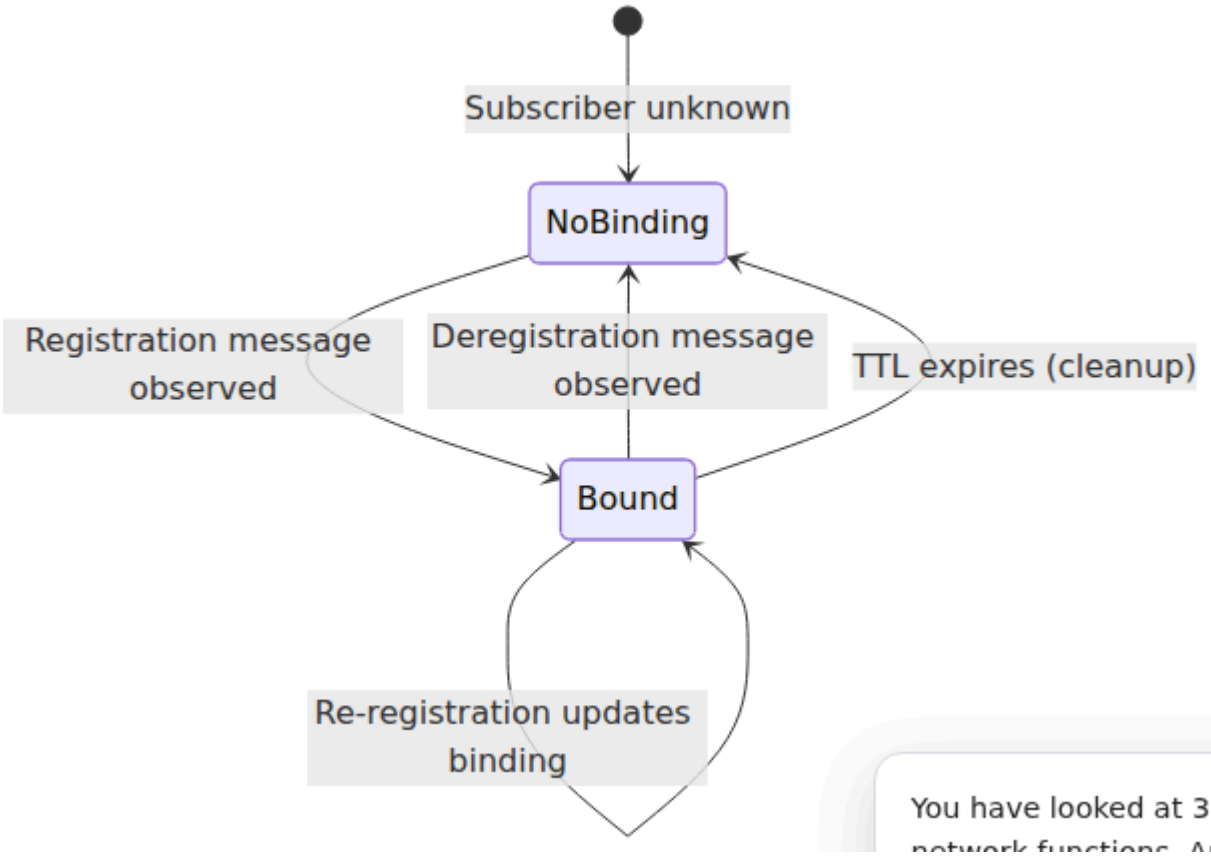
You have looked at 3
network functions. Ar
questions and we ser
estimate.
Get an est
No than



绑定生命周期

学习：创建绑定的消息

模块从以下 Diameter 消息中学习订阅者到服务节点的绑定：



接口	消息	命令码	创建的绑定	**IMSI** 来源
S6a	Update-Location-Request (ULR)	316	`serving_mme`	User-Name AVP (1)
Gx	Credit-Control-Request Initial (CCR-I)	272 (CC-Request-Type=1)	`serving_pgw`	Subscription-Id AVP (443)
Cx	Server-Assignment-Request (SAR)	301 (Server-Assignment-Type=1)	`serving_cscf`	Public-Identity AVP (601)

学习：移除绑定的消息

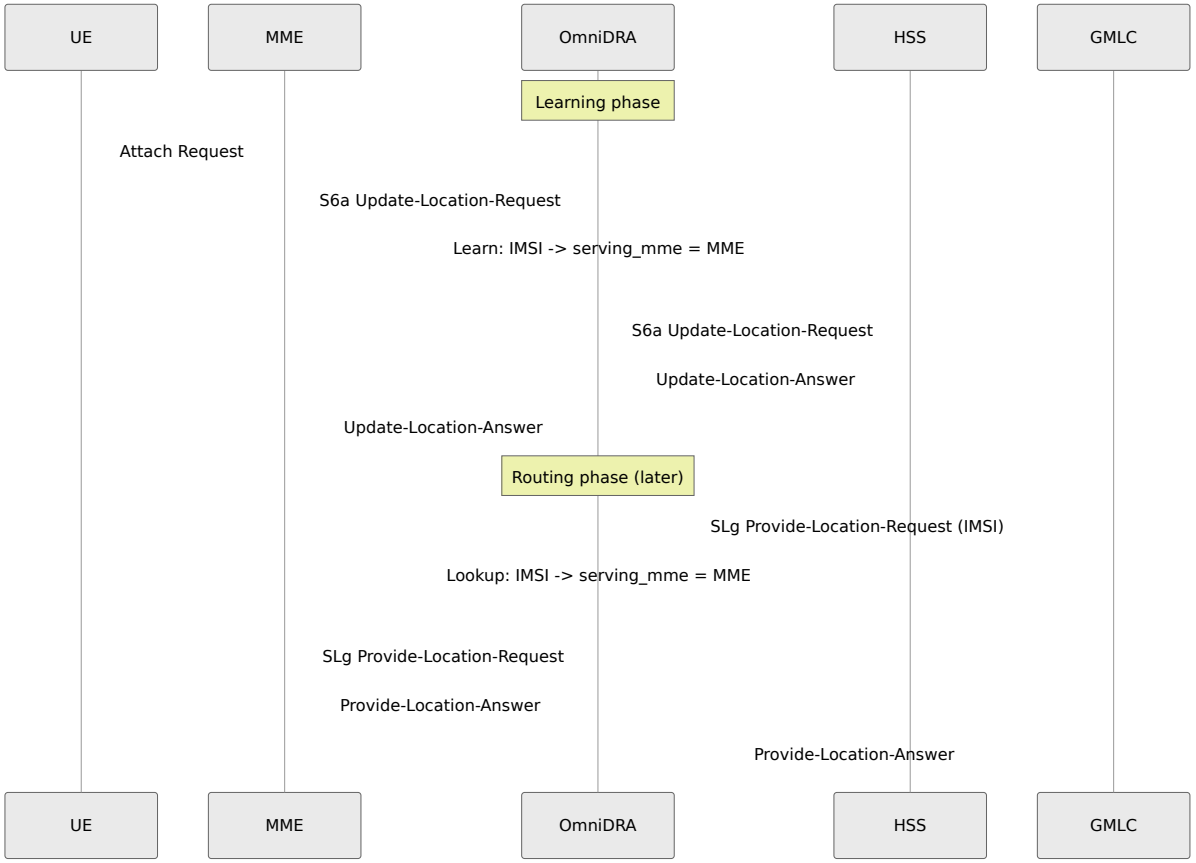
接口	消息	命令码	移除的绑定
S6a	Cancel-Location-Request (CLR)	317	`serving_mme`
S6a	Purge-UE-Request (PUR)	321	`serving_mme`
Gx	Credit-Control-Request Termination (CCR-T)	272 (CC-Request-Type=3)	`serving_pgw`
Cx	Server-Assignment-Request (SAR)	301 (Server-Assignment-Type=4)	`serving_cscf`

当模块移除订阅者的最后一个绑定时，它会从表中删除整个订阅者记录。

路由：使用绑定的消息

接口	消息	命令码	使用的绑定
SLg	Provide-Location-Request (PLR)	8388620	`serving_mme`
SLh	LCS-Routing-Info-Request (RIR)	8388622	`serving_mme`

DRA 将通过学习到的绑定路由的请求转发到服务节点。它使用来自 `diameter.request_timeout` 设置的基本转发超时（���参见 Peer Connectivity 或 Configuration）。该模块不为这些路由请求定义自己的超时。



示例流程：位置服务请求

管道位置

订阅者查找在请求处理管道中在□级路由之前运行。如果 SLF 找到绑定并覆盖路由，□级路由仍然会运行。SLF 路由具有优先权。

配置

在 `config/runtime.exs` 中的 `module_subscriber_lookup` 下配置该模块。

```
module_subscriber_lookup: %{
  # □用或禁用模块
  enabled: false,

  # 绑定过期前保持多长时间（秒）
  binding_ttl_seconds: 86400
}
```

参数

参数	类型	必需	默认	描述
`enabled`	布尔值	是	`false`	启用或禁用订阅者查找模块。当 `false` 时，所有请求未经修块不学习任何绑定。
`children`	列表	否	`[]`	启用时 DRA 主管启动的进程。将其设置为 `[DRA.Router.Modules.SubscriberL` 以便绑定表进程运行。
`binding_ttl_seconds`	整数	否	`86400`	订阅者绑定过期前的时间（秒）。清理然后将其移除。默认值为 2 理在 TTL 间隔的一半运行，最小为 60 秒。

配置示例

标准部署

在大多数网络中使用此配置，其中订阅者在 24 小时内重新注册。

```
module_subscriber_lookup: %{
  enabled: true,
  binding_ttl_seconds: 86400
}
```

工作原理：该模块从通过的流量中学习 MME/PGW/CSCF 绑定，并保持 24 小时。DRA 自动将 SLg 和 SLh 请求路由到正确的服务 MME。每当观察到同一订阅者的新注册消息时，模块会刷新绑定。

用例：具有位置服务 (LCS) 需求的网络。GMLC 必须在没有静态订阅者到 MME 映射的情况下到达服务 MME。

流动环境

在频繁移动的订阅者网络中使用此配置。过时的绑定必须快速过期。

```
module_subscriber_lookup: %{
  enabled: true,
  binding_ttl_seconds: 3600
}
```

工作原理：绑定在 1 小时后过期。当订阅者经常在 MMEs 之间移动时使用此配置。过时的绑定会导致位置请求错误定向。清理间隔每 30 分钟运行。

用例：密集城市网络或具有订阅者流动性的事件。

指标

绑定更新

指标：`diameter.subscriber_lookup.binding.update`

类型：计数器

描述：DRA 每次创建或更新订阅者绑定时递增此计数器。

标签：

- `imsi` - 订阅者的 IMSI
- `binding_type` - 绑定类型：`serving_mme`、`serving_pgw` 或 `serving_cscf`
- `serving_host` - 服务网络元素的 Origin-Host

绑定删除

指标：`diameter.subscriber_lookup.binding.delete`

类型：计数器

描述：DRA 每次注销消息移除订阅者绑定时递增此计数器。

标签：

- `imsi` - 订阅者的 IMSI
- `binding_type` - 移除的绑定类型

路由请求

指标：`diameter.subscriber_lookup.route.count`

类型：计数器

描述：DRA 每次使用学习到的绑定路由 SLg/SLh 请求时递增此计数器。

标签：

- `imsi` - 订阅者的 IMSI
- `binding_type` - DRA 用于路由的绑定类型（目前始终为 `serving_mme`）
- `serving_host` - DRA 路由请求的对等方的 Origin-Host

故障排除

位置请求未路由到服务 **MME**

症状：DRA 不会将 SLg Provide-Location-Requests 或 SLh LCS-Routing-Info-Requests 路由到预期的 MME。它们会转到默认路由。

可能的原因：

- 订阅者尚未通过此 DRA 注册，因此不存在绑定。
- 绑定已过期。TTL 超过。
- 服务 MME 对等方未连接到此 DRA。
- `enabled` 设置为 `false`。

解决方案：

1. 确认模块在配置中已启用。
2. 检查订阅者的 S6a ULR 流量是否通过此 DRA。模块仅从观察到的流量中学习绑定。
3. 确认 `binding_ttl_seconds` 足够长，以覆盖注册和位置请求之间的间隔。

4. 确认服务 MME 作为对等方连接。

绑定未被学习

症状：尽管 S6a/Gx/Cx 流量通过 DRA，绑定表仍然为空。

可能的原因：

- 模块未启用。
- 模块进程未运行。检查主管。
- 消息未在预期的 AVP 中包含有效的 IMSI。

解决方案：

1. 确认配置中 `enabled: true`。
2. 确认 DRA 在配置更改后已重启。
3. 检查 DRA 调试日志中的 `SubscriberLookup: Learned` 条目。这些条目确认了绑定活动。

过时的绑定导致请求错误路由

症状：DRA 将位置请求路由到不再为订阅者提供服务的 MME。

可能的原因：

- Cancel-Location-Request (CLR) 或 Purge-UE-Request (PUR) 未通过此 DRA。
- `binding_ttl_seconds` 对于网络的流动模式设置得过。

解决方案：

1. 将 `binding_ttl_seconds` 减少到与预期的订阅者重新注册间隔相匹配。
2. 确认所有 S6a 注销流量都通过此 DRA。

参考

应用程序 ID

ID	接口	描述	参考
16777251	S6a/S6d	MME/SGSN 到 HSS 的认证和订阅管理	3GPP TS 29.272
16777238	Gx	PCEF 到 PCRF 的策略和计费控制	3GPP TS 29.212
16777216	Cx	I-CSCF/S-CSCF 到 HSS 的 IMS 注册	3GPP TS 29.229
16777255	SLg	GMLC 到 MME 的位置服务	3GPP TS 29.172
16777291	SLh	GMLC 到 HSS/DRA 的 LCS 路由信息	3GPP TS 29.173

命令码

代码	名称	接口	描述
272	Credit-Control-Request/Answer (CCR/CCA)	Gx	会话级别的策略和计费控制
301	Server-Assignment-Request/Answer (SAR/SAA)	Cx	IMS 注册和注销
316	Update-Location-Request/Answer (ULR/ULA)	S6a	在 MME 处更新订阅者位置
317	Cancel-Location-Request/Answer (CLR/CLA)	S6a	HSS 发起的位置取消
321	Purge-UE-Request/Answer (PUR/PUA)	S6a	MME 发起的 UE 清除
8388620	Provide-Location-Request/Answer (PLR/PLA)	SLg	向服务 MME 的位置服务请求
8388622	LCS-Routing-Info-Request/Answer (RIR/RIA)	SLh	LCS 路由信息查找

绑定类型

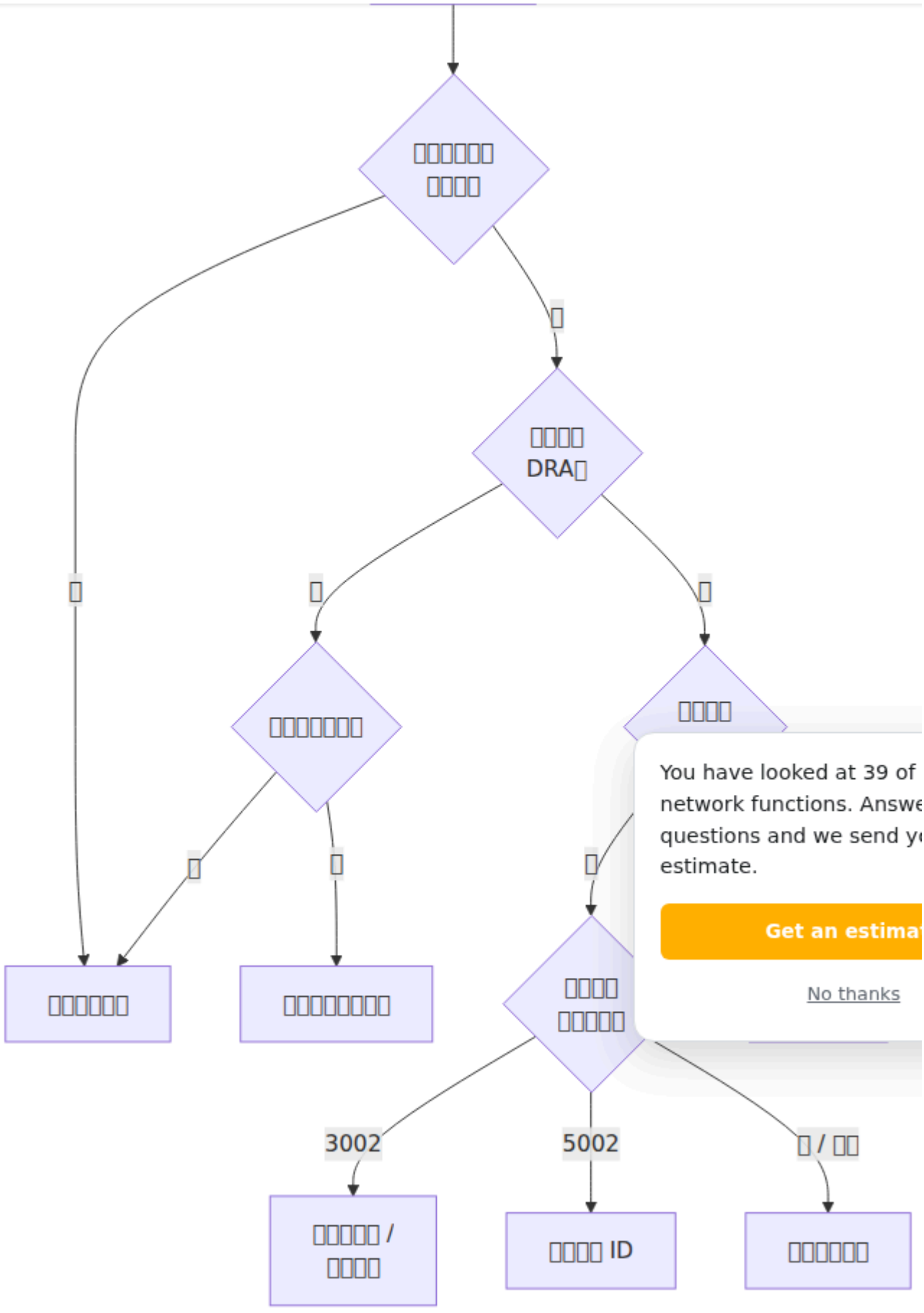
绑定类型	来源	使用	描述
`serving_mme`	S6a ULR	SLg PLR, SLh RIR	当前为订阅者提供服务的 MME
`serving_pgw`	Gx CCR-I	-	处理订阅者会话的 PGW（保留用于未来路由）
`serving_cscf`	Cx SAR (Registration)	-	为订阅者的 IMS 注册提供服务的 S-CSCF（保留用于未来路由）

故障排除

本指南涵盖了 OmniDRA 操作中最常见的问题。每个条目列出了操作员观察到的症状、可能的原因和一组有序的步骤。使用这些步骤来隔离和解决问题。有关本指南中的指标名称，请参见 Metrics。

诊断决策树

使用此流程图查找问题所在，然后再转到特定部分。



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

路由规则

规则不匹配

症状：配置的□级路由规则从未触发。流量落入标准路由或后面的规则中。

可能的原因：

- 一个或多个过滤条件与实际消息内容不匹配。
- 过滤器中的 AVP 代码与相关的 Diameter 应用程序使用的 AVP 不匹配。
- 正则表达式模式不正确或过度/不足锚定。
- 消息类型超出了规则的 `match` 范围。例如，仅请求过滤器应用于答案。
- 规则使用了错误的模块过滤器类型。

解决方案：

1. 检查每个过滤条件与捕获的消息是否匹配。规则中的所有条件必须匹配才能触发规则。
2. 确认 AVP 代码与 Diameter 应用程序匹配。请参见 Advanced Routing 中的 AVP 代码参考。
3. 在部署之前，针对真实的 AVP 值测试正则表达式模式。
4. 确认消息类型在规则的 `match` 范围内。
5. 查看 Advanced Routing 中可用的过滤器类型。确认您为模块使用了正确的过滤器类型。

意外路由

症状：DRA 转发请求，但转发到不同的对等方。或者，当您期望应用规则时，DRA 使用标准路由。

可能的原因：

- 规则排序。□级路由使用首次匹配优先，因此较早的更广泛的规则可能会遮蔽后面的更具体的规则。
- 目标对等方名称不正确，或者对等方不可达。
- 两个规则���有重叠的过滤器，错误的规则获胜。
- 没有规则匹配，因此 Standard Routing 接管。

解决方案：

1. 检查规则顺序。将更具体的规则放在更广泛的规则之前。第一个匹配的规则获胜。请参见 Advanced Routing。
2. 确认对等方名称与配置完全匹配，并且目标对等方已连接（`diameter_peer_status` 报告 `1`）。
3. 检查是否存在具有重叠过滤器的冲突规则。重新排序或收紧它们。
4. 确认 Standard Routing 中的降级行为与您观察到的没有规则应用时的行为相匹配。

转换未应用

症状：路由规则匹配，但 AVP 添加/编辑/删除转换未对消息生效。

可能的原因：

- 转换目标的 AVP 代码对于该用例不正确。
- `edit` 操作在消息中不存在的 AVP 上运行。`edit` 修改现有的 AVP，但从不创建一个。
- 规则的过滤器与预期的消息流不匹配，因此消息从未到达转换。
- 数据包类型过滤器（`:request` 与 `:answer`）排除了消息。
- 该操作不支持数据包类型。`edit` 仅适用于请求。

解决方案：

1. 确认 AVP 代码对于您的用例是正确的。请参见 Advanced Routing 中的 AVP 代码参考。
2. 对于 `edit` 操作，确认消息中已存在 AVP。如果 AVP 可能缺失，请使用 `add`。
3. 确认规则的过滤器与预期的消息流匹配。
4. 确认任何数据包类型过滤器（`:request` 与 `:answer`）对于您转换的方向是正确的。
5. 确认该操作支持数据包类型。`edit` 仅适用于请求。

对等连接性

对等连接失败

症状：配置的对等方从未达到连接状态。`diameter_peer_status{origin_host="..."}` 保持为 `0`，并且路由到该对等方失败。

可能的原因：

- 防火墙阻止配置的 `listen_port`（默认 `3868`）或对等方的端口。

- 对等方配置中的 `ip`、`port` 或 `transport` 不正确。
- 传输不匹配。一侧配置为 `:diameter_tcp`，另一侧配置为 `:diameter_sctp`。
- 对等方广告的 Origin-Host 与配置的 `host` 不匹配。身份必须完全匹配。
- `initiate_connection` 在两端设置相同，因此两侧都不拨号或都拨号。
- 对等方在能力交换（CER/CEA）期间未广告支持的应用程序。
- 对于出站对等方，对等服务未运行或未监听。

解决方案：

1. 确认防火墙规则允许在配置的传输上通过 `listen_port` 和对等方的端口的流量。SCTP 需要内核 SCTP 模块和允许 SCTP 的防火墙规则，而不仅仅是 TCP。
2. 确认对等方的 `ip`、`port` 和 `transport` 值与对等方的实际端点完全匹配。
3. 确认两端使用相同的传输协议。
4. 确认对等方广告的 Origin-Host 与配置的 `host` 匹配。Diameter 身份必须完全匹配才能路由。
5. 检查 `initiate_connection` 角色。必须有一侧发起。将 `initiate_connection: true` 设置在拨号的一侧（通常是 DRA 指向 HSS 或 PCRF）。将 `false` 设置在等待的一侧（通常指向 MME、SGSN 或 P-GW）。
6. 确认对等方在 CER/CEA 期间广告至少一个 DRA 服务的应用程序。没有共同应用程序的对等方无法用于路由。
7. 对于出站对等方，确认远程服务正在运行并监听。DRA 会自动重试不可达的对等方。

对等方抖动 / 重新连接后过早发送流量

症状：在对等方重新连接后，DRA 向其调度请求，而这些请求立即失败或超时，而对等方仍在稳定中。或者，最近恢复的对等方看似可用，但丢失流量。

可能的原因：

- 对等方在网络层不稳定并重复重新连接（抖动）。
- 应用级稳定时间对该对等方来说太短，因此 DRA 在对等方准备好之前路由流量。
- 过长的稳定时间延迟了合法的恢复并延长了停机窗口。

解决方案：

1. 首先调查基础传输不稳定性（网络路径、SCTP 路径故障转移或对等方侧重启）。抖动的对等方是症状，而不是路由配置问题。
2. 调整稳定时间，以便新连接的对等方必须在定义的时间间隔内保持在线，然后 DRA 才能向其路由流量。在此窗口内断开连接的对等方永远不会进入可路由集合。这可以避免流量落入抖动的对等方。

```
config :dra, :gateway,
  # 在向其路由流量之前，保持新连接的对等方此时长（毫秒）。
  # 0 = 一旦 CER/CEA 完成立即路由。
  peer_settle_time_ms: 2_000,

  # RFC 3539 看门狗定时器 Tw 的可选覆盖（毫秒）。
  # watchdog_timer_ms: 30_000
```

参数	类型	必需	默认	描述
`peer_settle_time_ms`	整数	否	`0`	新连接的对等方必须保持在线的毫秒数，DRA 才能向其路由流量。`0` 在 CER/CEA 完成后立即路由。如果对等方在此窗口内断开，DRA 永远不会将其标记为可路由。这可以避免流量落入抖动的对等方。
`watchdog_timer_ms`	整数	否	`30000`	RFC 3539 看门狗定时器 Tw 的可选覆盖，设备看门狗（DWR/DWA）��活间隔。除非您有特定原因更改存活频率，否则请保持未设置。

3. `diameter_peer_status` 反映实际的可路由性。处于稳定窗口内的对等方在指标中仍报告为尚未可路由。使用此信息确认稳定行为是否按预期工作。

4. 对于长期抖动的对等方，增加 `peer_settle_time_ms` 以抑制过早调度。对于稳定的对等方，将其保持在较低（或 `0`）以最小化恢复时间。

> DRA 在应用级稳定时间到期后立即将重新连接的对等方提升为可路由。它不会等待 RFC 3539 看门狗的隐式重新连接延迟。上述稳定时间是 DRA 持有新连接的明确控制。

未授权的对等连接

症状：DRA 拒绝对等方的连接。或者，`diameter_peer_unauthorized_connection_count_total` 随着意外的 `origin_host` / `peer_ip` 增加。

可能的原因：

- 连接的对等方不在 `peers` 配置中，并且 `allow_undefined_peers_to_connect` 为 `false`。

- 对等方广告的 Origin-Host 与任何配置的 `host` 不匹配。
- 一个真正未授权或配置错误的节点尝试连接。

解决方案：

1. 确认是否允许该对等方。如果允许，请将其添加到 `peers` 列表中，并提供其确切的 `host`、`realm`、`ip`、`port` 和 `transport`。
2. 确认对等方广告的 Origin-Host 与配置的 `host` 完全匹配。DRA 将不匹配视为未知对等方。
3. 在生产环境中保持 `allow_undefined_peers_to_connect: false`，并显式添加对等方。仅在任何对等方可能连接的受控环境中启用它。
4. 保持 `log_unauthorized_peer_connection_attempts: true`，以便 DRA 记录被拒绝的尝试以进行安全审查。
5. 调查 `diameter_peer_unauthorized_connection_count_total` 上的 `peer_ip` 和声明的 `origin_host` 标签。来自未知来源的持续速率需要进行安全调查。

参数	类型	必需	默认	描述
`allow_undefined_peers_to_connect`	布尔	否	`false`	当为 `true` 时，DRA 接受来自未配置中的对方的连接。生产环境中持为 `false`
`log_unauthorized_peer_connection_attempts`	布尔	否	-	当为 `true` 时，DRA 记录来自未权对等方的接尝试。

SCTP 关联问题

症状：SCTP 对等方未能建立关联。或者，已建立的关联意外掉线或故障转移。与同一主机上的 TCP 对等方正常连接。

可能的原因：

- 主机上未加载内核 SCTP 模块。
- 防火墙规则允许 TCP，但不允许配置地址上的 SCTP。
- 多宿主集中的一个或多个地址无法从��等方路由。
- 主 IP 不在对等方的 `additional_ips` 列表中。
- 仅为多宿主配置了一个端点，因此 DRA 无法实现完全冗余。
- 内核 SCTP 参数控制路径故障转移的时机，并且与预期不同。

解决方案：

1. 确认 SCTP 内核模块已加载（Linux 上的 `lksctp-tools` 包）。
2. 确认防火墙规则允许在每个配置的 IP 上通过 SCTP 流量，而不仅仅是 TCP。
3. 确认多宿主集中所有 IP 地址在 DRA 和对等方之间双向可路由。
4. 对于 SCTP 对等方，将主 IP 包含在 `additional_ips` 列表中。对于 TCP 对等方，DRA 仅使用 `ip`，因为 TCP 不支持多宿主。
5. 为完整路径冗余配置两个端点的多宿主。
6. 如果故障转移时机与预期不符，请检查内核 SCTP 参数。路径故障转移时机取决于它们，而不是 DRA 配置。
7. 请参见 SCTP Multihoming 以获取完整的传输配置及其限制。

消息交付

消息未路由 / 3002 无法交付

症状：请求到达 DRA，但 DRA 用结果代码 `3002`（DIAMETER_UNABLE_TO_DELIVER）回答它们。或者，DRA 根本不将它们转发到预期的目的地。

可能的原因：

- 没有连接的对等方与请求的 Destination-Host 或 Destination-Realm 匹配。
- 目标对等方已断开连接，或者它在其稳定窗口内尚未可路由。

- 目标对等方未广告请求中的应用程序支持。
- 级路由规则选择了一个不可达的对等方。
- 没有对等方提供请求的领域。这将产生 `3003`（DIAMETER_REALM_NOT_SERVED），而不是 `3002`。

解决方案：

1. 确认至少有一个匹配 Destination-Host 或 Destination-Realm 的对等方已连接。该对等方的 `diameter_peer_status` 必须为 `1`。
2. 检查目标对等方是否在其稳定窗口内。刚重新连接的对等方在 `peer_settle_time_ms` 到期之前故意不可路由。请参见 Peer Flapping。
3. 确认对等方在 CER/CEA 期间广告请求中的应用程序。DRA 会排除未广告该应用程序的对等方。
4. 如果级路由规则针对特定对等方，请确认该对等方已连接。DRA 会跳过未连接的对等方。请参见 Advanced Routing。
5. 区分 `3002` 和 `3003`。`3002` 意味着 DRA 无法选择可达的对等方。`3003` 意味着没有对等方提供请求的领域。如果您观察到 `3003`，请更正领域配置。
6. 跟踪 `3002` 答复的速率以发现系统性交失败：

```
rate(diameter_peer_message_result_code_count_total{result_code="3
[5m])
```

未答复的请求（超时）

症状：DRA 将请求转发给对等方，但在超时内没有返回答案。`diameter_peer_unanswered_request_count_total` 增加，`diameter_peer_last_response_delay` 上升。

可能的原因：

- 对等方接收了请求，但未在 `request_timeout`（默认 `5000` 毫秒）内回答。
- 对等方过载或响应缓慢。
- 网络路径恶化。这与 SCTP 多宿主在故障转移期间相关。
- 对等方处理了请求，但答案在传输中丢失。

解决方案：

1. 确定缓慢或无响应的对等方：

```
topk(5, sum by (routed_to) (
  rate(diameter_peer_unanswered_request_count_total[5m])
))
```

2. 与同一对等方的 `diameter_peer_last_response_delay` 进行比较。这显示它是缓慢而不是无响应。

3. 检查对等方自身的负载和健康状况。持续的超时通常表明下游问题，而不是 DRA 问题。

4. 对于 SCTP 对等方，检查与超时同时发生的路径故障转移事件。请参见 SCTP Multihoming。

5. 仅在对等方的合法处理时间超过默认值时调整 `request_timeout`。更□的值还会延长扩展指标 TTL（`request_timeout` + 5 秒）。

参数	类型	必需	默认	描述
`request_timeout`	整数	否	`5000`	在 DRA 将请求/答案对视为超时之前等待答案的毫秒数。

状态 Gx / Gy 上的 UNKNOWN_SESSION_ID

症状：状态会话（Gx 策略控制，Gy 在线计费）以结果代码 `5002`（DIAMETER_UNKNOWN_SESSION_ID）失败。通常，对等方拒绝更新（CCR-U）或终止（CCR-T），即使初始（CCR-I）成功。

可能的原因：

- DRA 将中间会话请求路由到与持有会话状态的对等方不同的对等方。
- 持有会话的对等方重新启动并丢失其会话表。
- DRA 路由到在恢复其状态之前重新连接的对等方。这是一个稳定窗口交互。
- 负载均衡将一个 Session-Id 的请求分散到多个对等方。

解决方案：

1. 确认所有具有相同 Session-Id 的请求都到达同一状态对等方。对于 Gx 和 Gy，会话亲和性至关重要。中间会话消息必须落在回答 CCR-I 的对等方上。优先使用基于 `Destination-Host` 的路由，或者使用将 Session-Id 家族固定到一个对等方的□级规则，而不是基于领域的负载均衡。请参见 Standard Routing 和 Advanced Routing。

2. 查看 `peer_selection_algorithm`。`:random` 在匹配的对等方之间分配请求，可能会打破状态应用程序的会话亲和性。`:failover` 将请求发送到列表中的第一个对等方。对于状态接口，基于 Destination-Host 进行路由。亲和性是明确的，不依赖于负载均衡。

3. 检查目标对等方在 `5002` 答复开始时是否重新启动。丢失会话表的对等方会拒绝进行中的会话。客户端必须重新建立这些会话。

4. 如果对等方最近重新连接，请确认 `peer_settle_time_ms` 足够大，以便 DRA 仅在对等方准备就绪后路由流量。请参见 Peer Flapping。

5. 跟踪 `5002` 答复的速率以与对等事件相关联：

```
rate(diameter_peer_message_result_code_count_total{result_code="5
[5m])
```

参考：常见结果代码

代码	名称	意义	参考
2001	DIAMETER_SUCCESS	DRA 成功完成请求。	RFC 6733 §7.1.1
3002	DIAMETER_UNABLE_TO_DELIVER	DRA 无法选择可达的对等方以交付请求。	RFC 6733 §7.1.3
3003	DIAMETER_REALM_NOT_SERVED	没有对等方提供请求的领域。	RFC 6733 §7.1.3
3004	DIAMETER_TOO_BUSY	选定的对等方过载。	RFC 6733 §7.1.3
5002	DIAMETER_UNKNOWN_SESSION_ID	对等方无法识别 Session-Id。这是一个丢失或错误路由的状态会话。	RFC 6733 §7.1.5
5012	DIAMETER_UNABLE_TO_COMPLY	DRA 无法按所示处理请求。	RFC 6733 §7.1.5

相关文档

- Metrics - 上述 Prometheus 指标和示例查询。
- Standard Routing - RFC 6733 优先路由、对等选择和答案路由。
- Advanced Routing - 基于规则的路由、过滤器、转换和对等目标。
- SCTP Multihoming - SCTP 传输、多宿主和路径故障转移。