← 述

OmniBSF 置参考

配置是从应用程序环境键 `:omnibsf` 中读取的。标准部署机制是 `config/runtime.exs`，它将每个值从操作系统环境变量映射到内置默认值。

```
config :omnibsf,
  sbi_scheme: System.get_env("SBI_SCHEME", "http"),
  sbi_addr:   System.get_env("SBI_ADDR", "127.0.0.15"),
  sbi_port:   String.to_integer(System.get_env("SBI_PORT",
"7777")),
  nrf_uri:    System.get_env("NRF_URI", "http://127.0.0.1:7777"),
  mcc:        System.get_env("MCC", "999"),
  mnc:        System.get_env("MNC", "70"),
  prometheus_metrics_port:
String.to_integer(System.get_env("PROMETHEUS_PORT", "9571")),
  heartbeat_interval:
String.to_integer(System.get_env("HEARTBEAT_INTERVAL", "10000"))

config :logger, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all
```

□□□

□□	□□	□□	□□	□
`sbi_scheme`	□□□	□	`"http"`	`SBI_SCHE`
`sbi_addr`	□□□	□	`"127.0.0.15"`	`SBI_ADDR`
`sbi_port`	□□	□	`7777`	`SBI_PORT`
`nrf_uri`	□□□	□	`"http://127.0.0.1:7777"`	`NRF_URI`
`mcc`	□□□	□	`"999"`	`MCC`
`mnc`	□□□	□	`"70"`	`MNC`

参数	类型	必需	默认	[
`prometheus_metrics_port`	整数	否	`9571`	`PROMETHE`
`heartbeat_interval`	整数（毫秒）	否	`10000`	`HEARTBEA`

> 注意：`runtime.exs` 中 `nrf_uri` 的默认值是 `http://127.0.0.1:7777`。当环境变量未设置且应用程序回退到编译的访问器默认值时，值为 `http://127.0.0.10:7777`。在生产环境中明确设置 `NRF_URI` 以避免歧义。

日志记录

日志记录器配置为通过 `OmniLogger.JsonFormatter` 发出结构化 JSON，并包含完整的元数据。活动日志级别可以通过管理 API 在运行时更改，而无需重启。有关运行时日志级别端点，请参见 管理 API。

绑定生命周期调优

绑定 TTL 和清理扫描间隔是内部默认值，并未在 `runtime.exs` 中作为环境变量暴露。如果部署需要非默认值，可以通过应用程序环境进行设置：

应用程序环境键	类型	默认	描述
`:binding_ttl_seconds`	整数（秒）	`86400`（24 小时）	在注册时应用于绑定的生存时间，并在每次 PATCH 时刷新；在此之后，条目有资格进行清理。
`:binding_cleanup_interval_ms`	整数（毫秒）	`60000`（60 秒）	后台清理程序扫描并删除过期绑定的频率。

TTL 在注册时被标记，并在每次 PATCH 时刷新。合并补丁 `UpdateIndPCFBinding`（PATCH）在补丁时重新计算绑定的到期时间，因此 PATCH 作为保持活动状态的操作，延长绑定的生命周期。定期触碰其绑定的注册者（例如，在 PDU 会话修改时）可以在会话活动期间保持该绑定处于活动��态。未被补丁的绑��在注册后最多只能存活 `:binding_ttl_seconds`。将 TTL 设置得高于保持活动触碰之间的间隔（或高于未发出保持活动时的最长预期 PDU 会话生命周期）；否则，仍处于活动状态的会话的绑定将在会话中被清除，后续发现查询返回 `204 No Content`，直到注册者重新注册。

清理间隔限制了过期绑定在删除之前的存在时间 - 条目可以在过期后最多存活一个完整的间隔。较短的间隔可以更快地回收内存并清除过期绑定，但会导致更频繁的绑定表完整扫描。删除操作会被记录（`[BindingCleaner] Removed N expired binding(s)`）；有关过期绑定增长的详细信息，请参见 故障排除。

NRF □置文件容量和优先级

NF 配置文件 OmniBSF 在 NRF 注册时携带 `capacity` 和 `priority`。通过 NRF 发现 BSF 的消费者使用这些进行选择和负载分配，依据 TS 29.510：在候选实例中，优先选择较低的 `priority`，而 `capacity` 则权衡发送到相同优先级实例的流量份额。它们并不会限制 OmniBSF 本身 - 它们是消费者选择逻辑的宣传提示，仅在注册多个 BSF 实例时才重要。

应用程序环境键	类型	默认	描述
`:nf_capacity`	整数	`100`	在 NF 配置文件中宣传的相对容量权重（典型范围 0–65535）。较高的值在相同优先级实例中吸引更大份额的流量。
`:nf_priority`	整数	`0`	在 NF 配置文件中宣传的选择优先级。较低的值更受消费者青睐。

这两个值都可以通过管理 API 在运行时更改，更新的配置文件将在下一个更新周期重新推送到 NRF；有关详细的端点参考，请参见 管理 API。运行时更改不会持久化，重启时将恢复为这些值。

管理和指标端口

除了 SBI 监听器，OmniBSF 还暴露：

- 在 `prometheus_metrics_port`（默认 `9571`）上提供 Prometheus 抓取端点。有关详细信息，请参见 指标与监控。
- 一个操作员管理 API（OAM），用于查看统计信息、检查/清除绑定、在运行时调整 NRF 配置文件的容量/优先级和日志级别，以及强制 NRF 重新注册。有关完整的端点参考，请参见 管理 API。

← 概述

OmniBSF 管理 API

管理 (OAM) API 是一个面向操作员的 REST 接口，与 5G SBI 分开。它公开状态、统计信息、绑定检查/移除和运行时配置控制。它通过 HTTPS 提供服务。

返回到 操作指南。

访问

属性	值
方案	HTTPS (TLS)
端口	`8443`
绑定地址	`0.0.0.0`

所有响应均为 JSON，封装在一个标准信封中，带有 `success` / `error` 状态。

状态端点

方法	路径	描述
`GET`	`/status/api`	存活探测。返回 `"ok"`。
`GET`	`/status/nrf`	NRF 注册状态：`nrf_status`，`nf_instance_id`，`nrf_uri`。如果注册过程不可用，则返回 `unknown`。
`GET`	`/status/nf`	NF 服务状态：`nf_type`，广告的 `nbsf-management` 服务，`sbi_addr`/`sbi_port`，以及当前活动的 PCF 绑定数量。

绑定检查

方法	路径	描述
`GET`	`/pcf_bindings`	列出活动的 PCF 绑定（摘要视图：绑定 ID，UE IP，DNN，S-NSSAI，SUPI，PCF FQDN）。
`GET`	`/api/bindings`	列出所有绑定，包括 UE 地址、DNN、S-NSSAI、SUPI、GPSI 和 PCF 地址信息，以及总计数。
`GET`	`/api/bindings/{id}`	按 ID 查看单个绑定的详细信息，包括框架路由列表和协商的 `suppFeat`。如果未知，则返回 404。
`GET`	`/api/statistics`	聚合统计信息：总绑定数和每个 DNN 的细分。

绑定管理

方法	路径	描述
`DELETE`	`/oam/binding/{id}`	按 ID 移除单个 PCF 绑定。如果绑定不存在，则返回 404。
`POST`	`/oam/binding`	清除所有 PCF 绑定。返回清除的绑定数量。这是破坏性的 - 活动会话在重新注册之前会失去其绑定。

运行时□置

方法	路径	描述
`GET`	`/oam/config`	查看当前运行时配置：`sbi_addr`，`sbi_port`，`nrf_uri`，`mcc`，`mnc`，NF `capacity`/`priority`，和日志级别。
`PATCH`	`/oam/config/{id}`	更新选定的运行时值。接受的键：`nrf_uri`，`nf_capacity`，`nf_priority`，`log_level`。无效或空的更新返回 400。
`POST`	`/oam/nrf/reregister`	强制 NF 重新注册到 NRF。返回结果注册状态和 NF 实例 ID。
`POST`	`/oam/log_level`	在运行时更改日志级别。主体：`{"level": "<level>"}`。

可□置值

键	类型	备注
`nrf_uri`	字符串	NRF 的基本 URI。在下一个注册/心跳周期生效。
`nf_capacity`	整数	在 NRF 配置文件中广告的 NF 容量；在更新时推送到 NRF。
`nf_priority`	整数	在 NRF 配置文件中广告的 NF 优先级；在更新时推送到 NRF。
`log_level`	字符串	取值之一：`emergency`，`alert`，`critical`，`error`，`warning`，`warn`，`notice`，`info`，`debug`。

通过此 API 进行的更改仅应用于运行实例，并不会持久化；在重启时将恢复为配置值。

← 概述

OmniBSF 指标与监控

指标在 `prometheus_metrics_port` 上公开（默认 `9571`）。

BSF 指标

指标	类型	标签	描[
`omni_bsf_pcf_binding_register_count`	counter	`dnn`	按 DNN PCF 绑定
`omni_bsf_pcf_binding_discover_count`	counter	`result`	按 `resu` （`found` `not_fo` 标记的发现[
`omni_bsf_pcf_binding_deregister_count`	counter	`binding_id`	按绑定 ID 销数量
`omni_bsf_binding_registers_total`	counter	-	聚合注册计[
`omni_bsf_binding_discovers_total`	counter	`result`	按 `resu` 聚合发现计[
`omni_bsf_binding_deregisters_total`	counter	-	聚合注销计[
`omni_bsf_active_bindings_count`	gauge	-	活动 PCF 数量。由遥[`[:omni` `:bindi` `:activ` 动，该事件[册、注销及[期绑定时发[跟踪当前活[量
`omni_bsf_nrf_registration_status`	gauge	`nf_type`	NRF 注[= 已注册[未注册）

> `omni_bsf_pcf_binding_deregister_count` 计数器按 `binding_id` 标记。由于绑定 ID 是唯一的 UUID，这会为每个绑定生成一个时间序列，并且具有高基数；对于聚合速率/警报，建议使用 `omni_bsf_binding_deregisters_total`。

> `omni_bsf_active_bindings_count` 计数器是一个实时计数器，由 `[:omnibsf, :bindings, :active]` 遥测事件驱动，BSF 在活动绑定集合发生变化时发出该事件：在绑定注册、注销以及每次 BindingCleaner 清理过期绑定时。因此，它反映当前活动绑定的数量，适合用于警报和仪表板。相同的数字也可以通过 OAM 统计信息端点（`GET /api/statistics`）和 NF 状态端点获得；请参见 管理 API。

仅遥测事件

OmniBSF 发出以下内部 `:telemetry` 事件，这些事件 当前 不作为 Prometheus 计数器导出。它们可供任何附加的遥测处理程序使用，但不会出现在指标抓取端点上。

事件	发出	描述
`[:omnibsf, :pcf_binding, :update]`	每次 PATCH（`UpdateIndPCFBinding`）一次	当绑定通过合并补丁更新时触发。携带受影响的 `binding_id`。
`[:omnibsf, :binding, :expired]`	每次清理过期条目时一次	背景清理程序触发，带有清理掉的绑定数量，清理的是那些超过其 TTL 的绑定。

BEAM VM 指标

指标	类型	描述
`beam_memory_total`	gauge	总 BEAM 内存（字节）
`beam_memory_processes`	gauge	分配给 Erlang 进程的内存
`beam_memory_processes_used`	gauge	进程实际使用的内存
`beam_memory_system`	gauge	系统内存
`beam_memory_atom`	gauge	总原子内存
`beam_memory_atom_used`	gauge	使用的原子内存
`beam_memory_binary`	gauge	二进制内存
`beam_memory_code`	gauge	代码内存
`beam_memory_ets`	gauge	ETS 表内存
`beam_processes_count`	gauge	Erlang 进程数量
`beam_ports_count`	gauge	Erlang 端口数量
`beam_atom_count`	gauge	原子数量
`beam_vm_uptime`	gauge	VM 运行时间（秒）

示例 PromQL

绑定注册速率（每秒，每个 DNN）：

```
sum by (dnn) (rate(omni_bsf_pcf_binding_register_count[5m]))
```

过去 5 分钟的发现未命中比率：

```
sum(rate(omni_bsf_binding_discovers_total{result="not_found"}
[5m]))
  /
sum(rate(omni_bsf_binding_discovers_total[5m]))
```

当 BSF 未与 NRF 注册时发出警报：

```
omni_bsf_nrf_registration_status{nf_type="bsf"} == 0
```

> 上述指标名称使用 Prometheus 点到下划线的规范化。`result` 的标签值报告为 `found` / `not_found`。

← 概述

OmniBSF 操作指南

概述

OmniBSF 实现了 5G 核心的绑定支持功能 (BSF)。BSF 维护一个 PCF 绑定的注册表：对于每个 PDU 会话，它存储 UE 身份 (IPv4 地址、IPv6 前缀、MAC 地址或 SUPI) 以及 DNN 和 S-NSSAI 与服务该会话策略的 PCF 实例之间的映射。其他网络功能 - AF/NEF、SMF 或另一个 PCF - 查询 BSF 以发现哪个 PCF 负责给定会话，以便该会话的所有策略信令都指向同一个 PCF 实例。

OmniBSF 在 SBI 基础路径 `/nbsf-management/v1` 下公开 `Nbsf_Management` 服务 (3GPP TS 29.521)。绑定完全保存在 ETS 表中并在内存中进行索引，以实现并发和低延迟查找。绑定具有可配置的生存时间 (TTL)；后台清理程序定期删除过期条目。OmniBSF 在启动时向 NRF 注册其 NF 配置文件，并定期发送心跳。

内部数据存储

绑定在运行时保存在六个 ETS 表中：主绑定表和五个专用查找索引。所有数据都在内存中，并且在进程重启后不会保留。

ETS 表	类型	键	值
`bsf_bindings`	`:set`	binding_id (UUID 字符串)	完整绑定映射
`bsf_ipv4_index`	`:set`	UE IPv4 地址 (字符串)	binding_id
`bsf_ipv6_index`	`:set`	UE IPv6 前缀 (字符串)	binding_id
`bsf_supi_index`	`:bag`	SUPI (字符串)	binding_id
`bsf_mac_index`	`:set`	UE MAC 地址 (小写字符串)	binding_id
`bsf_dnn_snssai_index`	`:bag`	`{dnn, sst}`	binding_id

每个查找键由其自己的索引提供服务，因此发现不会扫描主绑定表。`macAddr48` 查找通过 `bsf_mac_index` 解析（键被小写以便匹配不区分大小写），而 DNN+S-NSSAI 查找通过 `bsf_dnn_snssai_index` 解析，键为 `{dnn, sst}`（小

匹配桶随后根据可选的 S-NSSAI `sd` 进行过滤）。

`bsf_supi_index` 是一个 `:bag`：单个 SUPI 可以持有多个并发绑定（例如，多个不同 DNN 的 PDU 会话），因此一个 SUPI 可以映射到多个绑定 ID。通过 `supi` 的发现每次返回一个 `PcfBinding`，根据 TS 29.521，因此返回第一个匹配的绑定。

3GPP 角色和规范参考

规范	相关性
TS 23.501	系统架构 - BSF 网络功能角色
TS 23.503	策略和计费控制框架 - 策略关联的会话绑定上下文
TS 29.521	`Nbsf_Management` 服务 API (PCF 绑定注册/发现/更新/注销)，`PcfBinding` 数据模型、支持的特性协商
TS 29.500	5G SBI 通用框架 (ProblemDetails, HTTP 语义)
TS 29.510	NF 注册和与 NRF 的心跳

当 PCF 部署为多个实例（或作为 PCF 集合）时，BSF 变得相关。它的角色是会话绑定：记录哪个 PCF 服务每个 PDU 会话，以便后续的策略交互（来自 AF 通过 NEF，或来自 SMF）到达正确的 PCF。

SBI 端点

所有端点都在基础 URL `{sbi_scheme}://{sbi_addr}:{sbi_port}` 下提供，基础路径为 `/nbsf-management/v1`。

方法	路径	描述	成功
`POST`	`/pcfBindings`	为 PDU 会话注册 PCF 绑定。返回指向创建的绑定的 `Location` 头。	201 Crea
`GET`	`/pcfBindings`	发现与提供的查询参数匹配的 PCF 绑定。	200 OK, `PcfBind` 如果没有匹配则 204 No Content
`PATCH`	`/pcfBindings/{bindingId}`	更新现有绑定（`UpdateIndPCFBinding`）。主体必须是 `application/merge-patch+json`。	200 OK, ��新的 `PcfBind`
`DELETE`	`/pcfBindings/{bindingId}`	通过其 ID 注销 PCF 绑定。	204 No Content

任何其他路径返回 `404`，并带有 `ProblemDetails` 主体。

GET 查询参数

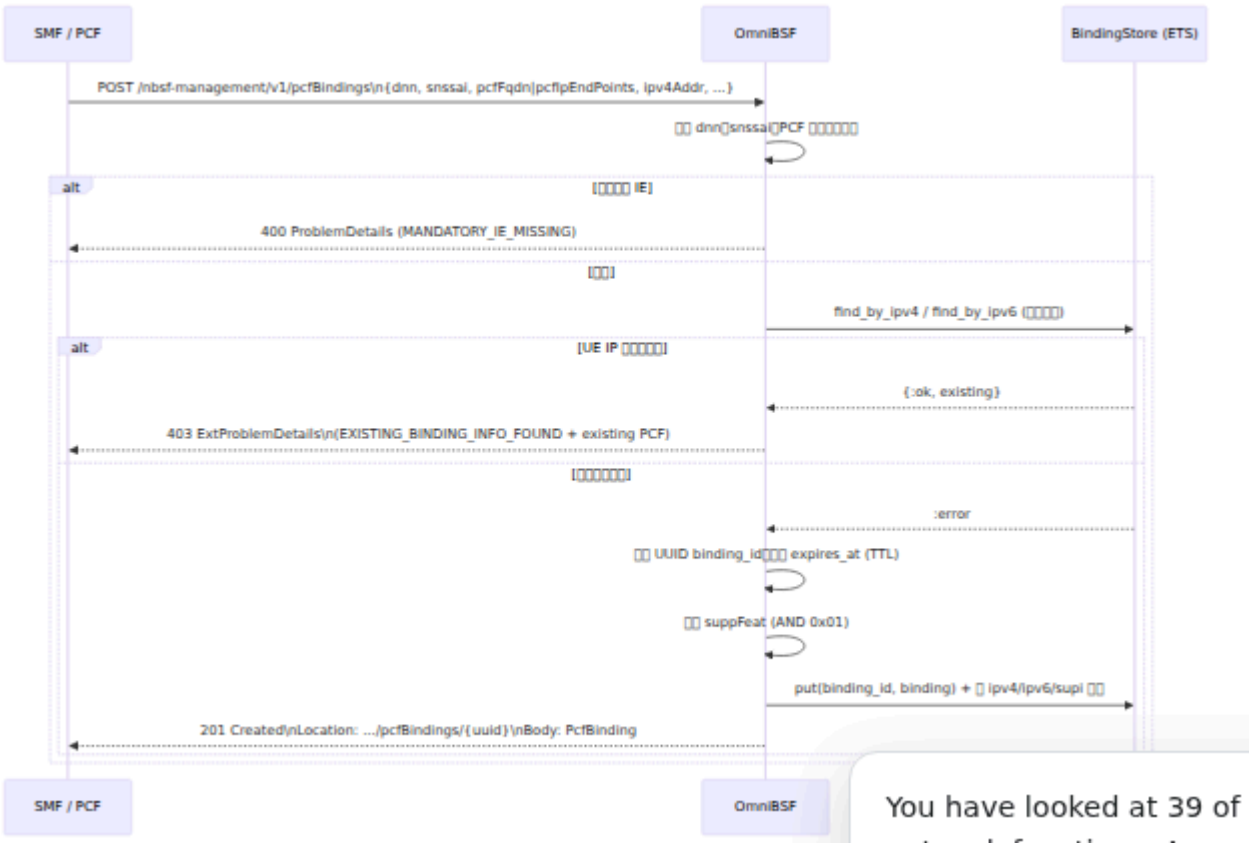
发现端点使用第一个匹配参数解析绑定，优先顺序如下：

优先级	参数	类型	查找	描述
1	`ipv4Addr`	字符串	索引	UE IPv4 地址（精确匹配）
2	`ipv6Prefix`	字符串	索引	UE IPv6 前缀（精确匹配）
3	`macAddr48`	字符串	索引	UE MAC 地址（通过 `bsf_mac_index` 不区分大小写匹配）
4	`supi`	字符串	索引	订阅者永久标识符（如果多个并发绑定共享 SUPI，则返回第一个匹配）
5	`dnn` + `snssai`	字符串 / JSON	索引	DNN 和 S-NSSAI 一起，通过 `bsf_dnn_snssai_index` 解析，键为 `{dnn, sst}`（`sst` 必须匹配；提供时匹配 `sd`）

当提供时，`pcfSetId`、`pcfRegionId` 和 `bindLevel` 查询参数进一步缩小结果范围，根据 TS 29.521：存储值不匹配提供的过滤器的候选项不会返回。查询中缺少的过滤器将被忽略。

POST 请求主体 - 字段

字段	类型	必需	描述
`dnn`	字符串	是	数据网络名称
`snssai`	对象	是	S-NSSAI (`sst` 整数，选填 `sd` 字符串)
`pcfFqdn`	字符串	有条件	PCF FQDN - 如果缺少 `pcfIpEndPoints` 则必填
`pcfIpEndPoints`	数组	有条件	PCF IP 端点列表 - 如果缺少 `pcfFqdn` 则必填
`ipv4Addr`	字符串	否	UE IPv4 地址（已索引；也用于驱动重复检测）
`ipv6Prefix`	字符串	否	UE IPv6 前缀（已索引；也用于驱动重复检测）
`macAddr48`	字符串	否	UE MAC 地址
`supi`	字符串	否	订阅者永久标识符（已索引）
`gpsi`	字符串	否	通用公共订阅标识符
`ipv4FrameRouteList`	数组	否	IPv4 帧路由
`ipv6FrameRouteList`	数组	否	IPv6 帧路由
`pcfSetId`	字符串	否	PCF 集标识符。存储在绑定中，并在作为查询过滤器提供时用于缩小发现范围。
`pcfRegionId`	字符串	否	PCF 区域标识符。存储在绑定中，并在作为查询过滤器提供时用于缩小发现范围。
`bindLevel`	字符串	否	绑定级别。存储在绑定中，并在作为查询过滤器提供时用于缩小发现范围。



字段	类型	必需	描述
`suppFeat`	字符串（十六进制）	否	支持的特性位图；通过与特性 `0x01`（BindingUpdate）的位与协商

对重复注册的 `403` 响应携带 `ExtProblemDetails` 主体（`cause` = `EXISTING_BINDING_INFO_FOUND`），并在顶层内联现有 PCF 的 `pcfSmFqdn` / `pcfSmIpEndPoints`，根据 TS 29.521。

运营商管理 API

除了 SBI 监听器，OmniBSF 还公开了一个运营商管理（OAM）REST 接口，用于查看统计信息、检查/清除绑定、在运行时调整 NRF 配置文件容量/优先级和日志级别，以及强制 NRF 重新注册。有关完整的端点参考，请参见 管理 API。

关键程序

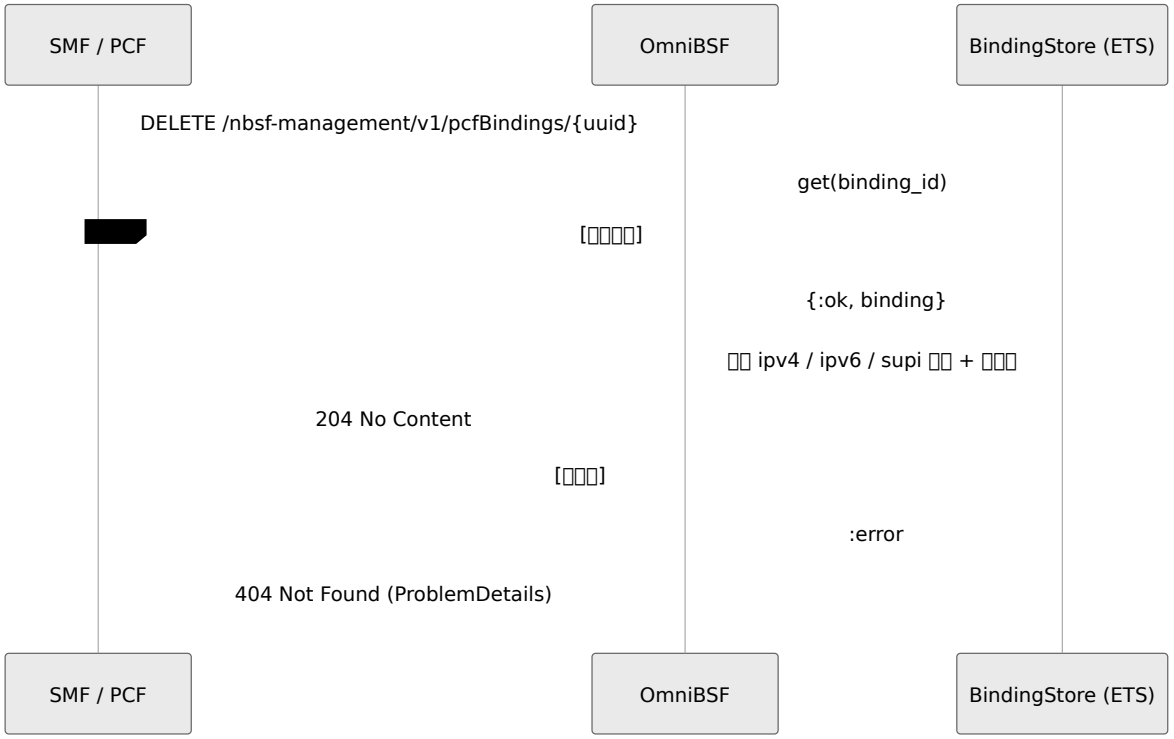
PCF 绑定注册（POST）

You have looked at 39 of

PCF 绑定发现（GET）

PCF 绑定更新（PATCH）

Merge-patch 仅适用于字段 `ipv4Addr`、`ipv6Prefix`、`macAddr48`、`pcfFqdn` 和 `pcfIpEndPoints`；因为 IPv4/IPv6/SUPI/MAC 和 DNN+S-NSSAI 索引基于绑定内容，旧记录及其索引条目将被删除，更新的绑定将被重新插入，以保持索引一致性。PATCH 还会从 `binding_ttl_seconds` 刷新绑定的 `expires_at`，因此它充当保持活动的机制，延长绑定的生命周期。请参见 绑定生命周期调整。



PCF 绑定注销（DELETE）

NRF 注册和心跳

在启动时，OmniBSF 向 NRF 注册一个 BSF NF 配置文件（来自 `nrf_uri`），广告其 SBI 地址/端口/方案、配置的 PLMN（`mcc`/`mnc`）和 `nbsf-management` v1 服务。然后，它每 `heartbeat_interval` 毫秒发送定期的 NFUpdate 心跳。可以通过管理 API 随时强制注册。

← 概述

OmniBSF 故障排除

期望绑定时返回 204

1. 确认查询键与注册的内容匹配。`ipv4Addr`、`ipv6Prefix` 和 `supi` 查找是精确字符串匹配 - 格式差异（前导零、前缀长度表示法）会导致匹配失败。
2. 检查日志中是否有 `PCF binding not found for query:` 以查看确切的查询。
3. 确认注册者已完成 POST（在日志中查找 `PCF binding registered:`）。
4. DNN+S-NSSAI 发现通过 `{dnn, sst}` 索引解析，并且 `sst` 必须完全匹配；原始 POST 主体中的整数与字符串类型不匹配将导致无法匹配。
5. 检查绑定的 TTL 是否已过期，清理程序是否已将其移除（`[BindingCleaner] Removed N expired binding(s)`）。

POST 返回 403

提供的 `ipv4Addr` / `ipv6Prefix` 已存在绑定。响应是一个 `ExtProblemDetails`，其中 `cause = EXISTING_BINDING_INFO_FOUND`，并包含现有 PCF 的 `pcfSmFqdn` / `pcfSmIpEndPoints`。如果确实打算重新绑定，请重用返回的 PCF，或者先 `DELETE` 现有绑定。

POST 返回 400

主体缺少一个强制 IE。`detail` 字段标识它：`No DNN`、`No S-NSSAI` 或 `No PCF address information (pcfFqdn or pcfIpEndPoints required)`。通用缺失字段路径报告 `MANDATORY_IE_MISSING`。

DELETE 或 PATCH 返回 404

绑定 ID 与任何活动绑定不匹配。可能已经被移除、其 TTL 已过期，或 OmniBSF 已重启（所有绑定都在内存中，重启时会丢失）。检查 `PCF binding not found for deregister:` / `PCF binding not found for update:`。

`PATCH` 返回 415

`PATCH` 处理程序需要 `Content-Type: application/merge-patch+json`。任何其他内容类型都会被拒绝，返回 `415 Unsupported Media Type`。

高内存 / 增长的绑定数量

通过 `omni_bsf_active_bindings_count` 指标监控实时绑定数量，该指标在绑定注册、注销和每次清理程序清扫时发出；OAM 统计信息端点（`GET /api/statistics`）和 NF 状态端点报告相同的数字。未通过 `DELETE` 终止的 PDU 会话的绑定会一直存在，直到其 TTL 过期并被清理程序移除。由于 PATCH 会刷新绑定的 TTL，因此持续对绑定进行 PATCH 的注册者会使其保持活跃；绑定仅在 `:binding_ttl_seconds` 内未收到 DELETE 或 PATCH 时才会过期。如果过时绑定的积累速度超过默认的 24 小时 TTL 清除速度，请缩短 `:binding_ttl_seconds` 或 `:binding_cleanup_interval_ms`。管理 API 还提供清除所有操作。

NRF 注册失败

检查 `omni_bsf_nrf_registration_status{nf_type="bsf"}` 和日志。验证 `nrf_uri` 是否可达，`mcc`/`mnc` 是否与 NRF 配置的 PLMN 匹配，以及从 `sbi_addr` 到 NRF 的连接是否正常。可以通过管理 API 强制重新注册。