← 述

OmniSCP 置

所有可调节的设置都通过 `config/runtime.exs` 提供，每个设置都有一个环境变量支持，以便在部署时可以覆盖值而无需编辑文件。下面显示的值是对应环境变量未设置时应用的默认值。

```
import Config

config :omniscp,
  sbi_scheme: System.get_env("SBI_SCHEME", "http"),
  sbi_addr: System.get_env("SBI_ADDR", "127.0.0.17"),
  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"),
  heartbeat_interval:
String.to_integer(System.get_env("HEARTBEAT_INTERVAL", "10000")),
  prometheus_metrics_port:
String.to_integer(System.get_env("PROMETHEUS_PORT", "9568")),
  discovery_cache_ttl:
String.to_integer(System.get_env("DISCOVERY_CACHE_TTL", "60000")),
  lb_strategy: String.to_atom(System.get_env("LB_STRATEGY",
"round_robin")),
  max_retries: String.to_integer(System.get_env("MAX_RETRIES",
"2")),
  upstream_timeout:
String.to_integer(System.get_env("UPSTREAM_TIMEOUT", "5000"))

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

`:omniscp` □□

□□	□□	□□	□□	□
`sbi_scheme`	□□□□	□	`"http"`	`SBI_SCHE`
`sbi_addr`	□□□□	□	`"127.0.0.17"`	`SBI_ADDR`
`sbi_port`	□□□	□	`7777`	`SBI_PORT`
`sbi_certfile`	□□□□□	□	*(□□□)*	`SBI_CERT`
`sbi_keyfile`	□□□□□	□	*(□□□)*	`SBI_KEYF`
`nrf_uri`	□□□	□	`"http://127.0.0.1:7777"`	`NRF_URI`

□□	□□	□□	□□	□
	□			
`mcc`	□□□□	□	`"999"`	`MCC`
`mnc`	□□□□	□	`"70"`	`MNC`
`heartbeat_interval`	□□(□□)	□	`10000`	`HEARTBEA`
`prometheus_metrics_port`	□□	□	`9568`	`PROMETHE`
`discovery_cache_ttl`	□□(□□)	□	`60000`	`DISCOVER`
`lb_strategy`	□□	□	`round_robin`	`LB_STRAT`

参数	类型	必需	默认	[
`max_retries`	整数	否	`2`	`MAX_RETR`
`upstream_timeout`	整数(毫秒)	否	`5000`	`UPSTREAM`

NRF □置文件容量/优先级

这两个值是 SCP 自身 NRF 配置文件的一部分。它们不在 `runtime.exs` 中设置，也没有专用的环境变量，因此通常采用默认值，但可以通过 管理 API 进行运行时调节（`PATCH /api/oam/config/{id}`），这也会触发 NRF 重新注册，以便宣传新值。

参数	类型	必需	默认	环境变量	描述
`nf_capacity`	整数 (0-65535)	否	`100`	(无，OAM 仅)	在 SCP 的 NRF 配置文件中宣传的容量值。消费者/对等方可以在加权 SCP 时使用。
`nf_priority`	整数 (0-65535)	否	`0`	(无，OAM 仅)	在 SCP 的 NRF 配置文件中宣传的优先级值（值越低优先级越高）。

> 编译的回退默认值。应用程序还携带编译的回退默认值，仅在 `:omniscp` 键完全缺失时使用，这些值与上述 `runtime.exs` 环境默认值不同（例如 `sbi_addr` 回退为 `127.0.0.200`，`nrf_uri` 回退为 `http://127.0.0.10:7777`，`max_retries` 回退为 `1`）。在正常操作中，`runtime.exs` 提供每个键，因此上表中的值是有效的默认值。

负载均衡器和过载调节

这些 `:omniscp` 应用程序配置键控制基于故障的自动健康跟踪和过载控制剔除。它们没有环境变量，也不通过 OAM API 暴露，因此仅通过应用程序配置设置。每个键在运行时实时读取，因此对正在运行的应用程序配置的更改将在下一个请求路径使用时生效，无需重新编译。未设置时应用的默认值如下。

参数	类型	必需	默认	描述
`health_check_interval`	整数（毫秒）	否	`15000`	对生产者实例进行的后台健康/恢复清扫的间隔。
`failure_threshold`	整数	否	`3`	连续转发失败次数，超过此次数后实例将被标记为不健康并被移出轮换。
`recovery_cooldown_ms`	整数（毫秒）	否	`30000`	不健康实例在此冷却后可以重新进入轮换。
`oci_reduction_threshold`	整数	否	`50`	过载控制阈值，控制何时从候选集剔除过载生产者（根据其 `3gpp-Sbi-Oci` 减少指标）。请参见 操作 → 路由和发现。

SBI 监听器调节

SBI 代理通过 HTTP/2 提供服务。以下传输参数在应用程序中是固定的（不可通过环��调节），并在此记录，因为它们控制 SCP 在负载下的并发行为：

设置	值	描述
`max_concurrent_streams`	`5000`	每个入站连接的最大并发 HTTP/2 流。限制单个消费者连接通过代理复用的在途 SBI 请求数量。
`num_acceptors`	`100`	SBI 监听器的接受者池的大小，用于接受新的入站 TCP 连接。

对生产者的出站（上游）请求使用由公共 Omni5G 运行时提供的共享 Finch 连接池，在所有转发流量中重用，并按生产者主机进行键控。其池大小未作为 OmniSCP 设置暴露，因此上游并发性由该共享池以及 `upstream_timeout`（限制每个单独的转发请求）和上述入站 `max_concurrent_streams` 限制共同控制。要扩展繁忙的 SCP，您可以控制的杠杆是 `upstream_timeout`（较短的超时更快地将连接释放回池）和 `discovery_cache_ttl`（更高的命中率从请求路径中移除 NRF 往返）；`max_concurrent_streams` 限制单个消费者连接通过代理复用的数量。

管理 / OAM API

OmniSCP 通过 **HTTPS** 在 **8443** 端口提供管理/OAM API，用于只读状态和实时操作控制。这些端点不承载用户流量；请限制对操作网络的访问。以下所有路径都在管理监听器的 `/api` ��础路径下提供。

状态端点（只读）

方法	路径	描述
`GET`	`/api/status/api`	API 服务状态。
`GET`	`/api/status/license`	许可证状态。
`GET`	`/api/status/nrf`	SCP 的 NRF 注册状态。
`GET`	`/api/status/nf`	SCP NF 状态：SBI 地址/端口、发现缓存大小、活动 `lb_strategy` 和每个实例的负载均衡器健康。
`GET`	`/api/api/cache`	发现缓存内容：每个条目的目标 NF 类型、服务名称、实例计数和剩余 TTL。
`GET`	`/api/api/health`	负载均衡器健康：每个跟踪实例的健康标志和失败计数，以及健康/不健康的总数。
`GET`	`/api/api/statistics`	SCP 统计信息：发现缓存大小和跟踪/健康/不健康实例计数。
`GET`	`/api/oam/config`	当前有效的运行时配置（SBI、NRF、缓存 TTL、`lb_strategy`、`max_retries`、`upstream_timeout`、`nf_capacity`、`nf_priority`、日志级别）。
`GET`	`/api/oam/cache`	发现缓存视图（与 `/api/api/cache` 相同）。
`GET`	`/api/oam/upstream/drain`	列出当前被排除的上游 NF 实例。

控制端点

方法	路径	描述
`PATCH`	`/api/oam/config/{id}`	实时更新选定的运行时设置：`discovery_cache_ttl`、`lb_strategy`、`max_retries`、`upstream_timeout`、`nf_capacity`、`nf_priority`、`log_level`。值会被验证（`lb_strategy` 必须是 `round_robin`、`weighted`、`priority` 之一；整数键必须是整数）。更改 `nf_capacity`/`nf_priority` 还会触发 NRF 重新注册。
`DELETE`	`/api/oam/cache/{id}`	刷新特定 NF 类型的发现缓存，或传递 `all` 刷新整个缓存。
`POST`	`/api/oam/upstream/drain`	排除一个上游，以便代理停止选择它。主体：`{"nf_instance_id": "..."}`（别名 `"target"`），可选 `"reason"`。
`DELETE`	`/api/oam/upstream/drain/{id}`	通过 NF 实例 ID 取消排除一个上游，使其再次符合选择条件。
`POST`	`/api/oam/log_level`	在运行时更改活动日志级别。主体：`{"level": "..."}`（`emergency`、`alert`、`critical`、`error`、`warning`/`warn`、`notice`、`info`、`debug`）。
`POST`	`/api/oam/nrf/reregister`	强制 SCP 与 NRF 立即重新注册。

上游排除是管理状态，与基于故障的自动健康跟踪分开。被排除的实例在负载均衡器选择之前从候选列表中过滤，因此被排除的生产者永远不会接收转发流量。如果某个服务的每个候选者都被排除，代理将没有可用的下一跳，请求该服务将返回 `504 NF_DISCOVERY_FAILURE`（请参见 操作 → 代理错误响应 和 故障排除）。

日志记录

参数	类型	必需	默认	描述
`format`	元组	否	`{OmniLogger.JsonFormatter, :format}`	通过共享的 `OmniLogger` 格式化程序将日志行格式化为结构化 JSON，适合被日志管道摄取。
`metadata`	原子 / 列表	否	`:all`	要发出的日志元数据。`:all` 包括附加到日志条目的每个元数据键。

活动日志级别可以通过 管理 API 在运行时更改而无需重启，使用 `POST /api/oam/log_level` 或 `PATCH /api/oam/config/{id}` 以及 `log_level` 值。

← 概述

OmniSCP 指标

指标通过 Prometheus 监听器（`prometheus_metrics_port`，默认 `9568`）在 `/metrics` 上公开。代理指标的结果标签值为：`success`（2xx/3xx）、`client_error`（4xx）、`server_error`（5xx）、`error`（连接/超时）。

代理指标

指标：`omni_scp_proxy_requests_count`

类型：计数器

描述：每个请求的代理结果计数。

标签：`target_nf_type`，`result`

指标：`omni_scp_proxy_requests_duration_ms`

类型：摘要

描述：每个请求的代理持续时间（以毫秒为单位）。

标签：`target_nf_type`

指标：`omni_scp_proxy_requests_total`

类型：计数器

描述：按目标 NF 类型和结果的总代理请求数。

标签：`target_nf_type`，`result`

指标：`omni_scp_proxy_request_duration_ms`

类型：直方图（分布）

描述：代理请求持续时间。桶：1、5、10、25、50、100、250、500、1000 ms。

标签：`target_nf_type`

SCP 是一个无状态的转发代理，不保持每个关联的状态，因此不公开“活动关联”指标。通过上述代理请求计数器（速率、错误比例）和 OAM API 上的负载均衡器健康数据观察实时活动。

```
# 目标 NF 类型的代理请求速率
sum by (target_nf_type) (rate(omni_scp_proxy_requests_total[5m]))

# 所有生产者的服务器错误比例
sum(rate(omni_scp_proxy_requests_total{result="server_error"}
[5m]))
  / sum(rate(omni_scp_proxy_requests_total[5m]))

# NF 类型的 p95 代理延迟
histogram_quantile(0.95, sum by (le, target_nf_type)
(rate(omni_scp_proxy_request_duration_ms_bucket[5m])))
```

发现缓存指标

指标：`omni_scp_discovery_cache_hits`

类型：计数器

描述：每个服务的发现缓存命中数。

标签：`target_nf_type`，`service_name`

指标：`omni_scp_discovery_cache_misses`

类型：计数器

描述：每个服务的发现缓存未命中数。

标签：`target_nf_type`，`service_name`

指标：`omni_scp_cache_hits_total`

类型：计数器

描述：聚合发现缓存命中数。

标签：无

指标：`omni_scp_cache_misses_total`

类型：计数器

描述：聚合发现缓存未命中数。

标签：无

```
# 整缓存命中率（目标 > 0.9 在稳定拓扑中）
sum(rate(omni_scp_cache_hits_total[5m]))
  / (sum(rate(omni_scp_cache_hits_total[5m])) +
sum(rate(omni_scp_cache_misses_total[5m])))

# 每个服务的未命中率（高值意味着频繁的 NRF 重新发现）
sum by (target_nf_type, service_name)
(rate(omni_scp_discovery_cache_misses[5m]))
```

NRF 注册指标

指标：`omni_scp_nrf_registration_status`

类型：计量器

描述：SCP 与 NRF 的注册状态（`1` = 已注册，`0` = 未注册）。

标签：`nf_type`

```
# 如果 SCP 失去 NRF 注册则发出警报
min(omni_scp_nrf_registration_status) < 1
```

BEAM VM 指标

运行时还导出主机/VM 计量器：`beam_memory_total`，`beam_memory_processes`，`beam_memory_processes_used`，`beam_memory_system`，`beam_memory_atom`，`beam_memory_atom_used`，`beam_memory_binary`，`beam_memory_code`，`beam_memory_ets`（所有字节），以及 `beam_processes_count`，`beam_ports_count`，`beam_atom_count`，`beam_vm_uptime`（秒）。这些对于 SCP 进程本身的容量和泄漏监控非常有用。

> 上述指标名称使用 Prometheus 点到下划线的规范化；直方图系列公开通常的 `_bucket`、`_sum` 和 `_count` 后缀。

← 述

OmniSCP 操作

本指南涵盖了SCP的3GPP角色、它所暴露的接口和端点、其请求路由和发现行为，以及关键的运行时程序和序列图。有关每个可调设置，请参见配置；有关指标，请参见指标；有关故障处理，请参见故障排除。

3GPP角色和规范参考

SCP提供间接通信，如TS 23.501中所定义。在模型**C**中，消费者自行执行发现（通过NRF），但将请求转发委托给SCP。在模型**D**中，消费者将发现和转发都委托给SCP：它使用`3gpp-Sbi-Discovery-*`参数发送请求，SCP代表其执行NRF发现。OmniSCP支持这两种模型，以及当消费者已经知道目标时的直接转发。

规范	节	相关性
TS 23.501	§6.3.1, 附录E	SCP角色和5G SBA中的间接通信模型（C和D）。
TS 23.501	§7.3	SCP在SBA中的请求处理行为。
TS 29.500	§6.10	通过SCP的SBI��由；`3gpp-Sbi-*`路由/发现头。
TS 29.500	§6.10.3	在委托发现请求中携带的`3gpp-Sbi-Discovery-*`头集。
TS 29.500	§6.10.3.2	`3gpp-Sbi-Target-apiRoot`头（直接路由目标）；主要服务名称选择。
TS 29.500	§6.10.3.3	`3gpp-Sbi-Producer-Id`响应头（消费者与生产者绑定）。
TS 29.500	§5.2.2.2	`User-Agent`头传达请求者的NF类型。
TS 29.510	§6.2	Nnrf_NFDiscovery，SCP使用的NRF发现服务。
TS 29.510	§6.1	Nnrf_NFManagement，SCP自注册和NF状态通知。

参考：TS 23.501，TS 29.500，TS 29.510。

接口和监听器

OmniSCP是一个通用的HTTP反向代理。NF消费者将其SBI流量指向SCP的SBI监听器，而不是直接指向生产者。对于每个请求，SCP解析目标生产者（从请求头、缓存的NRF发现结果或通过查询NRF），通过配置的负载均衡策略选择一个实例，转发请求，并在失败时重试不同的实例。

监听器	默认端口	协议	目的
SBI代理	`7777` (`sbi_port`)	HTTP / HTTPS (`sbi_scheme`)	接受来自消费者的SBI请求知。所有非通知路径都被代[`sbi_scheme=ht` `sbi_certfile/` 以在监听器上终止TLS。
Prometheus 指标	`9568` (`prometheus_metrics_port`)	HTTP	提供 `/metrics` 抓取[
管理/OAM API	`8443`	HTTPS	只读状态和OAM控制端点 API。

SBI监听器通过HTTP/2提供，具有每连接流限制和固定的接收池；这些传输调优值在配置 → SBI监听器调优中描述。

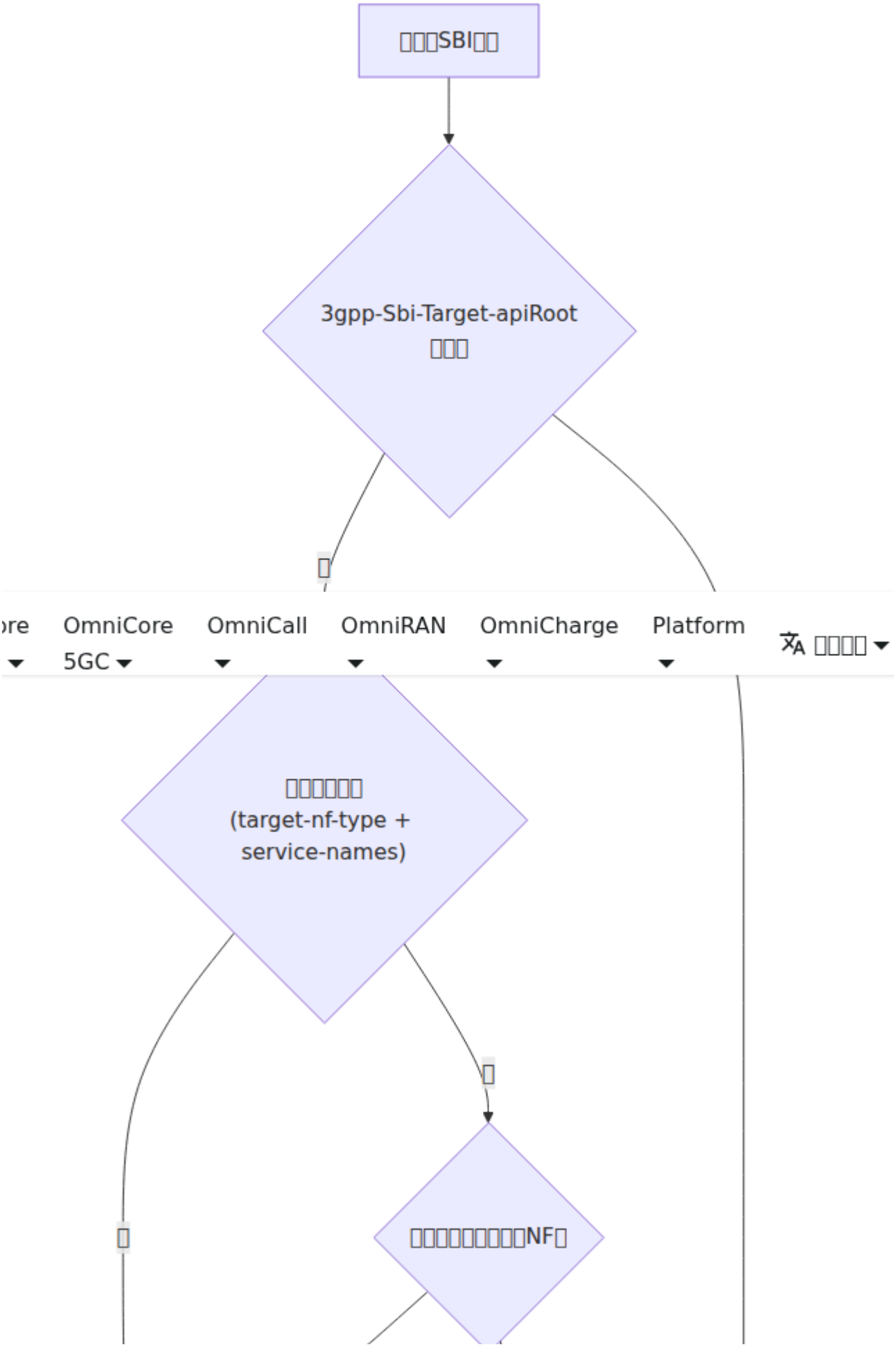
SBI端点表

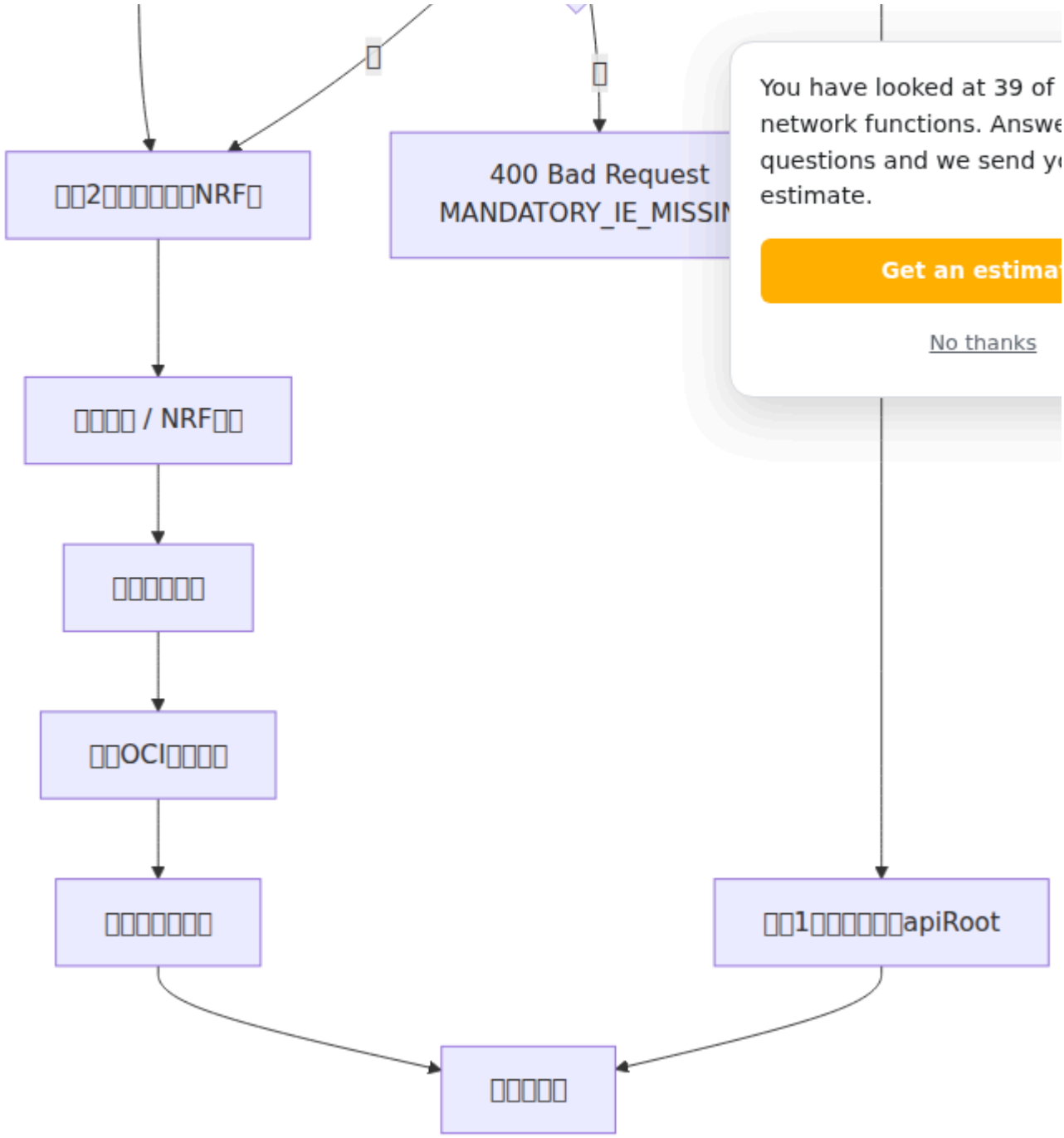
OmniSCP是透明的：确切地说，只有一个路径在本地处理，所有其他请求都被代理。

方法	路径	本地处理	描述
`POST`	`/nnrf-nfm/v1/nf-status-notify`	是	接收NRF NF状态通知。在`NF_DEREGISTERED`或`NF_PROFILE_CHANGED`时，发现缓存会因其`nfInstanceId`标识的单个受影响实例而失效（如果没有实例ID，则回退到仅驱逐一个NF类型，如果两者都不存在则不执行任何操作）。整个缓存不会被清除。返回`204 No Content`。
`*`	所有其他路径	否（代理）	根据活动路由模式转发到解析的生产者。

路由和发现

在每个传入请求中，SCP检查请求头和路径以决定如何解析目标生产者���按优先顺序评估三种路由模式：


□□□SBI□□
3gpp-Sbi-Target-apiRoot
□□□
□
ore
OmniCore
5GC
OmniCall
OmniRAN
OmniCharge
Platform
□□□□
□□□□□□□
(target-nf-type +
service-names)
□
□
□□□□□□□□□□NF□



模式**1**：直接转发

当消费者发送`3gpp-Sbi-Target-apiRoot`时，它已经知道生产者的API根。SCP剥离SCP特定的头并直接将请求转发到该基本URI，不进行NRF查找、缓存或负载均衡。

模式**2**：委托发现（模型**C/D**）

当消费者发送`3gpp-Sbi-Discovery-target-nf-type`和`3gpp-Sbi-Discovery-service-names`时，SCP自行解析生产者集：

1. 缓存查找：以`{target NF type, primary service name}`为键。如果在TTL内命中，使用缓存的实例列表，不进行NRF查询。
2. **NRF**发现：在未命中时，SCP查询NRF（Nnrf_NFDiscovery），传递其接收到的发现参数：请求者NF���型、目标PLMN列表、请求者S-NSSAI列表、NF集ID和目标NF实例ID。结果在`discovery_cache_ttl`下使用相同的键缓存。
3. 排水过滤：在选择之前，任何其`nfInstanceId`被行政排水的实例（请参见配置 → 管理API）会从候选集中移除。
4. 过载控制剔除：任何当前通过其`3gpp-Sbi-Oci`（过载控制信息）响应头信号过载的实例，其有效期内达到或超过`oci_reduction_threshold`，会从候选集中移除（TS 29.500 §6.3）。这是尽力而为：如果每个候选都过载，则保持完整集合，而不是丢弃请求。
5. 负载均衡器选择：从剩余候选中使用`lb_strategy`选择一个实例。
6. 转发并重试：请求被转发；在失败时尝试不同的实例，最多重试`max_retries`。在每个响应中，记录生产者广告的`3gpp-Sbi-Oci`，以便后续请求可以从中剔除负载。

模式3：基于路径的推断

当没有路由头时，SCP从请求路径的服务名称前缀推断目标NF类型（路径格式`/<service-name>/<version>/...`），然后完全按照模式2进行处理。

路径前缀	**NF**类型
`nudm-`	UDM
`nausf-`	AUSF
`namf-`	AMF
`nsmf-`	SMF
`npcf-`	PCF
`nudr-`	UDR
`nnssf-`	NSSF
`nbsf-`	BSF
`nnrf-`	NRF
`nchf-`	CHF
`nnef-`	NEF
`naf-`	AF

如果路径前缀不在此表中，SCP返回`400 Bad Request`，原因是`MANDATORY_IE_MISSING`。

负载均衡策略

选择策略由`lb_strategy`设置，适用于模式2/3（发现的实例集）：

策略	选择逻辑
`round_robin`	按顺序循环通过健康的候选实例。每个 `{nfType, serviceName}` 组保持一个单独的轮换计数器，以便不同服务不会相互干扰。这是默认值。
`weighted`	从其NRF配置中选择具有最佳（最低）`load - capacity` 分数的实例，优先选择高容量、低负载的生产者。缺失的 `load` 默认为50，缺失的 `capacity` 默认为100。
`priority`	从其NRF配置中选择具有最低 `priority` 值（最高优先级）的实例。缺失的 `priority` 默认为65535。适合主动/备用部署。

> 权重来源。 `weighted` 和 `priority` 策略从每个生产者自己的**NRF**置中读取 `load` 、 `capacity` 和 `priority` ；SCP没有自己的每个生产者权重配置。要偏向选择，您需要在生产者（或NRF）上调整这些值，而不是在SCP上。SCP的 `nf_capacity` / `nf_priority` 设置（配置 → NRF配置容量/优先级）是个单独的念：它们描述 *SCP* 在其自己的配置中，以便对等方可以对SCP进行加权，而对SCP如何加权生产者没有影响。

基于故障的健康跟踪。不论策略如何，负载均衡器跟踪每个实例的结果。一个实例在连续**3**次失败后被标记为不健康，并从选择中排除；每**15**秒运行一次后台健康扫描，一旦自上次失败以来经过**30**秒冷却时间，该实例将恢复服务。如果每个候选都不健康，SCP将回退到完整列表，而不是直接失败（优先考虑可用性而不是严格排除）。这种自动健康跟踪与运营商驱动的排水状态不同。

这三个阈值（失败计数、扫描间隔和冷却时间）未作为环境变量暴露，默认为上述值，但它们是实时读取的真实 `:omniscp` 应用程序配置键（ `failure_threshold` 、 `health_check_interval` 、 `recovery_cooldown_ms` ）。没有OAM端点在运行时更改它们。请参见配置 → 负载均衡器和过载可调设置。要按计划将生产者排除在轮换之外（而不是等待失败阈值），请使用行政排水。此目的的失败是5xx响应或连接/超时错误；任何状态低于500的响应（包括4xx）都计为成功并重置实例的失败计数。

SBI / 代理行为

消耗的头（并在转发前剥离）

这些控制路由，永远不会转发给生产者。

头	目的
`3gpp-Sbi-Target-apiRoot`	模式1直接路由目标。
`3gpp-Sbi-Discovery-target-nf-type`	要发现的NF类型（例如`UDM`）。
`3gpp-Sbi-Discovery-service-names`	以逗号分隔的服务名称；第一个是主要服务（用作缓存键和NRF查询服务）。
`3gpp-Sbi-Discovery-requester-nf-type`	请求者NF类型；限制NRF查询的范围。
`3gpp-Sbi-Discovery-target-plmn-list`	目标PLMN列表；传递给NRF查询。
`3gpp-Sbi-Discovery-requester-snssai-list`	请求者S-NSSAI列表；传递给NRF查询。
`3gpp-Sbi-Discovery-nf-set-id`	NF集ID过滤器；传递给NRF查询。
`3gpp-Sbi-Discovery-target-nf-instance-id`	特定目标NF实例ID；传递给NRF查询。
`3gpp-Sbi-Discovery-requester-nf-instance-id`	请求者实例ID。被消耗和剥离，但当前不包含在NRF发现查询中。

`User-Agent`头也会被检查：根据TS 29.500 §5.2.2.2，它以请求者的NF类型（例如`AMF-...`）开头，如果没有显式的发现头提供，则用作请求者NF类型。如果两者都不存在，SCP将请求者NF类型默认为`SCP`以进行NRF查询。

生成的头

头	目的
`3gpp-Sbi-Producer-Id`	添加到响应中，携带服务请求的生产者的`nfInstanceId`，以便消费者可以将后续请求绑定到同一生产者（TS 29.500 §6.10.3.3）。
`3gpp-Sbi-Lci`	生产者负载控制信息，从生产者的响应中端到端转发，以便消费者通过SCP接收其负载提示（TS 29.500 §6.3）。SCP转发生产者值，而不合成自己的值。
`3gpp-Sbi-Load`	`3gpp-Sbi-Lci`负载伴随头，从生产者的响应中端到端转发，连同`3gpp-Sbi-Lci`一起（TS 29.500 §6.3）。

逐跳响应头（`transfer-encoding`、`connection`、`content-length`）不会转发回消费者。

代理错误响应

当SCP无法完成操作时，它会根据TS 29.500返回RFC 7807 `ProblemDetails`主体。

HTTP状态	原因	条件
`400 Bad Request`	`MANDATORY_IE_MISSING`	没有路由信息：没有目标-apiRoot，没有发现头，并且路径前缀映射到没有已知服务。
`502 Bad Gateway`	`TARGET_NF_NOT_REACHABLE`	无法从发现的NF配置中解析出SBI URI。
`504 Gateway Timeout`	`NF_DISCOVERY_FAILURE`	发现（NRF + 缓存）未产生可用的NF实例，包括当每个服务的候选实例都被行政排水时。
`500 Internal Server Error`	`SYSTEM_FAILURE`	实际上未预期的内部代���错误，未被上述更具体的原因覆盖。

> 关于所有排水情况的说明。如果服务的每个候选都被排水，代理没有可用的下一跳，返回`504 NF_DISCOVERY_FAILURE`（而不是`500`）。至少排水一个实例以恢复服务。请参见故障排除。

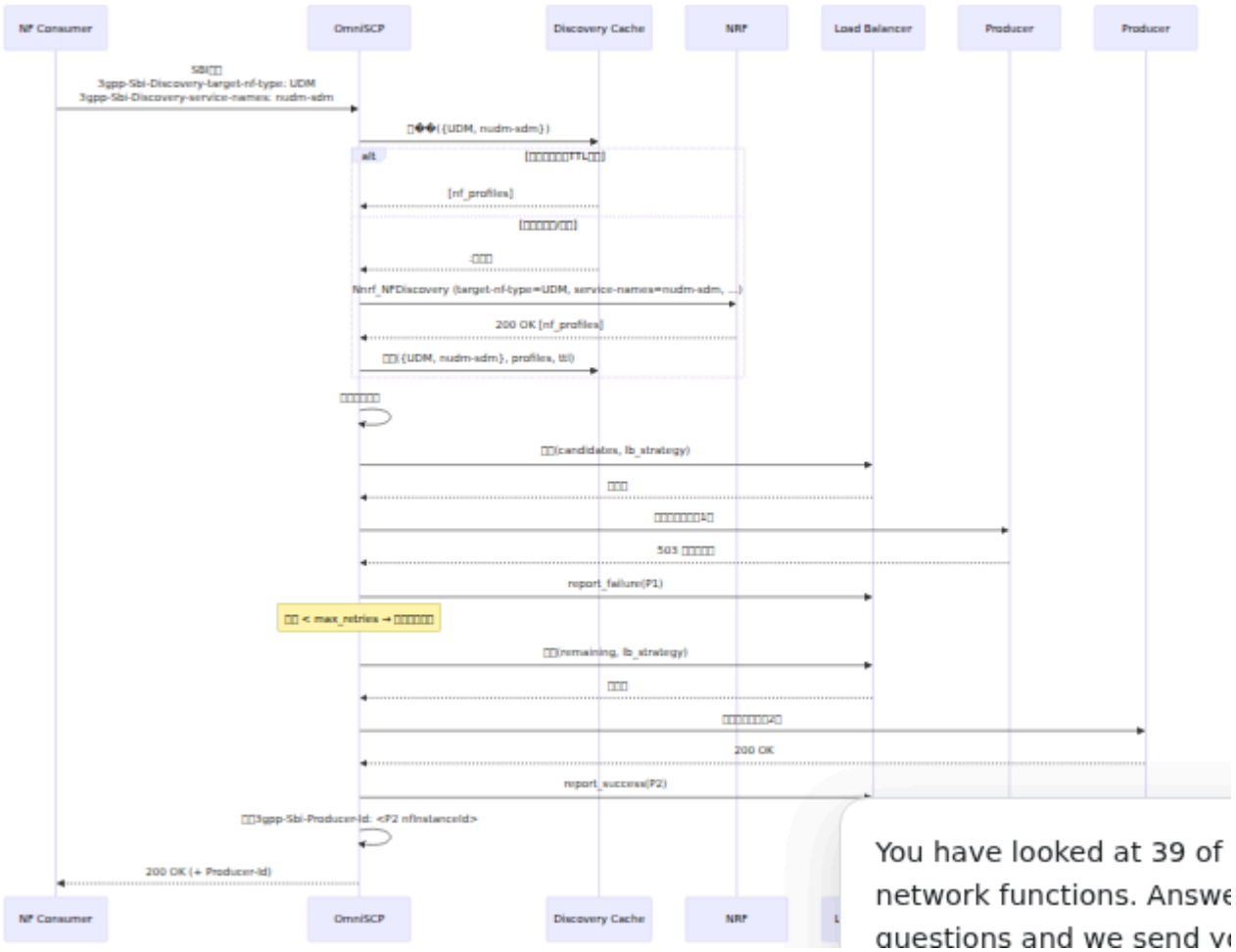
重试语义。生产者响应状态**< 500**（包括4xx）被视为有效答案，并原样返回给消费者。**5xx**响应或连接**/**超时错误计为失败：失败的实例报告给负载均衡器并从候选集中移除，尝试不同的实例。重试继续，直到获得非失败响应、达到`max_retries`或候选列表耗尽，此时返回最后的响应（或错误）。

关键程序

代理的、负载均衡的请求与重试（模式**2/3**）

下面的序列显示了委托发现、负载均衡器选择、第一次尝试失败（返回5xx）以及对第二个实例的成功重试。

NF Consumer
OmniSCP
Discovery Cache
NRF
Load Balancer
Producer
Producer
SBI请求
3gpp-Sbi-Discovery-target-nf-type: UDM
3gpp-Sbi-Discovery-service-names: nudm-sdm
查��({UDM, nudm-sdm})
[缓存命中（在TTL内）]
[nf_profiles]
[缓存未命中/过期]
:未命中
Nnrf_NFDiscovery (target-nf-type=UDM, service-names=nudm-sdm, ...)
200 OK [nf_profiles]
存储({UDM, nudm-sdm}, profiles, ttl)
移除排水实例
选择(candidates, lb_strategy)
生产者
转发请求（尝试1）
503 服务不可用
report_failure(P1)
尝试 < max_retries → 重试剩余实例
选择(remaining, lb_strategy)
生产者
转发请求（尝试2）
200 OK
report_success(P2)
添加3gpp-Sbi-Producer-Id: <P2 nfInstanceId>
200 OK (+ Producer-Id)
NF Consumer
OmniSCP
Discovery Cache
NRF
Load Balancer
Producer
Producer



直接转发（模式**1**）

You have looked at 39 of
network functions. Answe
questions and we send v

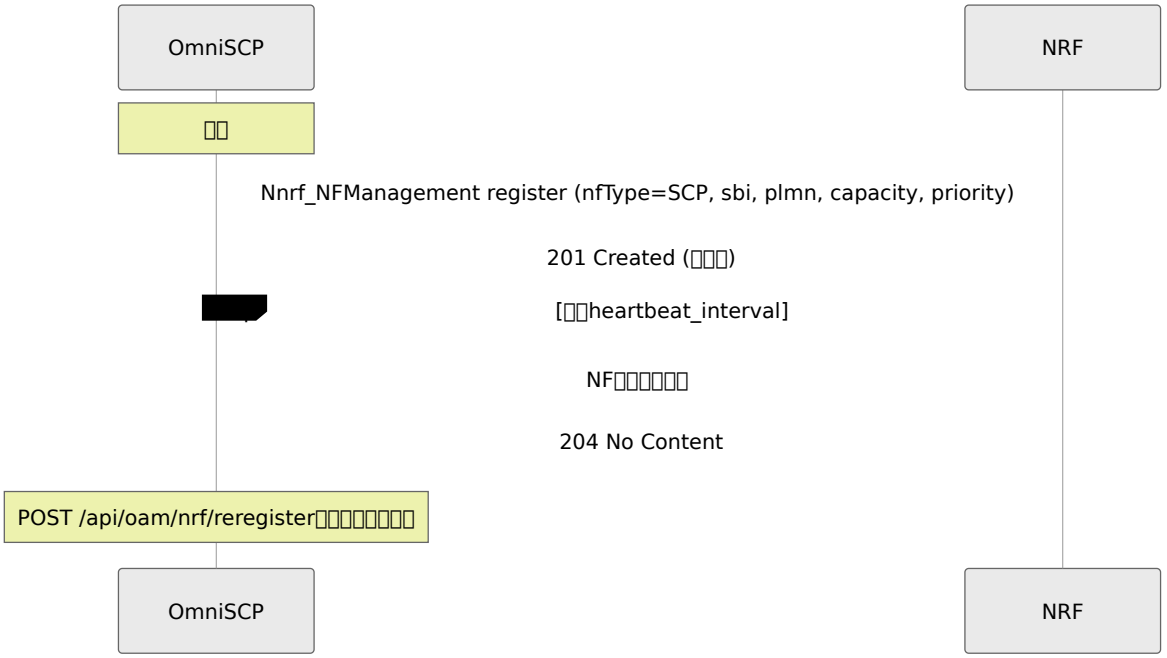


NRF状态通知（缓存失效）

NRF
OmniSCP
Discovery Cache
POST /nnrf-nfm/v1/nf-status-notify
{event, nfInstanceUri, nfProfile.nfInstanceId, nfProfile.nfType}
[event = NF_DEREGISTERED或NF_PROFILE_CHANGED]
[nfInstanceId存在（或可从nfInstanceUri推导）]
驱逐该单个实例
[实例ID缺失，nfType存在]
驱逐该NF类型的条目
[两者都缺失]
无缓存操作（整个缓存从未清除）
[其他事件]
无缓存操作
204 No Content
NRF
OmniSCP
Discovery Cache

NRF自注册和心跳

SCP在启动时作为`SCP` NF向NRF注册，并通过定期心跳保持注册。广告的配置携带SBI地址/端口/协议、PLMN（`mcc`/`mnc`）和NF `capacity`/`priority`。可以通过OAM API强制立即注册。

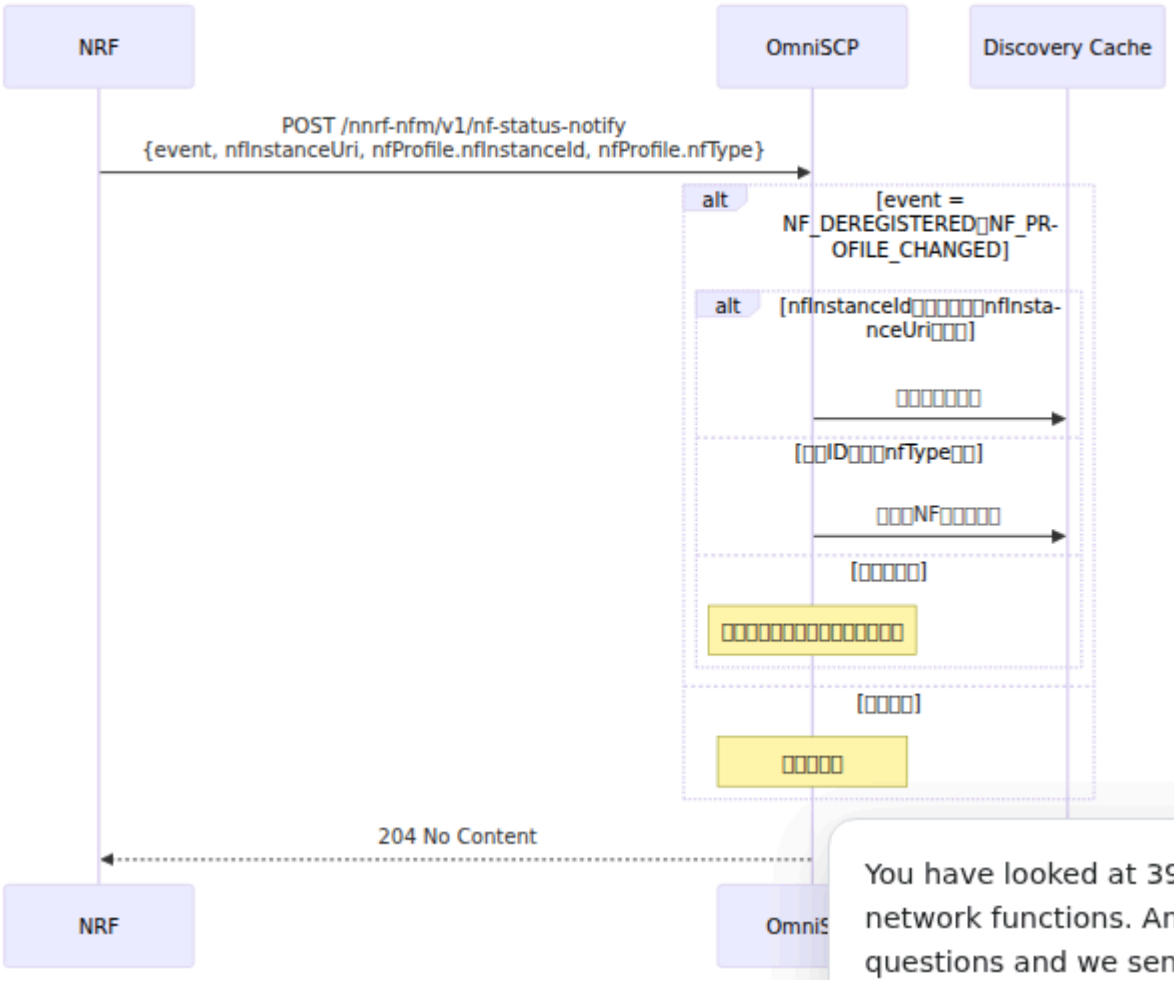


[redacted]

[redacted]

[redacted] [redacted]NRF[redacted] `nfInstanceId` [redacted]ID[redacted] `reason` [redacted] `POST /api/oam/upstream/drain` [redacted] [redacted] `GET /api/oam/upstream/drain` [redacted] `DELETE /api/oam/upstream/{id}` [redacted]

> [redacted] [redacted]ETS[redacted]**SCP**[redacted]SCP[redacted]
> [redacted]SCP[redacted] `GET /api/oam/upstream/drain` [redacted]



You have looked at 39 of
network functions. Answe
questions and we send y

← 概述

OmniSCP 故障排除

此处提到的错误原因和状态代码在 操作 → 代理错误响应 中进行了描述。配置键和 OAM 端点在 配置 中；指标在 指标 中。

400 错误请求：MANDATORY_IE_MISSING

症状：消费者收到 `400`，原因是 `MANDATORY_IE_MISSING`。

可能原因：

- 请求中既没有携带 `3gpp-Sbi-Target-apiRoot`，也没有携带 `3gpp-Sbi-Discovery-target-nf-type` + `3gpp-Sbi-Discovery-service-names` 对。
- 依赖于基于路径的推断（模式 3）来获取不在内置映射中的服务前缀。

解决方案：

1. 确认消费者发送了 target-apiRoot 或两个发现头。
2. 如果依赖于路径推断，请验证路径前缀是否在 模式 3 表 中。对于不在该表中的前缀，请发送显式发现头。

504 网关超时：NF_DISCOVERY_FAILURE

症状：消费者收到 `504`，原因是 `NF_DISCOVERY_FAILURE`。

可能原因：

- NRF 对请求的 NF 类型/服务未返回任何实例。
- NRF 无法访问。

解决方案：

1. 验证 `nrf_uri` 和到 NRF 的网络可达性。
2. 确认目标 NF 类型已在 NRF 中注册。
3. 检查最近的 NRF 状态通知是否使该 NF 类型的缓存失效，并且生产者尚未重新注册。

504 网关超时：NF_DISCOVERY_FAILURE（所有实例已排空）

症状：消费者收到 `504`，原因是 `NF_DISCOVERY_FAILURE`，而该服务有注册的生产者。

可能原因：

- 服务的每个候选实例都被管理性排空，因此代理没有可用的下一跳。此条件映射到 `504 NF_DISCOVERY_FAILURE`，与发现未返回实例的状态相同。

解决方案：

1. 检查 `GET /api/oam/upstream/drain`。如果该服务的所有实例都已排空，通过 `DELETE /api/oam/upstream/drain/{id}` 取消排空所需的实例。
2. 如果没有实例被排空且服务已注册，则将其视为发现失败案例（请参见上面的 NF_DISCOVERY_FAILURE 部分）。

500 内部服务器错误：SYSTEM_FAILURE

症状：消费者收到 `500`，原因是 `SYSTEM_FAILURE`。

可能原因：

- 代理内部出现了真正意外的错误，未被上述更具体的原因（`400`、`502`、`504`）覆盖。

解决方案：

1. 查看 SCP 日志以获取内部错误详细信息。

502 错误网关：TARGET_NF_NOT_REACHABLE

症状：消费者收到 `502`，原因是 `TARGET_NF_NOT_REACHABLE`。

可能原因：

- 无法从发现的 NF 配置文件中解析出任何 SBI URI。

- 所有选定的生产者在重试后返回连接错误。

解决方案：

1. 确认生产者正在运行并且可以通过其 NRF 配置文件中的 SBI 地址访问。
2. 如果生产者响应缓慢，请增加 `upstream_timeout`。
3. 查看 `max_retries`；`0` 会将单次失败转变为立即错误。
4. 检查 `GET /api/api/health`，查看在多次失败后标记为不健康的实例。

上游超时 / 高代理延迟

症状：`omni_scp_proxy_request_duration_ms` 升高；请求间歇性失败，结果为 `error`。

可能原因：

- 生产者响应慢于 `upstream_timeout`。
- 低缓存命中率迫使许多请求进行 NRF 往返。

解决方案：

1. 检查延迟直方图和每种 NF 类型的错误率。
2. 比较 `omni_scp_cache_hits_total` 与 `omni_scp_cache_misses_total`；高未命中率意味着频繁的 NRF 发现，因此考虑增加 `discovery_cache_ttl`。
3. 调整 `upstream_timeout` 以匹配实际的生产者响应时间。请记住，���时的尝试会在重试之前将整个超时添加到端到端延迟中。

陈旧路由 / 发现缓存陈旧

症状：请求被转发到已移动或重启的生产者，导致错误，直到条目过期。

可能原因：

- 生产者更改了地址或重启而未从 NRF 注销，因此 SCP 保持陈旧的配置文件，直到 TTL 过期。

解决方案：

1. 减少 `discovery_cache_ttl` 以缩短陈旧窗口。

2. 确保生产者在关闭时从 NRF 注销，以便 SCP 接收 `NF_DEREGISTERED` 通知并使受影响的 NF 类型失效。
3. 通过 `DELETE /api/oam/cache/{nf_type}`（或 `DELETE /api/oam/cache/all`）显式刷新。

NRF 注册未维护

症状：`omni_scp_nrf_registration_status` 显示为 `0`。

可能原因：

- 不正确的 `nrf_uri` 或 NRF 无法访问。
- PLMN 与 NRF 配置不匹配（`mcc`/`mnc`）。

解决方案：

1. 验证 `nrf_uri` 和连接性。
2. 确认 `mcc`/`mnc` 与 NRF PLMN 配置匹配。
3. 使用 `POST /api/oam/nrf/reregister` 强制重新注册，并查看启动日志。