

OmniNWDAF □置参考

□述 | 操作 | 指标 | 故障排除

本参考文档记录了每个 OmniNWDAF 配置参数。所有参数都位于 `:omninwdaf` 应用程序范围内。环境变量仅适用于生产运行时配置（`runtime.exs`）；仅文件调优的参数没有环境变量，必须在应用程序配置文件中设置。

□置结构

完整置示例

```
config :omninwdaf,
  # SBI binding and advertised endpoint
  sbi_scheme: "http",
  sbi_addr: "127.0.0.23",
  sbi_port: 7777,

  # NRF registration target and heartbeat cadence
  nrf_uri: "http://127.0.0.10:7777",
  heartbeat_interval: 10_000,

  # Served PLMN (override the 999-70 test default per deployment)
  mcc: "999",
  mnc: "70",

  # Validity window applied to returned analytics expiry (seconds)
  analytics_validity_seconds: 3600,

  # Notification delivery retry tuning
  notification_max_attempts: 4,
  notification_backoff_ms: 500,

  # Durable subscription store file (defaults under the system
temp directory)
  subscription_store_path: "/var/lib/omninwdaf/subscriptions.dets"
```

生产环境的 `runtime.exs` 从环境中读取环境支持的子集，回退到以下默认值：

```
config :omninwdaf,
  sbi_scheme: System.get_env("NWDAF_SBI_SCHEME", "http"),
  sbi_addr: System.get_env("NWDAF_SBI_ADDR", "127.0.0.23"),
  sbi_port: String.to_integer(System.get_env("NWDAF_SBI_PORT",
"7777")),
  nrf_uri: System.get_env("NWDAF_NRF_URI",
"http://127.0.0.10:7777"),
  mcc: System.get_env("NWDAF_MCC", "999"),
  mnc: System.get_env("NWDAF_MNC", "70")
```

SBI 绑定参数

这些参数控制 OmniNWDAF 为其 SBI 服务绑定的地址和端口，以及它向 NRF 和费者广告的方案。

参数	类型	必需	默认	环境变量	描述
`sbi_scheme`	字符串	否	`http`	`NWDAF_SBI_SCHEME`	SBI 端点和广告 NF 配置文件的 URL 方案。
`sbi_addr`	字符串	否	`127.0.0.23`	`NWDAF_SBI_ADDR`	OmniNWDAF 为其 SBI 服务绑定的 IP 地址，并向 NRF 和消费者广告。
`sbi_port`	整数	否	`7777`	`NWDAF_SBI_PORT`	OmniNWDAF 监听 SBI 请求的 TCP 端口。

NRF 参数

这些参数控制与 NRF 的注册和心跳频率。请参见 操作: NRF 注册和心跳。NRF 交互遵循 TS 29.510。

参数	类型	必需	默认	环境变量
`nrf_uri`	字符串	否	`http://127.0.0.10:7777`	`NWDAF_NRF_URI`
`heartbeat_interval`	整数（毫秒）	否	`10000`	（无）

服务 **PLMN** 参数

这些参数标识 OmniNWDAF 在其 NF 配置文件中广告的 PLMN。

参数	类型	必需	默认	环境变量	描述
`mcc`	字符串	否	`999`	`NWDAF_MCC`	服务 PLMN 的移动国家代码。默认的 `999` 是一个测试值：每次部署时覆盖。
`mnc`	字符串	否	`70`	`NWDAF_MNC`	服务 PLMN 的移动网络代码。默认的 `70` 是一个测试值：每次部署时覆盖。

分析参数

参数	类型	必需	默认	环境变量	描述
`analytics_validity_seconds`	整数（秒）	否	`3600`	（无）	应用于返回 AnalyticsData 的 `expiry` 时间戳的有效性窗口：`expiry = now + this value`。仅限配置文件；没有环境变量。

通知重试参数

这些参数调整每个事件订阅通知（创建时间、定期和修改后）的尽力而为交付。请参见 操作：通知交付和重试。

参数	类型	必需	默认	环境变量	描述
`notification_max_attempts`	整数	否	`4`	（无）	每个通知的最大交付尝试次数，超过此次数将被丢弃。仅限配置文件；没有环境变量。
`notification_backoff_ms`	整数（毫秒）	否	`500`	（无）	重试前的基本退避时间，单位为毫秒。延迟会随着每次尝试呈指数增长。仅限配置文件；没有环境变量。

订阅存储参数

这些参数控制事件订阅的持久存储位置。订阅保存在热内存 ETS 表中，并镜像到 DETS 文件中，在重启时重新加载。请参见 操作: Nnwdaf_EventsSubscription。

参数	类型	必需	默认	环境[量
`subscription_store_path`	字符串（路径）	否	`omninwdaf_subscriptions.dets` 在系统临时目录下	（无[

关于仅文件参数的说明

`heartbeat_interval`、`analytics_validity_seconds`、`notification_max_attempts`、`notification_backoff_ms` 和 `subscription_store_path` 没有环境变量，并在应用程序配置文件中设置。

OmniNWDAF 指标与监控

□述 | 操作 | 配置 | 故障排除

可观察性：结构化日志

可观察性通过 结构化应用日志 和标准请求日志提供。操作员应监控 OmniNWDAF 日志以获取以下信号。

订阅生命周期

日志条目涵盖事件订阅生命周期：

- 订阅的创建及其分配的订阅 ID。
- 启动时持久订阅的重新加载，包括从存储文件恢复的数量。
- 拒绝无效订阅（缺少 `notificationURI` 或没有支持的事件）。
- 订阅的删除，包括对未知 ID 的删除尝试。

通知投递

日志条目涵盖每个通知的投递（创建时间、定期和修改后；参见 操作：通知投递和重试）：

- 每次对订阅者 `notificationURI` 的投递尝试。
- 每次重试，遵循有界指数退避计划。
- 最后一次尝试后的放弃，记录为 警告。

请求日志

标准 HTTP 请求日志涵盖 SBI 端点，包括 AnalyticsInfo 检索、订阅的创建、读取、修改和删除调用，以及存活状态端点。使用此日志确认消费者是否到达 OmniNWDAF，并观察响应状态码，例如 `400 MANDATORY_QUERY_PARAM_MISSING`、`404 UNSUPPORTED_EVENT` 和 `400 MANDATORY_IE_MISSING`。

推荐的监控方法

在添加 NF 特定指标之前，通过日志和存活端点监控 OmniNWDAF：

信号	位置	告诉你什么
存活性	`GET /nnwdaf-analyticsinfo/v1/status` 返回 `200 {"status":"UP"}`	服务正在运行并处理 SBI 请求。
NRF 注册	启动时的日志	注册到 NRF 是否成功。
通知放弃	最后投递尝试后的警告日志	无法到达的订阅者的 `notificationURI`。
请求错误	状态为 `4xx` 的请求日志	格式错误或不支持的分析请求和订阅。

健康检查

存活端点是推荐的就绪和存活探测目标：

- 端点：`GET /nnwdaf-analyticsinfo/v1/status`
- 健康响应：`200`，主体为 `{"status":"UP"}`

健康状态确认 SBI 监听器正在处理请求。它本身并不能确认 NRF 注册是最新的：请检查启动和心跳日志以获取该信息。

OmniNWDAF 操作指南

□述 | 配置 | 指标 | 故障排除

本指南描述了 OmniNWDAF 在运行时的行为以及如何操作它。OmniNWDAF 是 5G 核心网络数据分析功能（3GPP TS 23.501 §6.2.18, TS 23.288），暴露了 TS 29.520 的 Nnwdaf 服务。

目录

- 分析覆盖
- 架构□述
- 服务端点
- NRF 注册和心跳
- Nnwdaf_AnalyticsInfo: 请求 / 响应
- Nnwdaf_EventsSubscription: 订阅 / 通知
- 通知交付和重试
- 分析事件 ID 参考
- 操作行为总结

分析覆盖

负载导向事件是从 OmniNWDAF 从 NRF 发现的实时数据计算得出的，即每个 NF 的注册状态及其报告的 `load`（0-100，TS 29.510 NFProfile）：

- `NF_LOAD` - 每个发现的 NF 一个 `NfLoadLevelInformation` 条目，携带其报告的平均和峰值负载及其状态。当提供单个 `nfType` 过滤器时，仅发现该类型。
- `SLICE_LOAD_LEVEL` 和 `NSI_LOAD_LEVEL` - 服务于请求的 S-NSSAI 的 NFs 的平均报告负载。
- `NETWORK_PERFORMANCE` - 网络范围的汇总：注册 NFs 的比例（`GNB_ACTIVE_RATIO`）和平均报告负载（`GNB_COMPUTING_USAGE`, `GNB_MEMORY_USAGE`）。

对于这些事件，`confidence` 反映了支持结果的实时数据量（具有可用 `load` 的 NF 数量），当没有数据可用时为 `0`。

架构概述

订阅存储保持一个热的内存 ETS 表，后备于一个 持久的 **DETS** 文件。订阅在重启后仍然有效：它们在启动时被重新加载，并且其定期通知被重新启动。

服务端点

OmniNWDAF 暴露以下 HTTP SBI 端点。所有路径在由 `sbi_scheme`、`sbi_addr` 和 `sbi_port` 配置的 SBI 绑定下提供（请参见 配置）。

方法	路径	
GET	`/nnwdaf-analyticsinfo/v1/analytics`	检索一个[`id` 选[
POST	`/nnwdaf-eventssubscription/v1/subscriptions`	创建一个[`noti` `even`
GET	`/nnwdaf-eventssubscription/v1/subscriptions/{subscriptionId}`	读取存储[
PATCH	`/nnwdaf-eventssubscription/v1/subscriptions/{subscriptionId}`	修改一个[中。
DELETE	`/nnwdaf-eventssubscription/v1/subscriptions/{subscriptionId}`	删除一个[
GET	`/nnwdaf-analyticsinfo/v1/status`	存活检查[

这些映射到 TS 29.520 中的 Nnwdaf 服务操作。提供创建、读取、修改（PATCH）和删除单个订阅的功能。

NRF 注册和心跳

在启动时，OmniNWDAF 以 NF 类型 `NWDAF` 在 `nrf_uri` 处向 NRF 注册，广告两个 Nnwdaf 服务（`nnwdaf-analyticsinfo` 和 `nnwdaf-eventssubscription`，均为 `v1`）和服务的 PLMN（`mcc`/`mnc`）。然后每 `heartbeat_interval` 毫秒发送一次定期心跳以保持配置文件的活跃。NRF 注册和心跳遵循 TS 29.510。

除了注册，OmniNWDAF 还将 NRF 作为数据源：当提供负载导向分析请求或通知时，它从 NRF 发现相关的 NF 实例，并读取它们报告的 `load` 和状态以计算结果（请参见 分析覆盖）。发现是尽力而为的：发现失败会导致空的 NF 集，受影响的分析返回 `confidence` 为 `0` 的结果，而不是错误。

除了注册和心跳，OmniNWDAF 还执行 NRF 发现，以读取其他 NFs 的负载和状态，以进行负载导向分析。它不消耗任何其他生产者接口。

Nnwdaf_AnalyticsInfo: 请求 / 响应

消费者通过调用 `GET /nnwdaf-analyticsinfo/v1/analytics` 并带上 `event-id` 查询参数来请求单个事件的分析。额外的查询参数（例如 S-NSSAI、DNN、SUPI、NF 类型）作为过滤器。

OmniNWDAF 返回一个 `AnalyticsData` 对象：

- `start` 和 `timeStampGen` 设置为当前时间。
- `expiry` 设置为当前时间加上 `analytics_validity_seconds`。
- 对于四个负载导向事件（`NF_LOAD`、`SLICE_LOAD_LEVEL`、`NSI_LOAD_LEVEL`、`NETWORK_PERFORMANCE`），OmniNWDAF 从 NRF 发现 NF 实例，并根据它们报告的负载和状态计算分析。`confidence` 反映具有可用负载数据的 NFs 数量（当没有时为 `0`）。
- 对于其余事件，调用者的过滤值在响应中回显。

工作原理：负载导向事件反映了 NRF 报告的实时 NF 人口。对于其余事件，响应携带标准时间戳并回显调用者的过滤值。

Nnwdaf_EventsSubscription: 订阅 / 通知

消费者通过 POST 到 `/nnwdaf-eventssubscription/v1/subscriptions` 创建一个订阅，主体包含 `notificationURI` 和一个或多个 `eventSubscriptions`。

创建时的行为：

- 订阅存储在热的 ETS 表中，并镜像到 持久的 **DETS** 文件，并分配一个订阅 ID。
- 返回 `201 Created`，并带有指向新订阅的 `Location` 头和响应中的存储主体。
- 订阅中不支持的事件会被静默丢弃。包含 没有支持事件 的订阅会被拒绝，返回 `400 MANDATORY_IE_MISSING`。缺少 `notificationURI` 也会被拒绝，返回 `400 MANDATORY_IE_MISSING`。
- 在创建时会立即向 `notificationURI` 发送通知（每个支持的订阅事件一个 `eventNotification`）。

定期通知。当订阅携带报告要求，具有 `repPeriod`（秒），无论是在订阅级别（`evtReq`）还是每个事件（`eventReq`），OmniNWDAF 会在该周期内重新发送通知，直到订阅被删除。找到的最小正 `repPeriod` 被使用。如果没有 `repPeriod`，则订阅为一次性：仅发送创建时的通知。没有阈值或数据变化触发。

读取和修改。可以使用 `GET .../subscriptions/{id}` 读取存储的订阅，并使用 `PATCH .../subscriptions/{id}` 修改。修改时将补丁主体浅合并到存储的订阅中（保留 `subscriptionId`），立即以更

新的定义重新通知，并重新启动重复计时器。

持久性。订阅及其定期通知在重启后仍然有效：存储在启动时从 DETS 文件重新加载，并���新启动重复计时器。

Consumer
OmniNWDAF
notificationURI
POST /nnwdaf-eventssubscription/v1/subscriptions {notificationURI, eventSubscriptions}
[redacted]
[没有 notificationURI 或没有支持的事件]
400 MANDATORY_IE_MISSING
[有效]
丢弃不支持的事件
存储在 ETS + 持久 DETS
分配 subscriptionId
201 Created (Location + 存储的主体)
POST 通知（分析）
2xx（已交付）或重试
[redacted]
[repPeriod 存在]
[redacted]
[每个重复周期直到删除]
POST 通知（当前分析）
PATCH /nnwdaf-eventssubscription/v1/subscriptions/{id} {patch}
[redacted]
[已知 id]
合并补丁，重新通知，重新启动计时器
200 OK（更新的订阅）
[未知 id]
404 Not Found
DELETE /nnwdaf-eventssubscription/v1/subscriptions/{id}
[redacted]
[已知 id]
204 无内容
[未知 id]
404 Not Found
Consumer
OmniNWDAF
notificationURI

通知交付和重试

每个通知（创建时的通知、每个定期通知和修改后的通知）都是 尽力而为 地交付，并具有有界的指数退避重试：

- 最多进行 `notification_max_attempts` 次交付尝试。

- 每次重试之前的延迟从 `notification_backoff_ms` 开始，并随着尝试次数的增加而指数增长。
- 如果在最后一次尝试后，订阅者仍未接受通知，OmniNWDAF 将放弃并记录警告。订阅仍然存储；不会自动删除。

有关重试参数，请参见 配置，有关交付尝试、重试和放弃如何在日志中显示，请参见 指标。

分析事件 ID 参考

OmniNWDAF 广告并接受来自 TS 29.520 的 `NwdafEvent` 枚举中的所有 16 个标准化分析事件 ID。

事件 ID	分析	参考
`NF_LOAD`	特定 NF 的负载分析	TS 29.520
`SLICE_LOAD_LEVEL`	网络切片的负载水平信息	TS 29.520
`NSI_LOAD_LEVEL`	网络切片实例负载水平	TS 29.520
`NETWORK_PERFORMANCE`	网络性能分析	TS 29.520
`SERVICE_EXPERIENCE`	服务体验分析	TS 29.520
`UE_MOBILITY`	UE 移动性分析	TS 29.520
`UE_COMMUNICATION`	UE 通信分析	TS 29.520
`QOS_SUSTAINABILITY`	QoS 可持续性分析	TS 29.520
`ABNORMAL_BEHAVIOUR`	异常行为分析	TS 29.520
`USER_DATA_CONGESTION`	用户数据拥塞分析	TS 29.520
`SM_CONGESTION`	会话管理拥塞分析	TS 29.520
`DISPERSION`	数据和移动性分散分析	TS 29.520
`RED_TRANS_EXP`	冗余传输体验分析	TS 29.520
`WLAN_PERFORMANCE`	WLAN 性能分析	TS 29.520
`DN_PERFORMANCE`	数据网络性能分析	TS 29.520
`PDU_SESSION_TRAFFIC`	PDU 会话流量分析	TS 29.520

在此集合之外的 `event-id` 在 AnalyticsInfo 上返回 `404 UNSUPPORTED_EVENT`，并在事件订阅中出现时被静默丢弃。

操作行为总结

行为	细节
负载导向分析	`NF_LOAD`、`SLICE_LOAD_LEVEL`、`NSI_LOAD_LEVEL`、`NETWORK_PERFORMANCE` 从实时 NRF 数据计算；`confidence` = 具有可用负载的 NFs 数量
其他分析内容	过滤器回显
分析时间戳	`start`/`timeStampGen` = 现在；`expiry` = 现在 + `analytics_validity_seconds`
订阅存储	热 ETS 表镜像到持久 DETS 文件；在重启时重新加载
订阅操作	创建、读取（GET）、修改（PATCH）、删除
订阅通知	创建时立即通知；当存在 `repPeriod` 时定期通知；修改时重新通知；没有阈值或数据变化触发
不支持的事件	在订阅中静默丢弃；在 AnalyticsInfo 上返回 `404 UNSUPPORTED_EVENT`
空订阅	被拒绝，返回 `400 MANDATORY_IE_MISSING`
通知交付	尽力而为，有界的指数退避重试；在最后一次尝试后记录放弃
NRF	注册、定期心跳和 NF 发现用于负载导向分析
数据收集	NRF 发现（每个 NF 的负载 + 状态）为负载导向分析提供数据

OmniNWDAF 故障排除

□述 | 操作 | 配置 | 指标

本指南涵盖常见问题及其解决方案。

常见问题

AnalyticsInfo 返回 400 MANDATORY_QUERY_PARAM_MISSING

症状：`GET /nnwdaf-analyticsinfo/v1/analytics` 返回 `400` 和 `MANDATORY_QUERY_PARAM_MISSING`。

可能原因：

- 缺少 `event-id` 查询参数。它是强制性的：分析始终是针对单个事件进行检索。

解决方案：

1. 添加一个 `event-id` 查询参数，设置为支持的事件 ID 之一（请参见 操作：分析事件 ID 参考）。

AnalyticsInfo 返回 404 UNSUPPORTED_EVENT

症状：`GET /nnwdaf-analyticsinfo/v1/analytics` 返回 `404` 和 `UNSUPPORTED_EVENT`。

可能原因：

- `event-id` 值不是 16 个标准化的分析事件 ID 之一。

解决方案：

1. 检查该值是否与 分析事件 ID 参考 相符。事件 ID 是区分大小写的，必须与 TS 29.520 的 `NwdafEvent` 名称完全匹配。

订阅创建返回 400 MANDATORY_IE_MISSING

症状：`POST /nnwdaf-eventssubscription/v1/subscriptions` 返回 `400` 和 `MANDATORY_IE_MISSING`。

可能原因：

- 请求体中没有 `notificationURI`。
- 请求体的 `eventSubscriptions` 中没有支持的事件。不支持的事件会被静默丢弃，因此仅列出不支持事件的请求体会被视为空并被拒绝。

解决方案：

1. 确保存在 `notificationURI`。
2. 确保至少有一个 `eventSubscriptions` 条目引用一个支持的事件 ID。

订阅者从未收到通知

症状：创建了一个订阅（`201`），但订阅者的 `notificationURI` 从未收到回调。

可能原因：

- `notificationURI` 从 OmniNWDAF 无法访问，或者订阅者在每次尝试时返回非 `2xx` 状态。
- 所有交付尝试（最多 `notification_max_attempts` 次）已用尽；OmniNWDAF 放弃并记录了警告。

解决方案：

1. 确认 `notificationURI` 从 OmniNWDAF 可访问，并且订阅者在接收时返回 `2xx`。
2. 检查 OmniNWDAF 日志以获取交付尝试和最终放弃警告（请参见 指标）。
3. 如���必要，增加 `notification_max_attempts` 或 `notification_backoff_ms`（请参见 配置）。错过的创建时间通知会根据交付计划进行重试；如果订阅携带 `repPeriod`，它也将在每个周期中重新尝试。否则，修改（`PATCH`）或重新创建订阅以触发新的交付。

订阅在重启后消失

症状：之前创建的订阅在读取或删除时返回 `404`，或在重启后不再存在。

可能原因：

- 订阅被持久化到耐用的 DETS 文件中，并在启动时重新加载。如果它们消失了，存储文件没有被保留：它可能位于临时存储中（默认路径在系统临时目录下），或者配置的 `subscription_store_path` 在运行之间发生了变化。

解决方案：

1. 将 `subscription_store_path` 指向跨重启持久化的耐用存储，并在运行之间保持路径稳定（请参见 配置）。
2. 如果存储文件丢失，重新创建受影响的订阅。

OmniNWDAF 不出现在 NRF 中

症状：消费者无法找到 OmniNWDAF，或者 NRF 显示没有 NWDAF 配置文件。

可能原因：

- `nrf_uri` 指向错误的 NRF 或无法访问的地址。
- 启动时注册失败（检查日志）。
- ���跳超时，NRF 过期了配置文件。

解决方案：

1. 验证 `nrf_uri` 是否正确且可访问（请参见 配置）。
2. 检查启动日志以确认注册成功。
3. 确认心跳正在运行；如果 `heartbeat_interval` 长于 NRF 的有效期，请缩短它。

DELETE 返回 404

症状：`DELETE /nnwdaf-eventssubscription/v1/subscriptions/{id}` 返回 `404`。

可能原因：

- 订阅 ID 是未知的，可能是因为它从未存在、已被删除，或者耐用存储文件在重启之间没有被保留。

解决方案：

1. 确认 ID 是否与创建时返回的 `Location` 头中的 ID 匹配。如果它在重启之前存在，请验证 `subscription_store_path` 是否指向耐用存储（请参见 订阅在重启后消失）。