

OmniEIR Configuration Reference

[Overview](#) | [Operations Guide](#) | [Metrics and Monitoring](#) | [Troubleshooting](#)

All OmniEIR configuration lives under the `:omnieir` application key. A subset of keys can be overridden by environment variables, but that only takes effect in the **production runtime configuration** (`config/runtime.exs`).

Configuration Structure

```
config :omnieir,
  # SBI listener identity (advertised to the NRF)
  sbi_scheme: "http",
  sbi_addr: "127.0.0.23",
  sbi_port: 7777,

  # NRF registration
  nrf_uri: "http://127.0.0.10:7777",
  heartbeat_interval: 10_000,

  # Served PLMN (override per deployment)
  mcc: "999",
  mnc: "70",

  # OmniHSS REST API holding the operator's EIR rules
  hss_api_base_url: "https://127.0.0.1:8443",

  # Verdict returned when no rule matches the presented equipment
  unknown_equipment_status: "WHITELISTED"
```

Core Parameters

Parameter	Type	Default	
sbi_scheme	string	"http"	EIR_S
sbi_addr	string	"127.0.0.23"	EIR_S
sbi_port	integer	7777	EIR_S
nrf_uri	string	"http://127.0.0.10:7777"	EIR_N
heartbeat_interval	integer (ms)	10_000	none
mcc	string	"999"	EIR_M
mnc	string	"70"	EIR_M
hss_api_base_url	string	"https://127.0.0.1:8443"	HSS_A
unknown_equipment_status	string	"WHITELISTED"	EIR_U

The self-signed TLS certificate of the lab OmniHSS is accepted (`verify_none`), matching the other HSS-REST consumers (OmniUDR). Terminate the HSS behind a trusted CA in production.

Provisioning the rules (OmniHSS)

OmniEIR has **no provisioning API of its own**. Rules are created against OmniHSS, exactly as for the 4G S13 interface, e.g.:

```
curl -k -X POST https://<hss>:8443/api/eir/rule \
-H 'content-type: application/json' \
-d '{"action":"blacklist","regex":"^35316005"}'
```

- `action` is one of `whitelist`, `blacklist`, `greylist`.
- `regex` is matched against the equipment identity string. Rules are evaluated in the order OmniHSS returns them; the first match wins.

AMF-side configuration (turning the check on)

The equipment check is performed by the **AMF**, and is **disabled by default** so existing deployments are unaffected. Enable it under the `:omniamf` key:

Parameter	Type	Default	Description
<code>eir_enabled</code>	boolean	<code>false</code>	When <code>true</code> , the AMF queries the 5G-EIR during registration and rejects blacklisted PEIs.
<code>eir_uri</code>	string nil	<code>nil</code>	Static 5G-EIR SBI URI. When <code>nil</code> , the AMF discovers the EIR via the NRF (NFType <code>5G_EIR</code>).

```
config :omniamf,
  eir_enabled: true
  # eir_uri: "http://127.0.0.23:7777"    # optional; omit to use
  NRF discovery
```

OmniEIR Metrics and Monitoring

[Overview](#) | [Configuration Reference](#) | [Operations Guide](#) | [Troubleshooting](#)

Prometheus endpoint

OmniEIR exposes a Prometheus scrape endpoint via

`TelemetryMetricsPrometheus` on `prometheus_metrics_port` (default **9572**):

```
GET http://<eir-host>:9572/metrics
```

All labels are low-cardinality (`status` / `result` / `nf_type`) so the series count stays bounded.

Metrics

Metric	Type	Labels	Description
omni_eir_equipment_check_count	counter	status	Equipment check count. EquipmentStatus (WHITELISTED/BLOCKED)
omni_eir_hss_request_count	counter	result	OmniHSS GET / outcome (success/failure)
omni_eir_hss_request_duration_ms	gauge	result	Latency of the rule read.
omni_eir_nrf_registration_status	gauge	nf_type	NRF registration status (success/failure = not).
beam_*	gauge	none	Standard BEAM process/port count

Underlying telemetry events

Event	Measurements	Metadata	Emitted by
<code>[:omnieir, :equipment_check]</code>	<code>count</code>	<code>status</code>	OmniEIR on each resolved verdict
<code>[:omnieir, :hss, :request]</code>	<code>count</code> , <code>duration_ms</code>	<code>endpoint</code> , <code>result</code>	OmniEIR on each <code>GET /api/eir/rule</code> to OmniHSS
<code>[:omniamf, :eir, :check]</code>	<code>count</code>	<code>result</code> (<code>:whitelisted</code> <code>:greylisted</code> <code>:blacklisted</code> <code>:unknown</code> <code>:error</code>)	The AMF on each equipment check

The AMF also emits its existing `[ :omniamf, :registration, :result ]` event with `result: :rejected` when a blacklisted PEI bars a registration. (The `[ :omniamf, :eir, :check ]` series is exposed on the **AMF's** Prometheus endpoint, not OmniEIR's.)

What to watch

- A rising rate of `[ :omnieir, :hss, :request ]` with `result: :failure` means OmniEIR cannot reach OmniHSS. The AMF then fails open (every device allowed). Alert on this.
- `[ :omniamf, :eir, :check ]` with `result: :blacklisted` counts barred devices; `:error` counts fail-open events at the AMF.

Logs

OmniEIR and the AMF log each decision. Notable lines:

- AMF: 5G-EIR returned BLACKLISTED for PEI <pei> (<supi>); barring registration
- AMF: 5G-EIR check failed for <supi> (<reason>); failing open
- OmniEIR: Could not read EIR rules from HSS: <reason>

OmniEIR Operations Guide

[Overview](#) | [Configuration Reference](#) | [Metrics and Monitoring](#) | [Troubleshooting](#)

Role

OmniEIR is the 5G Equipment Identity Register. During UE registration the AMF asks it whether the device's PEI (IMEI/IMEISV) is permitted; OmniEIR returns `WHITELISTED`, `BLACKLISTED`, or `GREYLISTED`. The verdict is computed from the operator's rules held in OmniHSS.

Endpoints

The SBI listener exposes the N5g-eir_EquipmentIdentityCheck service (TS 29.511):

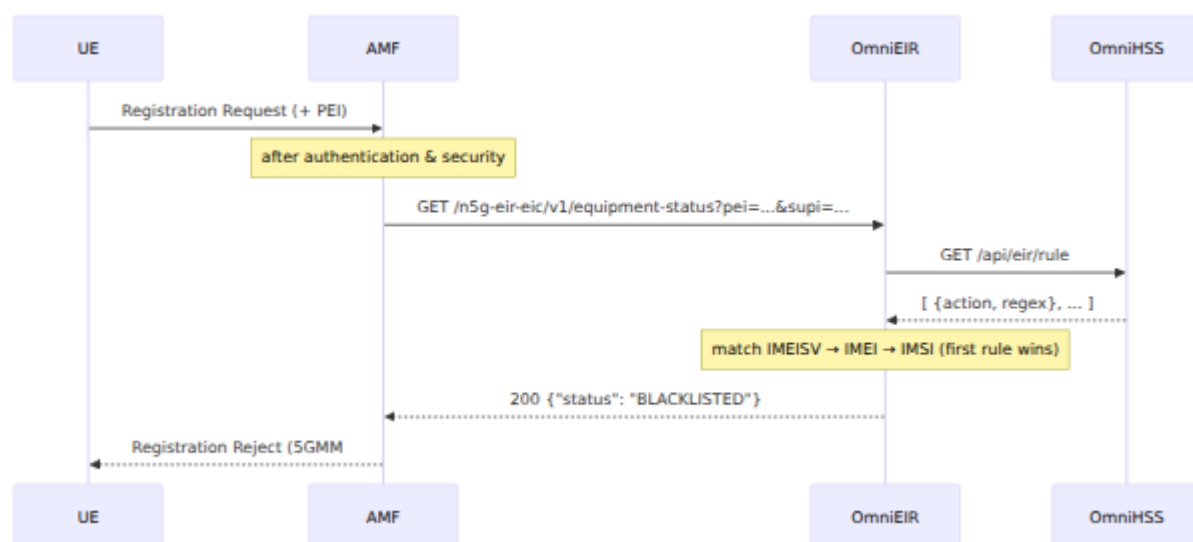
Method	Path	Description
GET	<code>/n5g-eir-eic/v1/equipment-status?pei={pei}&supi={supi}</code>	Returns <code>EirResponseData</code> : <code>{"status": "<EquipmentStatus>"}</code> . <code>pei</code> is mandatory; <code>supi</code> is optional.
GET	<code>/n5g-eir-eic/v1/status</code>	Liveness probe (<code>{"status": "UP"}</code>).

`pei` is a PEI in TS 23.003 form: `imei-<15 digits>` or `imeisv-<16 digits>`.
`supi` is `imsi-<digits>`.

Example:

```
curl 'http://127.0.0.23:7777/n5g-eir-eic/v1/equipment-status?
pei=imei-353160051234567'
# -> {"status":"WHITELISTED"}
```

Message Flow



For a **WHITELISTED** or **GREYLISTED** verdict (or any EIR error), the AMF proceeds with registration.

Matching logic

OmniEIR mirrors the 4G S13 decision so a single rule set governs both radios. Given the parsed equipment, it tries, in order, and stops at the first rule whose **regex** matches:

1. the full **IMEISV** digit string (IMEI + software version), then
2. the **IMEI** alone, then
3. the **IMSI** (from the SUPI).

If nothing matches, the configured **unknown_equipment_status** is returned.

Fail-open behaviour

The AMF-side gate is deliberately **fail-open**: if the EIR is unreachable, returns an error, or the UE presented no PEI, registration proceeds. Only an explicit `BLACKLISTED` verdict bars a device. This prevents a broken EIR or HSS from stranding every subscriber. A deployment that requires fail-closed semantics sets `unknown_equipment_status: "BLACKLISTED"` on OmniEIR (so unmatched devices are barred). Note that a total EIR outage still fails open at the AMF by design.

Enabling in a deployment

1. Provision rules in OmniHSS (`POST /api/eir/rule`).
2. Start OmniEIR; confirm it registers with the NRF as `5G_EIR`.
3. Set `config :omniamf, eir_enabled: true` and restart the AMF.
4. Verify with a phone reboot / UE re-attach and watch the AMF logs for the `5G-EIR` decision line.

OmniEIR

Troubleshooting

[Overview](#) | [Configuration Reference](#) | [Operations Guide](#) | [Metrics and Monitoring](#)

Common issues

Every device is allowed even though I provisioned a blacklist rule

Check, in order:

1. **Is the AMF check enabled?** The gate is off by default. Set `config:omniamf, eir_enabled: true` and restart the AMF. Without it the AMF never queries OmniEIR.
2. **Can OmniEIR reach OmniHSS?** A failure to read `GET /api/eir/rule` fails **open** (device allowed). Watch for `Could not read EIR rules from HSS` in the OmniEIR logs and `[ :omnieir, :hss, :request ]` telemetry with `result: :failure`. Confirm `hss_api_base_url`.
3. **Does the regex match the PEI from the UE presents?** The rule regex is matched against the digit string. A rule written for a 14-digit IMEI will not match a 15-digit `imei-` PEI unless the regex is anchored to allow it. Test the regex against the exact PEI digits.
4. **Did the AMF actually receive a PEI?** If the UE presented no PEI, the AMF logs `EIR enabled but UE ... presented no PEI; allowing` and proceeds.

The AMF rejects a device I did not intend to bar

- A rule's `regex` is broader than intended (e.g. `^3531` matches a whole TAC range). Narrow it.
- `unknown_equipment_status` is set to `BLACKLISTED` (fail-closed) and the device matches no rule. Either add a `whitelist` rule for it or revert to the

default `WHITELISTED`.

400 MANDATORY_QUERY_PARAM_INCORRECT

The `pei` query parameter was missing from the request to `/n5g-eir-eic/v1/equipment-status`. It is mandatory (TS 29.511).

500 SYSTEM_FAILURE from OmniEIR

OmniEIR could not read the rule set from OmniHSS. This is an upstream (HSS) problem; the AMF treats a non-2xx as fail-open. Check OmniHSS health and `hss_api_base_url`.

Architecture notes

OmniEIR has no provisioning API of its own: the IMEI allow/block/grey rules live in OmniHSS (one rule set shared with the 4G S13 interface), and OmniEIR reads them over the HSS REST API and applies the identical matching logic. The AMF gate is fail-open by design (see [Operations](#)), and the N5g-eir service is a single equipment-status Get per TS 29.511. Terminate TLS at an external proxy in front of the SBI listener.

