

Metrics Verbosity Behaviour Changes

Scope

The behaviour changes in this chapter apply to these components:

- app-infra
- cn-ipam
- bng-dhcp
- bng-common
- radius-ep
- OAM and YANG

- Metrics verbosity framework, on page 1
- Metric gating changes by component, on page 3
- OAM and CLI behaviour changes, on page 6
- Configure application verbosity to enable Debug and Trace metrics, on page 6

Metrics verbosity framework

A metrics verbosity framework is a monitoring control mechanism that

- classifies application-layer metrics into verbosity levels,
- determines emission of metrics during a Prometheus scrape cycle, and
- enables configuration of metric visibility using OAM CLI and ConfigMap settings.

Each pod uses the **app-infra metrics verbosity framework** to determine whether a Prometheus metric is emitted during a scrape cycle. Metrics are classified into the following verbosity levels:

Table 1: Verbosity levels

Level	Integer value	Description
Production	0	Essential production metrics; always emitted.

Level	Integer value	Description
Debug	1	Investigation-oriented metrics; emitted when verbosity level is greater than or equal to Debug.
Trace	2	High-frequency or verbose metrics; emitted when verbosity level is greater than or equal to Trace.
Off	-1	Never emitted.

Verbosity level for each pod is configured through the OAM CLI and stored in the `infra-system-conf` ConfigMap. The **podType** field determines which type of pod is targeted.

Table 2: Verbosity configuration by pod type

podType	Applies to	Controls
service	cn-ipam, bng-dhcp, bng-smf, ...	Infra-internal metrics (framework metrics)
protocol	bng-n4-protocol, ...	Infra-internal metrics
load-balancer	LB pods	Infra-internal metrics
application	All pods	Application-layer metrics (for example, ipam-pool-stats, dhcp-* metrics)

Note A single pod responds to two verbosity settings: one for its podType (such as **service**) that controls infrastructure-internal metrics, and another for application, which controls application-layer metrics. If you want to adjust both types of metrics, you must configure both verbosity settings.

From Release 2026.02.0 onwards, the fallback level for application-layer metrics changed when no `podType: application` entry is present in the ConfigMap.

In `app-infra/src/app-infra/infra/Types.go`, the fallback level changed from

```
DefApplicationVerboseLevel = MetricsVerboseLevelDebug
```

to

```
DefApplicationVerboseLevel = MetricsVerboseLevelProduction.
```

Table 3: Application metric behaviour when no application entry is configured

Application metric gate	Before Release 2026.02.0	From Release 2026.02.0 onwards
Production (0)	Emitted	Emitted (no change)
Debug (1)	Emitted (default was Debug)	Suppressed (default is now Production)
Trace (2)	Suppressed	Suppressed (no change)



Important If your deployment previously showed application-level debug metrics without a configmap entry, you now need to explicitly add a `podType: application` entry to make these metrics visible.

Metric gating changes by component

Use the following tables to review metric gating changes for each component.

cn-ipam

File: `src/ipam/statsAPI.go`

Metrics that were previously always emitted now require configuration to be enabled.

Prometheus Metric	Old Behaviour	New Behaviour	Configuration Needed to Restore
`ipam_address_events_total`	Always emitted (no gate)	Emitted only when application level ≥ **Debug**	`podType: application, level: debug`
`ipam_static_request_statistics`	Always emitted (no gate)	Emitted only when application level ≥ **Debug**	`podType: application, level: debug`
`ipam_static_db_statistics`	Always emitted (no gate)	Emitted only when application level ≥ **Debug**	`podType: application, level: debug`
`ipam_threshold_event`	Always emitted (no gate)	Emitted only when application level ≥ **Debug**	`podType: application, level: debug`
`records_in_cdl_for_startrange`	Always emitted (no gate)	Emitted only when application level ≥ **Debug**	`podType: application, level: debug`
`ipam_msg_turn_around_time`	Always emitted (no gate)	Emitted only when application level ≥ **Trace**	`podType: application, level: trace`
`ipam_address_upf_total`	Always emitted (no gate)	Emitted only when application level ≥ **Debug**	`podType: application, level: debug`



Note

- The `metrics id_manager_cp_audit_validate_key_id` and `id_manager_key_service_statistics` are production-gated and remain unchanged—they continue to be emitted by default.
- The metric `ipam_handle_start_terminate` has its record function commented out, so it is registered but never emitted; there is no change to its behavior.

bng-dhcp

File: `bng-dhcp/images/bng_dhcp/src/bng-dhcp/metrics/metrics.go`

Six metrics that were previously always emitted are now gated at the Trace level. To emit these metrics, you must set **podType: application** and **level: trace** in the configuration.

Prometheus Metric	Old Behaviour	New Behaviour
`dhcp_ha_packet_drop_stats`	Always emitted (no gate)	Emitted only at application level ≥ **Trace**
`dhcp_n4_ipc_total`	Always emitted (no gate)	Emitted only at application level ≥ **Trace**
`dhcp_pkt_aborted_total`	Always emitted (no gate)	Emitted only at application level ≥ **Trace**
`dhcp_sess_update_reason`	Always emitted (no gate)	Emitted only at application level ≥ **Trace**
`dhcp_v4_packet_error_stats`	Always emitted (no gate)	Emitted only at application level ≥ **Trace**
`dhcp_up_inactive_packet_drop_stats`	Always emitted (no gate)	Emitted only at application level ≥ **Trace**

bng-common

File: `bng-common/src/bng-common/reconcp/stats.go`

Prometheus Metric	Old Behaviour	New Behaviour	Configuration Needed
`recon_cp_events_total`	Always emitted (no gate)	Emitted only at application level ≥ **Debug**	`podType: application, level: debug`

radius-ep

File: `radius-ep/images/radius_ep/src/radius-ep/app/statsAPI.go`

Prometheus Metric	Old Behaviour	New Behaviour
`radius_least_out_standing_server`	Suppressed unless application level ≥ Debug	**Always emitted** — now Production level

Summary

Component	Prometheus Metric	Before (level / gate)	After (level / gate)	Visible by Default?
cn-ipam	`ipam_address_events_total`	No gate — always emitted	Debug gate	**No (was Yes)**
	`ipam_static_request_statistics`	No gate — always emitted	Debug gate	**No (was Yes)**
	`ipam_static_db_statistics`	No gate — always emitted	Debug gate	**No (was Yes)**
	`ipam_threshold_event`	No gate — always emitted	Debug gate	**No (was Yes)**
	`records_in_cdl_for_startrange`	No gate — always emitted	Debug gate	**No (was Yes)**
	`ipam_msg_turn_around_time`	No gate — always emitted	Trace gate	**No (was Yes)**
	`ipam_address_upf_total`	No gate — always emitted	Debug gate	**No (was Yes)**
bng-dhcp	`dhcp_ha_packet_drop_stats`	No gate — always emitted	Trace gate	**No (was Yes)**
	`dhcp_n4_ipc_total`	No gate — always emitted	Trace gate	**No (was Yes)**
	`dhcp_pkt_aborted_total`	No gate — always emitted	Trace gate	**No (was Yes)**
	`dhcp_sess_update_reason`	No gate — always emitted	Trace gate	**No (was Yes)**
	`dhcp_v4_packet_error_stats`	No gate — always emitted	Trace gate	**No (was Yes)**
	`dhcp_up_inactive_packet_drop_stats`	No gate — always emitted	Trace gate	**No (was Yes)**
bng-common	`recon_cp_events_total`	No gate — always emitted	Debug gate	**No (was Yes)**
radius-ep	`radius_least_out_standing_server`	Debug gate — suppressed by default	No gate (Production)	**Yes (was No)**

Note 14 metrics that were previously always visible are now suppressed by default. One metric, `radius_least_out_standing_server`, which was previously suppressed, is now always visible.

OAM and CLI behaviour changes

Behaviour changes:

- That YANG default has been removed.

  The `level` leaf in `tailf-mobile-infra.yang` previously had the default value `trace`. This default has been removed. The render template now owns the default.

New render template defaults

When an operator configures a verboseLevel entry without specifying the level, the render template now automatically applies a default value based on the type.

podType	Level when omitted (Before)	Level when omitted (After)
`service`	trace (2) via YANG default	trace (2) via render — *no change*
`protocol`	trace (2) via YANG default	trace (2) via render — *no change*
`load-balancer`	trace (2) via YANG default	trace (2) via render — *no change*
`application`	trace (2) via YANG default	**production (0)** via render — *changed*

Note **Operator Impact:**

Any existing CLI or OAM script that configures metrics `verbose-levels pod-type application` without specifying a level will now default to LEVEL: PRODUCTION (0) instead of TRACE (2). To keep the previous behavior, you must explicitly set `level trace`.

Configure application verbosity to enable Debug and Trace metrics

Procedure

Step 1 To restore visibility of cn-ipam Debug metrics (5 metrics), execute the following command on the OAM CLI:

Example:

```
metrics verbose-levels pod-type application level debug
```

This enables: `ipam_address_events_total`, `ipam_static_request_statistics`, `ipam_static_db_statistics`, `ipam_threshold_event`, and `records_in_cdl_for_startrange`.

Step 2 To restore visibility of cn-ipam Trace metric (1 metric), execute the following command:

Example:

```
metrics verbose-levels pod-type application level trace
```

This enables all cn-ipam debug metrics, as well as the `ipam_msg_turn_around_time` metric.

Step 3 To restore visibility of bng-dhcp Trace metrics (6 metrics), execute the following command on the bng-dhcp pods:

Example:

```
metrics verbose-levels pod-type application level trace
```

This enables `dhcp_ha_packet_drop_stats`, `dhcp_n4_ipc_total`, `dhcp_pkt_aborted_total`, `dhcp_sess_update_reason`, `dhcp_v4_packet_error_stats`, and `dhcp_up_inactive_packet_drop_stats` metrics.

Step 4 To restore visibility of bng-common Debug metric (1 metric), execute the following command:

Example:

```
metrics verbose-levels pod-type application level debug
```

This enables `recon_cp_events_total` metric.

Step 5 For `radius-ep` metrics, no CLI configuration is needed.

The metric `radius_least_out_standing_server` is always visible.

Step 6 After executing these commands, your ConfigMap structure should contain the following:

Example:

```
metrics:
  verboseLevels:
  - podType: service       # infra-internal metrics for service pods
    level: 2               # trace
  - podType: application   # application metrics (cn-ipam, bng-dhcp, etc.)
    level: 2               # trace — restores all Debug + Trace app metrics
```



Note Changes take effect immediately (live reload via file watcher); no pod restart required.